

CASUAL INTERACTION

Devices and Techniques for Low-Engagement Interaction



Von der Fakultät für Elektrotechnik und Informatik
der Gottfried Wilhelm Leibniz Universität Hannover
zur Erlangung des Grades

DOKTOR DER NATURWISSENSCHAFTEN
— Dr. rer. nat. —

genehmigte Dissertation

von

HENNING POHL, M.Sc.



2017

KOMMISSION

REFERENT

Prof. Dr. Michael Rohs

KOREFERENT

Prof. Dr. Roderick Murray-Smith

TAG DER PROMOTION

7. März 2017

Abstract

Interactive systems in use today are commonly built around an assumption of focused and engaged effort of the user. However, many situations call for a less engaged, more casual way for users to control their devices. Consider, for example, the difference between sitting down at a desk to write an email and sending a quick text while on public transport. In the former scenario, focused interaction is much more likely while the later scenario can benefit from interaction that is less demanding. Such lower-engagement interactions form the basis for casual interaction.

Users might desire less engaged interactions because they are occupied, are tired, focused elsewhere, or just lazy. We explore these and other reasons that make lower-engagement interaction desirable. In particular, we also look at evidence that users see a need or desire systems that allow them to engage less. In the process of engaging less, users delegate some control to the system. Hence, casual interaction systems require models to *fill in* for users that are disengaged. What form this takes depends on the specific system. For example, we explore a text entry system where users can adapt the level of correction applied to their texts. In this case, the underlying model incorporates language-based and touch-based input disambiguation to ease the workload on users should they desire correction of their texts.

Starting from a conceptual view of casual interaction, this dissertation describes a range of concrete instantiations of casual interaction systems. This work spans from around-device interaction, via smart homes and text input, to systems that offer more casual forms of output. How to translate the desire for lower-engagement devices to actual systems has no straightforward answer. Hence, each system described herein draws upon engagement and control aspects specific to the target scenario for design of lower-engagement alternatives.

With a growing number of interactive devices on and around users, focused interaction with all of them at all times is not sustainable. Casual interaction strives to empower users to choose lower-engagement interactions as they see fit and lower their interaction burden accordingly.

KEYWORDS:

Casual interaction, Interactive Systems, User Engagement, Control, User Interfaces

Zusammenfassung

Interaktive Systeme sind häufig unter der Annahme entworfen, dass Nutzer bei ihrer Bedienung aufmerksam und mit vollem Einsatz vorgehen. Es gibt allerdings Situationen in denen eine legerere Interaktion angemessener wäre. Zum Vergleich eignen sich hier das Schreiben einer Email am Schreibtisch und das Versenden einer kurzen Textnachricht während der Fahrt in der U-Bahn. Während das Emails Schreiben gut zum fokussierten Arbeiten passt wäre eine weniger fordernde Interaktion für die U-Bahn angemessener. Solche legeren Interaktionen wollen wir hier näher untersuchen.

Die Gründe aus denen Nutzer legerere Interaktion vorziehen könnten sind vielfältig. Nutzer welche beschäftigt sind, müde sind, abgelenkt sind, oder einfach nur faul sind, haben Vorteile von Geräten die kein konzentriertes oder aufwendiges Arbeiten bedingen. In dieser Dissertation schauen wir uns diese und weitere Gründe für legerere Interaktion genauer an. Vor allem Nachweise für einen Wunsch oder Zwang zu solcher Interaktion stehen dabei im Vordergrund.

Im Zuge einer legereren Interaktion geben Nutzer zwangsläufig Kontrolle an das System ab. Eine entscheidende Frage solcher Systeme ist daher welche Modelle genutzt werden können, um Nutzer, die weniger in die Interaktion involviert sein wollen, zu unterstützen. Die Art des zu verwendenden Modells hängt dabei stark vom spezifischen System ab. Ein Beispielszenario in dieser Dissertation ist ein Autokorrektursystem, bei dem Nutzer den Grad der Korrektur steuern können. In diesem Fall werden ein Sprachmodell und ein Modell der Eingabe auf Touchscreens herangezogen, um Benutzereingaben zu korrigieren, falls diese sich weniger selber um korrekte Buchstabierung kümmern wollen.

In dieser Dissertation werfen wir zunächst einen Blick auf die Konzepte hinter legerer Interaktion. Darauf aufbauend stehen dann konkrete Beispiele von Systemen der legeren Interaktion im Fokus. Diese kommen aus einem breiten Spektrum an Anwendungsfeldern wie Smart Homes, Texteingabe, Eingabe im freien Raum um Geräte herum, oder legerer Ausgabe. Die Überführung von Konzepten der legeren Interaktion zu tatsächlich existierenden Systemen ist dabei komplex und beeinflusst von den spezifischen Anforderungen der Szenarien an Kontrolle und Nutzereinbindung.

Mit der wachsenden Anzahl an interaktiven Geräten an und um uns herum wird konzentrierte und fokussierte Interaktion zu jeder Zeit nicht mehr aufrechtzuerhalten sein. Legere Interaktion versucht daher Nutzer zu ermächtigen, an Stellen wo sie es als passend erachten, Kontrolle abzugeben und damit die ihre Interaktionslast abzumildern.

SCHLAGWÖRTER:

Legere Interaktion, Interaktive Systeme, Nutzereinsatz, Kontrolle, Benutzerschnittstellen

Acknowledgments

First of all, I would like to thank Michael for the four years in Hannover. This job was a great chance to help build up a new lab, while also moving back closer to friends & family. You gave me the freedom to pursue a wide range of my research ideas, providing ample support where needed.

Many thanks also to Rod who got me started on my work on casual interaction. Much of the work presented here goes back to meetings and conversations back in Glasgow. Me continuing on towards the PhD is in no small part due to Rod giving me the chance to spend the summer over there on a vacation scholarship.

I would also like to thank my collaborators over the years, in particular Christian Holz, Markus Krause, Daryl Weir, and Simon Rogers. Among my co-authors, there are also many students who helped with papers, often as part of their own thesis. In particular, I would like to highlight the great contributions by Christian Domin, Dennis Stanke, Justyna Medrek, Peter Brandes, and Sven Greiner. But my thanks also extends to all the other students with whom I worked together on theses or class projects.

I was very fortunate to have had the chance to discuss my research agenda with several people that provided great feedback. In particular, I remember engaging lunch and floor meetings with Daniel Ashbrook and Jürgen Steimle. I also enjoyed a few stimulating conversations over conference dinners and would like to especially thank Vincent Levesque and Aaron Quigley for those. Finally, I would like to thank all the organizers and fellow participants of the CHI 2015 doctoral consortium, in particular Alice Oh, for the insightful two days.

Thanks also to the many lab mates, visitors, and conference buddies who I have had a chance and pleasure to hang out with over the years: Sean Gustafson, Stephan Richter, John Williamson, Julie Williamson, Max Pfeiffer, Tim Dünthe, Daniel Buschek, Christian Loclair, Liwei Chan, Allesandro Mulloni, Eric Marsh, Alyson Young, Florian Daiber, Daniel Boland, Frederik Rudeck, Torsten Becker, Christian Steins, Florian Heller, Hugues Salamin, Don McMillan, Markus Löchtefeld, Ross McLachlan, Lauren Norrie, and many others.

Finally, I would like to thank my family and friends for their continued support and encouragement. Special thanks to Mareike Leppin for proofreading this dissertation.

Foreword

This dissertation is the result of several years of research on casual interaction concepts and systems. Hence, it brings together a diverse set of projects in one document. Many of these projects have already been previously published in conference proceedings, as book chapters, or as journal articles. These previous publications are incorporated, extended, and put into a larger context here. Namely, this dissertation includes:

- Henning Pohl and Roderick Murray-Smith. “Focused and Casual Interactions: Allowing Users to Vary Their Level of Engagement.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '13*. New York, New York, USA: ACM Press, 2013, pp. 2223–2232. DOI: 10.1145/2470654.2481307
- Daryl Weir, Henning Pohl, Simon Rogers, Keith Vertanen, and Per Ola Kristensson. “Uncertain Text Entry on Mobile Devices.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '14*. New York, New York, USA: ACM Press, 2014, pp. 2307–2316. DOI: 10.1145/2556288.2557412
- Henning Pohl and Michael Rohs. “Around-Device Devices: My Coffee Mug is a Volume Dial.” In: *Proceedings of the 16th International Conference on Human-Computer Interaction with Mobile Devices and Services - MobileHCI '14*. 2014. DOI: 10.1145/2628363.2628401
- Henning Pohl, Michael Rohs, and Roderick Murray-Smith. “Casual Interaction: Scaling Fidelity for Low-Engagement Interactions.” In: *Workshop on Peripheral Interaction: Shaping the Research and Design Space at CHI 2014*. 2014
- Henning Pohl. “Casual Interaction: Scaling Interaction for Multiple Levels of Engagement.” In: *CHI '15 Extended Abstracts on Human Factors in Computing Systems - CHI EA '15*. New York, New York, USA: ACM Press, 2015, pp. 223–226. DOI: 10.1145/2702613.2702625
- Henning Pohl, Dennis Becke, Eugen Wagner, Maximilian Schrapel, and Michael Rohs. “Wrist Compression Feedback by Pneumatic Actuation.” In: *CHI '15 Extended Abstracts on Human Factors in Computing Systems on - CHI EA '15*. 2015. DOI: 10.1145/2702613.2725427

- Henning Pohl, Markus Krause, and Michael Rohs. “One-Button Recognizer: Exploiting Button Pressing Behavior for User Differentiation.” In: *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing - UbiComp '15*. New York, New York, USA: ACM Press, 2015, pp. 403–407. DOI: 10.1145/2750858.2804270
- Henning Pohl. “Casual Interaction: Moving Between Peripheral and High Engagement Interactions.” In: *Peripheral Interaction: Challenges and Opportunities for HCI in the Periphery of Attention*. Ed. by Saskia Bakker, Doris Hausen, and Ted Selker. Berlin, Heidelberg: Springer, 2016, pp. 117–135. DOI: 10.1007/978-3-319-29523-7_6
- Henning Pohl, Dennis Stanke, and Michael Rohs. “EmojiZoom: Emoji Entry via Large Overview Maps 🗺️.” In: *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services companion - MobileHCI '16*. 2016. DOI: 10.1145/2935334.2935382
- Henning Pohl, Justyna Medrek, and Michael Rohs. “ScatterWatch: Subtle Notifications via Indirect Illumination Scattered in the Skin.” In: *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services companion - MobileHCI '16*. 2016. DOI: 10.1145/2935334.2935351
- Henning Pohl, Bastian Krefeld, and Michael Rohs. “Multi-Level Interaction with an LED-Matrix Edge Display.” In: *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services adjunct - MobileHCI '16 Adjunct*. 2016. DOI: 10.1145/2957265.2961855
- Henning Pohl, Peter Brandes, Hung Ngo Quang, and Michael Rohs. “Squeezeback: Pneumatic Compression for Notifications.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '17*. New York, New York, USA: ACM Press, 2017. DOI: 10.1145/3025453.3025526
- Henning Pohl, Franziska Hoheisel, and Michael Rohs. “Inhibiting Freedom of Movement with Compression Feedback.” In: *CHI '17 Extended Abstracts on Human Factors in Computing Systems - CHI EA '17*. New York, New York, USA: ACM Press, 2017. DOI: 10.1145/3027063.3053081
- Henning Pohl, Christian Domin, and Michael Rohs. “Beyond Just Text: Semantic Emoji Similarity Modeling to Support Expressive Communication 🗺️📱😊.” In: *ACM Transactions on Computer-Human Interaction* 24.1 (2017). DOI: 10.1145/3039685

Furthermore, I have supervised many student theses during my time in Hannover, each exploring a specific casual interaction aspect. Some of these theses have led to publications, but all of them have led to new insights and perspectives. These theses are:

- Eike Clas Schulze. “Punktwolkenvisualisierung mittels Bewegungsparallaxe.” Master Thesis. Leibniz Universität Hannover, 2014
- Sven Karsten Greiner. “Direkte Interaktion im Nahbereich mobiler Geräte.” Bachelor Thesis. Leibniz Universität Hannover, 2014
- Karoline Busse. “Casual Interaction with a Bracelet.” Master Thesis. Leibniz Universität Hannover, 2014
- Christoph Manuel Lenz. “Eye Tracking für die beiläufige Interaktion.” Bachelor Thesis. Leibniz Universität Hannover, 2015
- Bastian Krefeld. “Variable Ausgabeauflösung für den Nahbereich Mobiler Geräte.” Bachelor Thesis. Leibniz Universität Hannover, 2015
- Franziska Hoheisel. “Kompressions-Feedback für Pervasive Games.” Bachelor Thesis. Leibniz Universität Hannover, 2015
- Philipp Seelig. “Attribute-Driven Soft Keyboard for Emoji Entry.” Bachelor Thesis. Leibniz Universität Hannover, 2015
- Justyna Medrek. “Indirekte Beleuchtung für Smartwatch-Benachrichtigungen.” Bachelor Thesis. Leibniz Universität Hannover, 2015
- Dennis Stanke. “Ein Zoombares Emoji Keyboard.” Bachelor Thesis. Leibniz Universität Hannover, 2016
- Sven Röttering. “Casual Interaction with a Smartwatch.” Master Thesis. Leibniz Universität Hannover, 2016
- Peter Brandes. “Interaction with Compression Feedback on the Wrist.” Bachelor Thesis. Leibniz Universität Hannover, 2016
- Marco Deneke. “Sammeln von Emoji-Bewertungen mit einem kompetitiven Spiel.” Bachelor Thesis. Leibniz Universität Hannover, 2016
- Sezer Dursun. “Grob- und Feininteraktion mit einem ambienten Display.” Bachelor Thesis. Leibniz Universität Hannover, 2016
- Hung Ngo Quang. “Compression Feedback for Notifications.” Master Thesis. Leibniz Universität Hannover, 2016

Contents

1	Introduction	1
1.1	Contributions	3
1.2	Dissertation Structure	3
2	Casual Interaction	5
2.1	Overview	5
2.2	Defining Casual	7
2.2.1	Definitions of Casual in Other Works	8
2.3	Defining Engagement	10
2.4	Defining Control	13
2.4.1	The H-Metaphor	14
2.4.2	An Information-Theoretic View	15
2.4.3	Dealing with Reduced Control	16
2.5	Related Concepts	17
2.5.1	Levels of Interaction	18
2.6	Reasons for Casual Interaction	19
2.6.1	Physical Reasons for Casual Interaction	19
2.6.2	Social Reasons for Casual Interaction	20
2.6.3	Mental Reasons for Casual Interaction	22
2.7	Evidence for Least-Effort Behavior in Interaction	24
2.7.1	Apparatus	25
2.7.2	Participants	27
2.7.3	Results	28
2.7.4	Discussion	29
2.8	In Situ Qualitative Study of Control and Effort	29
2.8.1	Method	30
2.8.2	Results	34
2.8.3	Discussion	38
2.9	Towards Casual Interaction Systems	39
3	Interacting Around a Device	41
3.1	Introduction	42
3.2	Related Work	43
3.2.1	Sensing Around-Device Input	43
3.2.2	Interaction Techniques for Around-Device Interaction	45

3.2.3	Around-Device Tangibles	46
3.3	Moving Mobile Interaction to Proxies	47
3.3.1	Around-Device Devices	48
3.3.2	Around-Device Device Types	51
3.4	Environments for Around-Device Interactions	54
3.4.1	Envisioned Settings	54
3.4.2	Phone Context Ground Truth	55
3.4.3	Discussion	58
3.5	Eliciting Gestures for Around-Device Devices	60
3.5.1	Participants	60
3.5.2	Task	60
3.5.3	Procedure	61
3.5.4	Results	62
3.5.5	Discussion	64
3.6	Notification Access Around a Phone	65
3.6.1	Prototype	66
3.6.2	Interaction Design	68
3.6.3	Evaluation	69
3.6.4	Discussion	72
3.6.5	Summary	72
3.7	Around-Device Interaction Performance	73
3.7.1	Apparatus	74
3.7.2	Design	75
3.7.3	Participants	76
3.7.4	Procedure	76
3.7.5	Results	77
3.7.6	Discussion	81
3.7.7	The Influence of the Visualization	82
3.7.8	Summary	83
3.8	Chapter Summary	83

4 Casual Interaction in the Smart Home 85

4.1	Introduction	86
4.2	Related Work	87
4.2.1	Smart Homes	87
4.2.2	Low-effort User Recognition	88
4.2.3	Sensing Furniture	89
4.3	Low-Effort User Recognition with One Button	90
4.3.1	One-Button Recognizer	91
4.3.2	Button Prototypes	92
4.3.3	Feature Extraction	93
4.3.4	Training the Recognizer	94

4.3.5	Evaluating Recognition Performance	95
4.3.6	Evaluating Prolonged Usage	96
4.3.7	Summary	99
4.4	Augmenting Furniture for Casual Interaction	100
4.4.1	A Prototype System for the Smart Living Room	100
4.4.2	Evaluation	101
4.4.3	Controlling Devices via Postures	102
4.4.4	Summary	103
4.5	Changing Lighting with Varying Control	104
4.5.1	Casual Interaction Bracelet Prototype	106
4.5.2	Electronics for the Light Control Bracelet	107
4.5.3	Input Methods for the Light Control Bracelet	109
4.5.4	Design Testing	110
4.5.5	Summary	111
4.6	Chapter Summary	111
5	Continuous Casual Control for Autocorrection	113
5.1	Introduction	114
5.2	Related Work	114
5.2.1	Text Entry Correction Approaches	115
5.2.2	Offset Modelling	115
5.2.3	Pressure Input	115
5.2.4	Autonomy Handover	116
5.3	Implicit Text Correction: GPType	116
5.3.1	Language Model	117
5.3.2	Touch Model	117
5.3.3	Decoder	118
5.4	The Autocorrect Trap	119
5.5	ForceType: Pressure as Certainty	122
5.5.1	Correction Model	123
5.6	Evaluation	124
5.6.1	Participants	124
5.6.2	Apparatus	125
5.6.3	Procedure	126
5.6.4	Results	127
5.7	Discussion	129
5.8	Summary	129
6	Casual Communication	131
6.1	A Definition of Casual Communication	132
6.2	Case Study: Apple Watch	133
6.3	Casual Messaging	136

6.4	Introduction to Emoji	137
6.4.1	The History of Emoji	138
6.4.2	The Nature of Emoji	140
6.4.3	A Combinatorial Explosion of Emoji	142
6.4.4	Common Usage of Emoji	143
6.5	Quantifying Emoji Usage	144
6.6	Evaluating the State of the Art of Emoji Entry	148
6.6.1	Challenges When Testing Emoji Keyboards	150
6.6.2	Participants	151
6.6.3	Procedure	152
6.6.4	Results	152
6.6.5	Discussion	156
6.7	Input Methods for Emoji & Large Character Sets	157
6.7.1	Enumeration	159
6.7.2	Querying and Prediction	159
6.7.3	Methods for East Asian Languages: Mapping to Latin Script	162
6.8	EmojiZoom: A Novel Emoji Entry Method	163
6.8.1	Related Work	164
6.8.2	Implementation	164
6.8.3	Evaluation	166
6.8.4	Longitudinal Evaluation	171
6.8.5	Emoji-Level Analysis	172
6.8.6	Summary	175
6.9	Motivating Emoji Similarity Modeling	176
6.10	Towards a Model of Emoji Similarity	178
6.10.1	Deriving Emoji Similarity From Unicode Annotations	178
6.10.2	Emoji Model Building from Tweets	179
6.10.3	Evaluating Model Performance	187
6.10.4	Comparing Tag-Based and Semantic Emoji Models	192
6.11	Summary	195
6.11.1	Outlook: Beyond Emoji	197
7	Feedback in Casual Interaction	201
7.1	What Makes Feedback Casual?	201
7.2	Implementing Casual Feedback	203
7.3	Related Work in Casual Output	203
7.3.1	Subtle Feedback	204
7.3.2	Feedback with Multiple Complexity Levels	204
7.4	Towards Specific Casual Output Systems	205
8	Casual Feedback: Indirect Light Feedback	207
8.1	Introduction	208

8.2	Related Work	209
8.2.1	LEDs for Feedback	209
8.2.2	Novel Feedback on the Wrist	209
8.3	Prototype	210
8.3.1	Choice of Light Color	212
8.3.2	Feedback Modes	214
8.4	Evaluation	217
8.4.1	Participants	219
8.4.2	Procedure	219
8.4.3	Apparatus	219
8.5	Results	220
8.5.1	Influence of Illumination Mode	222
8.5.2	Influence of Activity and Surroundings	224
8.5.3	Qualitative Ratings	226
8.5.4	Discussion	227
8.6	Summary	229

9 Casual Feedback: Compression Feedback 231

9.1	Introduction	232
9.2	Related Work	233
9.2.1	Pneumatic Actuation	233
9.2.2	Pressure Feedback	233
9.2.3	Pneumatic Input	234
9.3	Compression Feedback	234
9.4	Compression Feedback Systems	235
9.4.1	How Air-Pressure Relates to On-Arm Force	238
9.5	Properties of Compression Feedback	238
9.5.1	Background Feedback	239
9.5.2	Absolute Detection Threshold: In the Lab	239
9.5.3	Absolute Detection Threshold: In the Wild	241
9.5.4	Just-Noticeable-Differences	243
9.5.5	Discussion	246
9.6	Reactive Compression Feedback	247
9.6.1	Evaluating Reactive Notifications	248
9.6.2	Discussion	251
9.7	Vibration vs. Compression Feedback	252
9.7.1	Results	253
9.7.2	Discussion	254
9.8	Beyond Notifications	254
9.8.1	Inflation Profiles	254
9.8.2	Jamming Bodies with Compression Feedback	255
9.8.3	Case Study: Compression Feedback for a Jogging Game	256

9.8.4	Beyond the Wrist	258
9.9	Summary	261
10	Conclusion	263
10.1	Limitations	265
10.2	Future Work	267
10.3	Closing Remarks	269
	References	271
	Curriculum Vitae	309

List of Figures

1.1	Scenario: casual interaction for a secondary task	2
2.1	Runner with a casual interaction system	7
2.2	Closed-loop interactive system diagram	13
2.3	Around-device sensing prototype for the steering study	25
2.4	Levels used for the steering study	25
2.5	Index of difficulty for steering tasks	26
2.6	Share of touch and hover interactions for different steering tasks	27
2.7	Gestures used for steering tasks with different indices of difficulty	28
2.8	Motorola Moto 360 smartwatch	31
2.9	Participants' rating of control level of different input modalities	34
2.10	Participants' rating of required effort per input modality	35
2.11	Participants' rating of required attention per input modality	36
2.12	Relationship between control and effort/attention	37
2.13	Usage of input modalities dependent on required effort	38
3.1	Mockup of future phone with built-in depth sensing	44
3.2	Around-device devices concept	48
3.3	Mockup of defining around-device devices	50
3.4	Mockup of using virtual joystick next to phone	52
3.5	Crowdsourced photos of phones lying around	56
3.6	Map of photo crowdsourcing contributor locations	58
3.7	Example object arrangement on table	59
3.8	Scenario setup used for elicitation study	60
3.9	High brightness edge display prototype	66
3.10	Distance sensor used in the edge display prototype	66
3.11	Fibers routing LED light to the edge display	67
3.12	Side view of LED routing fibers	67
3.13	Notification display state for different modes and distances	68
3.14	Edge display showing additional information of a call notification	69
3.15	Study setup of edge display evaluation	70
3.16	Qualitative results of edge display evaluation	71
3.17	Around-device pointing illustration	73
3.18	Leap Motion phone attachment	74
3.19	Around-device pointing setup	75

3.20	Targets used in around-device pointing study	76
3.21	Target indicator used in around-device pointing study	77
3.22	Around-device pointing distribution	78
3.23	Per factor pointing error for around-device pointing	79
3.24	Per factor acquisition time for around-device pointing	80
3.25	Mean acquisition time for every target by pitch	80
3.26	Vizualization influences orientation of pointing distribution	82
3.27	Mahalanobis distance for skewed target distributions	82
4.1	One-button recognizer concept	90
4.2	Button pressing profiles of four users	91
4.3	Standalone button prototype	92
4.4	Wall-mounted button recognizer prototype	93
4.5	Features of button pressing trajectory	94
4.6	One-button recognizer study setup	95
4.7	Recognition rates for different group and training sizes	97
4.8	Button-recognizer longitudinal performance	98
4.9	Electrode layout of capacitive couch	101
4.10	Capactive couch sensor schematic	101
4.11	Scenario of casual interaction in the smart living room	103
4.12	Light control bracelet overview	104
4.13	Philips hue app for iOS	105
4.14	Philips hue app for the Apple Watch	105
4.15	Light control bracelet prototypes	106
4.16	Silicone prototype molds	107
4.17	Bracelet controller board	108
4.18	Arduino-based lamp controller	108
4.19	Interactions on the light control bracelet	108
5.1	Human rater judgement of autocorrect likelihood	122
5.2	Illustration of ForceType touch model	123
5.3	Synaptics ForcePad with screen attachment	126
5.4	Lower need for active correction of text with ForceType	128
5.5	Words per minute improvement with ForceType	129
6.1	Casio DBC310-1 Databank digital watch with hardware keyboard	133
6.2	Apple Watch modes for casual communication	134
6.3	Ways to respond to a text message with the Apple Watch	135
6.4	Growth of the number of available emoji	138
6.5	Emoji skin tone modifiers	141
6.6	Skin tone selection for emoji	142
6.7	Zero width joiner emoji sequences	143

6.8	Number of emoji per tweet	145
6.9	Comparison of emoji usage frequency	146
6.10	Google keyboard emoji categories	149
6.11	Google emoji keyboard evaluation application	151
6.12	Selection time and error rate for Google emoji keyboard	153
6.13	Selection time per emoji frequency rank	154
6.14	Page traces for emoji selection	155
6.15	Histogram of page transitions needed for emoji entry	155
6.16	EmojiZoom overview	163
6.17	Emoji selection and layout used for EmojiZoom	165
6.18	Google keyboard used in EmojiZoom evaluation	166
6.19	Space-filling strategy for emoji grid	166
6.20	EmojiZoom evaluation and design	168
6.21	EmojiZoom and Google keyboard performance comparison	169
6.22	Usage of zoom levels in EmojiZoom	169
6.23	EmojiZoom qualitative results	170
6.24	Longitudinal evaluation of EmojiZoom	172
6.25	Distribution of EmojiZoom selection times	173
6.26	Best and worst performing emoji in EmojiZoom	173
6.27	Emoji similarity embedding visualization	183
6.28	Emoji similarity model distance characteristics	185
6.29	Emoji relationship hierarchy	186
6.30	Correlation between human raters and models	191
6.31	Agreement between semantic and tag-based emoji models	192
6.32	Non-ASCII Emoticons	197
6.33	Unicode text stylization	198
6.34	GIF keyboards	198
8.1	The ScatterWatch LEDs create a subtle glow on the user’s arm	208
8.2	Front of the custom ScatterWatch PCB	210
8.3	ScatterWatch silicone layer for comfort	211
8.4	Influence of silicone layer on ScatterWatch illumination	212
8.5	ScatterWatch case with PCB attached	213
8.6	Light behavior inside the skin	214
8.7	Remittance in the skin at different wavelengths	215
8.8	Illumination patterns used by ScatterWatch	216
8.9	ScatterWatch study participants wore the device for a full day	217
8.10	ScatterWatch shines through some clothing	218
8.11	Temporal distribution of ScatterWatch trials	220
8.12	Reacted stimuli per time of day	220
8.13	Reacted stimuli for ScatterWatch	221
8.14	Reaction times for ScatterWatch	221

8.15	Reaction time response curve for ScatterWatch	222
8.16	Reacted stimuli per illumination pattern	222
8.17	Reaction time per illumination pattern	223
8.18	Confusion matrix for illumination patterns	224
8.19	Reaction time per user activity	225
8.20	Reaction time for different lighting conditions	225
8.21	Qualitative ratings for ScatterWatch	226
9.1	Comparison of different haptic feedback techniques	235
9.2	Blood pressure cuffs tried for compression feedback	236
9.3	Compression feedback device prototypes	237
9.4	Air pressure to on-arm force relationship	238
9.5	Interfaces for lab-based compression feedback studies	240
9.6	Absolute detection thresholds for compression feedback	241
9.7	Pouch worn during in the wild study of compression feedback	242
9.8	Absolute detection threshold for compression feedback in the wild	243
9.9	Just-noticeable difference results for compression feedback	245
9.10	Weber's law for compression feedback	245
9.11	Overview of pressures in compression feedback	246
9.12	Reactive compression feedback illustration	248
9.13	Reactive feedback patterns confusion matrix	250
9.14	Qualitative ratings of reactive feedback patterns	251
9.15	Qualitative comparison of compression and vibration feedback	253
9.16	Restricting joint movement with compression feedback	255
9.17	Compression feedback strap placements on leg	256
9.18	Testing restricting compression feedback with a jogging game	257
9.19	Body locations for compression feedback	259
9.20	Compression feedback on the finger	259
9.21	Pneumatic actuators integrated into shoes	260
9.22	Compression feedback integrated into bracelets	260

List of Tables

3.1	Most common phone locations in the photo dataset	57
3.2	Most common objects near phones in the photo dataset	57
3.3	Around-device pointing ANOVA	77
4.1	Expected number of required button presses for success	96
4.2	Confusion matrix for long-term one-button recognizer use	99
4.3	Classification accuracy for postures on capacitance-sensing couch	102
5.1	Comparison of expected and observed autocorrection behavior	120
6.1	Emoji differences between platforms	141
6.2	Differences between traditional and emoji text entry	158
6.3	Most failure-prone emoji	174
6.4	Most likely to succeed emoji	175
6.5	Jaccard similarity between emoji pairs	179
6.6	Most related emoji examples	184
6.7	Top 5 most related tokens and emoji for ten example tokens	188
6.8	Ranking comparison for emoji similarity models	194
9.1	Reactive feedback patterns	249

1 Introduction

*The more that you read,
The more things you will know.
The more that you learn,
The more places you'll go.*

— Dr. Seuss, *I Can Read With My Eyes Shut!*

For many people around the world today, interactive systems are ubiquitous in their daily lives. They are on their phones, have computers on their desks, and encounter a wide range of interactive objects (e.g., elevators, or TVs) as they go through their day. A common assumption underlying most of those systems is that users will be actively engaged with them, devoting their attention to the interaction. This shows in complex desktop applications, but can also be seen in devices meant for more ad-hoc interaction, such as ticket machines (which often require users to provide quite a lot of input). Yet, users are not always able or willing to fully engage with such devices. For example, a user might be tired and willing to accept less control over an interaction if she in turn has to put less effort into it. Such systems, that enable users to trade off some control in return for having to engage less with them, are the focus of this dissertation.

Casual interaction stands in contrast to focused interaction, i.e., interaction where users closely engage by, e.g., paying attention or concentrating on a task. But not every system that is low engagement is automatically a casual interaction system. A crucial aspect required of a casual interaction system is that it enables the user to trade control. Casual interaction thus circles around notions of *control* and *engagement*. Users engaging less give up some control, while users engaging more with a system can take more control. This can be a continuum between different control levels, but a system can also discretize to a more limited number of control levels. Different levels also need not necessarily be contained in one device. If two devices are present and the user can pick between one that offers more control, but is also more complicated, and one that is easy to use, yet does not allow the user to control every aspect, the two together also form a casual interaction system.

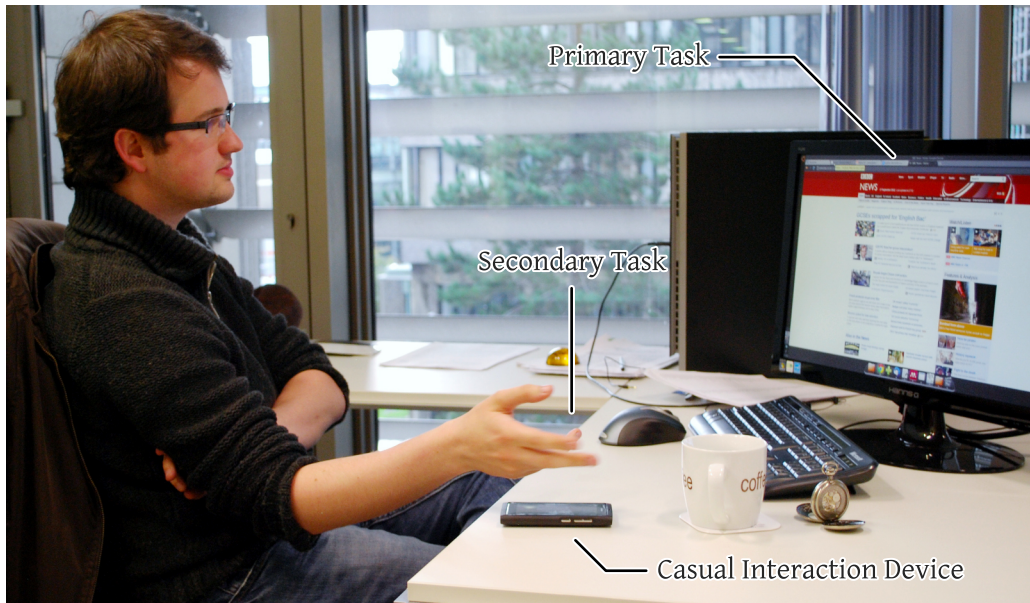


Figure 1.1: While Daniel is reading an article, he is listening to music on his phone. To change the song, without closely engaging with his phone, he just waves over it.

But before investigating casual interaction more closely, let us first take a look at a basic example scenario (also shown in Figure 1.1), to outline the basic approach:

Daniel is at his desk, doing research for an article he is writing. He is also listening to music on his phone. Just now, his phone has started playing a song he grew tired of recently. Hence, he wants to change the song and switch to something more enjoyable. However, he also does not want to disrupt his work. He knows that if he picks up his phone, unlocks it, and starts looking for the perfect song for right now, he will lose track of the article he is reading right now. In other words: while closely engaging with his phone would give him his desired control over the music, his current situation makes such interaction less desirable.

Instead, he decides to let the phone pick the next song to play. He waves over the phone to indicate he wants to hear a different song. The phone picks up on the gesture and, based on Daniel's music listening behavior, picks a new song that is (1) very different from the one currently playing, yet (2) also to Daniel's liking. By making this choice for Daniel, he was spared the effort of having to closely engage with the phone. On the other hand, he had to give up some control in what the next song to play would be. Enabling this trade-off between control and engagement is what casual interaction is all about. In this scenario, Daniel is happy with the automatic choice of the next song and can continue work on his article without ever having needed to divert his full attention and effort.

We can see that this user gave up some control, but was able to complete the interaction with the phone without closely engaging with it. Around-device interactions in this case allowed this reduction in control. While we will investigate those closer in a later chapter, they are but one way to offer lower-engagement ways to interact with a system. In fact, while the principle of casual interaction is easy to understand, finding concrete ways to translate a desire for lower engagement to concrete systems and techniques is not as straightforward. A large part of this dissertation is thus dedicated to just this: showing ways to offer lower-engagement alternatives, augmenting existing systems and environments to give users a chance to be more casual.

1.1 Contributions

This dissertation touches upon several different interactive systems. Each one of these improves or investigates a smaller aspect of casual interaction. Overall, this dissertation makes two main contributions:

1. It describes the concept of casual interaction and shows evidence for a need for said casual interactions. Such evidence comes primarily from related work (e.g., the law of less work from psychology), but also from two of our own empirical studies on effort-avoidance in interaction.
2. It provides concrete examples of how the concept of casual interaction translates to actual systems, thus showing how to augment current focused interaction system with lower-engagement alternatives. An important part of this contribution are the discussions of the area-specific forms casual interaction takes on.

1.2 Dissertation Structure

This dissertation is roughly composed of two components: one discussing the concept of casual interaction and one investigating specific casual interaction systems. The focus is on the individual casual interaction systems, which also provide a context to discuss specific aspects of casual interaction more in detail. Hence, chapters providing a more conceptual view are intermixed with chapters that describe concrete systems.

To start off, Chapter 2 describes the overall concept of casual interaction. In the process, we explore definitions for both *control* and *engagement* which together form the base of casual interaction. This is also the place where we compare casual interaction to related concepts and discuss where casual interaction differs. Yet, note that more applied related work is moved to chapters on specific aspects of casual interaction.

After having explored the casual interaction concept, subsequent chapters detail specific application areas and foci. We begin with work on around-device interaction in Chapter 3. Around-device interactions are a straightforward way to decrease engagement with a device. As we move away from a device, our ability to control it or perceive feedback naturally deteriorates anyway. In particular, we will take a look at moving interaction off the phone to either objects or the space around it. We end this chapter with a concrete example system for around-device notification access and a quantitative study investigating how granular interactions for this setup can be.

Next, we take a look at smart homes, an area well suited for casual interactions, in Chapter 4. As the home is supposed to be a place to relax and regenerate, systems requiring a lot of engagement can be ill equipped for it. We explore one system in depth: a casual interaction alternative for identifying users. By integrating the identification into a push button, no extra effort is needed to tell the system who triggered an action. If more complex or secure authentication is needed (e.g., for the front door), users can fall back to a less casual authentication mechanism. We also take a brief look at augmenting furniture to support casual interactions and light control with casual interactions.

While the systems described so far use discrete levels of control, Chapter 5 presents a system with gradual change of control. Here control is taken or relinquished to an autocorrection mechanism used in text entry. Built around a force sensitive trackpad, this system allows users to gradually change how much correction they want to allow as they type, by varying their typing pressure.

In Chapter 6 we stay with text entry. However, here we look at messaging at a higher level and discuss how casual interaction applies to messaging in general. Building on this exploration, we focus on one specific means for supporting more casual interaction messaging: emoji. By improving emoji entry, the presented systems strive to allow users to engage in more casual communication.

As a final area of casual interaction, we take a closer look at output in casual interaction systems. Starting from a more general overview of casual interaction output in Chapter 7, we then explore two modalities in more detail. First, Chapter 8 details work on using indirect light feedback as a more subtle and less disruptive form of feedback. Then, Chapter 9 describes how pneumatic compression as a feedback method allows systems to scale from subtle all the way to intense feedback.

We end this dissertation with an overall conclusion and an outlook on future directions for research on casual interaction in Chapter 10.

2 Casual Interaction

I'm too careless. I don't put out enough effort. I'm tired.

— Charles Bukowski, *South of No North*

In this dissertation we will take a look at several different casual interaction systems, i.e., systems that do not require users to fully engage with them. But while much of this dissertation focuses on those systems, how they are built and how well they perform, we first need to more closely define what is actually meant by the term *casual interaction*. Starting from a definition of *casual*, we thus investigate the contrast between *control* and *engagement*. This necessitates more clarification of what those two terms describe within this dissertation. Furthermore, this chapter discusses reasons why users might desire or require casual interaction. Finally, we will take a look at evidence a preference for lower engagement interaction exists in the context of user interfaces.

This chapter is partly based on a paper published at *CHI 2013*, a supervised thesis¹, and a book chapter published as part of a book on peripheral interaction:

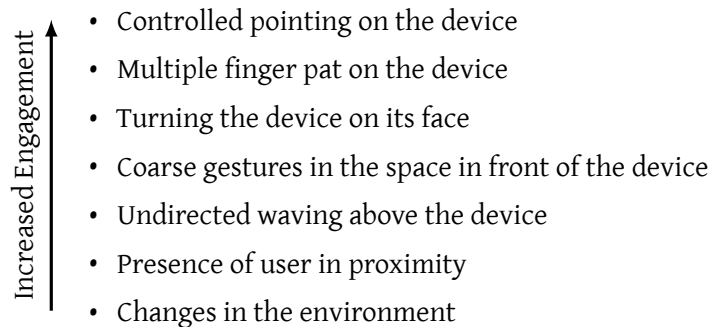
- Henning Pohl and Roderick Murray-Smith. “Focused and Casual Interactions: Allowing Users to Vary Their Level of Engagement.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '13*. New York, New York, USA: ACM Press, 2013, pp. 2223–2232. DOI: 10.1145/2470654.2481307
- Henning Pohl. “Casual Interaction: Moving Between Peripheral and High Engagement Interactions.” In: *Peripheral Interaction: Challenges and Opportunities for HCI in the Periphery of Attention*. Ed. by Saskia Bakker, Doris Hausen, and Ted Selker. Berlin, Heidelberg: Springer, 2016, pp. 117–135. DOI: 10.1007/978-3-319-29523-7_6

2.1 Overview

In Chapter 1 we already saw one example of a casual interaction scenario. Casual interaction ultimately is all about the trade-off between engagement and control. If we engage less with an interactive system we yield some control to it while closer engagement allows us a higher level of control over the interaction. In Sections 2.3 and 2.4, we will later define those two terms more precisely.

¹Sven Röttering's Master thesis [260]

For an initial overview, though, we can look at some means of interaction and reflect on how much user engagement they require. As an example, consider the seven ways of interacting given below:

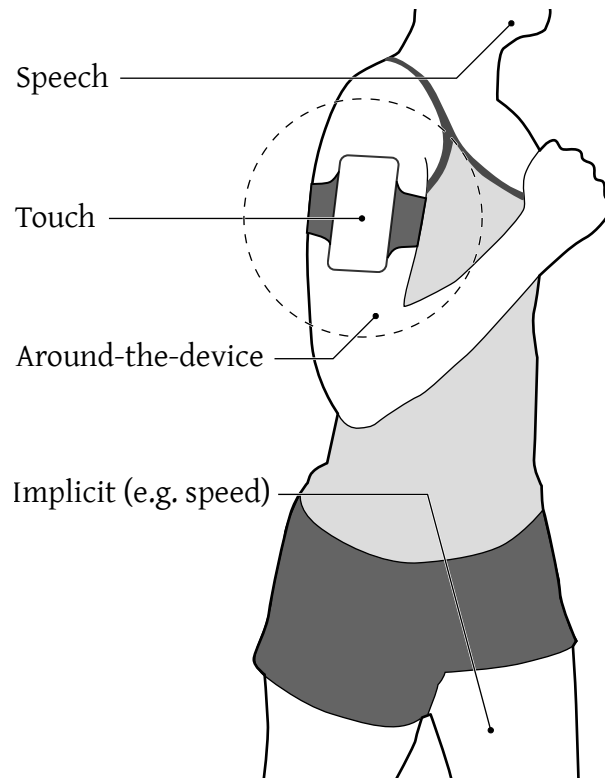


If we look at the extremes, then environmental changes or general proximity are interactions where a user would not even need to pick up the device. On the other hand, controlled pointing not only requires holding the device, but also involves non-trivial targeting and movement effort. Thus the *physical* demands on the interaction increase with more engagement. Higher engagement with the interaction thus necessarily means lower capability to exert control elsewhere. As we will learn later in this chapter, physical constraints are but one reason for lower engagement. Users could also face social or mental constraints on their interaction, incentivizing them to also go for lower engagement.

In casual interaction, systems should cover a wider range of engagement levels to help users deal with these constraints. This can mean building devices with several different sensors to, e.g., sense both touch and around-device interactions. An example of this is shown in Figure 2.1, where a runner is given multiple ways of control. Just the way she is running could be used as an input mechanism (e.g., as picked up by an accelerometer), but she could also use touch input. However, touch input is hard to use while still running and would require her to stop for most interactions. Hence, this is another example of the trade-off facing users with casual interaction systems.

If we go back to the engagement scale shown above, we can see that as users engage more, the way information is transmitted during the interaction also changes. While, e.g., presence is a low-fidelity input channel, touch interaction can transmit many more bits per second. When controlling a complex system, high bandwidth input can be needed for full control. For example, an interactive 3d position and orientation task requires fast updates for each axis for a high quality interaction. Correspondingly, low bandwidth input in such a setting forces the system to infer and *fill in* the gap in input fidelity. Hence, engagement and control are closely related—something we will discuss in-depth in the remainder of this chapter.

Figure 2.1: This illustrated casual interaction setup for runners shows how many different input modalities can be combined into one system. Each of the modalities requires a different level of engagement by the user. Where implicit interactions (e.g., running faster) allow for convenient input, they offer less control over the interaction than, say, touch input. However, touch input on the other hand is hard to use while running and requires users to stop to interact.



In the remainder of this chapter, we first define the terms around casual interaction. We then look at related concepts and how they differ from casual interaction. Furthermore, we detail the types of constraints that motivate use of more casual interactions. Finally, we investigate how and when users choose casual interactions instead of more focused interactions.

2.2 Defining Casual

With this thesis revolving around casual interaction, it is sensible to define this term more rigorously. A good start for this is the term *casual* itself, which can take on several different meanings. Hence, there is also some ambiguity in what *casual interaction* describes. We thus need to be careful in stating which notion of casual to refer to when talking about casual interactions. As we will see later in this section, other work has already used the term casual interaction differently. This all the more shows the necessity for distinguishing the various interpretations of what casual can mean in the context of interactions.

The term casual has four main interpretations (the one most related to casual interaction is highlighted below), which (per the New Oxford American Dictionary [283]) are:

1. **Relaxed and unconcerned**
 - a) **Made or done without much thought or premeditation**
 - b) **Done or acting in a desultory way**
 - c) **Done or acting without sufficient care or thoroughness**
2. *Not regular or permanent, in particular:*
 - a) *Employed or established on a temporary or irregular basis*
 - b) *(Of a sexual relationship or encounter) occurring between people who are not regular or established sexual partners*
3. *Happening by chance; accidental*
4. *Without formality of style or manner, in particular (of clothing) suitable for everyday wear rather than formal occasions*

As highlighted above, casual interaction, for the purposes of this dissertation, concentrates on the *relaxed and unconcerned* meaning of casual. In fact, the three more detailed specifications to that point directly link to the notion of engagement (to be explored in Section 2.3). Users who are not engaged with an interaction do not give it much thought or are not precise and accurate with their input.

The dictionary definition of *casual* already highlights several reasons on *why* users might desire casual interactions: because they do not want to give it much thought, lack the enthusiasm to put effort into it, or are careless. We explore these and other reasons to desire lower engagement interaction later in Section 2.6.

Note that while we focus on the highlighted aspect here, there is some interconnection with other definitions. For example, in informal settings it is more acceptable to put less care into communication than in more formal ones. Thus a lack of formality can be a reason for resorting to more casual and disengaged input in general.

2.2.1 Definitions of Casual in Other Works

While this dissertation builds upon the above-mentioned definition of *casual*, previous work has used the term casual interaction differently. Based on the notion of casual as relating to informality, the focus of these casual interaction system was on communication and in particular groupware systems. An early example from 1991 is Borning and Travers' work on enabling non-collocated users to have more informal encounters [23]. They designed virtual rooms, *vrooms*, where users are represented by images. When two images are moved close together, an audiovisual channel is opened and the users can have a quick conversation. The authors' stated goal with their system is to: "*support informal interactions: the sorts of conversations that occur around coffee pots, mailboxes, in the hallways, and the like.*" [23]

Similarly, Cockburn and Greenberg also frame casual interaction around informal communication [53]. In particular, they focus on how people start communicating and choose between different communication channels. Greenberg later investigated presence indicators as a way to facilitate such casual interactions [89]. Again connecting this to group communication, he states that: “opportunities for casual interaction happen when people are aware that others are available for communication” [89]. Nine years later, McEwan and Greenberg built the *Community Bar* to further explore this concept [193]. Here, they state that casual interactions are “unplanned, brief, frequent, and usually engage small groups of people familiar with one another.”

The term casual interaction has more recently also been used in papers not related to communication. For example, Wilson et al. mention casual interactions in the context of physics-based interactions on interactive surfaces [318]. However, they do not provide a definition of what exactly they mean by casual interaction. Similarly, Robinson et al. also refer to casual interaction without clarifying the meaning [258]. They built an application to mark geographical locations with different interaction styles. Their use of the term casual incorporates notions of ad hoc and imprecise interaction, where users point in the general direction of a location instead of precisely pinpointing it on a map. Just recently, Klamka and Dachsel referred to casual interaction in their discussion of their elastic controller device [153]. Unfortunately, it is not clear which of the previous definitions of casual interaction they are building upon. All these examples are unfortunately using the term *casual* rather “casually”.

Echtler et al. use casual interaction with respect to playing casual games [72]. Casual games are designed to have simpler gameplay and to require less effort to play. This includes, e.g., games designed to be played during a short wait for the next bus. Correspondingly, Echtler et al.’s work focuses on enabling ad hoc games between players without much preparation.

As we discuss the meaning of casual in relation to interactions, what makes a game casual has received similar attention. A summary of this discussion is given by Kuitinen et al., who provide an overview of different interpretations [167]. They find that one of the best descriptions is defining casual by the “*easiness of the game experience*”. This does not only refer to a game itself, but also covers aspects such as “*player attitude or availability of the game*”. In the notion of casualness, there is some overlap with casual interactions (e.g., designing for low cognitive demand). However, casual games describe a more specific niche while casual interaction is a concept targeting interactions in a more general context.

2.3 Defining Engagement

In our definition of casual interaction, how much influence a user has on an interactive system depends on her *level of engagement*. We thus need to define what is meant by *engagement* before proceeding. Unfortunately, the term is rather elusive and is used to describe many different things. *Engaging* with something could mean devoting ones attention to it, being interested, or putting more than usual thought into it. Engaging others can mean drawing them into a performance or conversation, but also could just describe the initial approach of them. Engagements need not be desirable, e.g., two battalions can engage in battle (according to “rules of engagement”). In general, engagement is active rather than passive—one does rarely accidentally engage (drunk marriage proposals notwithstanding).

But for interactive systems we need to make this definition slightly more concrete. Helpfully, engagement is something that has interested psychology and human-computer interaction researchers for a while. In fact, there are multiple theories of engagement. However, instead of extending these individually, we shall build upon the definition of O’Brien and Toms [214, 215], which we will discuss here in detail. They in turn built their engagement model on top of those and tried to unify the concept. In their notion of engagement, users are engaged if they are “*affectively involved, motivated, and perceive themselves to be in control over the interaction*” [215]. With regards to casual interaction, aspects of *involvement* and *control* are of particular importance, while *motivation* is less central. While *involvement* captures how much users are invested in an interaction, *control* describes how they can act on that (or at least feel they can). *Motivation*, on the other hand, describes aspects of why users engaged with an interface in the first place. While this is important as well, it does not immediately impact the actual interaction.

In their initial investigation of engagement, O’Brien and Toms review previous approaches and then propose their own definition of engagement as:

a category of user experience characterized by attributes of challenge, positive affect, durability, aesthetics and sensory appeal, attention, feedback, variety/novelty, interactivity, and perceived control. [215]

In their definition and research they specifically target four application areas: online shopping, web searching, educational webcasting, and video games. These areas differ slightly from the interactive systems we will look at in this dissertation. Where casual interaction takes us to mobiles, smart homes, or around-device interaction, their areas focus more on desktop computing scenarios.

O’Brien and Tom then conducted semistructured interviews with 17 participants to determine whether there were any factors they missed in their literature review. Among other things, participants were asked to recall times were they were so involved in the activities that they, e.g., lost track of time. All interviews were transcribed and coded, resulting in 52 codes, such as *serendipity*, and *interruptibility*.

As a result of their study, they identify four stages of engagement: *point of engagement*, *engagement*, *disengagement*, and (possibly) *re-engagement*. Each of these stages has different relevant attributes of engagement. For example, attributes for the point of engagement describe what drives users to engage with a system initially. This can, e.g., be goal driven, because people they trust recommended it, because an interface is aesthetic, or because they are simply interested in the offered content. Whilst already engaged, users are more motivated to sustain their engagement when they felt in control, continued to receive interesting content/feedback, and kept their attention on the task. As O'Brien and Toms put it: "*Users must be made to feel part of the interaction through an awareness of what the system is doing (feedback) and feeling connected to the technology (interactivity) or to other people (social awareness), and in control over what is happening.*" [215] These attributes of sustained engagement directly relate back to the notion of a control–engagement trade-off in casual interaction. Users only sustain engagement if they have an adequate level of control and are more likely to disengage when this is not the case. Control and engagement are thus intrinsically linked.

In light of their interview results, O'Brien and Toms update their engagement description accordingly (new attributes not highlighted in source): "*Engagement is a quality of user experience with technology that is characterized by challenge, aesthetic and sensory appeal, feedback, novelty, interactivity, perceived control and time, **awareness, motivation, interest, and affect.***" [215]

The work by O'Brien and Toms discussed so far has focused on a qualitative description of engagement. However, this still begs the question of how to actually measure engagement and how the attributes of engagement are interconnected. Correspondingly, later work by them set out to define a "user engagement scale" [214].

It should be noted that their work is not the first to attempt to measure engagement, but one of many approaches. One common approach is just to ask users. At the most basic, this can be a Likert scale where users indicate whether they agree or disagree that an experience was engaging. A more sophisticated approach is to ask for ratings of individual engagement attributes. For example, Webster and Ho wanted to find out whether viewers found a presentation engaging and asked them for ratings of *challenge*, *feedback*, *control*, *variety*, *attention focus*, *curiosity*, and *intrinsic interest* [307]. We can immediately see how this aligns well with the attributes of engagement we have seen so far.

Some of the attributes of engagement, such as *attention* and *affect* also lend themselves well to automatic monitoring. For example, a user's gaze could be tracked to determine how much attention an application received or galvanic skin response measurements could indicate a user's arousal level. Using such measurements avoids the more subjective nature of questionnaires, but comes with other problems. Primarily, these measurements only cover some aspects of engagement. For example, it is not clear how aesthetic or novelty attributes could be automatically measured as well. Hence, such physiological metrics only offer a more limited view of engagement.

O'Brien and Toms also used surveys to measure engagement. However, in contrast to earlier work, they try to expand the number of attributes covered by their questionnaire in order to cover their wider definition of engagement. For each of their attributes they tried to find existing scales in the literature and also reused interview statements from their previous work. Overall, this process yielded 459 questions which they had a second researcher go through and eliminate ones non-representative for their respective attributes. After pre-testing and eliminating further questions, this left them with 123 questions, where each is associated with an attribute of engagement. In this iteration they focused on engagement in the context of online shopping and thus all questions related to that. For example, the scale included questions such as "I was absorbed in my shopping task", or "I felt in control of my shopping experience."

They gathered responses from 440 users to evaluate their user engagement score. The intent here was to find out whether questions within and between attributes correlated. In a first step, questions with low reliability were eliminated. Exploratory factor analysis was then used to find relationships between questions and hence also attributes. This data was used to find a smaller subset of the questions that still has strong variance. After multiple iterations and a final examination, this left them with 31 questions and six factors: *focused attention*, *perceived usability*, *aesthetics*, *endurability*, *novelty*, and *felt involvement*. Note that this removed several of the earlier identified attributes from their user engagement scale.

The validity of these 31 questions was confirmed in a second survey study with 802 participants. In this study they also investigated relationships between engagement factors and, e.g., found that the level of focused attention predicts the level of felt involvement. They also found that aesthetics predict perceived usability, a connection supported by previous research [250, 297].

In general, O'Brien and Toms' work is primarily concerned with engagement as a property of an experience. The attributes of engagement they identify thus cover a wider range than necessary for this dissertation. For example, endurability is less central for this dissertation, while focused attention and felt involvement are a better fit. In fact, it is worthwhile to consider another view of engagement that plays into our use of engagement as well. The starting point here is the observation that higher engagement comes with increased monopolization of user resources. Mathur et al., in reference to Wickens' *multiple resource theory* [313], e.g., put this as: "*while driving a car, if a user receives an interesting content, he/she still may not engage with it because the user does not have sufficient visual and motor resources.*" [190]

Another issue with the survey-driven approach by O'Brien and Toms is that it is less suitable for mobile and dynamic situations, as explored in this dissertation. While it is easier to evaluate the overall engagement of, e.g., an online shopping experience, engagement in casual interaction can be varying a lot. One approach here, as mentioned above, is to use physiological measurements. But interestingly, this is not always necessary and

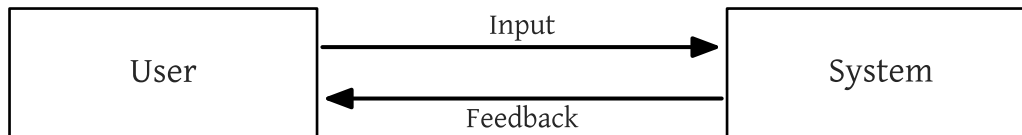


Figure 2.2: Interaction with computing systems happens in a closed-loop. Users receive feedback from the system and in turn provide inputs to it. If the feedback quality deteriorates (e.g., due to noise), this negatively impacts how well and fast users can react.

interaction context can be sufficient to predict and investigate engagement. Just recently, Mathur et al. have shown that this works reasonably well for mobile phones [190]. They found that they could predict whether a user would be more or less engaged in a phone session depending, e.g., on how much time has passed since the last call, or whether they used a lot of apps in the last hour. Their work hence shows that how users engage with their phones is a complicated affair and desired engagement changes dynamically.

Overall, engagement is a complex issue and measuring it poses significant challenges. However, while the *concept* of engagement is important for this dissertation, it is less important to derive a quantitative view of engagement for casual interaction. After all, the underlying principle is to have users pick how much engagement/control they desire. Judging the level of engagement is hence up to them. Casual interaction systems only need to have an input mode available that fits a user's current desires. If we think back to the attributes of engagement, then a user looking for an interaction with *focused attention* and *felt involvement* should have an input mode available to support this. On the other hand, if a users desires an interaction that allows her to be less attentive or involved, this should be available as well.

2.4 Defining Control

Control is the other side in the trade-off with engagement. When users engage more, they receive more control and the other way around. There are areas of human-computer interaction, where the goal is to automate system behavior and take control away from the user. For example, Erickson discusses this with a focus on context-aware computing, and points out how such automation is bound to run into problems, such as when a car incorrectly determines the doors should be locked [76]. He argues that "*rather than trying to take humans out of the control loop, we keep them in the loop,*" which is also the goal of casual interaction and a principle that runs through the entirety of this dissertation. As with our discussion of engagement, we need to closely investigate the notion of control as well. The notion of control in this dissertation is based on dynamic systems [134], e.g., described by the closed loop between users and their interactive devices (see Figure 2.2).

One basic example of a closed-loop interactive system is a computer's volume control. Users can move a slider to the left or right, changing the volume of the music playing. During their manipulation of the slider they receive feedback over multiple channels. First of all, the volume of the music changes as the slider is moved. Users can also track the position of the slider knob along the slider to observe the set volume. Finally, a call out or label might also show a numeric representation of the volume level. All three feedbacks are representative of the internal volume state of the computer.

As users move the slider, they monitor the feedback and adjust their motion accordingly. If they, e.g., use a mouse to drag the slider they have to use the right acceleration, movement, and deceleration to move the knob to the desired position. As the relationship between mouse movement and knob movement can vary (depending on the specific system, but also, e.g., due to mouse acceleration settings) they need to observe the knob in order to adapt their motion. This connection between the input signal (mouse movement) and the feedback (knob position) is what characterizes a closed-loop system.

How well users can control such a system depends, among other things, on the quality of the feedback. For example, consider if the slider only updated five times a second. In this case, users would need to anticipate and accordingly extrapolate their motion in the breaks between updates of the feedback. Similarly, if users move the slider based on their auditive perception of the music, then the level of environmental noise would have an impact on how well they can adjust the volume. Previous studies have shown that, in general, the quality of user input in motor tasks goes down as users' uncertainty in the system's internal state increases [156, 285].

2.4.1 The H-Metaphor

A core aspect of casual interaction is allowing the user to yield control. This builds on previous work by Flemish et al., which they named the *H-Metaphor*. Their work concentrates on driving and in particular automated vehicles. They summarize their concept as: “*You can let your vehicle go without being completely out-of-the-loop, or you can reassert a more direct command, for example, by taking a tighter grip on your haptic interface.*” [79]

The example Flemish et al. use to illustrate the H-metaphor is horse riding. A rider can “*loosen or tighten the reins*” to change how much control to exert on the horse. Tightening the reins can, e.g., mean making more deliberate and decisive movements or interacting with the horse more frequently. When the reins are loose, the horse is given more freedom to decide where to go. By tightening the reins, a rider can take back control and steer the horse more closely. This metaphor does not just apply to the literal handling of the reins, but translates to other rider behavior. For example, “*loosening the reins*” could also mean that riders only use subtle commands or only communicate with the horse infrequently. The horse itself contributes to the task and follows trained behavior when

not given overriding commands. It can, e.g., see a path ahead and follow it even under loose reins. External cues thus also inform the behavior of the horse. However, the rider retains the option to tighten the reins and steer off the path if so desired. Similarly, casual interaction systems are designed to allow looser reins when using a device—yielding some control to it as desired to offload some of the effort of the interaction.

2.4.2 An Information-Theoretic View

We already saw how deterioration of feedback impacts how well users can control a system. As feedback, e.g., becomes noisier, users have to pay more attention to the system to correct for that noise. Relating such systems back to Shannon's information theory [272] provides a good way to frame this problem.

A central notion here is the concept of channel bandwidth or channel capacity. It describes how much can at most be transmitted over a channel. This can be limited by noise (e.g., a digital-to-analog converter can not create an arbitrarily precise output waveform), or by the capabilities of the sender and receiver. For example, while a wide range of sounds can be created, the human ear is only able to pick up sounds in a smaller range of frequencies. The notion of channel capacity need not be a physical one. A feedback channel to a user paying less attention would also be of smaller capacity than one to an attentive user.

A high bandwidth channel for interaction thus allows for more complex and/or faster input and output. For example, a computer screen has a high output bandwidth (many pixels and a fast refresh rate) compared to a single LED. On the input side, a proximity sensor, e.g., has a lower bandwidth than a video camera. Note that the bandwidth is not just given by the technical limits of the used technology. A three-axis accelerometer, attached to a user's wrist, is able to pick up movements at very high sample rates—commonly at hundreds of hertz. Yet, if this sensor were used to track in-air hand gestures, this would severely oversample the gesture signal. The bandwidth of such a hand gesture channel would hence be set by the information content of the gestures and not by the potential data available to the accelerometer.

But the fidelity of interaction is not just determined by the theoretical bandwidth of the channel. Instead, how a channel is used makes most of the difference. A computer screen might offer high bandwidth feedback, but if a user only glances at it from time to time the effective channel bandwidth is much smaller. Similarly, a very tired user might only be able to process a smaller share of the information on a channel. The same holds for user input as well. For example, users can focus on performing a gesture, providing more accurate input, or be sloppy in their execution of a gesture, adding noise to the signal. In general, the level of attention to an interaction thus changes the interaction bandwidth. Correspondingly, physical restrictions, such as inebriation, have the same effect on the effective channel capacity.

We can relate this back to the concept of casual interaction. Here, focused interactions are characterized by a higher bandwidth channel between the user and the computer. Users will tend to frequently and attentively sample the feedback provided, and quickly respond with precise inputs of their own. Casual interactions, on the other hand, have control loops that are partially or fully restricted in some way. For example, users might only respond sporadically and with low-fidelity input instead of being focused on the system. Such intermittent interaction could also be caused by the computing device only being able to provide coarse feedback, e.g., due to resource contention. The interaction might also be inhibited by environmental factors, such as noise or distraction. A partial restriction occurs if only one side of the interaction is hindered, such as when an attentive and focused user is faced with a sluggish system. Similarly, a system providing ample means for input and high quality output does not guarantee a high bandwidth channel if the user is not paying much attention to it. Should any such restriction be the case, the amount of control that can be exerted goes down as well.

2.4.3 Dealing with Reduced Control

As we have seen, casual interaction inherently means that the channel capacity between the user and the system is reduced. Because fewer bits can be transmitted, this prevents high bandwidth interaction, e.g., with high fidelity graphics and precise input. If we focus on the input side then the lower channel bandwidth has to result in one of two outcomes: (1) users still attempt precise control, which now takes longer to complete, (2) users only get a more limited set of inputs, but keep the time needed for interaction roughly equal to focused interaction. The first case can, e.g., be observed in the area of brain-computer interfaces, where the signal-to-noise ratio is often so poor extensive sampling is needed to get stable inputs [88]. The second case is more interesting in the context of this dissertation. By limiting the available inputs, some control is delegated to the system (after all, users have a reduced set of options), but the interaction is maintained, even for low-fidelity channels. Note that options in this case need not be discrete (like the number of buttons to click), but can be a continuous measure. For example, if a system applies more correction to touch input, this reduces noise (such as from hand tremor), but also smooths the input and thus eliminates details.

However, limiting options need not be the only way to handle reduced control levels. Instead, a system could offer higher level control alternatives. We already encountered an example of this in the introductory example in Chapter 1. There, the user wanted to change the audio track, but did not want to closely engage with the phone to do so. Instead, he just swiped in the air above the phone to signal a desire for a different song. Note that while picking up the phone would have allowed him to *pick* the next song, swiping relegated this decision to the system. It is this a more higher level abstraction of song changing interaction, where the fine detail of direct change are replaced by a more coarse interface to only signal general change.

2.5 Related Concepts

Casual interaction builds on or is related to several existing concepts. The earliest notable one is Buxton's foreground/background model [33]. He denotes the foreground as those activities that are intentional and "in the fore of human consciousness". Background activities, on the other hand, are described as "tasks that take place in the periphery – 'behind' those in the foreground." Buxton discusses this from a telecommunications perspective and notes that "nearly all" existing communications technology is designed for foreground interaction. Hence, he presents the porthole system, which provides background awareness of possible video chat participants. His model also incorporates the notion of transitioning between different sectors, e.g., detecting presence of others in the background and then initiating a video call with them, thus moving the interaction to the foreground. As in our discussion of control in Section 2.4, he brings in a bandwidth perspective, noting that background tasks seem to be low bandwidth but persistent, while foreground tasks are high bandwidth and come in bursts. Yet, this model discretizes interactions to either the fore- or background. In casual interaction, there is no such clear distinction—trading off control and engagement can happen to any degree.

Hinckley et al. built on Buxton's work and provide a more in-depth discussion of how mobile devices, their sensors in particular, can support this kind of interactions [118]. In their view, "the distinguishing characteristic of the foreground is that it requires the user's direct attention [...]." In casual interaction, attention is also an important characteristic, however, as discussed in Section 2.3, attention is only one component of engagement. Correspondingly, in their model, background interactions are those that are performed without much attention or on-the-fly, e.g., automatically rotating the screen content when a device rotation is detected.

A specific notion of background interaction, incidental interaction, was investigated by Dix [69]. Incidental interactions happen unbeknownst and secondary to an intended primary interaction, or as Dix puts it: "In incidental interaction, the context of the user's current purposeful interaction is used to drive a secondary interaction." For example, the air conditioning might turn off when a window is opened or the light turn on when a room is entered. Users do not intend to perform such an interaction, but an "intelligent" system senses what the user is doing and affects other changes in the world because of that. In his framework, as users become more experienced with incidental interaction, they eventually transition to expected and finally intended interactions. For example, users noticing that opening the car door turns on the car's lighting might deliberately open the door as to turn on the light. It is thus the users' experience that determines the level of user intent, instead of users' freely choosing their desired level of engagement as in casual interactions.

Two years earlier, Schmidt proposed the related concept of implicit interaction [267]. He defines this as: “an action performed by the user that is not primarily aimed to interact with a computerised system but which such a system understands as input.” As an example application, he describes a word processor on a PalmPilot where, e.g., the font size increases if the device is moving (to make reading easier). The two critical concepts required for devices to enable such interactions, according to Schmidt, are perception and interpretation. They need to be able to track their situational context and interpret it adequately to then offer reasonable automation. Inherent to implicit interaction are automatic, rule-based, interactions, where the user is not necessarily in the loop. A slightly different notion of implicit interaction was used by Ju et al. for their *implicit interaction framework* [140]. Here Buxton’s foreground/background model is extended with an initiative dimension. Hence, while attentional demand moves interaction between fore- and background, a user’s initiative determines whether the system is *reactive* or *proactive*. This framework thus also incorporates agents (proactive in the background).

Peripheral interaction is another concept closely related to casual interaction. According to Bakker et al., in peripheral interaction, “interactions with technology could be designed to shift between center and periphery of the attention” to “enable digital technologies to better blend into our everyday lives” [9]. This, e.g., results in “objects that could drift between the focus and periphery of a user’s attention according to the momentary demands of their activity” [74]. Where peripheral interaction focuses on aspects of physical placement and attention, casual interaction builds on a user’s desired level of control. An interaction is *casual* when control is yielded to the system, whereas it is *peripheral* if low attention is given to the interaction. Those two aspects can overlap, e.g., when yielding control means using coarser interactions at the side, but at other times those two views diverge. Users can, e.g., have an interaction in the center of their attention, yet choose to give up control (e.g., by providing ambiguous input, expecting the system to partly take over). Recently, Bakker and Niemantsverdriet have extended the peripheral interaction concept to incorporate a more flexible range of attention in what they call the *interaction-attention continuum* [11]. This is reminiscent of the focused-casual continuum in casual interaction.

2.5.1 Levels of Interaction

A central aspect of casual interaction is how users can choose and possibly transition between interaction at different levels of engagement. Such levels and transitions between them have previously been explored for different systems. Vogel and Balakrishnan, e.g., outline design principles for public displays that offer four interaction zones from ambient to personal interaction [304]. They note how interaction can adapt based on the level of attention of the user (e.g., angle of the user to the display). Their prototype also

enables all interaction levels at any point, giving users the freedom to choose how much to engage. However, they have looked at engagement mostly as a function of distance, while casual interfaces have a user-centric view on engagement. Similarly, Clark et al. allowed users to interact with a display using touch from up close and pointing from further away [50]. They combined this with a zoomable interface approach, enabling fine grained selection up close with a more coarse range of options when at a distance. The *Range whiteboard* prototype, designed around the above-mentioned implicit interaction framework, also uses interaction zones to switch between different kinds of interaction.

In the context of interactive surfaces, Marquardt et al. tried to unify designs for interaction on and above the table [188]. Instead of treating those separately, they propose designing for the *continuous interaction space*, where interaction moves smoothly between them. They identified 14 different “interaction categories” for such setups. Most of these are not related to changes in control or engagement levels, though. However, one technique—adjusting the control–display ratio depending on the height above the surface—does change how much control users have. A similar technique was also described by Yu et al., whose *Air Finger* system segmented the space above a surface into different layers, each with a different control–display ratio [326].

2.6 Reasons for Casual Interaction

So far we have mostly concentrated on defining what is meant by casual interactions. However, while this makes clear the general idea of yielding and taking control, another question is *why* a user would want to do so. After all, designing for this change of control is more complicated and if there is no need, this cost would be superfluous. Instead, one could just design systems for fully engaged users and not consider lower engagement at all. In fact, this is how most current systems are designed. Yet, there are many scenarios where users are not able to, or do not want to, fully engage with their devices. In general, inhibiting factors are of (1) physical, (2) social, or (3) mental nature.

2.6.1 Physical Reasons for Casual Interaction

Physical reasons not to fully engage with a system are perhaps the most straightforward. Whether users have their hands full or are far away from the device, such constraints directly inhibit how well any interaction can take place. Some physical constraints are permanent, such as user disabilities or device sizes (e.g., a device might be too heavy and bulky to be held with only one hand). Yet, other constraints are temporary or situational (e.g., a user carrying shopping bags can not use his hands as freely).

Interaction with mobile devices is one research area where the impact of encumbrance has received close attention over the years. For example, Ng and colleagues have looked at pointing and gesturing accuracy in encumbered and walking situations [209, 210]. Holding objects or walking around was shown to significantly decrease user performance. In fact, as shown by Bergstrom-Lehtovirta et al., there is a non-linear relationship between users' preferred walking speed and targeting performance [16]. Previously Kane et al. showed that these impairing effects of walking can be countered by adapting the size of the interface [142]. Overall, this indicates that the simple act of walking already negatively impacts common interactions. Yet, walking is a common activity and thus there have been attempts to design walking-specific interfaces, such as keyboards [83].

This connection between encumbrance and interaction has not just been shown for mobile interaction. For example, Oulasvirta and Bergstrom-Lehtovirta investigated multitasking performance (with a mouse, trackpoint, or touchpad) under conditions such as holding objects of different sizes [219]. Compared to the unconstrained baseline, the constrained conditions exhibited significantly worse performance. This shows that working at a desk on a laptop is also affected by physical constraints, e.g., when holding a mug. In general, physical impairment inhibits our achievable accuracy, for example, Keates et al. showed that movement error in mouse usage is larger for motion-impaired users [146]. In some cases, users can obtain physical relief, and overcome strain or fatigue by changing the level of interaction.

While physical disabilities or impairments from, e.g., walking are unavoidable to users (unless they adopt a completely sedentary lifestyle), they generally have more choice when it comes to temporary impairments. For example, when impaired by wearing gloves, users can choose themselves between giving up control (sometimes completely) or relieving the impairment (taking off the gloves) for a more full engagement (but cold hands). Casual interaction systems try to enable interaction even in such situations where users can not engage manually as needed in focused interaction.

2.6.2 Social Reasons for Casual Interaction

Compared to physical impairment, social constraints on interaction are more subtle. They primarily come into play when using technology in company of others (such as on public transport). For example, gestures users are willing to perform in private can be uncomfortable for them when used in public, as shown by Rico and Brewster [256]. Location and audience had a significant impact on whether a gestural interaction was considered acceptable. But social acceptability of specific interactions is just one aspect of interactive systems. For casual interaction, the more central question is how much engagement with a device is acceptable in a given situation or company. While acceptability of engagement and, e.g., acceptability of specific gestures can be related this is not necessarily the case.

Social acceptability of engagement with a device is commonly situational. For example, while taking a phone out of the pocket and checking stock quotes is perfectly acceptable when alone, this interaction would be considered rude when performed during a date. Phone use within a family is one scenario where this influence of social aspects is readily apparent. As reported by Hiniker et al., e.g., household rules commonly forbid phone use in context such as during dinner [120]. Yet, they also find that it can be hard for users to temporarily not use their phones (7 % of parents reported this about their children's phone use). The core concern with phone use in this context is that no attention is paid to other members of the family. Note though, that while phones might inhibit social connection to those immediately around us, they can instead help connecting to others elsewhere (e.g., by sending messages or photos). Lower engagement interaction where the majority of attention is still devoted to others around the user might help find a common ground between no phone use and no connection to the ones around us.

Going back to the situation of dating, previous research has investigated the impact of technology on couples. McDaniel and Coyne, e.g., found that interruptions due to technology had a significant negative correlation with relationship satisfaction of women [192]. Their results also show that interruptions by technology are common, e.g., 70 % of their participants reported at least sometimes having their interactions with their partner interrupted by a phone. Similar results are also reported by Roberts and David who found that being ignored due to a partner's phone use negatively impacted relationship satisfaction [257]. Overall, this shows that use of technology can have a strong adverse impact on people's relationships. If devices are designed to capture our full engagement it is understandable that those close to us can sometimes feel left out.

While social settings limit some interactions, they can also strengthen others. Users might, for example, be more inclined to perform interactions, that make them look relaxed and laid back. The desire to seem as if one is not putting in too much of an effort in order to be seen as *cool* has, e.g., been described for 14–16 year old students by Warrington et al. [306]. Casual interaction might be more desirable in social settings, because it can reinforce some users' desired image of being in control yet laid back and relaxed about it. The view of casual interaction in the social sphere fits well into the concepts of *front stage* and *back stage* by Goffman [85]. He notes that people change their behavior based on what they imagine people around them are thinking about them—putting on an *everyday performance* when on the *front stage*. As people put on a character, they are more likely to engage in behavior that they see as potentially appealing. Note that casual interactions are, however, not necessarily more *subtle interactions* (as, e.g., understood in [57]), as the casual action might be more overt than a focused one.

2.6.3 Mental Reasons for Casual Interaction

While there are physical and social reasons to go for casual interactions, probably the most prominent reasons are related to mental state. Users might, e.g., be tired/exhausted, concentrating on something else, generally lazy, or just not very interested in a given interaction. But there is also the inherent desire by people to put in the least amount of effort required for a task. This concept is sometimes called the *law of less work* and is particularly relevant to casual interaction. Before diving into effort-avoiding behavior, however, we will first discuss the other mental reasons.

One reason not to engage much in an interaction is focus on something else. If users already work on a primary task, this leaves little attention for other interactions. But this division between tasks is not necessarily clear cut. For example, Bakker et al. found that even primary tasks shifted back and forth between the center and the periphery of users' attention [9]. This suggests that there is room for interfaces that enable users to carry over this behavior for their interactions. When influenced by mental constraints, users might still *want* to engage with their devices, just sometimes with lower effort and precision, or in a more sporadic fashion. Casual interaction mechanisms should hence allow users to control the *level* of their engagement.

Another reason not to engage is exhaustion or fatigue. After a long day at work, one might be less inclined to expend much more effort in the evening. But tiredness is only one aspect at play here. In general, the capacity to make active choices declines over the day, suggesting that users could especially benefit from casual interactions later in the day. This concept, named *ego depletion* by Baumeister et al. [15], postulates that making active choices depletes some kind of internal resource. Restraining oneself or picking between different options costs us and makes us less able to self-regulate and make active choices afterwards. Baumeister et al. link this idea back to Freud's notion of the *ego* expending effort to counteract the *id* and *superego*. They state that "*Freud was fond of the analogy of horse and rider, because as he said the rider (analogous to the ego) is generally in charge of steering but is sometimes unable to prevent the horse from going where it wants to go.*" This is reminiscent of the H-metaphor discussed above, but hints that the means of control might be limited and not universally available.

However, the primary motivator for casual interactions is not user being tired or occupied elsewhere. Instead, the basic desire to *do less*, to get away with as little effort as possible is driving force in limiting focused and thus overly engaged interaction. We take a closer look at this aspect below.

2.6.3.1 Law of Less Work

Intuitively, one could assume that people act in a way that minimizes their required effort. Hull named this the *law of less work* and states:

If two or more behavioral sequences, each involving a different amount of energy consumption or work, have been equally well reinforced and equal number of times, the organism will gradually learn to choose the less laborious behavior sequence leading to the attainment of the reinforcing state of affairs. [127, p. 294]

Hull focused on physical effort and, in fact, much of the work following his went along the same lines. While this is important, here we are primarily interested in minimizing *cognitive* effort. This has often been assumed to be covered by the law of less work as well.

Kool et al. formally evaluated whether the law of less work also applies to cognitive effort with six behavioral experiments [155]. They consistently found that participants will pick the lower demand option when given a choice to do so. For example, in their first experiment, participants were shown two stacks of cards on a screen. They could pick from the left or the right one as they saw fit. After picking a card, it revealed a number out of {1, 2, 3, 4, 6, 7, 8, 9}, either colored blue or purple. If the number was blue, participants had to respond whether the number was less than five. On the other hand, with purple numbers, participants had to respond whether the number was even or odd. Switching between these two tasks incurs a cost, as participants have to change their answering strategy. But the two stacks were not equal. While cards from one stack stayed at one color 90 % of the time (lower effort), the other stack switched often, only showing the same subsequent color 10 % of the time (higher effort). Participants over time tended more and more towards drawing cards from the low-effort stack, significantly more often than random chance would suggest.

This is only the result from one experiment. Their other investigations confirm this trend while controlling for different aspects of the experimental setup. The law of less work is therefore extended to the *law of least mental effort* by Kool et al. Overall, there is hence strong evidence for such effort avoiding behavior in psychological studies. Yet, Kool et al. state that “*the law of least mental effort stipulates that anticipated cognitive demand weighs as a cost in the cost-benefit analyses that underlie decision making.*” It is thus only one factor of many and other desires (such as a wish to impress a potential mate) can override the general desire to not expend effort.

A related concept is discussed by Zipf, who names his the *principle of least effort* [337]. He summarizes his principle by stating that:

[a person] will strive to solve his problems in such a way as to minimize the *total work* that he must expend in solving *both* his immediate problems *and* his probable future problems. That in turn means that the person will strive to minimize the *probable average rate of his work-expenditure* (over time). And in so doing he will be minimizing his *effort*, by our definition of effort. Least effort, therefore, is a variant of least work. [337, p. 1]

He hence takes a broader view on effort than the law of least mental effort, by ascribing a much larger perspective to effort avoidance. This explains Zipf's work on tying his principle to the structure of language. According to Zipf's law, the frequency of a word is inversely proportional to its rank (i.e., a few common words make up most of a language's use). He thought that the resulting distribution is the result of an optimization process of language. The underlying idea here is that, while speakers favor the use of as little vocabulary as possible, listeners prefer as expressive a language as possible. This is due to speakers trying to expend little effort on composing the message, which in turn puts the onus on the listener to figure out what was meant. Instead, listeners would prefer receiving a more detailed and specific message, in order to reduce their effort of parsing the message. The compromise between those two desires then manifests itself in the language's resulting word distribution.

While Zipf's work underlines the general pattern of effort avoidance, for the purposes of casual interaction, the law of least mental effort provides a more fitting frame. After all: it has been evaluated directly with concrete tasks, while the principle of least effort has a more post hoc heritage. However, Zipf's work is still noteworthy, as it hints at the fact that effort avoidance might have a larger impact than just direct interactions. It might very well have shaped many of the systems around us.

2.7 Evidence for Least-Effort Behavior in Interaction

As described in Section 2.6.3, there is strong evidence for a general trend towards effort-avoiding behavior. We set out to collect further evidence for such behavior, specifically in the context of interactive systems. Given three input methods with different levels of control, we observe how users switch between them while trying to complete steering task of varying difficulty. We hypothesize that with increasing index of difficulty, players will take more control of the interaction, while being more casual in easier tasks.



Figure 2.3: Ceiling mounted Kinect (left) and SHAKE SK7-ExtCS1 capacitive sensing extension board (right) used for the study prototype. This setup allowed for the necessary on and above the device sensing.

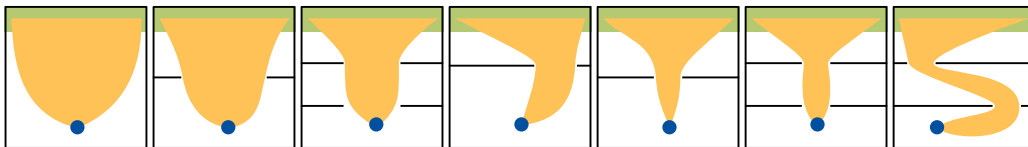


Figure 2.4: Overview of all levels used in the evaluation, sorted from lowest to highest index of difficulty. Participants were instructed to move the ball from the bottom start position inside the green target area at the top. The orange shape highlights the path used to compute the index of difficulty and is not visible to participants during the evaluation.

2.7.1 Apparatus

For our evaluation, we designed a prototype (shown in Figure 2.3) that allows for above-the-device interaction as well as precise touch interaction. A SHAKE SK7 sensor pack with SK7-ExtCS1 capacitive sensing extension board from SAMH Engineering Services is used as the foundation. It provides 24 capacitive sensor pads (each 7 mm^2 in size) in a 4×6 arrangement ($57 \text{ mm} \times 36 \text{ mm}$ overall). Those sensors provide accurate on-the-device touch tracking and also can be used to track input up to 1 mm away from the surface. In the future, combining different pad sizes could enable sensing at much larger distances, while preserving the capability for accurate on-the-surface touch tracking [259]. Future devices could also incorporate range sensors for coarse above-the-device tracking [159]. In our current prototype we use a Kinect sensor to emulate such future devices. The Kinect is mounted on the ceiling, looking down on the sensor pack. We use computer vision techniques to extract the hand from the depth image by thresholding and connected-components analysis. Sensor data is combined on a PC, mapping the Kinect data to the coordinate space of the device.

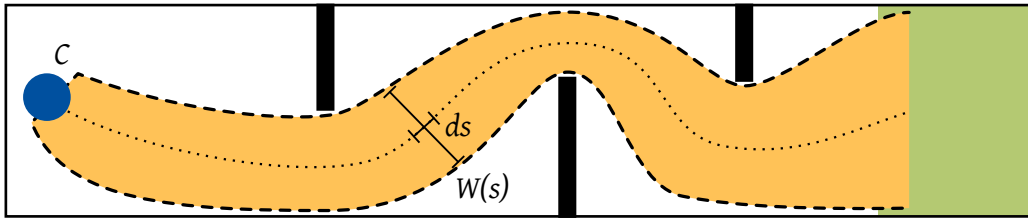


Figure 2.5: The index of difficulty for a game is defined as the integral of the reciprocal of the width along the required path. Hence tighter paths have higher indices of difficulty. Path tightening is a result of obstacles, restricting how the ball can move through the level.

For our study, we use an obstacle course navigation type game that requires players to steer a ball to a target area. On the way, players need to stay away from the walls. If contact with a wall is made, the ball is reset to the starting position. Figure 2.7.3 shows the used levels with different indices of difficulty. The ball in the game has momentum and is simulated using a physics engine.

We compute the index of difficulty for each level according to Accot and Zhai’s steering law [2]. Where Fitt’s law describes pointing performance, this law was designed for trajectory-based tasks. Such tasks occur, e.g., when users navigate a menu and need to stay within the menu bounds. For navigation along a curved path C , parameterized by s and with width changing as $W(s)$, the path’s index of difficulty is thus defined as (see also Figure 2.5):

$$ID_C = \int_C \frac{ds}{W(s)} \quad (2.1)$$

While there is a linear relationship of the index of difficulty to task time, this aspect was not of interest for this experiment. We did not ask participants to be fast, nor did we show a timer or any ranking based on time. This was done to prevent participants from trying to finish levels as fast as possible. Instead of changes in task time, we were interested in changes of engagement depending on the varying index of difficulty.

For control of the ball, participants had to use our casual interaction prototype system (shown in Figure 2.3). This allowed them to exercise three levels of control:

- Touch** allowed for the most precise control. After touching the surface, any finger movement directly results in a corresponding ball translation (at a low control-display ratio). By using clutching, the ball can be moved to any point in the level.
- Hover** provides a higher control-display ratio rate control mechanism. Here users move their hand in the space just above the device. The zero point is set to the vertical axis through the center of the device. Deviation from this center results in a force of proportional amplitude being applied to the ball.

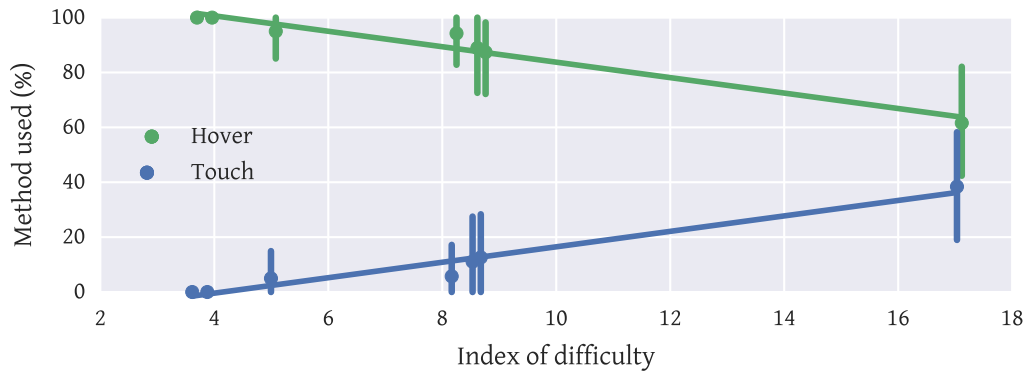


Figure 2.6: The study showed an increase in the share of direct touch interaction for higher indices of difficulty. Ratios are computed per participant and averaged over all participants. Error bars show bootstrapped 95 % confidence intervals.

Gestures can be used further atop the device, beyond the range of hover interactions. Here, users indicate a general *direction* they would like the ball to move in. This is the least precise of all three methods. Execution of a gesture results in an impulse of constant strength and in the indicated direction being applied to the ball.

While the direct touch mode allowed for more precise control, it required participants to closely engage with the device. The hover option on the other hand is less precise, but requires no such close engagement. Finally, the gesture option allows participants to potentially “solve” a level with one ballistic action, but it is also the least precise option available—offering no feedback control ability.

2.7.2 Participants

We recruited 12 participants (2 female, age 24–48 $\bar{x} = 29.75$, $SD = 6.37$) from Glasgow University’s School of Computing Science. They were given a chance to familiarize themselves with the game in a special training level. Participants played freely until they felt confident in their ability to control the game and switch between techniques. Afterwards, they played through a randomized sequence of 7 levels. Participants were instructed to decide themselves how much they wanted to engage with the game at any given situation. They were also assured that obstacle collisions were not a problem in this study and that they would not be evaluated based on their performance. As mentioned earlier, we made sure they did not feel the need to finish a level as fast as possible.

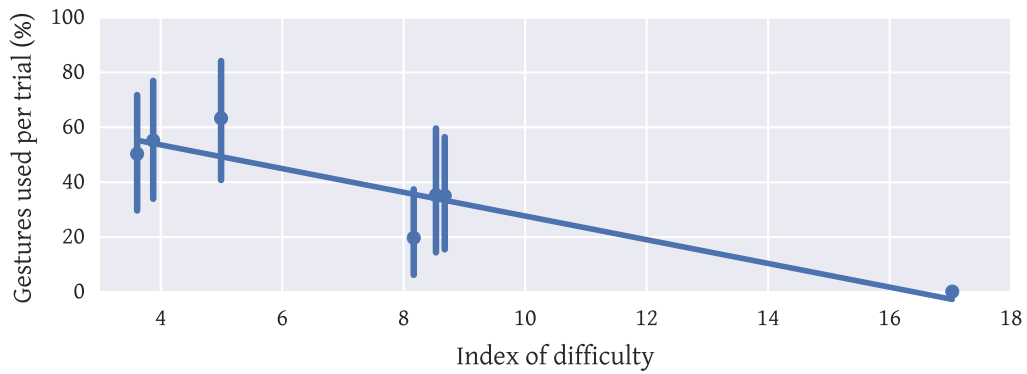


Figure 2.7: The study showed a decrease of gesture usage in levels with higher indices of difficulty. Error bars show bootstrapped 95 % confidence intervals. No gestures were used for the hardest level.

2.7.3 Results

We find that for low difficulty levels, participants exclusively used hover and gesture control (see Figure 2.6). Only at the highest index of difficulty was a noticeable number of interactions performed by direct touch control. Note, though, that *hover* can be regarded as the default mode. When a user's hand is near the device this registers as a hover interaction. Most users occasionally let go of control and allowed the ball to move around, as there was no penalty for obstacle collisions. The ratio of touch interactions shown here should therefore be regarded as a lower bound. Linear regression shows a significant ($p < 0.001$) but weak ($R^2 = 0.24$) influence of the index of difficulty on the percentage of touch interactions. Correspondingly, the same holds for the linear regression of hover usage over index of difficulty. While there is a clear preference for hover interactions in easier levels, in the harder level this shifts and touch interactions are used much more.

Looking at the number of times participants used gesture control for a level (shown in Figure 2.7), we see a similar trend with the number of gesture invocations declining with increasing index of difficulty. As with hover and touch usage, linear regression shows a significant ($p < 0.001$) but weak ($R^2 = 0.21$) influence of the index of difficulty on the number of gestures used. An interesting condition is the third easiest one (also the third level from the left in Figure). Even though the level was harder than the two before, participants tried to use gestures slightly more. This is likely because a direct vertical movement was sufficient to reach the target, which participants noticed and often kept trying until succeeding. This indicates that Accot and Zhai's steering law, while a good initial estimate, is not sufficient for capturing the actual level of difficulty for the set of input techniques used here. It is also interesting to note that for a case such as condition 5, if users can get away with multiple attempts at a low-effort gesture, they appeared to prefer this over a higher engagement interaction style.

2.7.4 Discussion

The participants indeed adapted their behavior based on the difficulty of a level. If the index of difficulty was higher they were more likely to use precise touch interactions. Correspondingly, in lower difficulty levels they were more open to using gesture or hover interactions. These results supports one basic premise of casual interaction: users are willing to use more casual input if the level of control is still sufficient to achieve their desired goal. This aligns well with the law of less work, as discussed in Section 2.6.3.1. Where participants a chance to minimize their expended effort they did.

A subsequent example of this pattern of effort avoidance can be found in work by Jakobsen et al. [135]. In their study, they investigated how users pick between touch and mid-air gestures when interacting with a large display wall. Most of the participants rated touch as their preferred interaction modality, which also performed better and had lower fatigue effects than mid-air interaction. However, because touch interaction required them to move closer to the display it was still often avoided by users when given a choice. Several participants pointed out that the benefit of mid-air interaction lies in how it requires little movement effort. So even though mid-air interaction is slower and more prone to errors, this was accepted by these participants as long as they got to save on effort. This hence is further validation of the results obtained previously by us.

2.8 In Situ Qualitative Study of Control and Effort

The study discussed above looked at effort avoiding behavior in a lab setting. However, realistic situations are much more complex and interaction preferences can likely not be reduced to the difficulty of a steering task. To gather some data in more realistic settings, we thus conducted another study to investigate participants' interaction preferences in a more in situ setup.

As task for this investigation we chose music control. Listening to music is a very common activity and entails several different actions listeners might perform. They could, e.g., be picking an album to play, adjusting the volume, or skipping a song. If listening to music on a phone, those actions are commonly performed via a graphical user interface on said phone. Yet, this requires closer engagement with the device and might not be suitable or possible in all situations.

Instead, listeners could benefit from ways to control music that require lower engagement with their devices. An example of this is work by Boland and Murray-Smith on *casual music interaction* where they present a lower-engagement control method for selecting the next song [21]. Users select their desired track by tapping its rhythm on their phone. This can be done without taking out the phone and closely engaging with it, e.g., facilitating interaction while walking.

Here, we do not focus on novel, more casual interaction techniques. Instead, we compare three existing interaction methods: touch, gestures, and speech. Where touch requires closer interaction with the device, both gesture and speech input do not. In fact, for speech input the device only needs to be nearby for interaction, yet the environment can restrict whether this is a suitable modality. Gesturing falls somewhere between those two methods—requiring not as much engagement as touch, yet not being decoupled from the device as much as speech.

2.8.1 Method

We conducted our experiment in a semi-controlled setting. While we wanted to observe participants in natural environments and not in the lab, we curated the situations they were in. For example, participants were riding on their bicycle outside, but we picked the area where they were doing so.

2.8.1.1 Participants

We recruited 10 participants (2 female, age 21–29 $\bar{x} = 25.9$, $SD = 2.5$) for this experiment. All participants owned a smartphone, but only one also owned a smartwatch (primarily used for notifications, navigation, and to tell the time). We asked participants how they would rate their music listening behavior on a five-point scale from “listening to whatever comes up” (e.g., shuffling their music collection) to “picking each song individually.” Seven participants located themselves right in the middle of that scale (3), one stated listening to whatever comes up (1), and two trended towards desiring more control over their music (4).

2.8.1.2 Apparatus

As device to use we picked a *Moto 360* smartwatch by Motorola (see Figure 2.8). This allows having all three modalities handled by the same device. Furthermore, smartwatches are a good example of an always-available device and thus well suited to explore how interaction changes in a wider range of scenarios. The *Moto 360* is paired with a Samsung *Galaxy S6*, which provides the internet connectivity required to stream music. Participants also wear headphones, plugged into the phone, to listen to the music.

Streaming music and metadata is retrieved from Spotify². This, e.g., allows access to playlists, music by specific artist, or music from a category (such as *workout*, *pop*, or *party*). Spotify also offers descriptive ratings for individual tracks. For example, each song has numeric *danceability*, *tempo*, *energy*, and *valence* values. Hence, this information enables music players to, e.g., create a playlist of very danceable songs.

²Accessed via the Spotify Web API and the Spotify Android SDK: <https://developer.spotify.com/>



Figure 2.8: Motorola Moto 360 smartwatch (2nd generation) that ran software to allow music control with gesture, touch, and speech input.

For each of the three input modalities, we created specific interactions to control the music playback. Because the modalities each are suited for different levels of complexity, some actions are not available for all of them. Each of these modalities is described in the following.

Touch For the touch mode we created a graphical user interface to run on the smartwatch. Due to the small screen size of the smartwatch, the user interface is split into multiple pages that users can transition between. The main page shows media controls (previous track, play/pause, next track), as well as volume controls and a shuffle button. It also displays status information, such as the currently playing track. By swiping to the left, users can access additional pages. On these other pages, users can select lists of songs, albums, categories, artists, or playlists. If a user, e.g., selected the album list, the view then would switch to a new list where individual albums can be selected for playback. Touch mode thus allows precise control of which songs are played. Users can browse all available songs in several different ways (e.g., all songs from the *party* category) and pick just the one they like.

Speech Speech recognition is handled by Android's built-in speech recognition service. Speech is thus not processed directly on the smartwatch, but the audio data is streamed to a server that performs the recognition. Users do not need to explicitly start speech recognition. While the music control app is active on the smartwatch, users can just use voice commands. If the app is not active, they only need to "wake up" the smartwatch by, e.g., tapping on the screen. While the speech input is free-form, users need to end their commands with the word "please". This is done to help determine what actually is a valid command, reduce spurious detections, and create an overall more friendly interaction experience for both, user and computer.

Speech commands vary in complexity, from simple one-keyword commands to more complex sentence-based commands. Actions encoded by one-word commands are: pause, resume, next, previous, louder, quieter, shuffle. Saying the respective keyword followed by a “please” hence issues the command. Users can also play back a specific item by saying “play <item>, please”, where <item> can, e.g., be a song, artist, or playlist name. Finally, users can use more abstract commands to access music based on its features (as provided by Spotify). We do not require exact matching of requested items, but use the Levenstein string distance metric to find the best match in the music library. Users can thus utter commands, such as “A little bit more tempo please”, to change to a song that is a bit faster than the currently playing one. This is enabled for the *tempo*, *energy*, *loudness*, and *valence* features. Users can request either *more* or *less* of the indicated quantity and optionally prefix this for *a little bit* or *much more/less* of a change.

Gesture While touch and speech commands can be used for more intricate control (e.g., picking a specific song), gesture controls offer a more coarse form of input. User can use them to skip a song (swing arm to right), return to the previous song (swing arm to left), increase/decrease the volume (swing arm up/down), and pause/resume playback (punch forward). Before inputting any gesture, users have to perform an activation gesture, by holding the watch with its face down (i.e., rotating the raised wrist halfway). This is acknowledged with a short vibrotactile stimulus. Afterwards, users have a 750 ms window for performing the actual gesture. This helps with gesture recognition by clearly demarcating the time frame where a gesture is potentially taking place.

Gesture recognition itself is performed via the *wiigee*³ library. Gesture data is collected with the smartwatch’s built-in three-axis accelerometer (part of an *InvenSense MPU-6051*), sampled at 50 Hz. Each accelerometer sample is quantized to one of 14 preset directions. Gesture recognition then is performed by hidden Markov models together with a Bayes-classifier. Training data for the gestures was recorded by the experimenter and used for all participants (i.e., participants did not train personalized gesture models).

2.8.1.3 Procedure

We first gave participants 15 minutes for training with the system. This gave them a chance to try out the different input methods and familiarize themselves with the available music library. Afterwards, participants had to perform eight different interactions in each of six different scenarios. Scenario order was counterbalanced with a Latin square, while interaction order remained constant. During the study, participants wore the smartwatch on their left wrist.

³<http://wiigee.org/>

The six scenarios were designed to provide a diverse set of physical and mental impediments. This allows gathering more ecologically valid data on participants reactions and opinions on the three interaction methods in these scenarios. The different scenarios participants experienced were:

Riding a bicycle We gave participants a course on a parking lot. They zigzag through the different aisles while listening to and controlling their music.

Jogging For jogging we reused the course from the bicycle scenario.

Reading Participants sat on a couch inside a private living room, reading a chapter from a book.

Work at desk Participants sat down at an office desk and had to work on a laptop. Their task was to enter text from a provided paper sheet into the laptop.

Walking Participants walked a course (almost all of it pedestrianized) while playing a memory game on a phone.

Walking with bag Participants walked the same course as above, but had to carry a bag in their right hand this time.

In each scenario, participants had to perform the same sequence of interactions with the music player. They could choose which interaction method to use to perform these (note that not all actions could be completed with the more limited gesture condition). The interactions they were asked to perform were:

1. Start playback of a playlist (e.g., dance music)
2. Change the volume of the music
3. Toggle shuffle on or off
4. Skip one or more songs
5. Change to a song via an audio feature (e.g., pick a song with more energy)
6. Skip one or more songs
7. Play a song from a different category
8. Pause the music playback

During the scenarios, the experimenter made note of participants' interaction choices. We also logged all interaction events. After finishing all scenarios, we also asked participants to fill out a questionnaire. Here we collected qualitative ratings on the different interaction modalities and the scenarios. In particular, we were interested in how they would rate how much effort and attention they felt the modalities required in the different scenarios. We also asked participants to rank the interaction modalities.

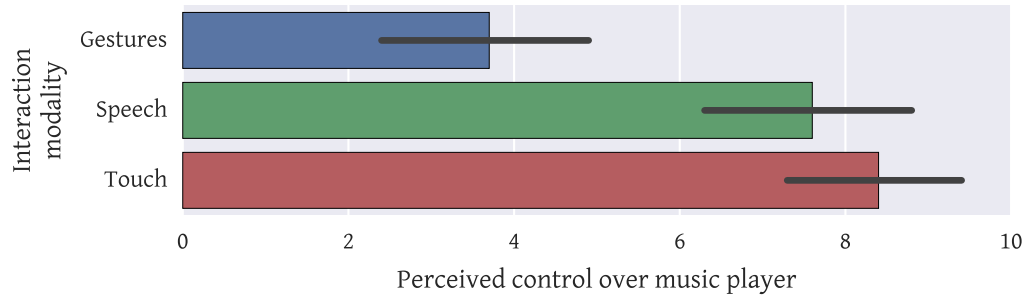


Figure 2.9: We asked participants to rate on a 0–10 scale how much control each interaction modality gave them when controlling music playback. They rated gestures as providing significantly less control than both speech and touch input. This is understandable as gestures only offer coarse input and, e.g., do not allow users to select a specific song.

2.8.2 Results

In this study we were primarily interested in participants’ *perception* of the different interaction modalities. We first look at how much they felt in control of the interaction, when using the different modalities. As shown in Figure 2.9, gesture input was perceived to offer much less control than both speech and touch input. A Friedman test confirmed a significant effect of modality on perceived control ($\chi^2(2) = 15.8, p < 0.001$). Post hoc testing with pairwise Wilcoxon rank sum tests with Bonferroni correction showed significant differences between gestures and both speech ($p < 0.01$) and touch ($p < 0.01$). However, there was no significant difference between the touch and speech modalities ($p > 0.05$).

Next we take a look at participants’ perceived effort (see Figure 2.10). We aggregate over the different scenarios and only look at differences due to modality. A Friedman test confirmed a significant effect of modality on the perceived required effort ($\chi^2(2) = 12.2, p < 0.01$). Post hoc testing with pairwise Wilcoxon rank sum tests with Bonferroni correction showed significant differences between speech and both gestures ($p < 0.05$), and touch ($p < 0.01$). However, there was no significant difference between the touch and gesture modalities ($p > 0.05$). We can also see that the scenario within which a modality is used has an influence on how participants rated the required effort. Particularly for the touch modality, there are large differences between, e.g., using touch while riding a bicycle and using touch while working at a desk. This is likely due to physical constraints in the bicycle scenario, restricting how well users can interact with the smartwatch.

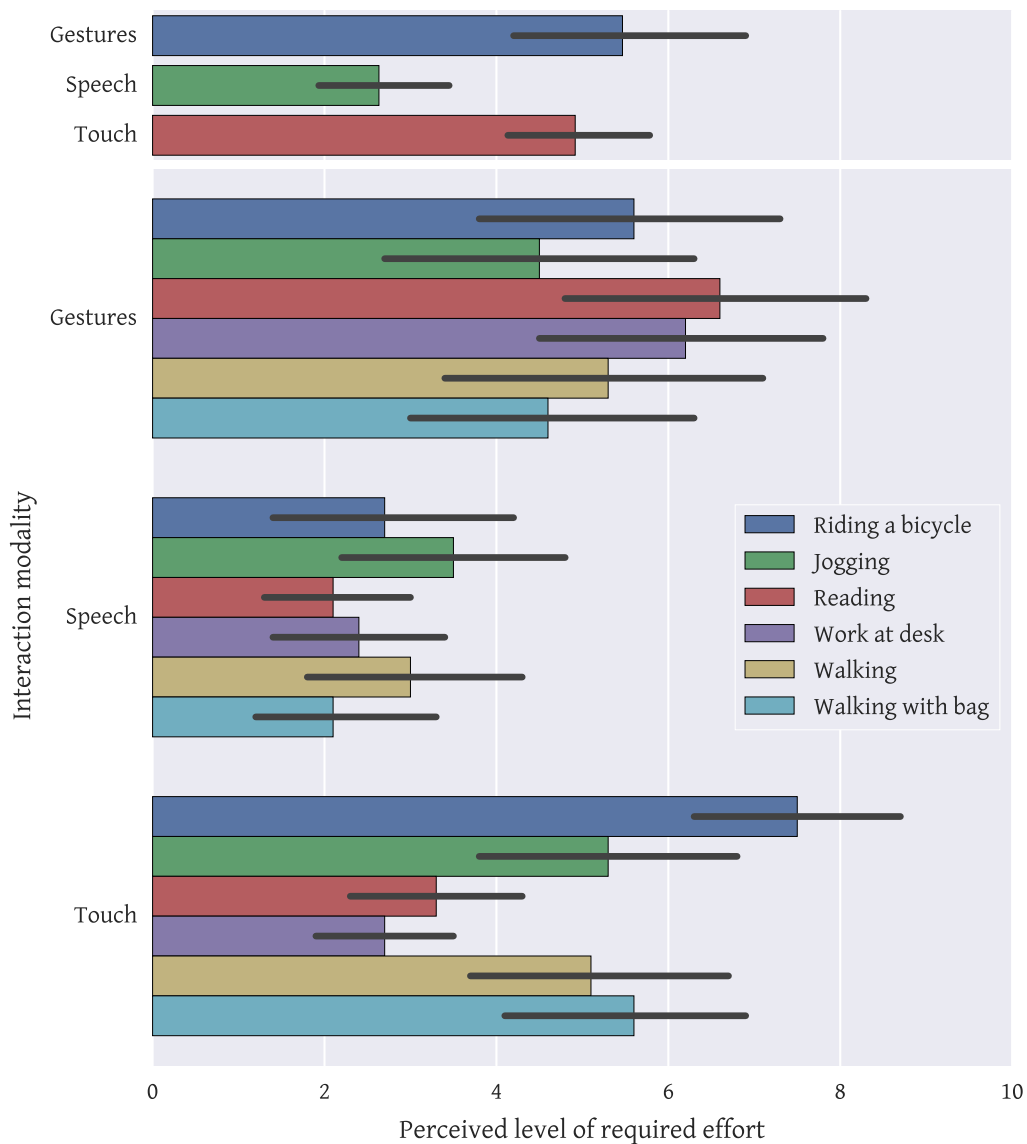


Figure 2.10: Participants, on a 0–10 scale, rated how much effort each interaction modality required when controlling music playback. They rated speech as more cumbersome than both gestures and touch. There are slight differences in the effort rating depending on the scenario. These are most pronounced for touch interaction, where scenarios like biking make it much harder to use the modality.

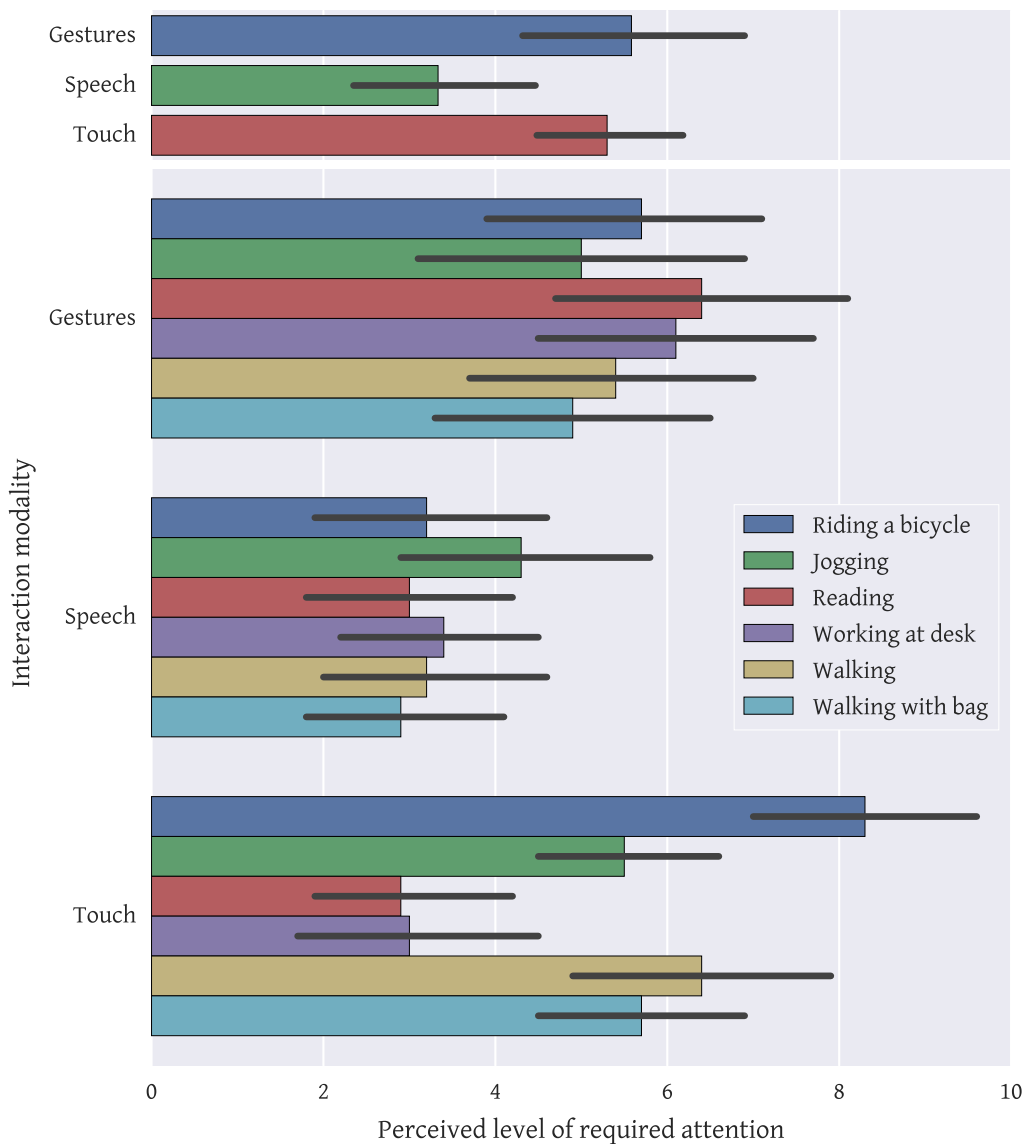


Figure 2.11: Participants, on a 0–10 scale, rated how much attention each interaction modality required when controlling music playback. They rated speech as requiring less attention than both gestures and touch. There are slight differences in the effort rating depending on the scenario. These are most pronounced for touch interaction, where biking stands out as requiring the most attention.

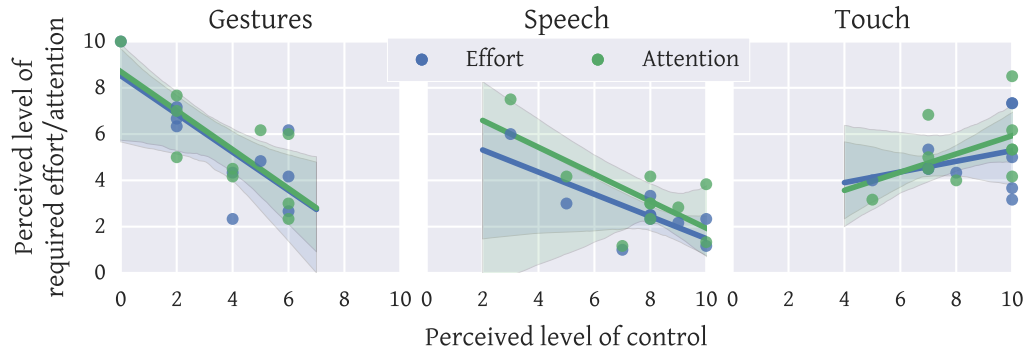


Figure 2.12: For the gesture and speech modalities, participants perceived requiring more effort and attention the less control they associated with a modality. This is the other way around for touch. Note that while effort and attention ratings were collected for each scenario, control ratings were only collected overall.

Finally, we can compare participants' ratings with respect to the required attention of each modality (see Figure 2.11). We again aggregate over the different scenarios and only look at differences due to modality. A Friedman test confirmed a significant effect of modality on the perceived required attention ($\chi^2(2) = 6.6, p < 0.05$). Post hoc testing with pairwise Wilcoxon rank sum tests with Bonferroni correction only showed a significant difference between the touch and speech modalities ($p < 0.05$). The speech and touch modalities showed no significant difference ($p > 0.05$), while the difference between the speech and gesture modalities was close to, but not significant ($p = 0.07$).

The results so far hint at a strong connection between effort and attention. Intuitively, this is to be expected, as a system requiring a lot of attention automatically also requires a lot of effort from the user. We tested this by computing correlation between the effort and attention ratings of each modality using Spearman's rank correlation coefficient r_s . This correlation is strong and significant for all three, the touch ($r_s = 0.71, p < 0.05$), speech ($r_s = 0.98, p < 0.0001$), and gesture ($r_s = 0.91, p < 0.001$) modalities. This is also visible in Figure 2.12.

So far we have only looked at the qualitative ratings of the participants. We can also relate those to how they actually interacted. For this we analyze the logs and compute the share of each modality from the total number of interactions. Figure 2.13 shows that the higher participants rated the required effort of a modality, the less they used it in a given scenario. However, the effect is weak and there no strong relationship. In fact, linear regressions for each modality do not show a significant prediction of usage from perceived effort ($p > 0.05$). One reason for this is the in situ setup of the study. Participants played around with the prototype and performed more interactions than the required minimum in each scenario. Compared to the qualitative data, the usage share data is thus comparably noisy.

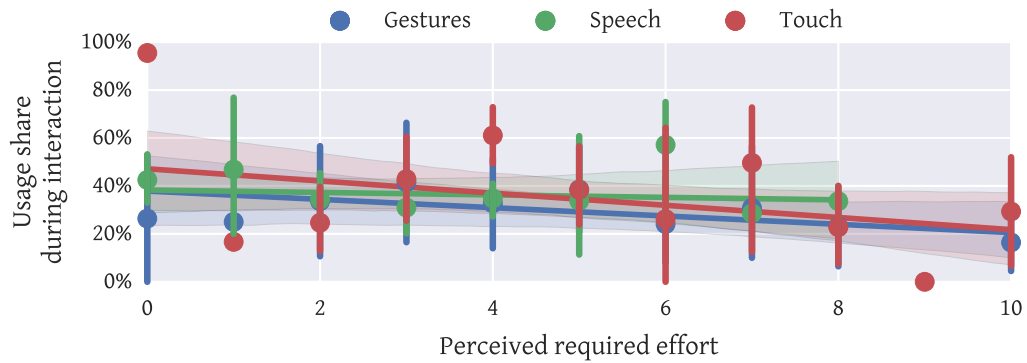


Figure 2.13: We can compare participants’ ratings of how much effort an input modality requires to how much they are actually used. The more effort requirement participants perceived, the less inclined they were to use it.

2.8.3 Discussion

The results paint a complex picture of participants’ perceptions. Two results stand out in particular: gestures were perceived as offering the least control and speech was perceived as requiring the least effort/attention. This is understandable given the used system: gesture input could not be used for the full range of interactions (e.g., it could not be used to switch to a specific song). Similarly, in some scenarios (e.g., riding a bicycle) speech input had a clear advantage as it does not suffer as much from physical constraints. The strongest influence of situational constraints can be seen with the touch modality. Where gesture and speech input was slightly impacted, touch input ratings vary much more. This indicates that even though touch input is the primary modality for most current mobile devices, having other input modalities available can be beneficial for users. Using a method with lower control might not be participants’ first choice, but is a potential alternative in case touch input is restricted.

An interesting pattern shows in Figure 2.12: gesture and speech input are rated as requiring less effort/attention the more control they are perceived to offer. This can also be read the other way around: if a participant felt those methods offer only little control, she was also likely to perceive them as requiring a lot of effort/attention. There might be a potential bias here where a low opinion of an interaction modality influences other ratings of it. However, we did not explore this further and thus the precise relationship between control and other subjective ratings at this point is still an open question.

2.9 Towards Casual Interaction Systems

In this chapter we have explored the concept of casual interaction. Starting from defining the relevant terminology, we have explored reasons for casual interaction and related approaches. We have also looked at two studies of casual interaction behavior. One gave evidence for effort-avoiding behavior in steering tasks, while the other qualitatively explored the relationship users perceive between effort and control.

However, a remaining challenge is how to relate those casual interaction concepts back to concrete systems. The music controller application, used in the in situ study, was a first example of such a system. In the remainder of this dissertation, we focus on several other systems that are built according to casual interaction concepts. While we have so far taken a more general look at casual interaction, those systems offer a chance to further explore specific aspects of casual interaction.

3 Interacting Around a Device

Touch-sensing and proximity-sensing technologies offer an inherent trade-off in intentional control versus the cognitive and physical burden of an input transaction. The progression from button-click, to touch, to hand-near-device is potentially accompanied by a decrease in intentional control by the user, and hence increases the inferential burden (and error rates) of interpretation.
— Ken Hinckley, [119]

Moving interaction away from a device and into the space around it is one way to allow lower engagement interaction. Instead of having to reach for the device, users can then just gesture towards it. In Chapter 2 we already saw one example of a casual interaction system that uses distance to switch between different input techniques. The further a user was away from that device, the less control she had over the input. In this chapter, we further explore how distance to a device and the space around the device can be used for casual interactions. Specifically, we look at moving interaction to proxies found around phones, instead of having to use phones directly. Instead of requiring proxies, we also investigate just using distance for interaction—notification access in particular. In more foundational work, we explore how accurate such interactions in front of a phone can be.

This chapter is based on a paper published at *MobileHCI 2014*, an extended abstract published at *MobileHCI 2016*, as well as two supervised theses¹:

- Henning Pohl and Michael Rohs. “Around-Device Devices: My Coffee Mug is a Volume Dial.” In: *Proceedings of the 16th International Conference on Human-Computer Interaction with Mobile Devices and Services - MobileHCI '14*. 2014. DOI: 10.1145/2628363.2628401
- Henning Pohl, Bastian Krefeld, and Michael Rohs. “Multi-Level Interaction with an LED-Matrix Edge Display.” In: *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services adjunct - MobileHCI '16 Adjunct*. 2016. DOI: 10.1145/2957265.2961855

¹The bachelor theses of Sven Karsten Greiner [90] and Bastian Krefeld [161]

3.1 Introduction

Having to physically interact directly with devices constrains us in several ways. First and foremost, such interaction requires either having devices sufficiently close to reach them or needing to expend additional effort for device retrieval. The further away a device is the higher the cost generally incurred to retrieve it. If we, e.g., imagine a phone lying on a table, then picking it up and using touch interactions requires more effort and precision than waving in the general direction of the device. This cost increases when the user is already engaged in a different task that she has to step away from. For example, consider writing a dissertation when a notification comes in at the phone lying further away on the desk. Pausing the writing to pick up and check the phone in this situation is more costly (in terms of interrupted workflow) than if the same occurred while sitting in a café.

Where focused interaction with current mobile devices commonly means touch interaction, making interaction more casual hence means moving away from the screen. This could still be interaction elsewhere on the phone [328]. However, by moving completely off the phone, to around-device interaction, frees users from interacting that closely with a device even more. Around-device interactions also open up a much larger physical space for interaction. Where the space for touch is limited by device size, around-device interactions are primarily limited by sensor range and occlusion. Having much more space available for input also allows for different kinds of gestures and inputs. Where touch gestures are limited by possible finger movements on the device, free-space gestures have inherently more degrees of freedom.

However, while around-device interaction allows more space for input, there are drawbacks as well. Not holding a device also means that any embedded screens or indicators are not as close and as visible as in direct interaction. Viewing angle, an aspect critical for some display technologies, is also not ideal for devices, e.g., lying around further away. As with output, input sensing capabilities also deteriorate with distance. For example, while a built-in camera (such as used by Surround-see [324]), can image close-by objects at high resolution, that resolution decreases rapidly for objects further away. Similarly, audio signals are more prone to noise the further the source and microphone are apart. Hence, the bandwidth available for interaction decreases with longer distances to the device.

Distance to the device hence is a good potential metric to use for deciding how casual an interaction is supposed to be. It is easily controlled by the user and also is suitable as an implicit control signal. Systems can either split the space around them into discrete zones (e.g., as in [304]), or use it to continuously adapt the level of engagement/control available. Switching between up-close interactions and around-device interaction incurs a cost, yet this cost is small enough not to make such hybrid interfaces infeasible [47].

In the remainder of this chapter, we explore specific aspects of around-device interaction that can support casual interactions. We start with conceptual work that explores moving interaction off of mobile devices and into the space and onto objects around them. This raises three questions which we then explore more in depth: (1) what objects are available for such externalization of interaction, (2) how would users map interactions to such objects, and (3) how well can users interact in the free space around a device if not resorting to available objects. We furthermore look at an example system for notification access on a mobile phone that uses distance to segment the interaction space into discrete zones and a quantitative analysis of pointing in this space.

3.2 Related Work

There is a large body of existing work on around-device interaction. While some papers focus more on sensing aspects of around-device interaction, a larger number of papers investigate interaction techniques and applications for such setups. In addition to general around-device work, we also look at work related to use of tangibles around devices.

3.2.1 Sensing Around-Device Input

Current mobile devices do not typically include the capability for sensing around-device interactions. Yet, there has been substantial previous work exploring how this could be done as well as new sensors in mobile form factors from industry.

3.2.1.1 Instrumentation

Instrumenting the user, the environment, or objects is one approach to sensing around-device interaction. Optical sensing can pick up colorful gloves [305] or markers. With *Digits*, users wear a device on their wrist that is able to track hand gestures [149].

Devices incorporating magnetometers, can track fingers wearing magnetic rings [104] or users holding magnets or objects containing magnets [147]. Hwang et al.'s *MagGetz* are physical widgets (e.g., buttons) with embedded magnets that can be placed around mobile devices for customizable around-device tangible controls [128]. *EyeRing* is a more sophisticated ring with a built-in camera, that was designed around pointing gesture interaction [208].

Researchers have also looked at using electromyography to sense muscle activity to, e.g., recognize hand gestures [266]. However, instrumentation restricts interaction to specific rooms or encumbers the user. Thus, we believe a different approach would be more suitable for around-device devices.

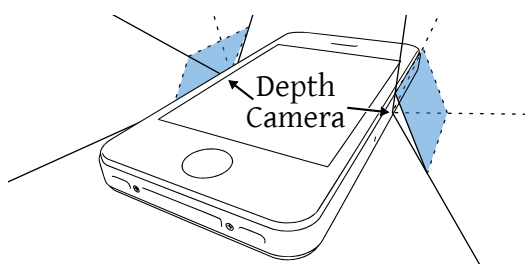


Figure 3.1: Future mobile devices are likely to include depth cameras with a large field of view in the sides of the device. The field of view shown here is equivalent to the one of a Leap Motion controller. Shading does not indicate sensor range but only highlights the field of view.

3.2.1.2 Electric Field Sensing

Many touchscreens on mobile devices make use of capacitive sensing. While this technology works well for 2D tracking, it can also be used to track conductive objects (including humans) in 3D. This approach has recently regained interest with researchers looking at tracking users in rooms [54], or fine grained hand tracking [91]. Le Goc et al., e.g., showed how this technology can be used to track fingers and gestures above the device [170].

To sense in 3D, electric field sensing requires usage of more than one electrode. Electrode placement has to be such that field lines pass through the desired sensing space. Thus, only having electrodes in a mobile device would most likely not be sufficient for tracking interactions in a larger space. One solution would be to have retractable electrodes that can be pulled out and placed where desired or integrate this technology in earpieces, attached to the phone anyway.

3.2.1.3 Optical Sensing

Most phones today already include cameras and we can envision future ones also integrating depth sensing sensors (see Figure 3.1). But even plain cameras can be augmented for around-device sensing. For example, Yang et al. attached an omnidirectional lens to a phone's built-in camera for their Surround-see project [324]. This allows tracking users' hands around the device, as well as objects and activity around the phone.

Kratz and Rohs attached infrared distance sensors along a phone's edge to detect hand gestures performed above [159]. Similarly, in *SideSight*, embedded infrared sensors along the sides of a mobile device enable finger tracking for tasks such as panning on-screen content [32]. Choi et al. arranged the same kind of sensors in a touchpad for *ThickPad*, to enable hover-tracking above it [45]. Instead of individual infrared sensors, *Z-Touch* uses infrared laser planes and detects fingers crossing them [291], an approach similar to the one used in *Mouseless* [199]. The proximity and ambient light sensors already built into phones can also be used for finger tracking in front of the phone. This was shown with the *UbiTouch*, by Wen et al. [312]. Their setup uses the built-in microphone to detect tap events, but is susceptible to light changes and not very accurate.

Depth cameras have also been shrinking and could soon be generally available in mobile form factors. For example, PrimeSense was offering the *Capri* sensor, an embedded design delivering VGA resolution from 0.9 m to 4.0 m before being bought by Apple. Occipital made the *Structure Sensor*², a mobile depth sensor attachment, available in late 2014. A more integrated solution is Google's *Project Tango*³, where a depth camera is directly embedded into a mobile device.

3.2.1.4 Other Approaches

A more exotic approaches to sense around-device input is *SoundWave*, where the doppler effect is used to detect gestures [95]. Ruan et al. extended this work to also estimate additional gesture parameters from the echo data [263]. Instead of sound waves, many other kinds of waves can also be used. For example, around-device sensing has been performed with GSM [332], or WiFi signal strength [1]. Another approach by Cassinelli et al. uses steerable lasers to track fingers in 3D [38].

3.2.2 Interaction Techniques for Around-Device Interaction

Having around-device input available opens up a large design space of possible interaction techniques. For example, Spindler et al. investigated interaction with lenses in above-the-surface layers [279]. As the number of layers increased, in-air tasks became more difficult for users. This shows that there is a delicate balance to be struck when moving interaction off the device: requiring fine positioning might make the overall interaction unpleasantly hard. Earlier work on interaction layers by Subramanian et al. focused on pen interaction [286]. In particular, they looked at how to correct for drift and what kind of selection technique is most appropriate for in-the-air interaction. Building on that work, Kattinakere et al. show how Accot and Zhai's steering law [2] applies to in-the-air interactions [144].

In *Mockup Builder* Araújo et al. provide a concrete example of a continuous interaction space, combining touch on the surface with finger tracking in the air [64]. This, e.g., allows intuitive extrusion of 3d shapes by finger movement through the air. Movement and shape of users' hands in the space just above a device were monitored by Jackson et al. to enable manipulation of 3d objects [133]. Similarly, Hilliges et al. implemented pinching and grasping techniques for seamless interaction on and above the surface [116]. Here, users can pick up virtual objects and drop them elsewhere, making object manipulations more "natural".

²<http://structure.io>

³<http://www.google.com/atap/projecttango>

Working specifically for mobile devices, Jones et al. make use of the larger interaction space around them for panning and zooming tasks [138]. Moving interaction to in-air techniques was found to be en-par with touchscreen interaction and was actually preferred by their participants. A larger design space is explored by Chen et al. in their *Air+Touch* project [42]. They propose tracking finger movement in the air *before*, *between*, and *after* touch events to detect in-air gestures. For example, users could perform a circular gesture and then touch down their finger to conjure an option menu for the object at the touch's location. In general, this is a way to make touches more expressive and reduce the need for mode switching buttons in mobile applications. Finally, Hasan et al. explore using the space around the phone to store and recall content [109]. The space is segmented into "bins", which users access by pointing at it.

3.2.3 Around-Device Tangibles

Both, *Portico* and *Bonfire*, are examples of systems that enable tangible interactions around portable devices. Camera-arms are attached to a tablet for *Portico*, where objects on and around the tablet are detected using classifier methods such as template matching [6]. Users can, e.g., roll a physical ball against the tablet to shoot a virtual ball in a game. In *Bonfire* cameras on a laptop detect objects (via color histograms) in two interaction zones to its sides [141]. While *Portico* is input only, *Bonfire* also includes projection output.

In general, tangibles [131] are a way to increase the affordance of an interface, making possible manipulations more apparent [117]. This need not be specifically designed objects. For example, Carvey et al. investigated using everyday objects as tangibles, differentiating objects by weight, so that placing an object on an attached scale triggers actions [37]. Corsten et al. used objects with modifiable weight to allow changing of the objects' haptic properties [56]. In their *Annexing Reality* system, Hettiarachchi and Wigdor explore how such everyday objects could be used specifically to provide haptic sensations for virtual objects in an augmented reality system [115]. This nicely circumvents the problem of missing haptic sensations for virtual objects.

Instead of tangibles, Henderson and Feiner used existing features in the environment to add passive haptics to augmented reality controls like sliders and buttons [111]. Buttons, e.g., are placed on existing protrusions and spinning, bending or sliding objects are used for 3D widgets. Cheng et al. in their *iCon* system added markers to everyday objects found on users' desks and enable toggle and consecutive control by moving or touching [43].

3.3 Moving Mobile Interaction to Proxies

As we have already established, the close engagement required by mobile phones can be problematic. But our phones are important everyday tools, with us most of the day, wherever we go. Like a swiss army knife, we use them for a wide range of tasks and roles. For example, phones can play the part of the alarm clock, remote control, cook book, level, game controller, or baby monitor. Their computing capabilities and range of sensors, give current phones enough flexibility for these scenarios. Yet, relying too much on these capabilities boxes us into the rather restrictive focused interaction mode.

In contrast to the phone, more specialized devices can offer tailored affordances. For example, a designated alarm clock can allow users to just slap it to turn it off. The phone, on the other hand, is usually a small brick-shaped device that requires the user to touch it for almost all interactions. Not only might this not be the best interaction for a given scenario, but this also requires the above mentioned close engagement with the device. In contrast, interaction with a specialized device can be more casual in nature.

To take advantage of specialized physical affordances, tangibles can be introduced to the interaction. These are often objects specifically designed for interaction with them. An example is *mixiTUI*, where Pedersen and Hornbæk built blocks with markers on them, which can be manipulated to control a music sequencer [228]. However, such purpose-built tangibles are not always readily available. Requiring users to bring along tangibles for each desired affordance would also encumber them too much for a mobile experience. Even when restricting themselves to a smaller subset of tangibles, the individual tangible could still be unacceptably bulky or heavy.

Instead, we propose repurposing objects and space around the phone for interactions (as illustrated in Figure 3.2). We notice that many of the objects that are in proximity to our phones over the course of a day offer good affordances for many common interaction tasks. For example, hacky sacks lying around in a room have a natural affordance for usage as push buttons, e.g., to snooze an alarm. Compared to virtual buttons on a phone screen, such physical *ad hoc buttons* allow pressing with less targeting effort (due to increased size) and allows for interactions with more varying force (we can hit the snooze button hard, if we want to).

Around-device devices are not necessarily bound to a physical object. Instead, such a device can also be “imaginary”, i.e., without an actual physical manifestation. This has previously been explored by Steins et al., who instructed users to use imaginary input controllers and infer the used controller from a user’s hand posture [282]. This enables fast device switching between the simulated controller set. While we cannot have real input devices floating in the air next to the user, we can thus have users pretend to interact with such a non-existing device. Such in-the-air input can also be gestures not directly tied to a specific device.

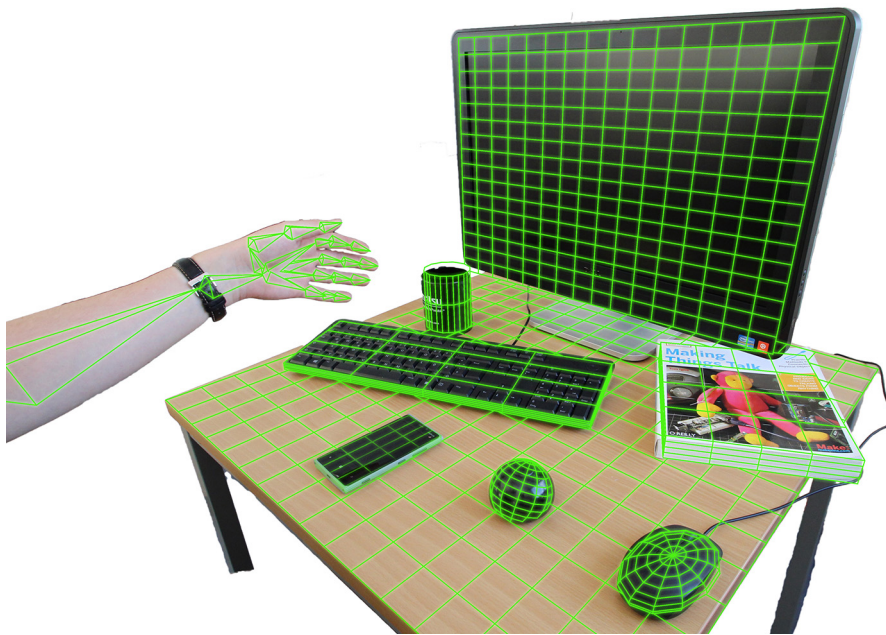


Figure 3.2: Instead of requiring the user to interact directly with the phone, around-device devices make use of the space and objects in the environment to offer more casual means of interaction. In this conceptual visualization the phone tracks the user's hand and detects objects nearby as geometric primitives, enabling interaction with them.

Around-device devices can provide better affordance, appropriateness, and casualness to phone interactions. On the other hand, such interactions around the device will usually not be as precise as direct touch interaction on the phone. Users do retain the choice to pick up their phone for more fine-grained interaction. They can pick the level of control they desire in the *casual interaction continuum*.

3.3.1 Around-Device Devices

Many current mobile devices constrain interaction to touch on the device. Around-device devices, on the other hand, strive to *repurpose* surrounding space and objects for interactions.

The idea of making a space “*active*” through the act of placing a device is similar to *PlayAnywhere*, where Wilson proposed a device to be placed on any table in order to enable interactions on it [317]. One could argue that many of the benefits of around-device devices described in this paper could be achieved by installing tracking systems in those locations we investigate, like the office. However, we believe there are intrinsic advantages to an approach that concentrates on sensors built into a device, observing the space around it.

First of all, instrumented rooms require a much larger amount of equipment. This increases costs and costs scale with the number of places in which a user would like to use around-device devices. Furthermore, there are important privacy concerns: would users, e.g., accept offices that constantly track them and would a date be accepting of a bedroom with cameras in the corners? By having the tracking inside their personal device, users can more conveniently customize their tracking (e.g., by defining different gestures) and ensure their setup always comes with them. This also ensures detection can be tailored better to a specific user, whereas room-based systems have to cater to the requirements of a larger number of potential users. Furthermore, containing the system in their personal devices allows users to retain a sense of control. It is clear *when* the system is tracking (the phone is on the table) and tracking can be easily blocked temporarily by turning the phone around, placing a hand over it or putting it away.

3.3.1.1 Benefits

Looking at the main benefits of around-device devices, they allow users to (1) choose interactors with more suitable affordances, (2) interact more casually when they do not want to grab their phone, and (3) extend their interaction space, no longer restricting them to interaction on the device itself.

Suitable Affordances Current phones cannot morph into different shapes, and thus cannot adapt to the affordance most appropriate at the moment. Tangibles, on the other hand, are purposely designed to offer specific affordances (e.g., to invite turning in a knob). Unfortunately, users do not always carry such tangibles with them or have them ready at home or at work. For many tasks, we can instead make use of objects already available nearby.

Casual Interactions Around-device devices can fill an important role in casual interactions. They essentially offer users a more casual interaction possibility for situations when they need less engaging interaction or are ok with relinquishing some control. For example, in situations where close interaction with a mobile phone is frowned upon (e.g., in a meeting), it might still be acceptable to interact with other objects nearby. This could be something as basic as scrolling through a list of upcoming appointments on a *glance screen*⁴ by turning a mug. While users might not universally be able to give detailed commands this way, they could still provide coarser input. Some coarse interactions such as slapping, more forceful touching or hitting are only acceptable with objects more resilient than mobile phones. This user-chosen trade-off between interaction fidelity and engagement defines casual interactions.

⁴<http://conversations.nokia.com/2013/06/25/a-closer-look-at-glance>

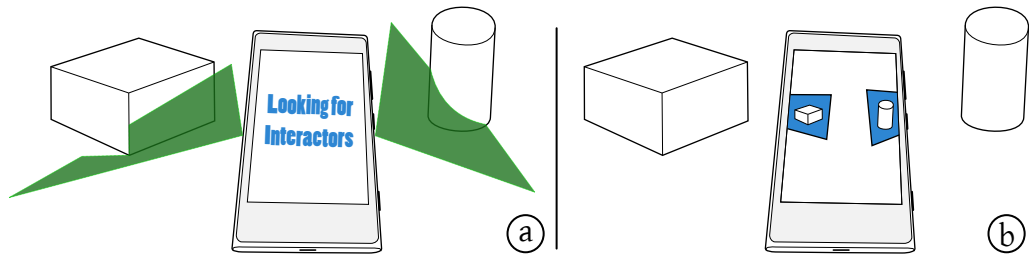


Figure 3.3: Here we show one way users can define around-device devices. When (a) the phone scans the surrounding space it finds two objects suitable for interaction and (b) uses *Wedge* [96] to inform the user about their type and location. Users can then click on an object to, e.g., associate it with an action in a next step.

Extended Interaction Space While phones have been getting bigger recently, the interaction space is still comparatively limited. Enabling input in the space around the device and with objects around it substantially increases the available space. That space is directly tied to the sensing in the phone, i.e., its location and extent is defined by the phone placement. As we are less inclined to place our phones next to strangers, this also means that this space is inherently personal in nature.

3.3.1.2 Challenges

A general problem with any around-device device (and also tangible and gestural interaction in general) is how users inform, e.g., their phone about an around-device device. Ideally, users would not need to retrain such an association every time they intend to reuse an around-device device. However, more ad hoc situations require a fast way to define associations.

When phones track objects around them, those can be offered to the user for associations. In Figure 3.3 we present an example, with off-screen visualization to show surrounding objects. Avrahami et al. indicate tracked around-device devices in a similar way [6]. This can be reactive—the phone detects an object nearby was pressed and asks the user whether an association should be made. Such a prompt is transient and disappears if the user does not engage further with the device.

In cases a more temporary association is desired, it will often be sufficient to offer the user to grab any object or just use their hands. Figure 3.4 shows an example of a game that on startup allows the user to either touch the virtual joystick on the screen or place the hands next to the device for an around-device joystick. The association in this case is established in a dedicated specification mode.

Mappings can also be implicit, offering a kind of plausible default mode, based on a state constraint. For example, when music is playing on a phone a user could expect nearby objects to automatically take on a role in the control of the audio. A nearby object with button affordance could, e.g., be assumed to be a play/pause button, a nearby tube to be usable as a volume slider. We can envision similar situations when an alarm is ringing, a call is coming in, or a beep signals an incoming message. For casual interactions it is important that the overall interaction costs are low. Thus, implicit and temporary mappings are more appropriate in that context.

Mobile projection from the phone could be used to indicate the default objects and possible interactors. Thus projection is also possible along the same optical path. Devices would not project on their surrounding space all the time, but would only do so when queried by the user. Projection can also be substituted by off-screen visualizations of the current mappings on the device itself.

Fundamental constraints are imposed by occlusion of in-device sensing. If the phone cannot see an object it cannot be tracked for interactions, which may not be clear to a user. While this is a general problem of vision based systems, around-device devices can avoid this limitation to some extent by assuming shape constraints. For example, when the phone sees the front of a bottle it can detect it as a cylindrical shape and *infer* interactions on the back. Thus, occlusion in some situations does not impede interaction. Projection could also demarcate a phone's sensing limits to the user. If users can see what is sensed they are less likely to be frustrated by failed interaction attempts.

3.3.2 Around-Device Device Types

In this section, we categorize around-device widgets and outline how they can improve interactions. Widgets in that respect relates back to graphical user interfaces, where common components are standardized to a set of building blocks. In the same way, we try to identify classes of interactors for control with around-device devices.

3.3.2.1 Around-Device Trigger Controls

Phone applications such as alarm clocks, stopwatches, or call response are centered around simple trigger interactions (e.g., stopping the alarm). With touch control, this requires acquiring targets on the screen and sometimes unlocking the screen to start the interaction. Unfortunately, touch buttons provide very little guidance and feedback. In contrast to physical buttons, there is no haptic sensation when pressing down and no tactile sensation when looking for the button.

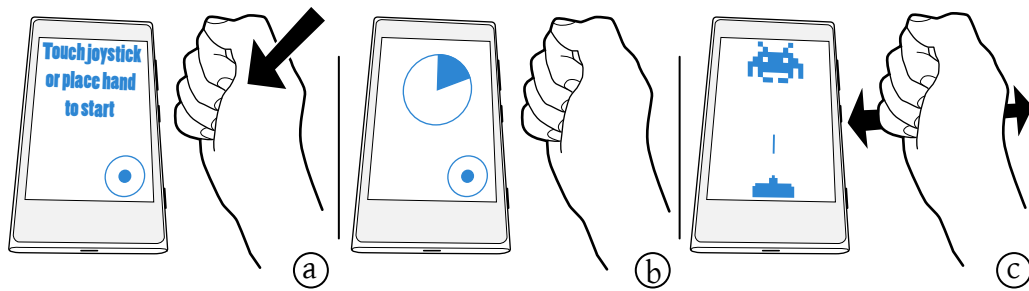


Figure 3.4: Mobile phones with sideways hand detection can offer players an alternative to on-screen joystick controls. (a) At the start of a game, the player is given the choice on what control to use. (b) When placing her hand next to the phone and holding it still for a short moment, the hand is detected as a joystick replacement. (c) The player can then control her ship’s movements by moving the hand next to the phone as if she was holding a physical joystick.

For around-device devices, we propose repurposing objects in the environment as trigger controls. A trigger could be any prominent protrusion [111], but would often ideally be an object inviting pushing. A hacky sack or stress ball, for example, invite slapping and allow users to apply substantially more force to the button than phones would allow (making turning off an alarm that much more satisfying). Apart from tapping or pushing on objects, actions could also be triggered via object arrangements or manipulations (e.g., flipping an object around).

3.3.2.2 Around-Device Linear Controls

Increasing or decreasing some value is a recurring task in many applications. Linear control is required for adjusting the screen brightness, seeking in media, or setting a numeric value with a spinner control. On top of graphical user interface controls, most mobile phones also offer dedicated physical buttons to control the volume level. An example of a tangible linear control are *Fillables* by Corsten et al. who, e.g., propose allowing users to scroll through video by moving a glass [56].

Henderson et al. have proposed utilizing edges, pipes, or cords as a base for around-device linear controls, making use of their haptic properties [111]. As we will show later, a number of objects found near mobile phones fit into this category (e.g., pens, candle holders, or cables). Any object that can be moved closer to or further away from the phone could also be used by mapping distance to value.

3.3.2.3 Around-Device Game Controllers

In contrast to dedicated gaming devices, phones do not offer physical controls. This constraint has yielded many games specifically designed around touch/gesture control. However, this is not always possible and thus we can find a plethora of on-screen buttons and joysticks. Compared to physical controls, those substitutes provide no haptic feedback, making zeroing and not overshooting challenging. Furthermore, large parts of the screen are occluded by the player's hand when using such on-screen controls.

Around-device joysticks or steering wheels could be just tracking the hands themselves (as shown in Figure 3.4). This has recently been explored by Steins et al. [282]. While this enables fast device switching, those *imaginary devices* do not provide haptic sensations. To get zeroing, plungers could be given to players to grab as around-device joysticks. Any round object (e.g., a plate) would be appropriate to form the base of a steering wheel controller. Thus, children could, e.g., be playing a driving game while waiting for lunch.

3.3.2.4 Around-Device Input Devices

Traditional desktop input controllers have also been recreated in imaginary form. Mistry and Maes built *Mouseless*, where a user's hand is tracked next to a device for an impromptu mouse emulation [199]. Terajima et al. show how fingers could be tracked for in-air typing [295]. Canesta in the past offered a commercial projection keyboard for ad hoc typing capability. In-air typing is also available for the Leap Motion controller through the *DexType*⁵ application. Instead of specific shapes for each device, *Visual Panel* tracks a quadrangle and maps user touches to pointing or typing interactions on a desktop [331].

We can envision other input devices being emulated as well. For example, tracking a pen or pen-shaped object around a phone could enable around-device writing or drawing [129]. Simple devices like remote controls can be replaced by gestures [12], but could also map to nearby objects with a more casual interface (i.e., restricting the set of operations, while retaining control of the main task, e.g., changing channels) [244].

3.3.2.5 Around-Device State Control

Instead of directly interacting with objects, their presence and arrangement itself can encode state. For example, if the phone were to detect several mugs together it could switch to a silent mode as not to disturb a conversation. Arranging common objects on a desk gives a subtle control opportunity, were, e.g., a bottle's proximity to the monitor could affect an application. Users could deliberately reveal or hide arrangements to put their phone in different modes.

⁵<http://dextype.com>

3.3.2.6 Around-Device Data Storage

While this paper focuses on controls around a device, the objects and space investigated here, could also be used for “storing” data (i.e., placing references to data objects). For example, Hasan et al. discretized the space in a plane around a device into bins and allowed users to store and browse content there [109]. In addition to such 2D approaches, storing data in 3D space around the user has also been explored [177].

3.4 Environments for Around-Device Interactions

While phones are designed for mobile use, we posit that most of the time they will, in fact, be placed somewhere. The around-device devices concept, as detailed above, is designed for just this kind of situation. So far, we have explored this scenario based on our assumptions of these placements. Here, we further explore environments for this kind of interaction, starting with an ideation phase. However, this is unsatisfactory and ideally we would like to have ground truth data available to inform what kind of interactions are actually feasible and/or desirable. Hence, we also collected ground truth data on actual device contexts via a crowdsourcing task.

3.4.1 Envisioned Settings

We initially brainstormed to collect a set of locations where we think phones would be lying around. This was done to inform subsequent steps of more in-depth evaluation. In the brainstorming, we identified four such locations:

Office: When at the office, people will often place their phone on their desk. While working on their computer the phone then stays within reach and users can still see notifications coming in.

Bedroom: Many people will be using their phones till the end of the day and/or use them as alarms, making placement on the nightstand near the bed a common option.

Kitchen: During breakfast or dinner, phones might be on the dining table or on kitchen counters. It is likely people take their phones with them as they start their day and do not just grab them on their way out of the house.

Living Room: While relaxing in the evening, phones might be placed on the couch or coffee table when not in use. This might be the time to watch TV, converse with friends or family, or maybe use the tablet instead of the phone.

For each of those locations, we then also determined what kind of objects we would expect to be available for interactions. We do not claim these lists are exhaustive, but they provide a rough bearing of what is characteristic for such places.

Office: staplers, monitors, keyboards, mice, landline phones, mugs, books, pens, office toys (e.g., Newton’s cradles).

Bedroom: pillows, books, lamps, glasses cases

Kitchen: cereal boxes, bowls, cups, mugs, glasses, silverware, candle holders

Living Room: remote controls, pillows, books, lamps, bowls, boxes

Depending on the location, different kinds of interactions are appropriate. In an office setting, around-device devices could, e.g., be useful for more casual interactions during a meeting (see earlier scenario) or to support peripheral tasks at the desk. At home, we see the role of around-device devices also primarily as enabler for casual interactions when sitting on a couch or lying in bed. When in the kitchen, users could also benefit from around-device interactions when not willing to interact with their device, e.g., because of dirty hands.

3.4.2 Phone Context Ground Truth

While we established a rough idea of places and objects for around-device devices with the brainstorming, we felt this does not give a full picture of the space. Previous work gave us additional data on those settings. For example, Harrison and Hudson visited ten participants at home and at work to look at placement locations of their phones [105]. Unfortunately, their list focuses on the materials at those locations and does not include details on the space and the objects available. Another six users were visited and interviewed by Carvey et al., who also surveyed nearby objects [37]. This investigation, however, does not focus on phone locations. On the other hand, they made observations on people’s object mappings, e.g., how in *transient couplings* users were inclined to repurpose any nearby object for simple and immediate tasks. Finally, Cheng et al. collected photos of 33 desks and the objects on them [43]. This study also focused on desktop scenarios, but the paper unfortunately does not provide detailed object information. However, they did learn that most participants placed phones on their desk.

To get some qualitative validation of our own for our envisioned scenarios, we set out to gather real-world data on the places where people lay down their phones. We used the Scoopshot⁶ photo crowdsourcing service to collect photos for this purpose (a subset of the photos can be seen in Figure 3.5). On Scoopshot, users can create tasks to ask other users for photos, matching the task’s description. We created several tasks asking users to send in “*photos of places where people usually have their phones lying around*”, where “*the phone and the context (e.g. objects around the phone) should both be visible in the photo*”. Tasks can be assigned to a geographic region, where users get a notification of the new task on their phone. However, users from elsewhere are able to submit photos as well when seeing the task on the Scoopshot website. We ran the task in several regions with high user density: around Amsterdam, Berlin, Copenhagen, Essen, Frankfurt, and Istanbul.

⁶<http://www.scoopshot.com>

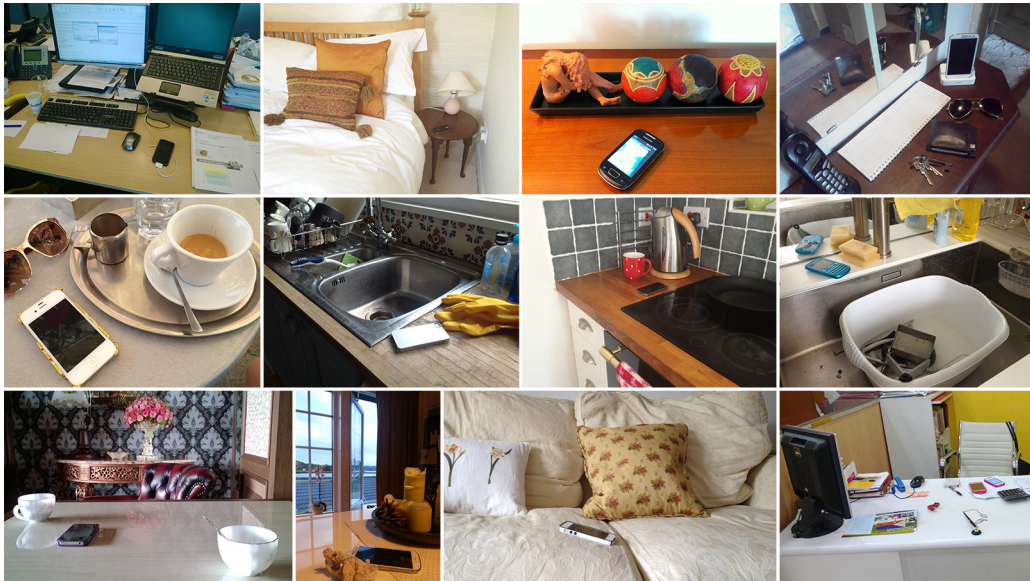


Figure 3.5: Photos depicting settings where phones are lying around. Overall 510 photos were collected via the Scoopshot crowdsourcing service. The photos depicted a wide range of situations, e.g., offices, bedrooms, kitchens, or cafés. Copyright of shown photos (numbered in row-major order): Ricardo Alves (3), Jens Bloszyk (4), Sarah Jenkins (2, 9, 11, 6, 10), Petra Larsson (8), Sem Lemmers (1), Brad McDougal (12), Rostislav Sedlacek (5), and Chaichan Srisawat (7).

We collected a total of 510 photos⁷ from 249 different users. We were able to collect photos from a wide range of locations (see Figure 3.6 for a map), not restricted to only those we advertised the task in (note that not all photos included location information). The majority of submissions still comes from Europe (91.9%, including Turkey), so there might be some cultural bias in the dataset. While the majority of users followed the task description, many sent in photos not in line with our requirements (e.g. no phone visible in the photo, or a closeup picture of a phone). Hence, we removed those contributions, leaving us with 286 valid photos.

For each photo, we manually annotated the location of the phone and the nearby objects. We only included objects visible in the photo and did not infer other objects (i.e., no mouse is assumed just because a keyboard and a computer are visible). We also only included objects in the range of the phone, if, e.g., an object is only visible in the far distance (beach, outside, ...) it was not counted. Table 3.1 shows the most common locations, while Table 3.2 shows the most common nearby objects.

⁷Photos for each task available at <http://www.scoopshot.com/v2/task/{bdwnmsxpvvhdd | cnqlcblts-gxvr | bxvhlmpwqcgx | bnnqhlrcbrlzg | lrdljhdppddxm | bxcjxxpvlbszk}> respectively.

Table 3.1: Most common phone locations in the photo dataset.

Location	Count	Location	Count	Location	Count
Table	94	Couch	10	Shelf	5
Desk	50	Restaurant	8	Dresser	5
Bed	20	Garden	8	Car	4
Kitchen	11	Bathroom	7	Bag	4

Table 3.2: Most common objects near phones in the photo dataset per manual annotation. Corresponding geometric primitives given where applicable. As can be seen many nearby objects can be reduced to simple shape primitives. However, some (such as flowers) are more complex in shape.

Object	Count	Object	Count
Cable	38	Tray	12
Phone	37	Container	12
Glass	35	Can	12
Box	33	Flower	11
Paper	32	Notebook	11
Bottle	30	Candle holder	11
Laptop	28	Lamp	11
Bowl	28	Charger	11
Cup	27	Ashtray	10
Keyboard	24	Saucer	10
Mouse	24	Headphones	10
Pillow	23	Figurine	9
Pen	18	Handle	9
Monitor	16	Spoon	8
Book	15	Fruit	8
Sunglasses	13	Pot	8
Tablet	12	Post-its	8
Mug	12	Vase	8
Plate	12	Keychain	8
Bag	12	Remote	7

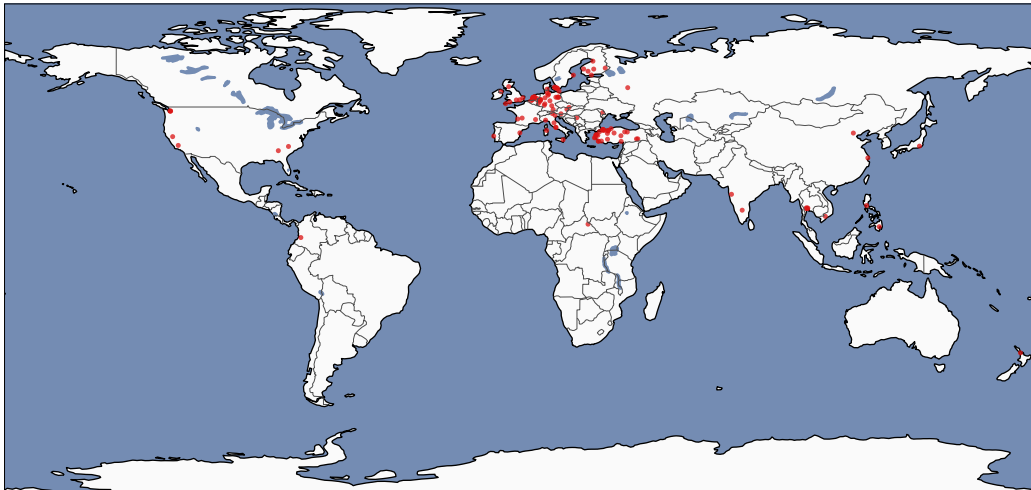


Figure 3.6: Map showing where the collected photos were taken for the 493 photos containing location information.

Many participants also left comments in a field reserved for the photo description. While mostly just describing their photo, some left more general comments, such as:

- “My mobile phone must lay beside the bed when i’m [sic] sleeping”
- “Wherever I am, my phone is too! It even comes with when I’m making food and is usually playing music!”
- “If I was cooking. My smartpohne [sic] is in the kitchen too.”
- “Das Smartphone liegt zwischen allem anderen auf dem Schreibtisch, immer griffbereit” (The phone is lying on the desk between all that other stuff, always available)
- “It is always near me”
- “Charging the Phone at work waiting to the end of my shift”
- “Under the pillow when sleeping”

3.4.3 Discussion

The locations we got from ground truth roughly match up with the set we expected. Tables and desks clearly are the most common locations to place a phone down. This is understandable, as these are locations at which users likely sit down for longer periods of time. Furthermore, the large, flat surface available also readily affords placing the phone there. Additionally, we also saw some locations that were more surprising to us, such as boats, bathrooms, or window ledges.



Figure 3.7: Exemplary object arrangement for a desk scenario, including all shape categories found in the ground truth study.

Objects found near phones also largely overlap with our expectations. Bottles, glasses, bowls, and lamps are rather common, while we did not expect the amount of nearby other phones (which supports collaborative usage scenarios such as [180]). Generalizing the found object set, we can sort the found objects into a set of five categories (also illustrated with proxies in Figure 3.7):

Spherical objects, such as balls or some fruits

Dome-like objects, such as plates, bowls, or cups

Cylindrical objects, such as bottles, cans, mugs, or candles

Rectangular cuboids, such as boxes, monitors, books, or other phones

Complex shapes, such as sunglasses, crumpled paper, or figurines

While each object may have additional semantic cues for how people perceive it to be used, their shape already imbues them with certain affordances. For example, spherical objects invite rolling, while cylindrical objects invite turning around the up-axis. These kind of objects could thus be more readily integrated into an around-device devices application, as the way they “want” to be used can be more easily perceived.

For many of the observed objects, we can make additional assumptions according to Gestalt law. For example, seeing the front of a glass one can assume that the back is similar. By focusing on interaction with objects corresponding to geometric primitives, we can use those assumptions to, e.g., alleviate occlusion problems somewhat. Just because a part of an object is not seen, does not mean it cannot be tracked or that touch on that part could not be inferred.



Figure 3.8: We asked participants how they would perform a series of ten tasks when limited to casual around-device interactions (object movement, object arrangement, gestures, coarse touch, ...). This was repeated for (a) living room, (b) café, and (c) office scenarios with different objects available.

3.5 Eliciting Gestures for Around-Device Devices

The collected images have provided us with a good overview of what objects would be *available* for users to incorporate into their interactions. However, most likely not all those objects would be considered equally for each interaction by actual users. For example, an object that invites *turning* (such as a cup) could potentially be perceived as more fitting for a task where a value needs changing (such as during volume control), than an object that invites *pushing*. Thus, we designed an elicitation study to collect additional data on which objects would potentially be *used* for interactions if available. This is similar to a previous study by Wobbrock et al., who asked users to come up with tabletop gestures to achieve a shown effect [322].

3.5.1 Participants

We recruited 15 participants (5 female, age 20–46, $\bar{x} = 29.07$, $SD = 6.88$) from our institution. The study took approximately 30 minutes and after completion, participants received a small non-monetary gratuity.

3.5.2 Task

We defined a set of ten tasks we felt were an appropriate subset of possible around-device device input scenarios. While covering many phone-specific tasks, we did not include tasks requiring prolonged interaction. Thus, the task list does, e.g., not include a task where participants could be expected to use an object as an ad hoc joystick. However, the used tasks allowed participants to potentially make use of around-device buttons, sliders, or state control.

During the study, participants were asked to find the interaction they deemed most suitable for:

1. Changing the volume of music already playing
2. Rejecting an incoming phone call
3. Dimming the lights
4. Check whether any (email, SMS, IM, ...) new message has arrived
5. Notifying their next appointment they would be a bit late
6. Skipping to the next song
7. Setting an alert/notification for when to leave to catch the next train
8. Querying their sport team's current score (game still running)
9. Querying the status of their ongoing ebay auction
10. Muting their phone

Participants were told that the system has any task-relevant context information available (e.g., calendar with upcoming appointments, information on favorite sport team, or train schedules). They would hence not need to relay context information and could limit their interaction to communicating intent. For example, in the task where they were asked to find out their team's current score, they did not need to tell the system which is their favorite team.

3.5.3 Procedure

We instructed participants not to consider speech input and close interactions with the phone (such as unlocking and starting an app). As this was an out of the ordinary requirement for many, we used a *priming* technique [204] and pointed out several other approaches they could use with objects instead: (1) moving or rotating, (2) changing arrangement, (3) placing them on top of others, (4) appropriating features or surfaces for touch, (5) performing gestures relative to them, or even (6) performing gestures independent of any object. We did not ask them to restrict themselves to these examples or indicated any preference.

As we were only interested in input, we told participants not to worry about output and just assume any feedback would be presented at an appropriate location or provided via audio. Furthermore, we assured participants any answer would be okay including reuse of previously performed interactions. To ensure participants would not concern themselves with recognizability, we told them to assume having previously defined that interaction and that sensing would be capable of picking up any interaction they perform.

In line with the most common settings determined earlier, we choose three different scenes for the study (shown in Figure 3.8): a living room, a café, and an office. We recreated each of these settings in our lab. The objects available in each varied, but there was always a mobile phone present in the scene. Other objects present in multiple scenes include: notebooks, bottles, books, umbrellas, pens and markers (see Figure 3.8 for shots of all present objects). Even though we found multiple phones were often present in the collected dataset, we chose to concentrate on single-user scenarios here where this is not the case. All of the ten tasks were repeated in every scene. We counterbalanced scene order using a reduced Latin square design and randomized task order for each scene.

3.5.4 Results

Overall, participants provided a wide range of interaction choices. There was substantial agreement for some tasks, but generally we observed many different approaches.

3.5.4.1 1. Change Music Volume

Participants almost exclusively followed one of three strategies: (1) turning some object (mostly the phone) on the table, (2) performing a circle gesture on a nearby object (often on the headphones, if available, or on furniture, especially armrests), or (3) using a hand up/down gesture. This behavior was sometimes coupled with an activation step such as clapping three times or pointing at speakers.

3.5.4.2 2. Reject Phone Call

The most common interaction used by participants was flipping the phone upside down. A close second is performing a “brush aside” gesture over or towards the phone. Some participants also knocked or patted the phone, placed an object on the phone, or performed a “put handset down” gesture.

3.5.4.3 3. Dim Lights

Here participants used interactions similar to those in task 1. However, they would often point at the lights first before initiating a gesture and use different objects, such as the candle. In the living room scene, several participants used parts of the lamp as base for sliders. Additionally, some participants used hand open/close gestures.

3.5.4.4 4. Check for New Messages

Most participants here chose to just tap the phone once, stating that any interaction should prompt the phone to display such information. When using gestures, participants would draw question marks on, or circles (like a “refresh” gesture) next to the phone. Many participants associated the notebook with checking for new messages and expected opening the notebook to display the status. Two participant used subtle cues such as glancing at the phone or turning it towards them.

3.5.4.5 5. Send Delay Message

Generally, participants made three mental connections here: (1) associating the task with leaving and thus, e.g., placing the umbrella or the phone itself ready to go (e.g., at table edge), (2) tapping on their watch or mimicking a clock hand with a pen, stressing the task’s association with time, or (3) interacting with the notebook (often described as calendar) because it is associated with the appointment itself.

3.5.4.6 6. Skip to Next Song

Here participants almost universally waved right, swiped right, or moved objects to the right. They would also tap on the right side of objects (e.g., headphones or chairs) or on the right of two objects (pillows). This was sometimes coupled with claps or pointing to speakers like in the volume changing task.

3.5.4.7 7. Catch-a-Train Notification

Similarly to task 5, many associations were formed around leaving, the most common one being placing the phone or the umbrella in a “ready to leave”-spot. Many participants also performed a “walking fingers” gesture for the same purpose. Some also placed objects so they would point to the door.

3.5.4.8 8. Sport Team Status

Most participants here thought of soccer and thus interacted with a ball (or other roundish object if no ball was available) to query the status. One participant arranged three objects in a row to query Formula One rankings. When not referring to a specific sport, participants often just tapped the phone, assuming this would also show game scores (similar to message status in task 4). Some also used notebooks, associating those with a general information lookup action.

3.5.4.9 9. Auction Status

There was little agreement in this task and participants used a diverse set of interactions. Some made connections to physical auctions, e.g., mimicking use of a gavel. Several times an object (often the camera) was picked and determined to be a stand-in for the auction. Often participants pointed out this task falls under a generic “query status” category and reused tap-on-phone gestures like in task 4.

3.5.4.10 10. Mute Phone

Almost all participants flipped the phone and placed it screen down for this task. Some also used variations of a shush gesture or placed objects on the phone to cover it.

3.5.5 Discussion

Looking at the interactions picked by participants, we see a strong preference for some (e.g., flipping the phone) and less consensus for others. Overall, we expected participants to make more use of available objects. Instead, participants often used gestures, frequently near or on specific objects, and placed objects in dedicated zones. We speculate this might be due to unfamiliarity with this kind of interaction and *legacy bias* [204]. Correspondingly, phone flipping might have been frequently used because it is already available in some phones for silencing an incoming call. However, we were impressed by the diverse set of associations participants were able to make. Some associations were very abstract, such as placing three paper clips in a row to represent a Formula One ranking. On the other hand, some associations were very concrete, such as writing “ebay” on the table with a finger on the table to check an auction’s status.

While we told participants to assume context awareness in the system, we still did not expect how much many participants relied on this. For tasks that can be grouped under *check status*, participants very often just looked for trigger interactions and expected the system to show all their desired information. It would be worthwhile to further investigate how we can design systems with more adaptive notification mechanisms that take those expectations into account. This also underlines the need for casual interaction systems to incorporate models for support lower-engagement. Only if we can predict which sport and team is likely meant by a generic “score” gesture can complete the interaction. Not requiring the user to specify everything when she delegates control to the system is a basic principle of casual interaction.

3.6 Notification Access Around a Phone

We have already learned in this chapter how around-device interactions are a means to have lower engagement interactions. Furthermore, we have looked at moving interaction off the phone and into the space and environment around it. While doing so, we have shown that one scenario is particularly common: having phones lying around on desks and tables. Users might, e.g., arrive at their offices in the morning and place their phone down in one corner of their desk.

Here, we take a closer look at a specific problem: checking notifications on a mobile device. This is a common interaction (out of curiosity or in reaction to an audible alert) and yet does regularly incur a high cost. Checking for notifications in the “phones lying around” scenario detailed above, regularly requires reaching for the device and unlocking it to read the available notifications. This also forces users to break their current work and devote close attention to their phone. For casual interactions, users should be able to access these notifications without this kind of close engagement.

As we discussed, further away from a device, sensing is less accurate and only limited feedback can be perceived. Thus, using distance to control the fidelity of notifications and interact with them is a good fit. As users get closer to the device, more information is revealed and additional input options become available. This allows users to control how much they want to engage with the device. For example, just checking whether there is any notification should be possible with very little effort. By giving users the choice of how much they need from the device in any given situation, we can enable them to only need to expend effort when really required to. Similar to our concept of varying control in interaction with a phone, previous work has explored changes in granularity for the smartwatch. Pasquero et al., e.g., discuss how users can get a coarse information fast and uncover additional details by prolonged interaction and additional user effort [224]. Similarly, Pearson et al., include granularity of displayed information as one dimension in their concept of smartwatches as public displays [227].

For this scenario, we are particularly interested in the feedback to be used with this kind of system. It needs to work in the periphery for quick glancing, but also scale up a bit to allow more granular feedback. We hence use a custom LED-matrix display prototype on the edge of the device, to emulate future devices that are designed for off-the-device information access. This allows for coarse, but bright, notifications in the periphery of attention, but scales up to allow for slightly higher resolution feedback as well. By combining an infrared distance sensor with this display, we enable both input and output in the periphery.



Figure 3.9: Our prototype emulates future phones with high brightness displays at their side. It consists of a red dot-matrix LED display and a proximity sensor mounted underneath a phone.

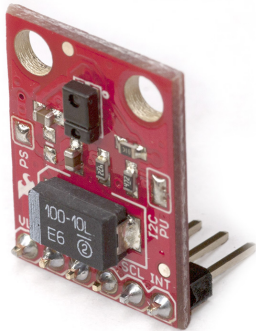


Figure 3.10: In our prototype, we use an Avago APDS-9960 distance and gesture sensor to measure how far the user's hand is away from the device.

3.6.1 Prototype

Our prototype is designed as a phone attachment (see Figure 3.9). It contains a dot-matrix display and a proximity sensor (see Figure 3.10)—emulating future devices that could have such capabilities build directly into their edges. Samsung already builds some mobiles in their *Galaxy Edge* series with a display that slightly slopes over one edge. While our prototype contains a phone, its addition is purely cosmetic and it is not used for any interaction.

We use an LED dot-matrix display to render visual feedback. This is an extension of previous work where a strip of LEDs was mounted around a phone [248]. Using individual LEDs allows for more brightness than a backlight and aids perception of the display when it is in the periphery. Instead of a single row of LEDs, having a whole dot-matrix of LEDs, allows us to display more detailed graphics while still maintaining the brightness levels achievable with individual LEDs.

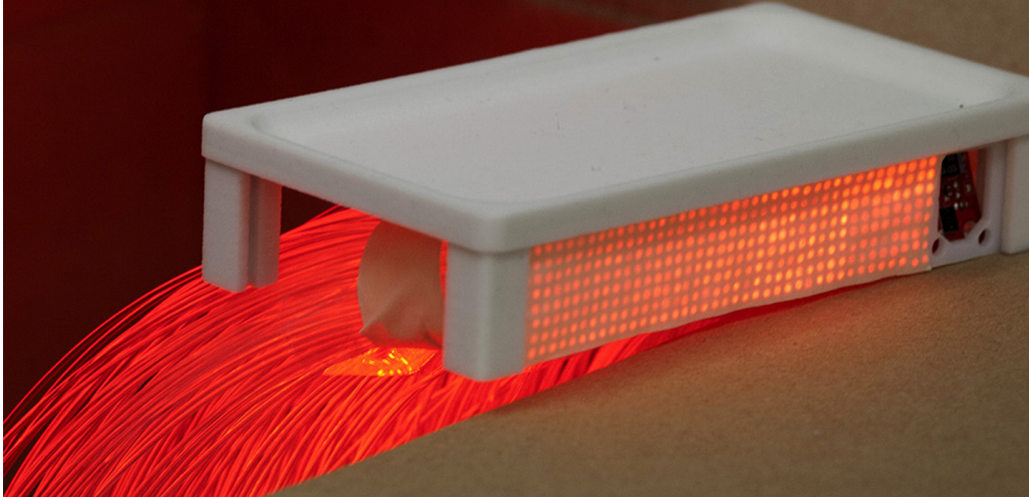
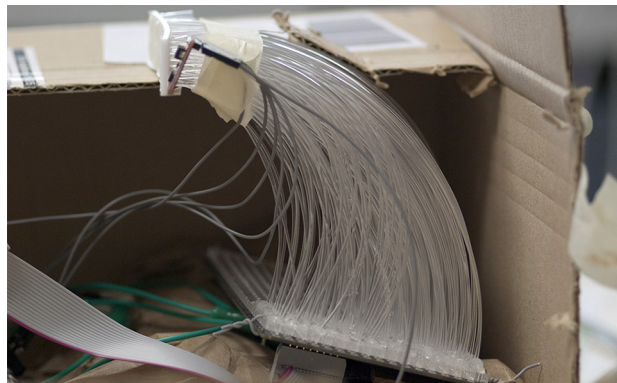


Figure 3.11: To achieve a higher dot density, we redirect the light from a dot-matrix display's LEDs to the prototype via optical fibers. A total of 320 strands of glass fiber connect the individual pixels. The visible display arranges the LEDs' dots in a 40×8 grid, held in place by a laser-cut acrylic stencil.

Figure 3.12: Glass fibers reroute the light from a large LED dot-matrix below the device to our edge display. This allows prototyping with denser LED grids than readily available ones.



Because we could not get an LED dot-matrix display in a suitable form factor for an edge display, we reroute the light to allow for a custom pixel arrangement. As shown in Figure 3.11, this is achieved with glass fibers that connect individual LEDs with our final pixel locations on the edge of the device. This requires space underneath the device (see Figure 3.12), but allows for a height reduction of the visible part of the prototype. We used slightly larger glass fibers for ease of prototyping, but future iterations could use much finer fibers for a further reduction in height.

The display is driven by an Arduino microcontroller, which also collects sensor readings from the proximity sensor. However, the Arduino merely acts as a relay, passing sensor readings to a laptop, which renders the next frame for the Arduino to show on the display. This enabled faster prototyping than running all code on the Arduino itself.



Figure 3.13: Display design for different notifications: messages, battery, clock, calls, alarm. Each row shows a different distance-dependent state with coarse feedback at the top. Thus, the lower the row, the more granular the feedback. For example, the messages first show just a message indicator animation, then an icon for each available message, then who a message is from, and finally the message subject.

3.6.2 Interaction Design

We designed interactions for several different notifications: incoming messages, battery level, clock, calls, and alarms (see also Figure 3.13). Interaction is split into four different levels, where level 0 is the default level when no interaction with the device is taking place. With increasing engagement, as the user comes closer to the device, the level increases. When no notification is active, the device defaults to clock mode and displays the time as the user approaches. Granularity here progresses from just showing a coarse watch to showing the exact time. Other modes become active once a corresponding event is triggered, such as a message coming in, an alarm triggering, or the battery depleting to a low level. In such an event, the device first plays back a notification-dependent animated icon (e.g., moving diagonal lines for the alarm). This informs the user about the type of upcoming notification and grabs attention.

Users can delete notifications by moving their hand close to the device and holding it in position for a brief moment. This activates deletion mode where users can confirm the deletion by swiping sideways. At this point, they can also move their hand back to cancel the deletion. This, e.g., allows dismissing calls without picking up the phone (see Figure 3.14). When dismissing an alarm, we added an additional challenge to the deletion. Instead of just holding in place then swiping, users have to complete a short minigame. Here they have to align a bar with markings by moving their hand to the respective distance from the device. This has to be done three times (the markings move to random positions for each challenge) to finally dismiss the alarm. We made this dismissal more challenging to experiment with forcing more engagement for more impactful interactions. A user might, e.g., commonly ignore alarms and could use this dismissal mode to force herself to give alarms more attention.

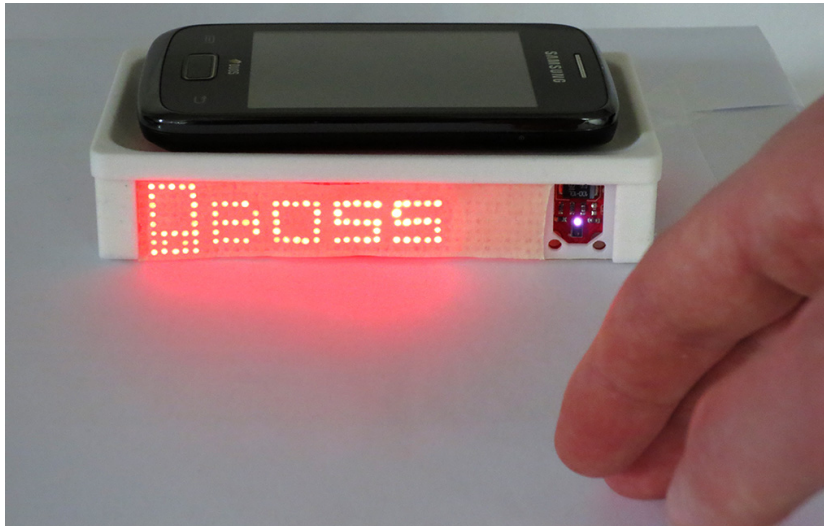


Figure 3.14: In call mode, users can reveal additional information about the caller by engaging more with the device. Here the name of the caller, the user’s boss, is shown and the user can then decide whether to dismiss the call or accept it.

3.6.3 Evaluation

We ran an evaluation to investigate how well the concept and prototype would be received by potential users. The evaluation was split in two phases: in an initial group phase, groups of three participants discussed the concept, while in a second phase, participants used the prototype individually. Overall, 9 participants (all male, age 23–34, $\bar{x} = 26.7$, $SD = 3.8$) took part in the study and were assigned to three different groups.

We explained the concept of using proximity as input for different levels of control to all participants before they engaged in their group discussions. We did not show the prototype and hence our concrete design to participants at this point, but instead tried to determine whether their designs would align with our own. In their group, participants then discussed the concept among themselves and tried to come up with possible mappings and applications for it.

All groups considered connecting to the Android notification center as source for content. One concept discussed was levels of display for time, showing only the time, only the date, or both, depending on the level of engagement. Group one brought up that this functionality could be used to control system settings such as volume in addition to showing notifications. Group two came up with a concept for message browsing. Here users can move from a coarse level, just showing message counts, to the inbox level, where message titles scroll through. Moving one level in control, users could peek at individual messages. Overall, we saw our design concept validated by the group discussions. The idea of “*drilling into*” content was regularly brought up.



Figure 3.15: We asked participants to sit down at a computer and copy a text from a webpage to a document. The prototype was located in the periphery of their visual field when copying the text and occasionally displayed notifications.

After the group phase, we had participants interact with the prototype individually. We showed and explained every function of the prototype to them, but left out a description for how to dismiss the alarm to see how they would react to that. We then asked participants to sit down at a computer and copy a text from a webpage to a document with the keyboard. The prototype was placed next to them, so it would still be visible in their periphery (see Figure 3.15). During their engagement with the primary task, the prototype would occasionally display a notification which the participants reacted to. Afterwards, we asked them to rate their experience.

All participants were able to successfully react to and interact with the notifications. In case of the alarm, which we did not explain to them, they were slightly surprised, but figured out how to dismiss it on their own. All but one participant could see himself using such a device in their day to day life. Participants gave mostly positive responses (see Figure 3.16) when asked to rate several statements. Asked for possible improvements, participants noted that the text scrolling speed could be faster, allowing them to skim more notifications in a given time. They also stated that color would be a great improvement, as it would make it easier to distinguish notification sources (e.g., blue for Facebook).

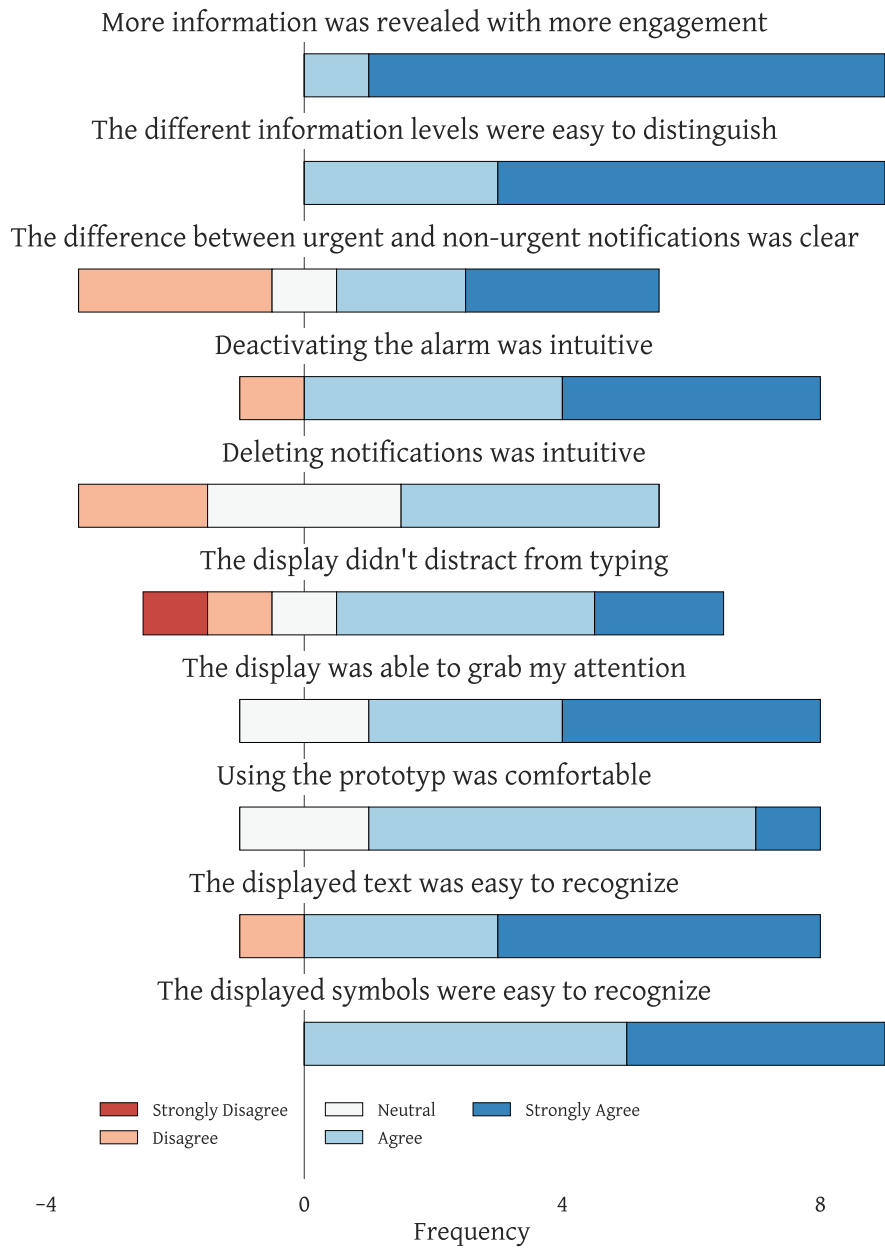


Figure 3.16: After participants interacted with the prototype, in a situation where they were engaged with a different primary task, we asked them to rate their experience on a 5-point Likert scale. Participants mostly rated the system favorably.

3.6.4 Discussion

Results from the evaluation were overall encouraging. The group discussion confirmed our design while each user was also able to effectively interact with the device. Users also rated interaction and concept favorably and indicated they would be open to using such a system if it were integrated in a future phone. We also found that the interactions we designed are discoverable. Even with alarm dismissal, where no instructions for that special more were provided, all participants were able to figure out how to react. This is likely due to the small gesture set and the clear mapping of distance to engagement. Discoverability could be further improved with feedback on where gesturing is expected [82].

Users positive view on the concept itself might be due to how they themselves use their devices. Seven of the nine participants stated that they place their phones on the table when sitting down to, e.g., work. This indicates that having the phone available for secondary interactions, such as checking notifications, is already common. Supporting this behavior by allowing users to scale back how much engagement they need to devote to the secondary task might thus be worthwhile.

3.6.5 Summary

We have presented a concept and design for a system that allows users to scale their level of interaction with notifications. By augmenting the edge of a mobile device that is lying around, users are enabled to interact with it without needing to pick the device up. This enables more casual interactions and also empowers users to only expend as much effort as they need: if they only need coarse information, they can get it with a brief and further away interaction. Only when they require additional details do they need to further approach the device and engage in closer interaction.

While our prototype is still quite large, we think such capabilities might come to future mobiles. We already see phones with one curved edge display and progress in display technology, particularly in OLEDs, could allow for much more flexible display placement. More powerful OLED technology would also be able to achieve the level of brightness we get from using individual LEDs. Integrating sensing for around-device interaction is a different challenge. However, our concept does not require precise finger or hand tracking and very coarse distance estimation would already be sufficient to enable the distance dependent interaction levels presented here.

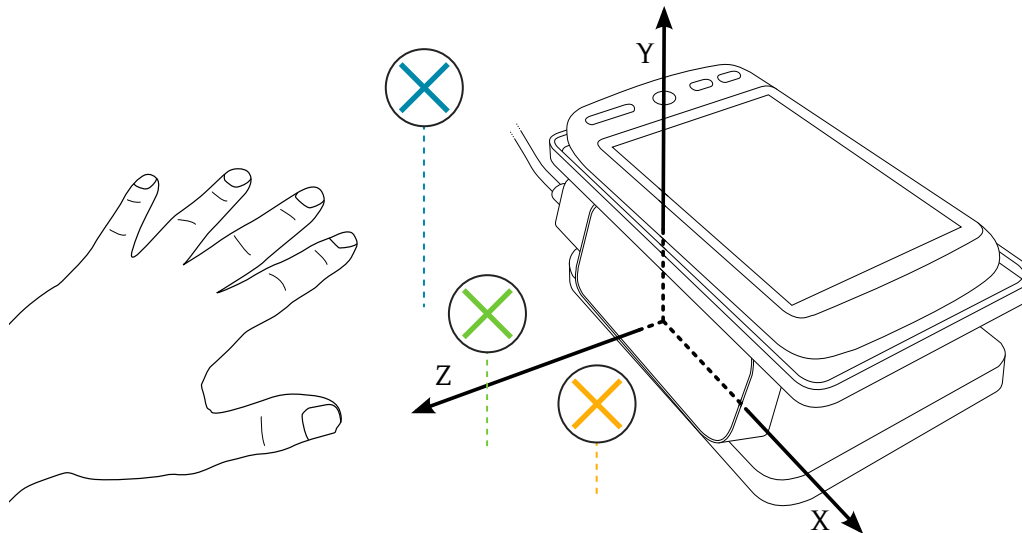


Figure 3.17: We envision future mobile devices to come with built-in sensing of their immediate surroundings. This enables moving interaction off the device, but begs the question how well such interaction can be performed. To investigate this, we use a prototype attachment of a Leap Motion sensor to a mobile phone. This emulates the envisioned device and allows us to measure how accurately users can select in-air around-device targets.

3.7 Around-Device Interaction Performance

As we have seen, moving interaction to the space around a device is a way to facilitate more casual interactions by easing the requirement to closely engage with the device. In Section 3.6 we have just looked at one example of this: notification access. But the interaction used in that example was simple, only making use of the distance of the hand from the device. Ideally, we would like to also have more complicated and intricate interactions in this space. But how much can actually be done when gesturing towards a phone that is lying around or pointing in the space around it? In this section we investigate this and measure how granular designs can be.

Figure 3.17 illustrates the goal of our investigation: reaching for targets in the space in front of a device. If users can be precise in one area of space this would enable more focused interactions there as well. On the other hand, if we find that some parts of the around-device space only allow for coarse selections, these areas would then also be only suitable for more casual interactions. Quantifying pointing error is hence a way to estimate the feasible balance between focused and casual for this specific around-device interaction scenario.

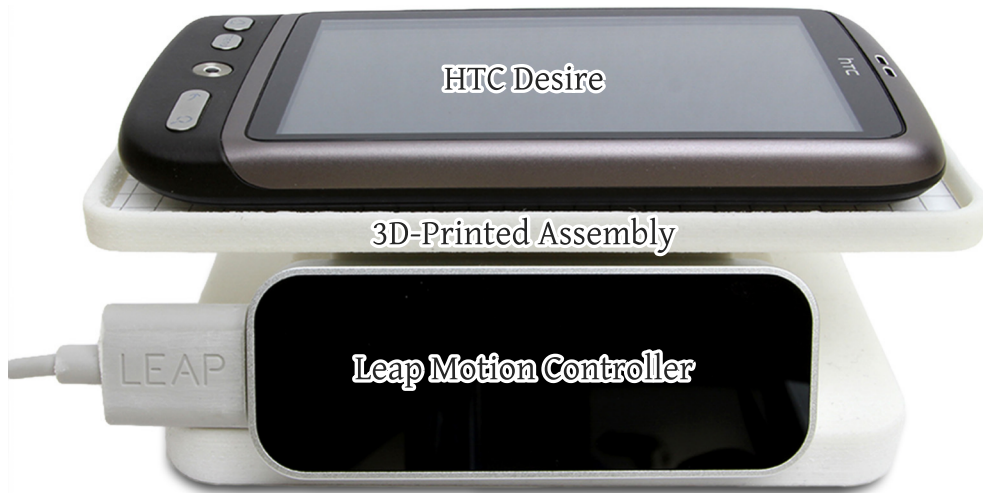


Figure 3.18: For our prototype we attached a Leap Motion controller to an HTC Desire smartphone using a custom 3d-printed mount.

However, while the accuracy of pointing for flat [124] or curved [261] surfaces has been investigated, it is currently less clear how targets for around-mobile pointing would need to be designed. Brown et al. have previously compared in-air techniques against using the mouse in a Fitt's law study and have shown significantly worse performance for in-air techniques [29]. However, their device setup is designed around focused interaction with in-air techniques and does not readily compare against the less engaged scenario we investigate here. For our given scenario, phones lying around, we can, e.g., expect an additional detrimental influence of the environment on the sensing and interaction. Compared to, e.g., an *optitrack* setup, observing the user from the device is, e.g., more susceptible to obstruction. Having the interaction in front of the device instead of above it also constraints user movement due to the table the device is lying on.

3.7.1 Apparatus

While we envision future mobile devices to incorporate depth sensors, enabling them to track hands and fingers, this is currently not generally available. We thus combined an off-the-shelf phone (*HTC Desire*) with a *Leap Motion* sensor⁸ using a 3d-printed mount (shown in Figure 3.18). In this configuration, the sensor is facing the user and is tilted 25° upwards. The sensor itself has a 132° horizontal and 115° vertical field of view and is accurate to up to 0.01 mm. This enabled us to emulate the desired device class. We use the hand tracking capability of the sensor with the cursor located at the center of the hand (selection emulates a “grab”—without closing the hand).

⁸<https://www.leapmotion.com>

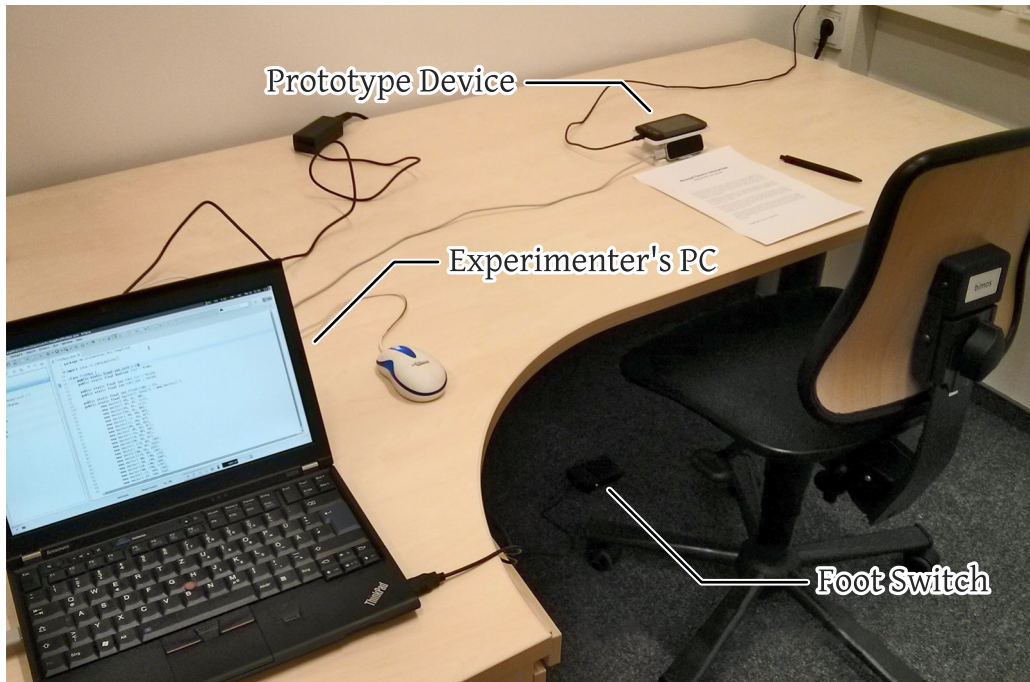


Figure 3.19: Setup used in the around-device pointing study. The prototype and positioning emulate a phone lying around on a desk. Participants committed selections with a foot switch.

Our phone-sensor combination was placed on a table and connected to a controller PC (see Figure 3.19). While the study code was running on this machine, participants could not see the PC screen. Instead, they were shown positional feedback on the phone's screen. As the screen was non-stereoscopic it did not provide further depth cues. The system also included a foot switch for confirmation.

3.7.2 Design

We used a $4 \times 3 \times 3$ factorial design with yaw (-30° , -10° , 10° , 30°), pitch (20° , 35° , 50°), and distance (12 cm, 24 cm, 36 cm) as independent variables (see Figure 3.20). This covers a large volume in front of the prototype device. As the minimum pitch angle was just 20° , some targets in this setup are quite close to the table. The ones at 12 cm distance, e.g., are only 4.1 cm off the table. Dependent variables are error (distance from target) and acquisition time. Target order was randomized and participants went through 10 repetitions per target.

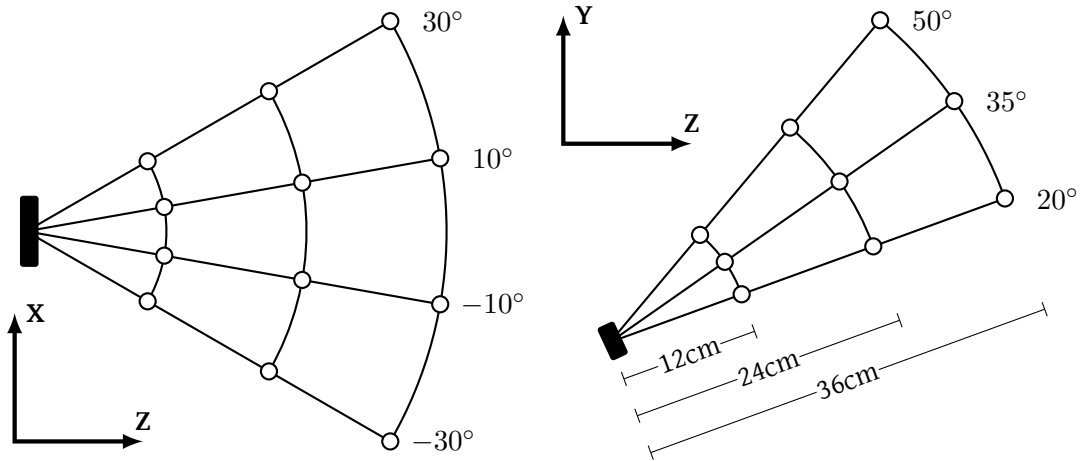


Figure 3.20: We placed targets in the space around the prototype at four different directions (-30° , -10° , 10° , 30°), three different inclinations (20° , 35° , 50°), and three distances (12 cm, 24 cm, 36 cm).

3.7.3 Participants

We recruited 13 participants (3 female, age 22-50 $\bar{x} = 27.5$, $SD = 7.37$) for the study. All but one participant were right-handed. Asked to rate their spatial sense on a 5-point Likert scale, most participants rated themselves highly ($\bar{x} = 4.0$). Also using a 5-point Likert scale, they indicated some experience with smartphones ($\bar{x} = 3.46$), but not much experience with using hand tracking systems ($\bar{x} = 2.15$).

3.7.4 Procedure

Participants sat down at a desk with the device in front of them (facing them as shown in Figure 3.19). Note that the device is placed far enough away that all targets fit in the space between the device and the user, yet sufficiently close that users would still be able to reach the targets only 12 cm away from the device. We gave our participants about 5 minutes to familiarize themselves with the system. During each trial, participants could see the target position and the current tracked hand position on the phone's screen (see Figure 3.21). They were instructed to move their right hand to the target position as accurately and as fast as possible. All targets were located above the desk surface, not in the open space in front of the desk. When reaching the target, participants would commit their position using a foot switch.

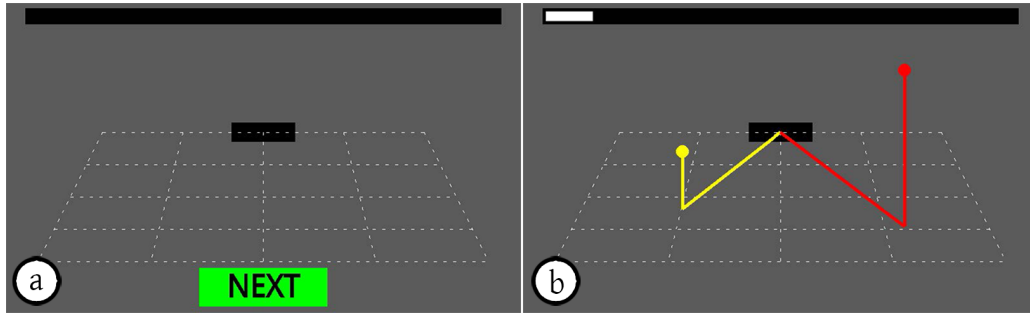


Figure 3.21: Visualization used for the study, showing (a) the screen participants initiated each trial on, and (b) a view during a trial with the hand position (red) and the target position (yellow) being displayed.

Table 3.3: ANOVA results for the around-device pointing study. Significant effects are marked * for $p < 0.05$, ** for $p < 0.01$ and *** for $p < 0.001$.

Effect	d.f.	Log error		Log time	
		F	p	F	p
(Y)aw	3, 36	3.24	*	17.26	***
(P)itch	2, 24	6.14	**	51.19	***
(D)istance	2, 24	10.77	***	63.65	***
Y×P	6, 72	3.33	**	7.43	***
Y×D	6, 72	10.1	***	4.36	***
P×D	4, 48	14.7	***	28.89	***
Y×P×D	12, 144	1.96	*	6.06	***

3.7.5 Results

From the raw data, we removed all erroneous selections (further than 3σ away). During statistical analysis, we log-transform both error and time. A summary of the results is shown in Table 3.3, while a visualization of raw pointing data can be seen in Figure 3.22. In this section we further describe the results with regards to *accuracy* and *time*. All main and interaction effects were analyzed via three-way repeated-measure ANOVA. We used pairwise permutational two-tailed t-tests with 1000 permutations for individual comparisons. P-values were adjusted for multiple comparisons with the Holm-Bonferroni method.

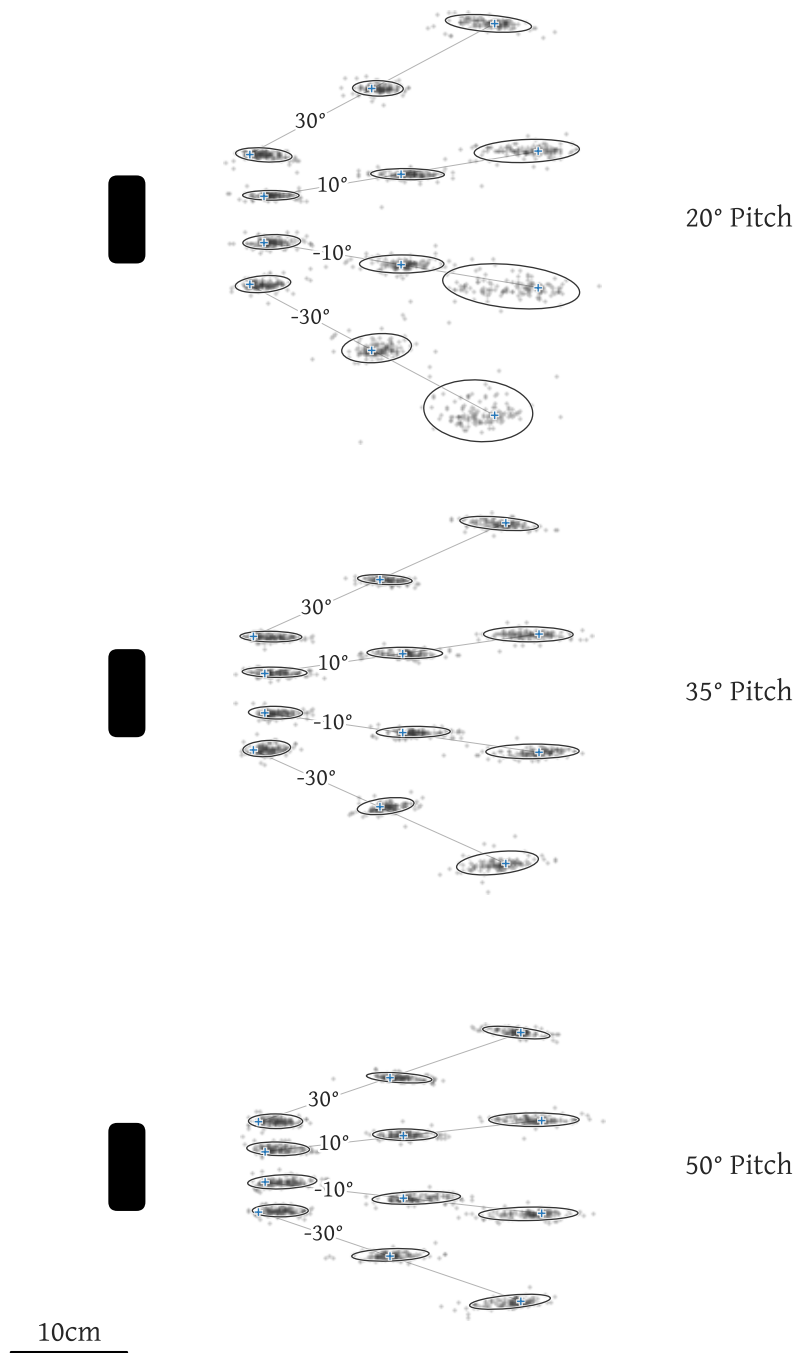


Figure 3.22: Raw selection data for all three pitch slices. View is normal to slice plane with sensor on the left. Ellipses are 95% confidence ellipses. Note that because ellipses are most skewed in the view direction, but this plot shows a view along the slice normal, skewdness cannot be directly compared.

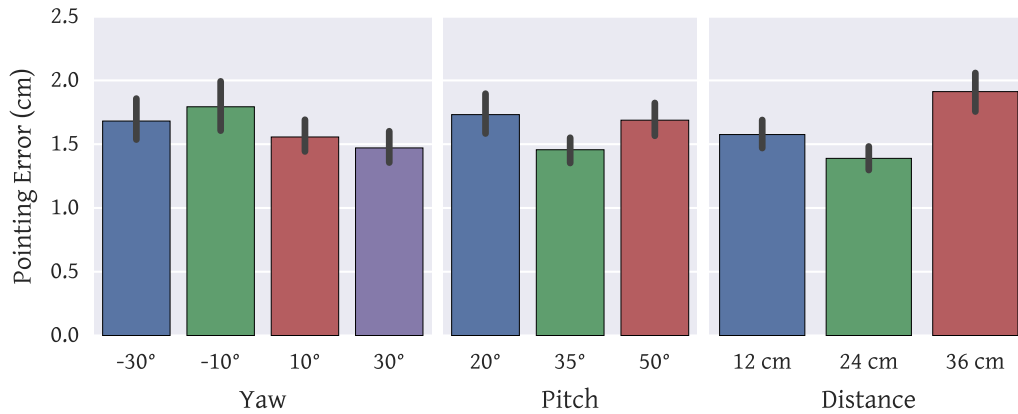


Figure 3.23: Mean pointing error for each factor. Targets further away and more towards the left resulted in larger pointing offsets. Error bars show bootstrapped 95 % confidence intervals.

3.7.5.1 Accuracy

We found a significant effect for each factor and all interactions on accuracy (see Figure 3.23 for a visualization of accuracy for each factor). Especially targets further away from the device resulted in a decrease of accuracy ($p < 0.01$ for targets 36 cm away against targets both 12 cm and 24 cm away). Targets at medium inclination fared significantly better than those lower down and higher up ($p < 0.05$ for 35° targets against 20° and 50° targets). However, there was no significant effect for yaw. Several $P \times D$ interactions were significant though, with targets low and far away or up and close yielding worse results than the others ($p < 0.05$). Some of the effect could be due to way users had to hold their hands for tracking: for lower targets, they would need to tilt it upwards while being close to the table, resulting in a slightly awkward pose. This is necessary because if the fingers were pointing forward, the Leap would only be able to see the fingertips. However, for accurate hand tracking more of the hand needs to be in view—having the palm point towards the device ensures this.

From the accuracy data we also computed minimum button sizes, as in [124, 309]. This size is given by the diameter of the sphere containing 95 % of the made selections. For each target, we fit an ellipsoid to the selections—its longest axis gives the minimum button size (i.e., we fit circular buttons to the data). A sliced view of raw data and 2σ (95 %) ellipses can be seen in Figure 3.22. As can be seen, the distributions for all targets are strongly skewed. We will return to this finding below. Between all targets, the minimum button size varies from 4.1 cm to 11.9 cm, with the average minimum size at 6.5 cm.

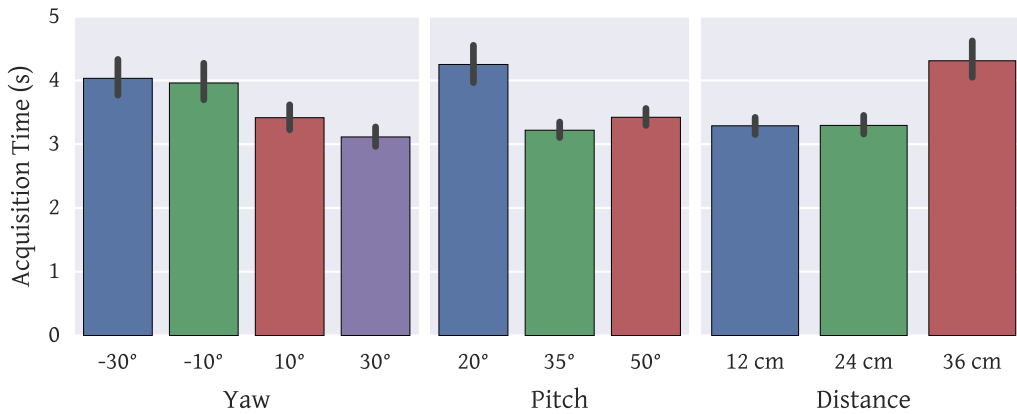


Figure 3.24: Mean acquisition time for each factor. The more to the left, the lower, and the further away a target was, the harder it was to acquire. Error bars show bootstrapped 95 % confidence intervals.

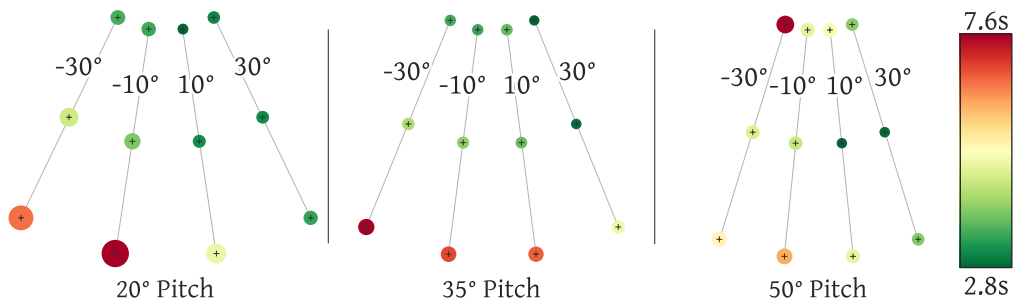


Figure 3.25: Mean acquisition time for every target by pitch with view normal to pitch plane. Circle size and color indicate acquisition time.

3.7.5.2 Acquisition Time

For time, all factors and interactions were significant as well. Distance was again a significant factor, with $p < 0.01$ for 36 cm compared to all other distances (Figure 3.24 shows times per factor). Furthermore, all inclinations were significantly different from each other ($p < 0.05$ for 35° and 50°, $p < 0.01$ for all other combinations). Targets low and far away combined both disadvantages and were significantly worse ($p < 0.05$) than other locations. All yaw directions but -10° vs. -30° were significantly different with targets on the side of the tracked hand being much faster to acquire. A per-target plot of acquisition time can be seen in Figure 3.25.

3.7.6 Discussion

With targets, on average, 6.5 cm large, around-device targeting is less accurate than touch. However, the space available for interaction is larger. The volume spanned by our targets (a subset of the total available sensor frustum), e.g., is 6413 cm³—enough for many targets. In-device tracking is less precise than lab tracking, one would thus expect our target sizes to be closer to realistic sizes than those from, e.g., an *OptiTrack* setup. For the desired usage scenario of our setup, conditions were ideal though: a footswitch for committing and the whole screen dedicated to the visualization. Target sizes are likely to be larger in more constrained setups. If no footswitch were available, a common way for selections is to use dwelling. In that case, the user would need to hold the hand in position for a given period of time. However, holding the hand in place introduces some noise due to tremor into the target estimation.

We find that the setup introduces some constraints that have not been observed in previous work. Most importantly, previous work such as [52] or [177] use optical trackers and place targets in free space. Such a setup inherently enables more accuracy than one with more constrained sensing. Where a stationary tracker can be deployed to ensure proper hand tracking in all poses, in-device solutions are not as agnostic. Here, the angle of the hand to the device can have a strong influence on the quality of the tracking. This needs to be considered when designing around-device systems. Note that our around-device notification system from Section 3.6 had a similar constraint. The used distance sensor works by observing infrared light reflecting from the hand. This works less well if the fingers are pointing towards the sensor and better if the flat hand faces it.

Furthermore, in our scenario the presence of the table proves troublesome for some targets. In contrast to selecting targets in a standing setup (e.g., [177]), the required approach trajectory for some targets is more restricted than for others. When the device is not in a users' hand (i.e., targets are in free space all-around the phone), as will be common in less engaged interactions, limiting obstruction is unavoidable. Resulting accuracy and acquisition time are dependent on the specific environment. While we had no other objects on the table, these would also inhibit sensing and pointing in the around-device space. For example, a bottle placed near a phone would occlude everything behind it from the device's sensors. But even if an object's occlusion is not a problem, it might still inhibit movement of the hand. For example, while an empty glass might be transparent for the sensor, the user could still not acquire targets where it is located. Hence, while the overall space is large, realistically only some of it will be available for interaction. Where previous work often studied those parameters in free space, new work is needed to quantify expected performance in common settings, such as the ones we identified in Section 3.4.

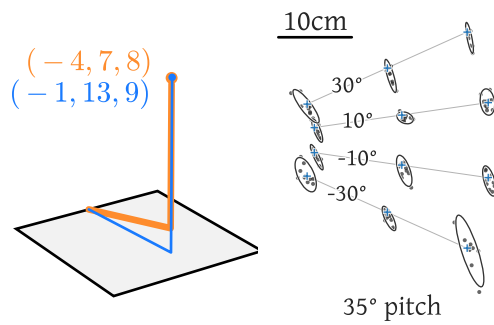
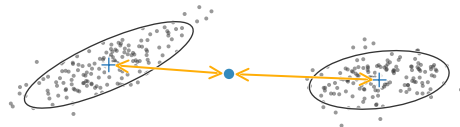


Figure 3.26: Target and cursor can overlap in the visualization, making it harder to judge depth and explaining the skewed distributions. Rotating the visualization 90° indeed results in the distribution also rotating correspondingly (data shown from 2 participants in explorative study).

Euclidean distance: 4.01

Euclidean distance: 5.00



Mahalanobis distance: 5.21

Mahalanobis distance: 4.55

Figure 3.27: Using Mahalanobis distance can account for the skewed target distributions when determining which target was closest to an input location.

3.7.7 The Influence of the Visualization

The distribution of selections is skewed (as shown in Figure 3.22). We hypothesized this might be due to the visualization: when target and cursor are view-aligned, it is hard to judge distance along the view direction, which in our setup equals the z-direction (see Figure 3.26). While the visualization's projection on the ground plane did show that information, participants might now have been able to make sense of this.

To validate this hypothesis, we tested two additional participants with a setup equivalent to the first study but with the visualization turned 90° around the y-axis. Instead of seeing targets and cursor from their perspective, participants would see a view from the left. Distributions of their pointing data show that the main axis does change correspondingly (35° pitch slice shown in Figure 3.26).

As this skewedness is a systematic effect of the system, it is possible to correct for it when disambiguating selections. For this, we compute covariance matrices per target (over all users). When determining the closest target, these matrices can be used to substitute the usual Euclidean distance measure with the Mahalanobis one (see Figure 3.27).

3.7.8 Summary

In this section, we investigated pointing accuracy for future mobiles with embedded around-device sensing. With size constraints in mobile devices, using the space around them offers a way for more and coarser interaction. The device itself can still be involved in the interaction, e.g., by showing feedback to help users acquire around-device targets. In contrast to non-visual imaginary interfaces [97], which rely on proprioception, this allows pointing in a tight control loop—allowing for smaller target sizes.

Tracking needs to be device- or user-centric, as instrumenting all places one would desire around-device interactions is not feasible. In-device tracking, however, differs from external trackers, e.g., deteriorating in quality when orientation towards the device is non-optimal. We know that the scenario we investigated, a mobile device sitting on a table, is the most common one. On average, 6.5 cm targets can be used here with some locations also feasible for 4.1 cm ones. Targets further off the table, towards the center or the dominant side of the user, and closer to the device are preferable.

We also found that the visualization has a strong influence on the selection distribution (with much less accuracy along the view axis), an effect, e.g., also observed by [93]. In the future, it would be interesting to investigate whether mobiles with stereoscopic screens or a fish tank VR visualization with head tracking could further improve minimum button sizes. It would also be worthwhile to combine the around-device pointing capabilities of this prototype with the edge-display capabilities of the notification system discussed earlier in Section 3.6. In that way more complex interaction than just distance to the device could be used to interact with notifications. For example, specific locations could serve as shortcuts to certain types of notifications, while space further away from the device could be reserved for coarser gesturing. Knowing where to best put shortcuts is important for informing such designs.

3.8 Chapter Summary

In this chapter we have taken a closer look at moving interaction away from devices. This alleviates the need for user to pick them up, possibly unlock them, and engage in focused interaction on the touchscreen. The around-device devices concept we discussed is one possibility for identifying options off the phone. We have shown how such devices have beneficial properties with respect to (1) affordance, (2) casual interactions, and (3) interaction space. Furthermore, we conducted a ground truth investigation into what objects would be available for the kind of interaction presented here. We hence gained a quantitative insight into what objects designers could assume to be most likely available. Furthermore, we conducted a study to determine which of those objects users would actually use for a range of tasks and three different settings.

Around-device devices enable users to imbue objects around them with rich interaction capabilities. With many objects sufficiently described by geometric primitives, this also allows for interaction in settings with light occlusion. Mobile phones, our everyday companion device, can be the sensing center for those interactions. This ensures a sense of control and personalization hard to achieve with instrumented rooms.

But physical objects are not always necessary and we have also explored an example of a notification access system that only works based on the user's distance to the device. It enables coarse access and information further away from the device, but allows users to "drill into" the available information if they see such a need. This raised the general questions of how accurate this around-device space actually is. We hence also took a look at the accuracy and speed for input around a mobile device laying around in the way envisioned in the notification prototype and validated in the ground truth investigation.

4 Casual Interaction in the Smart Home

As technology advances, it reverses the characteristics of every situation again and again. The age of automation is going to be the age of “do it yourself.”

— Marshall McLuhan, *Essential McLuhan*

In Mark Weiser’s vision of ubiquitous computing, “*future homes and offices will contain hundreds of [...] tiny computers*” [310]. We are not quite there yet, but the average modern home today already contains more computing devices than the average home in 1991, when Weiser made that statement. Currently, music systems, lighting, appliances, or climate control are all available in computerized versions and are becoming more readily available to consumers. In this context, things are commonly described as *connected*, or even *smart* (e.g., *smart cities*, *smart homes*, or *smart objects*). In fact, the smart prefix is so commonly used that hundreds of things exist in supposedly smart variants (see, e.g., Wikipedia’s long list of pages on smart things¹). Via the so-called *Internet of Things*, all the smart devices in the home and elsewhere are envisioned to talk with each other and connect into larger device ensembles. For example, changing the music might also influence the ambient lighting, or a central server can control both the fridge as well as the air conditioning.

If the smart home of the (possibly near) future indeed is connected to such an extent, this raises the question of how users control such a device ensemble. Some envision that artificial intelligence systems control such homes based on predictions of the inhabitants’ needs [62]. On the other end of the spectrum, each device could be individually controlled by the user. However, while the former takes much control away from the user, the later does not scale to the envisioned large number of devices. A critical question for the future smart home will thus be how to find a middle ground between complete automation and cognitive overload. In this chapter, we explore how casual interaction can support scenarios common to smart homes and fill out this middle ground. We start with a closer look at the smart home design space. We then discuss three individual projects that more closely investigate interaction with lighting, authenticating in the home, and augmenting furniture for casual interactions.

¹<https://en.wikipedia.org/wiki/Special:PrefixIndex/smart>

This chapter is based on a short paper and an extended abstract², both published at *UbiComp 2015*, as well as a supervised thesis³:

- Henning Pohl, Markus Krause, and Michael Rohs. “One-Button Recognizer: Exploiting Button Pressing Behavior for User Differentiation.” In: *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing - UbiComp '15*. New York, New York, USA: ACM Press, 2015, pp. 403–407. DOI: 10.1145/2750858.2804270
- Henning Pohl, Markus Hettig, Oliver Karras, Hatice Öztürk, and Michael Rohs. “CapCouch: Home Control with a Posture-Sensing Couch.” In: *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2015 ACM International Symposium on Wearable Computers - UbiComp '15*. New York, New York, USA: ACM Press, 2015, pp. 229–232. DOI: 10.1145/2800835.2800932

Statement of Original Work

Work on the *One-Button Recognizer* paper was done in collaboration with Markus Krause. While I focused more on the physical prototypes and the corresponding embedded code, Markus contributed most of the machine learning and evaluation work.

4.1 Introduction

Compared to interaction in the wider world, interaction in the home is more constrained. People are intimately familiar with their home and might even follow some daily routine. For example, a user might always get up at the same time each workday and start the day by taking a shower. But even without a routine, users do not need to expend the same amount of effort in navigating and interacting within their house as outside of it. This characteristic of being a somewhat controlled environment has made the house an attractive target for automation.

Casual interaction is one approach to this problem. Leveraging the full spectrum of engagement, users could yield control to their home as they see fit, yet take back control when they feel the need to. This user-driven choice of control is currently lacking from smart homes where often things either happen automatically or require quite a bit of effort. For example, a commonly automated aspect is light control, where lighting is switched on as somebody enters a room (via incidental interaction).

²On a project supervised in our winter 2013 *Aktuelle Themen der Mensch-Computer-Interaktion* class.

³The master thesis of Karoline Busse [31].

In this chapter, we discuss three projects around smart homes. The first one focuses on user recognition in the home (or office), something that commonly requires extensive instrumentation or user effort. For example, users might use a keycard or an RFID token to identify themselves. We designed a system that integrates user identification directly with user actions—specifically, pressing a button. In this way, light switches can identify who just walked into a room as the light is turned on. While this does not suffice for high security scenarios, it is adequate for many home or office scenarios.

Apart from this larger project, we explore two other, more concise, projects as well. One concerns augmenting furniture to sense users' posture and use this to control a room. Systems like this can allow users to affect their environment in more subtle ways. Consider cuddling on the couch and swiping on a specific part of the couch to slightly dim the lights—taking out a remote to change light settings might have unacceptably disrupted the situation. We stay with light control in the last project where we explore different levels of light control integrated into a wearable device.

4.2 Related Work

Apart from the general concept of smart homes, we investigate related work from two scenarios more closely: (1) low-effort user recognition, and (2) sensing furniture. Our third scenario, lighting control with a wearable, is discussed later in Section 4.5.

4.2.1 Smart Homes

While we explore more specific areas of smart home research below, the general concept itself has interesting connections to casual interaction. Two papers relate particularly well to casual interaction. Koskela et al.'s work, e.g., investigates how computers, phones, and TVs are used differently as means for home control [157]. Via focus groups, they find that users are not interested in full automation, but desire to retain some level of control. This highlights the need for different control mechanisms to serve this desire. They conducted a six month long study that had a couple live in a smart home equipped with a provided range of custom controllers. Here, the phone emerged as the controller of choice (mostly due to the increased convenience). But they also identify two overarching interaction paradigms for the smart home: (1) *pattern control*, where predetermined and planned sequences automate recurring everyday routines (e.g., dimming the lights at bedtime), and (2) *instant control*, where users desire to change things “right now”. While interfaces on a computer or phone are well suited for the former case, the latter is much better served by the phone (as users have it with them when they encounter such situations). These two paradigms correspond well to casual interaction principles, where an automation model exerts control as long as users do not desire otherwise. When they do, then more precise control needs to be available to them.

Rashidi and Cook studied how to identify patterns for potential automation [253]. However, they also differentiate between different levels of user involvement in pattern detection and modification. They identify four different levels of engagement, each different in how much control users have over the system behavior. For precise control, users can edit patterns in a graphical user interface via direct manipulation. If they do not desire to fine tune patterns, they can also just rate patterns identified elsewhere. This is a task requiring less effort while still giving them some say in how the systems behaves. More subtly, users can mark patterns for closer observation. The system will then check whether there were any changes in user behavior that require the rule to be updated. For example, if a user previously always went to bed at 22:00, but changed her schedule and now goes to bed at 24:00, this functionality can be used to force a fast update of the changed pattern. The method with least control by the users is based on datamining their behavior. By constantly observing the users and trying to detect patterns or pattern changes, the system constantly tries to adapt. However, this is a slower process and a bedtime change as above will take longer to be noticed with certainty than if an update is manually triggered. As with casual interaction, their users can thus pick how much influence on the system they want. Rashidi and Cook note that: “*Each method requires a certain balance between user collaboration and system autonomy, from no user involvement in the smart detection method to no autonomy in the direct-manipulation method.*” [253]

In addition to control issues, Dey and Newberger see a second challenge for users of context-aware systems (such as in the smart home) in *intelligibility* [67]. Users should hence be able to understand how such an application is working, e.g., to deal with unexpected system behavior. One example they provide is the smart home that left visitors in the dark as they did not align with learned behavior. Clearly, we cannot just built systems that only allow users to intervene and take control—there would also need to be a way for users to get feedback on the system and the actions available to them. However, in this dissertation we focus on the control aspects of smart home interaction. Ultimately, systems would also need to be sufficiently intelligible to users.

4.2.2 Low-effort User Recognition

There have been a number of approaches to low-cost user recognition. A focus of such work is often to find a method that integrates with the interaction. The way a touch gesture is performed, e.g., can be used to distinguish different users [65]. With more complex setups, it is possible to track fingerprints throughout touch interaction [123], allowing to identify users without any extra step. At any point in time, touch location and user identification go hand in hand. Similarly, tracking hand contours instead of fingerprints also allows for identification of users [268]. However, fingerprints and other biometric data can be privacy sensitive and users might have objections to providing it for a task such as getting a coffee.

An often explored biometric feature for identifying users is typing patterns [160, 173, 201, 202, 217, 226]. However, identification via typing is not applicable to short, transient interactions, where no text entry is required. This limits their usefulness in a smart home context, compared to, e.g., the office.

One approach would be to add cameras to the home and use computer vision methods to identify users. Face recognition is a common way to do this, but shoes [255], backs of hands [251], or in-air gestures [5] also can be used. Any method using vision relies on proper lighting though, restricting possible scenarios. Furthermore, mounting cameras inside the home raises questions of privacy.

If a home were outfitted with a pressure-sensitive floor, tracking walking behavior would be a way to identify users [290]. Attaching sensors to door knobs or other objects a user touches, allows distinguishing them based on their specific body capacitance [108]. Final systems need not necessarily pick only one of the above mentioned low-effort recognition methods. Instead, it is possible to combine many sensors for user identification [274]. If applied in a smart home context, every interaction and movement in the space would then add to the overall system estimation of who the user is. Lower-effort but higher uncertainty methods could also be mixed with higher-effort but more precise ones. High effort could then be mostly forgone, but with a fallback for more critical situations.

4.2.3 Sensing Furniture

One way to augment furniture with sensing is to add pressure sensors. A recent example is Papanikolaou et al.'s work where they embedded pressure sensors into custom designed chairs in order to distinguish five seating postures [222]. Similarly, Mutlu et al. added pressure sensors to a standard office chair and recognized ten different postures [207]. They achieved 78% accuracy using the logistic regression classifier from WEKA [98]. Another chair was prototyped by Forlizzi et al., who focus on supporting senior users [81]. Instead of adding active sensing to furniture, it can also be designed to relay pressure information to be picked up by a pressure-sensitive floor [25].

Another common method for augmenting furniture is adding capacitive sensors. An early exploration of this was conducted by Zimmerman et al. [336]. *CapTable* and *CapShelf* are later examples, where capacitive sensors are used to track gestures around and things on the augmented objects [320]. Instead, Große-Puppenthal et al. used capacitive proximity sensors in a couch [92]. They compare different classifiers for nine postures and achieved best results with radial basis function networks. An overview of several different pieces of furniture with capacitive sensing and corresponding applications is provided by Braun et al. [26]. Compared to existing work, we focus less on the sensors but more on how such furniture fits into casual interaction scenarios in the smart home.



Figure 4.1: Pushing the button next to the coffee machine, William is recognized and his coffee tally is updated. No extra interaction is needed to inform the system *who* is pressing the button. Recognition and action are integrated.

4.3 Low-Effort User Recognition with One Button

Differentiating who is using a device among a group of cooperating users is a frequent task in ubiquitous computing systems. We might, e.g., be interested in who just entered a room, who picked up a phone, or who is activating the dish washer. In such scenarios, security might be less important than convenience, i.e., minimizing the burden on users. Such convenience can be achieved by using biometrics or instrumenting users (e.g., RFID tags). However, this comes at a cost of privacy or can be impractical or out of place in some contexts (e.g., at home).

We propose differentiating users by how they press a physical button as a means for low-effort authentication. Because an explicit action is required, this avoids problems of accidental activation as in more automated systems. Furthermore, this is a common action—if every button in an environment has the capability to recognize who pressed it, a lot of context information is generated. Pressing a button does not require a lot of focus and can be done while engaged elsewhere. It thus builds on gross motor skills, inexact, and inattentive interaction [69, 126].

User recognition via button pressing is particularly suitable in one-off interactions, where users are not engaged for long. Sometimes the whole interaction can even be just that one button press (e.g., switching on the lights or calling an elevator). One such scenario, that we take as an example here, is tallying coffee consumption in an office (see Figure 4.1 for an illustration). Often this is done using a paper tally sheet where users mark every coffee. However, this requires tedious manual balancing at some point and requires cooperating users, as tally marks can easily be manipulated. We propose changing the task from making a mark on a paper sheet to pressing a button next to the coffee machine—and eventually replacing the button on the machine with a button that recognizes users.

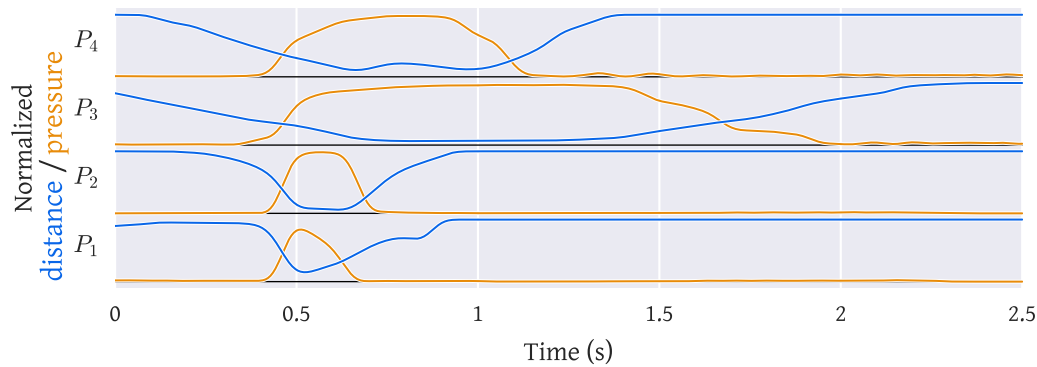


Figure 4.2: Comparing the normalized distance and pressure curves for 4 participants. The variation in button pressing behavior is apparent.

We built two prototypes of buttons that can recognize who pressed them: (1) a portable push button that integrates a pressure and a distance sensor, and (2) a wall-mounted version that extends a standard light-switch with a distance sensor. These sensors are low cost, compact, and robust to changing conditions. We did not use computer vision methods to make the approach resilient against lighting changes. This choice of sensors also allows for lightweight and transient interactions. Prolonged contact of the user with the interface is not required. The two prototypes are similar to the button designed by Spelmezan et al. [278], whose focus is on recognizing explicit gestures, rather than implicitly recognizing users.

In two evaluations, we investigate (1) recognition rates for different group sizes from a large set of 39 users, and (2) in-use system behavior over a 4 week period of time with a fixed set of 5 users. In our long term deployment, we observed fast improvement over time with over 94 % recognition rate in the last week. Users can correct mis-classifications with another button press—a low cost disambiguation approach. This extra action boosts recognition rates to 99 %.

4.3.1 One-Button Recognizer

Our recognizer builds on the observation that button-pressing behavior differ slightly between users. This was previously illustrated by Kim et al. [151], who used such pressure profiles to recreate haptic sensations for augmented reality buttons, in order to recreate the tactile feedback from depressing a physical button. Instead of recreating the button pressing sensation, we set out to use the nuances in button pressing for differentiating users. As shown in Figure 4.2, the sensor readings of a button press vary distinctively between different users.

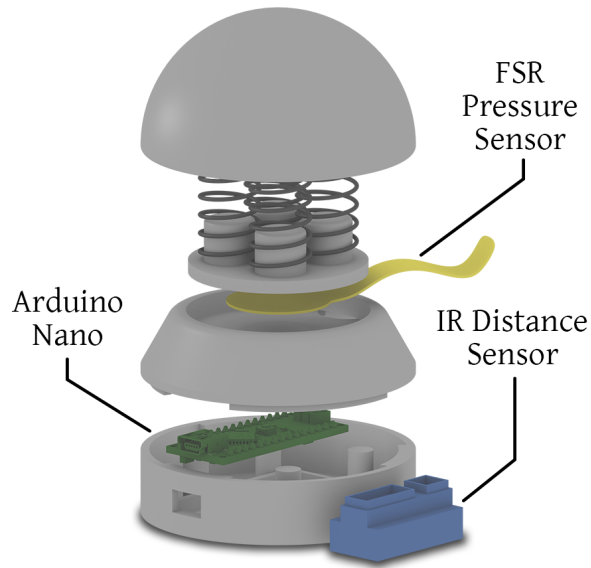


Figure 4.3: Our pressure-sensitive button contains an FSR sensor in a 3D-printed enclosure. Springs provide return force and push resistance. An IR distance sensor is attached at the side.

4.3.2 Button Prototypes

We designed two different button prototypes for two distinct form factors: (1) desk, and (2) wall-mounted. Both buttons include a *Sharp GP2Y0A41SK0F* infrared distance sensor (4–30 cm operating range) to detect how a hand approaches the button. The sensor works reliably over a wide range of lighting conditions. However, due to the large size of this sensor it cannot be integrated directly into either button. Instead, it sits just underneath both of them. Future designs could instead opt for smaller sensors, e.g., by making use of miniature time-of-flight sensors (such as the *VL6180X* by *STMicroelectronics*) that have become generally available recently. Both prototypes take pressure (where applicable) and distance readings via the 10 bit ADC of an *Arduino Nano*. Data is sampled at about 400 Hz and, via a serial connection, relayed to a host computer for further processing.

4.3.2.1 Desk Button

The desk button is 3D-printed in multiple parts (shown in Figure 4.3). Four springs separate the top of the button from the base to provide return force and give the button a proper feel. Force sensing is performed with an *Interlink FSR402* pressure sensor (sensitive in the 0.2–20 N range), attached below the spring contact plate. A small protrusion on the bottom of the contact plate focuses the force on the pressure sensor. The sensor gives us a pressure profile over the whole duration of a button press. However, we threshold the pressure values to derive a binary switch state. In this prototype the distance sensor is placed in front of the button.

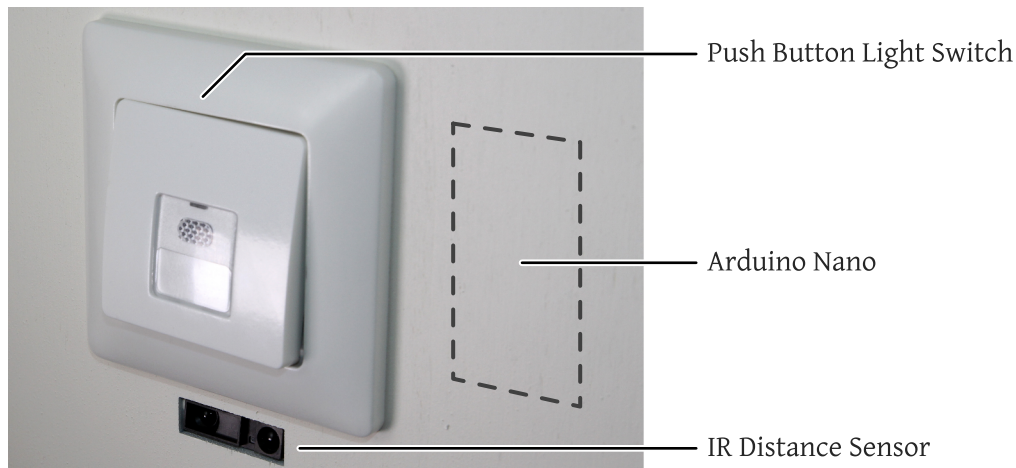


Figure 4.4: We augmented a light switch with a distance sensor to enable identification of who pressed it. Future switches could embed the sensor directly in the enclosure.

4.3.2.2 Wall-Mounted Button

For the wall-mounted prototype, we modified a *Voltomat MIKRO* push-button light switch. This is a standard product from a hardware store and not further modified by us. Hence, this prototype has strong ecological validity for the intended smart home setting. We mounted this button on a wall and embedded the distance sensor underneath (see Figure 4.4). Future versions could integrate the distance sensor inside the button itself. The center of the push-button is already hollow and designed to hold a small indicator light. This space could hence be repurposed to hold a custom distance sensor. Due to space constraints, this button did not integrate a pressure sensor underneath. Instead, the switch itself is directly connected to the Arduino which reads and relays the binary switch state.

4.3.3 Feature Extraction

We derive our features from six points along the trajectory of a button press (see Figure 4.5). Points t_1 and t_4 mark the beginning and end of the actual button press. In the wall button this is just the switch state. For the desk button, however, we threshold the continuous pressure values at 80 % of the maximum to derive binary button state.

Starting from those two points, we search for the start and end of the attack (t_0 and t_2) and release phase (t_3 and t_5) respectively. Those points are derived from the local extrema of the distance sensor signal around the start/end of a button press. Note that those points can be on either side of the button hit event, depending on the way the button is pressed.

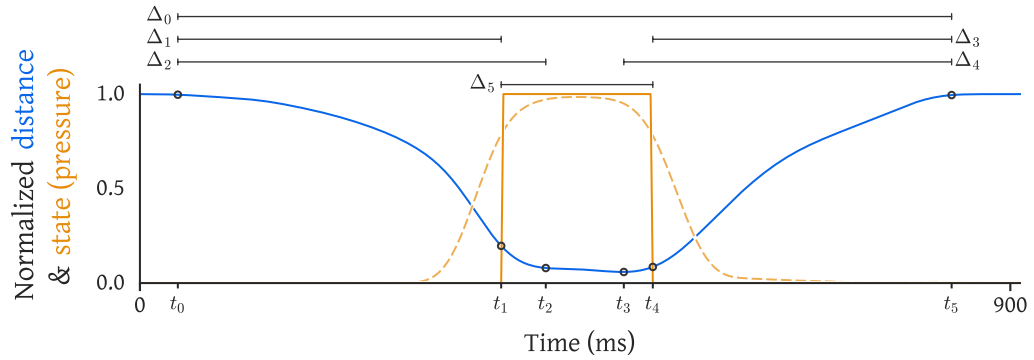


Figure 4.5: We build features from six times along a button press trajectory: attack start t_0 , press start t_1 , attack end t_2 , release start t_3 , press end t_4 , and release end t_5 . Features are then defined as time differences for six distinct phases of the button press. For the desk button, we threshold the pressure values (shown dashed) to compute button state.

From those points we derive six features. The overall interaction time Δ_0 and button down time Δ_5 capture measures of how fast the user was pressing the button. We also take into account two different measures of attack and release timing. For attack timing, we include the time it takes from the start of the approach to both (1) the button pressing Δ_1 , and (2) the end of the approach Δ_2 . Similarly, during release we use overall withdrawal time Δ_4 and the time from the release of the button Δ_3 . We use those features to discriminate two different button pressing behaviors: (1) Some users first press the button, then lower their palm. In this case Δ_1 is longer than Δ_2 . (2) Other users attack the button with their palm already lowered. In this case Δ_2 is longer than Δ_1 . This behavior holds inversely during the release phase.

4.3.4 Training the Recognizer

For our recognizer we use a random forest classifier. Random forest classifiers fit well to our problem, as they can directly handle multiple classes, and are less sensitive to outliers [49]. In our evaluation, we only used a fixed number of training samples per user. Our classifier generated 10 random trees per forest using Gini impurity [27] as split criterion. An alternative to our manual feature selection and the random forest classifier would be to use a convolutional neural network (which integrates feature selection into the training process and thus would not require manual definition of points of interest along the pressure or distance trajectories). Such a classifier has been shown to work well for time series data [333].

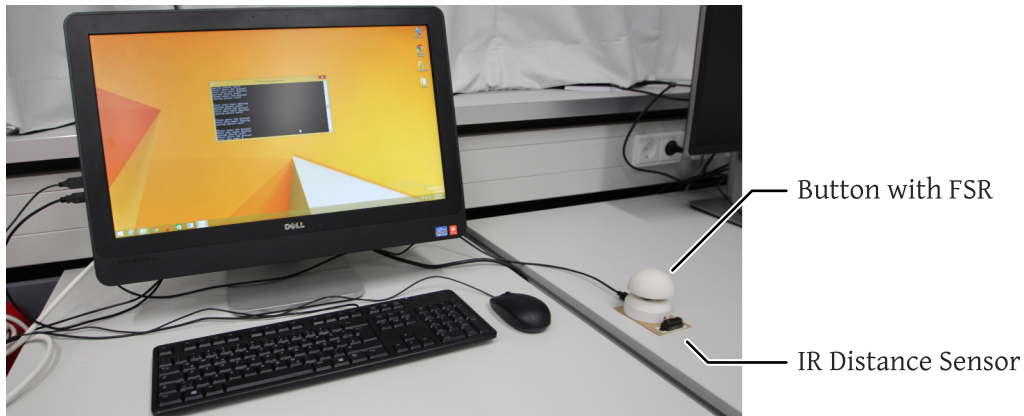


Figure 4.6: During the evaluation we set up the sensor-equipped button in our lab and temporarily also in a classroom. Over the duration of two weeks, participants would enter their name and then press the button when walking by it in the lab.

4.3.5 Evaluating Recognition Performance

In a first study, we investigate how well we can distinguish users in groups of different size and how many button presses would be needed for training. For this, we set up our desk prototype for two weeks in our lab (for use by staff and students, see Figure 4.6) and in a classroom (for use of students in a course). Participants each time entered their name on the connected laptop and then pressed the button. We did not instruct participants in any way as to how to press the button. Overall, we gathered data from 44 participants (7 female) in this way. However, in our analysis we remove all users who pressed the button less than 30 times (leaving us with 39 users).

We investigate the performance for group sizes of 2 to 10 users and with 2 to 30 button presses used as training data (at intervals of 2). For every combination of group size and button presses, we generate 1000 permutations from our data for subsequent testing. We do this to test the robustness of our classifier against group effects, such as subsets of users with similar button pressing behavior.

Accuracy testing for each permutation is done with leave-one-out sampling [145]. One button press sample is used as evaluation data and the other samples as training data. This yields binary results, which we aggregate to a mean accuracy score per permutation. Testing one sample against a classifier trained on all remaining data fits well with our intended use case: classifying one new button press at a time. For each combination, we aggregate these accuracies and calculate the 95% confidence interval of the mean via bootstrapping with 10000 iterations.

Table 4.1: The expected number of required presses and error rate δ go up with group size. It is unlikely to require more than 2 presses.

Group size	2	3	4	5	6	7	8
E [# of presses]	1.04	1.08	1.11	1.13	1.18	1.20	1.23
E [δ (1st press)]	0.04	0.08	0.13	0.14	0.15	0.17	0.19
E [δ (2nd press)]	-	0.00	0.01	0.02	0.02	0.03	0.04
E [δ (3rd press)]	-	-	0.00	0.00	0.00	0.01	0.01

4.3.5.1 Results

How well we can identify users largely depends on the size of the user set. In Figure 4.7—showing recognition rate for all tested combinations—this can be seen in the vertical orientation of the contour lines once a minimum amount of training data is used. When using 24 button presses for training, e.g., accuracy drops almost linearly from 95.0% (two users) to 78.2% (10 users). Overall, accuracy is less sensitive to the amount of training data. For groups of 5 users, e.g., recognition rates exhibit a quick rise from 69.4% for 2 samples per user to 82.3% for 10 samples. This is followed by slight improvement with 30 samples giving 90.3% recognition rate.

We saw high variance in accuracy when using small numbers of training samples. For example, using only 2 button presses is quite unstable and we observed $\pm 16\%$ standard deviation of bootstrapped recognition rates. The stability of the classifier goes up for larger training sets, e.g., being as low as $\approx \pm 3\%$ for groups of ten users when using more than 10 samples.

From this data we can compute the expected number of button presses during an interaction needed to reliably identify a user. Table 4.1 shows this number for different group sizes. For groups of 5 users with 30 samples each, a user is recognized at the first button press with 90.3% probability and at the second button press with 99.1% probability. After the button has been pressed three times, the expected probability of recognition is over 99.9%. Given the very low effort to correct a mistake (average time of a button press is $M = 0.4$ s, $SD = 0.1$ s), a possible additional button press takes but a mere instant.

4.3.6 Evaluating Prolonged Usage

In a second study we set out to evaluate an actual deployment of the button. We installed the wall-mounted prototype in our lab (hung in an office as shown in Figure 4.4) and provided it the names for a group of five users from our lab. Over a period of 4 weeks, those users would pass by to press the button every time they got a coffee.

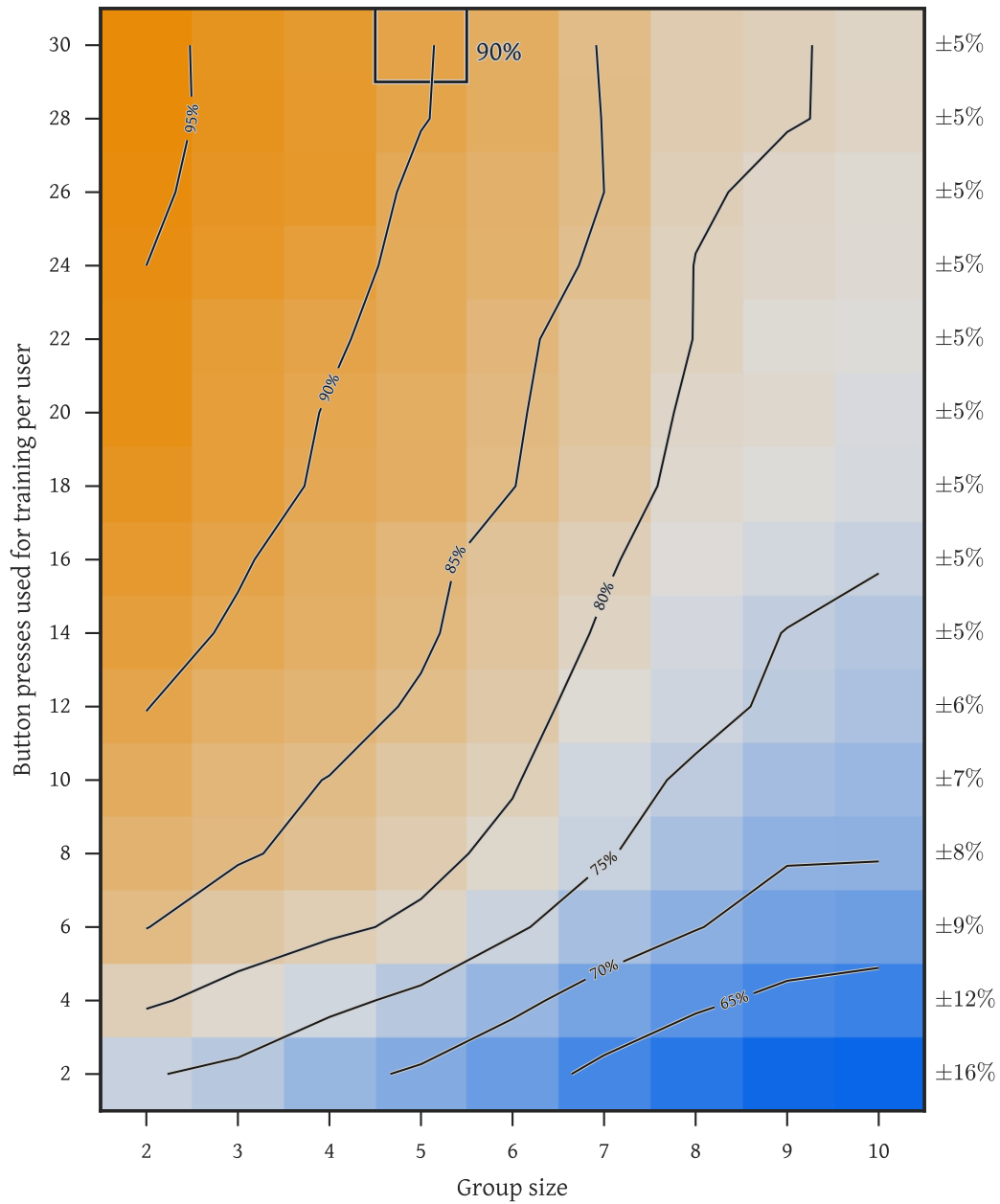


Figure 4.7: Recognition rates for group sizes from 2 to 10 users and button press data from 2 to 30 presses. Recognition rates were computed using 10000 bootstrap samples. On the right we show the average standard deviation for each row—training a recognizer with low numbers of button presses results in high variance, which goes down with more training data. Group size does not influence variability as much.

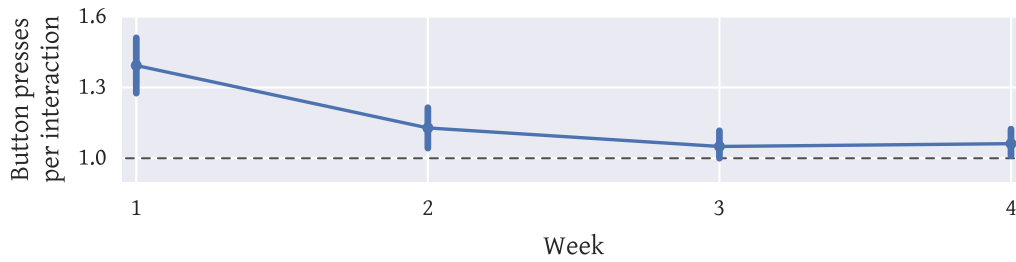


Figure 4.8: Over the course of 4 weeks, the number of times users had to press the button for one interaction (for disambiguation) goes down and approaches the theoretical minimum of 1. Error bars show 95 % confidence intervals.

While users in the first study did provide their name before a button press, this step was left out in this setup. Instead of providing ground truth before an interaction, we introduced a disambiguation mechanism in the second study to allow users a way to correct a wrong recognition. After a user pressed the button, audio feedback is provided via a text-to-speech engine (“Thanks <Name>”). If a user is not properly recognized, she can press the button again within the next 3 seconds to override that last recognition. Thus ground truth is provided by the final *user-accepted recognition*. We allowed cancellation by pressing the button down for at least 3 seconds, however this was not used at all during the deployment.

At first the button is untrained, i.e., has no recorded samples for any user. Thus, initially, performance is poor while the system is collecting samples (and users have to make use of the disambiguation mechanism by pressing the button again). We limit the number of training samples to 40 per user. This provides sufficient data for recognition but also allows the system to adapt to changing button pressing behavior.

4.3.6.1 Results

Overall, we logged 372 interactions with the button. We can see in Figure 4.8 that the number of required button presses goes down over time. During the first week, the system is still learning and users are getting used to it. However, even in that first week, users on average only need to disambiguate every third button press. Over the following weeks, the recognition rate improves and approaches the theoretical minimum of 1.

Overall, we were very pleased with the performance of the button in our lab setting. In the beginning the system was still adapting though, but quickly reached acceptable recognition levels. A confusion matrix for the last two weeks is shown in Table 4.2. With low instances of required disambiguation, using the button was mostly as easy as walking by, pressing it, and walking away.

Table 4.2: Confusion matrix for last two weeks of the longitudinal deployment test. While one user was always recognized correctly, the worst performing user was still recognized correctly 88.9 % of the time.

Actual \ Predicted	User 1	User 2	User 3	User 4	User 5
User 1	95.0%	0.0%	0.0%	5.0%	0.0%
User 2	0.0%	100.0%	0.0%	0.0%	0.0%
User 3	0.0%	7.1%	92.9%	0.0%	0.0%
User 4	6.2%	0.0%	3.1%	90.6%	0.0%
User 5	2.8%	0.0%	0.0%	8.3%	88.9%

4.3.7 Summary

We have investigated a low-effort user recognition approach that piggybacks on button presses. It links identity information to button presses and thus integrates recognition and action. The employed sensors and recognition system are simple enough to be integrated in commodity buttons, essentially disappearing into the environment [311]. Buttons enhanced in this way can serve sources of context information (as in [223]). The evaluations show that the prototypes are reliable enough to differentiate between users in small groups, such as in offices or at home.

The one-button recognizer concept addresses small cooperating groups of people who share a resource. Labs, e.g., often have shared computers for devices such as 3D printers, which could boot and load user profiles automatically when the power button is pressed. Similarly, families' home entertainment systems could switch profiles and play one's favorite music with a single button press.

Large buttons, like conventional light switches, enable quick, inexact, and inattentive action. They do not require fine motor skills. Our approach maintains these properties. However, it also gives users the opportunity to evolve their behavior. Users may deliberately modify their button-pressing behavior to communicate additional information or to become more distinct from others and thus easier to differentiate. Users could even “teach” their button-pressing behavior to others (akin to a secret handshake), allowing trusted users to impersonate them.

While we looked specifically at buttons, the same principle potentially also applies to other objects in the home. For example, users manipulating door handles, window handles, door hinges, or cabinets, might be differentiable as well. Each of those objects is reached for and then moved along one dimension of freedom. As with buttons that are pressed in different ways, users might also turn a door handle with different force profiles. User identification for actions on these objects thus enables a wide range of application possibilities.

Overall, the one-button recognizer shows that authentication does not need to be an involved affair. The home context already restricts the number of potential users and a system like the one-button recognizer can integrate into objects or walls for integrated user identification. This does not preclude users from moving to a more secure and complicated authentication method if required or desired to do so. However, the one-button recognizer enables a casual way to identify that allows users not to expend the effort if possible.

4.4 Augmenting Furniture for Casual Interaction

Control of devices like air-conditioning units, lighting, or media playback is a crucial aspect of smart homes. In current systems this often requires users to start and use an app on their phone. This can be cumbersome and puts demands on a user's attention or engagement, which they might not be willing or able to fulfill. An increasing number of "internet of things" devices in the home would exacerbate that problem—how many different apps or how complex an app would be acceptable? More casual interaction can be a way to decrease this demand, at a slight cost of control.

Here, we look at using augmented furniture to drive interactions in the living room. More specifically, capacitive sensors on a couch track posture changes, which are translated to interactions with a smart living room. By moving interaction off the phone and into the environment less engagement is required from users. Users can then, e.g., switch their TV to a fireplace scene when assuming a relaxed posture.

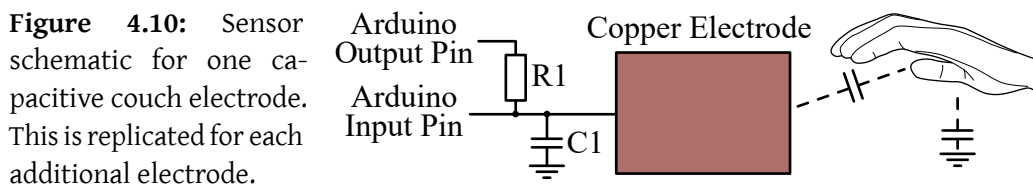
4.4.1 A Prototype System for the Smart Living Room

We built our living room prototype around a couch. For this we picked a *KLIPPAN* sofa from IKEA, which is designed for removable covers. This allowed embedding electrodes in the space between the couch and a *Dansbo* burgundy cover. The cover hides the electrodes, but does not inhibit capacitive sensing [265]. We attached six electrodes to the couch: on both halves of the couch three electrodes are placed on the back rest, near the front edge, and towards the back (see Figure 4.9).

We perform capacitive sensing with a simple resistor-capacitor network (see Figure 4.10), attached to an Arduino Uno. The Arduino is programmed to continually charge and discharge each electrode. During a charge cycle, the Arduino monitors the voltage on the electrode. Because of the capacitance of the electrode there is a delay between the time the charging voltage is applied and the voltage on the electrode reaches the same value. If a human comes close to the electrode the capacitance increases, thus also changing the time it takes for the electrode to charge. This difference in timing is used to detect proximity.



Figure 4.9: We attached six electrodes on the couch, normally hidden underneath the couch cover. Capacitive sensing is still possible through the fabric.



We equipped the room of the couch with several smart home devices. A lamp next to the couch is plugged into a Belkin *WeMo Switch*, allowing for remote control of the lighting. Speakers and a screen are wired up to allow over-the-network media playback. Sensor readings are collected on a PC which also orchestrates the connected devices.

4.4.2 Evaluation

Before implementing our smart home scenario, we set out to evaluate how well we could differentiate user's postures. In addition to six different seated postures, we investigated two lying down postures and also include an empty couch condition. Hence, our classes to be distinguished are:

- | | |
|---|--|
| Class 1 One person, right, on edge | Class 6 One person, left, lean back |
| Class 2 One person, right, upright | Class 7 One person, lying down to right |
| Class 3 One person, right, lean back | Class 8 One person, lying down to left |
| Class 4 One person, left, on edge | Class 9 Empty couch |
| Class 5 One person, left, upright | |

Table 4.3: A naive Bayes classifier recognized seating postures with 92.9% accuracy.

Class	1	2	3	4	5	6	7	8	9
1	78.2%	21.8%	0%	0%	0%	0%	0%	0%	0%
2	3.9%	84.9%	7.5%	0%	0%	0%	3.7%	0%	0%
3	0%	0.2%	99.8%	0%	0%	0%	0%	0%	0%
4	0%	0%	0%	89.9%	9.2%	0.4%	0%	0.5%	0%
5	0%	0%	0%	0.2%	98.6%	3%	0%	0.2%	0%
6	0%	0%	0%	0%	0%	100%	0%	0%	0%
7	0%	0%	0%	0%	0%	0%	100%	0%	0%
8	0%	0%	0%	0%	2%	0.9%	0%	97.1%	0%
9	0%	0%	0%	0%	0%	0%	10%	0.1%	89.9%

For each class, we recorded 100 samples (snapshots of raw capacitance sensor values while in the target posture) from 10 participants (2 female). We then analyzed the data with leave-one-subject-out cross-validation using WEKA [98]. Comparing a kNN (79.4% accuracy), a logistic regression (85.7% accuracy), and a naive Bayes (92.9% accuracy) classifier, we achieved the best results with the naive Bayes one. See Table 4.3 for a confusion matrix of the results when using the naive Bayes classifier.

Overall, we achieve good recognition rates with a simple sensing setup. More elaborate approaches (such as in [92], where electric field sensors are used) could potentially further increase accuracy. We also only used a small number of electrodes ([207], e.g., ran some tests with 2048 sensors on a smaller chair). Increasing the number of sensor locations on the couch could allow for even higher accuracy or distinguishing a larger set of postures (such as postures of multiple users). For our purposes though, the classification was good enough to prototype our envisioned smart home scenario.

4.4.3 Controlling Devices via Postures

We implemented a smart living room scenario with our prototype. When the user comes home and sits down, the lamp next to the couch switches on and the display shows a welcome message. Now, the user scoots back and starts checking his phone. This posture change is detected and the system starts to play some music. Once done with his phone, he leans back on the couch, assuming a relaxed position to watch some TV. Two things happen once the couch detects this: (1) music playback is suspended, and (2) the TV screen switches to display of broadcast TV (see Figure 4.11). As time passes, the user grows tired and lays down on the couch. This triggers the lights to switch off and the TV to stop running. Instead, a fireplace scene is displayed to make the room more relaxing. The system could also nudge users to move to the bedroom once such posture is detected.



Figure 4.11: Users can control lighting and media in a smart living room by adjusting their posture on the couch.

4.4.4 Summary

Our posture sensing couch enables implicit control of a smart living room. Note that users still retain the power to override the system. They can, e.g., use the remote to change channels or switch off the TV. However, posture-sensitive furniture, such as our couch, can play a role in supporting users' everyday behavior. They enable a low-engagement channel for interaction, while not prohibiting users from taking back control via devices such as their phone. The implicit behavior can either be pre-programmed, defined by the user, or trained based on users' behavior. Such provision of a low-engagement side-channel for interaction in a smart-home context is similar to the one-button recognizer, as discussed in Section 4.3.

An open question with systems such as the CapCouch is how to make sure implicit behavior is not annoying for users. Users, e.g., will not always want the lamp next to the couch to switch on when they sit down. One approach would be to add more kinds of sensors to be able to reason better on the user's intent. However, this is likely to still fail sometimes and remain annoying for users. Instead, we envision future versions of capacitive sensing furniture to be much higher resolution (either uniform or on more critical parts such as the armrest). Users can then use more intricate gestures for control and a larger set of postures could be available (increasing the entropy of any specific posture). Embedding feedback into the couch (like EmotoCouch [196]) could also support interaction.



Figure 4.12: Our light-control bracelet is made out of silicone, has an embedded micro-controller, and can sense touch and gestures. It allows changing the hue and intensity of a light at three different levels of control.

4.5 Changing Lighting with Varying Control

Lighting is a key component of any home and has gotten more flexible recently with LED lights and lighting control systems allowing for granular control. Instead of switching on a particular lamp, users can dim one corner of a room, change the hue in the other, or animate light progressions. With these new possibilities has come a need to move beyond light switches to more flexible software solutions. Systems like *Philips' hue* come with apps to allow control from users' mobile devices. However, while this gives users precise control of their lighting, such systems come at the cost of requiring close interaction with, e.g., a smartphone or remote—which might not always be desirable.

Offermans et al. have found that there is tension when using light control systems [216]. On one hand, “*participants generally wished to control lighting in [a] holistic fashion,*” but “*they also indicated that sometimes they wish to have a higher degree of control.*” Thus, neither solution, a light switch or an app, are appropriate in all situations. Offermans et al. state, that “*we believe that it is important to support the user with varying levels of control, depending on the context of use.*” This echoes the basic tenants of casual interaction, and shows that how much control users desire is highly dependent on the circumstances. While we might be fine with all lights just switching on when entering the house, a special occasion might, e.g., call for a more detailed light design.

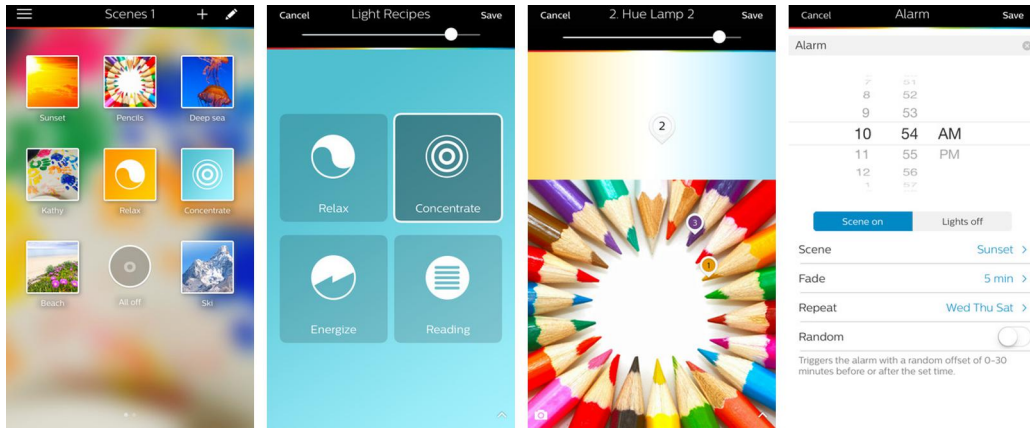


Figure 4.13: The *Philips hue* app for iOS allows changing the lighting in several ways. Shown here are (1) picking a *scene* (saved light setting for one or many lights), (2) choosing a *light recipe* (predefined light hues that are applied to all lights), (3) changing the hue of individual lights by dragging markers to specific colors, and (4) changing the light scene at a specific time. Image © Philips.

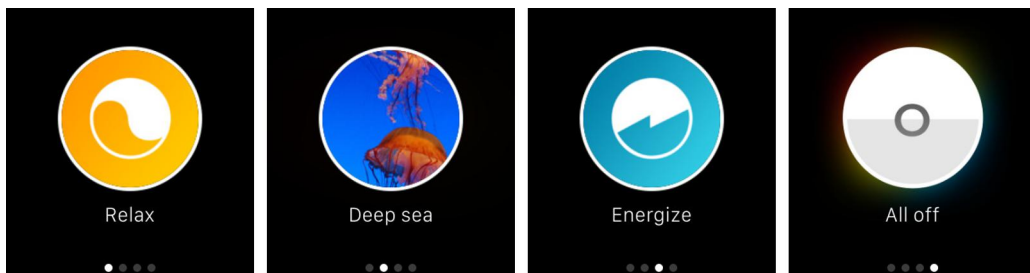


Figure 4.14: The *Philips* app for the *Apple Watch* only allows switching between light scenes, instead of granular control over each light. Image © Philips.

The *Philips hue* app (see Figure 4.13) actually supports different levels of control, but needing to grab and unlock a phone or tablet to run the app imposes an upfront cost. Imagine a couple sitting on a sofa and wanting to change the lights. Using a dimmer switch is more casual than taking out the phone, unlocking it, opening the app, and moving a slider. Instead, we propose moving interaction to a wearable that can cover a wider range of possible engagement and control (see Figure 4.12).

Philips also offers an app for the *Apple Watch* (see Figure 4.14). However, that app only allows one way of interaction: selecting predefined light scenes. Similar to that approach, we also explore a wrist-worn device for this purpose. The wrist is an ideal place for a controller, as it is easily reachable, in sight of the user, and has been found to be the preferred location for digital jewelry for both men and women [231].



Figure 4.15: Bracelet prototypes clockwise from top-left: (1) 3D-printed rigid open bracelet, (2) final silicone-cast bracelet with embedded electronics, (3) silicone-cast strap with magnetic clasp, (4) 3D-printed segmented bracelet, and (5) first closed silicone cast with electronics.

4.5.1 Casual Interaction Bracelet Prototype

Our bracelet design progressed from 3D-printing to silicone casting and from open designs to closed designs (see Figure 4.15). We ultimately found that silicone-cast closed bracelets offer a good trade-off between wearability/comfort and the ability to include all required electronic components. The latter is an especially important aspect when building functional prototypes. Using silicone also makes the bracelet conform to a wider range of wrist sizes, while 3d-printed bracelets were rigid and too loose/tight for some users.

Silicone casting allows for fabrication of more flexible and comfortable prototypes. The silicone bracelets feel smoother and stretch to fit the wrist. We used two different types of silicone, both from *Smooth-On: SORTA-Clear 37*⁴ and *Mold-Star 15 SLOW*⁵. While the first is translucent and harder, the latter is green and more flexible. Both silicones are easy to handle as they mix in a 1 : 1 volume ratio and exhibit low shrinkage when curing.

We initially tried silicone straps that could be closed with a magnetic clasp. These fit several wrist sizes and also were easy to cast. For such flat straps, we could 3D-print reusable molds (also shown in Figure 4.16). However, we found it hard to securely embed the magnets. As these prototypes were comparably thin, we found the harder silicone to work better as the softer one made the bracelets too flexible. But we also found we needed more space available in the bracelet and thus moved to thicker and closed bracelets.

⁴<https://www.smooth-on.com/product-line/sorta-clear/>

⁵<https://www.smooth-on.com/product-line/mold-star/>

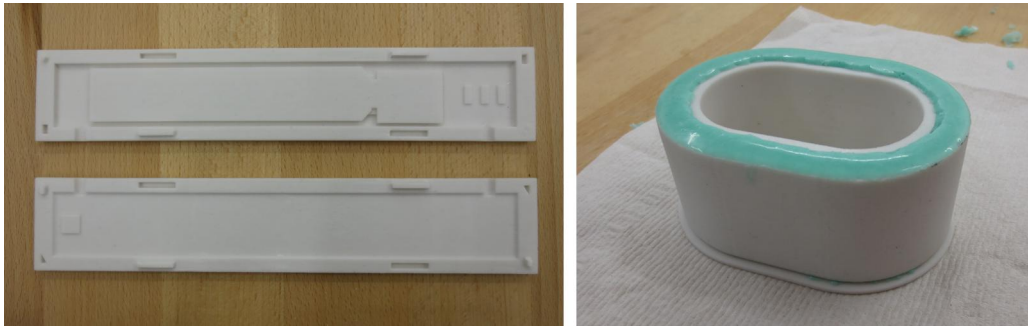


Figure 4.16: Molds for flat silicone straps were 3D printed in two pieces. The molds for closed bracelets were more complicated and consisted of three interlocking 3D-printed parts (outer, inner, bottom).

Our final one-piece silicone cast bracelets were flexible enough to stretch while putting them on, yet fit comfortable once around the wrist. By removing clasping mechanisms, we could also embed electronics anywhere on or in the bracelet. The molds (see Figure 4.16) for this kind of bracelet are more complicated though. This bracelet contains two cavities: one for a capacitive slider and one for the microcontroller and other electronics. The bracelet thickens at those locations to better accommodate the parts while retaining overall stretchability. The space for the capacitive slider is undercut to hold it in place while allowing for direct touch of the slider.

4.5.2 Electronics for the Light Control Bracelet

Our bracelet prototype is designed around a *Teensy 3.1* microcontroller (see Figure 4.17). We found the built-in 12-channel capacitive touch functionality particularly useful. The Teensy sits atop a custom shield with additional components: a 3-axis digital accelerometer, a Bluetooth module, and an RGB LED for feedback. Touch and gesture are processed at ~100 Hz.

We connected a seven-segment 2×9 cm touch slider to the Teensy. The slider is cut from copper tape with a vinyl cutter. By measuring capacitance for every triangular segment and interpolating, we derive a position value for the whole slider. When a user is covering the slider with the whole hand, this is detected as peaks from more than six individual electrodes.

The accelerometer is used for gesture and *force touch* detection. Tap and double tap detection runs directly on the accelerometer and the Teensy is notified when one occurs. We limit tap detection to the z-axis (up-down from the perspective of the Teensy). This prevents detection of arm movements to the sides but works well for taps on the device itself.

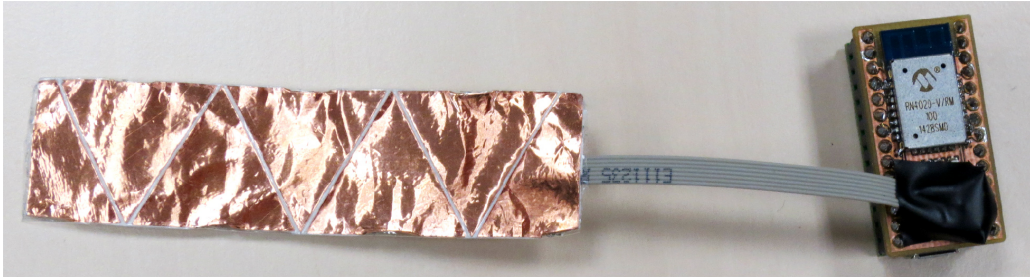


Figure 4.17: The bracelet contains a Teensy 3.1 with a custom shield (containing connectors, Bluetooth, an RGB LED and an accelerometer) that is connected to a seven-segment 2×9 cm capacitive touch slider.



Figure 4.18: Our prototype light setup uses a custom Arduino shield to drive an RGB LED strip inside a spherical lamp. The Arduino receives commands over Bluetooth and updates the LEDs' color accordingly.



Figure 4.19: We distinguish several different touch gestures (from left to right, top to bottom): (1) moving along the slider, (2) touching the leftmost segments of the slider, (3) touching the rightmost segments of the slider, (4) tapping on the device embedded in the back of the bracelet, and (5) covering the whole slider with a hand.

4.5.2.1 Custom Controllable Lamp

We built a custom remote controllable lamp for testing our prototype (see Figure 4.18). Illumination is provided by a 4300 lumen RGB LED strip, which is affixed inside a spherical *Brilliant TIMO* table lamp (25 cm diameter). The LED strip is plugged into a custom Arduino shield and can be controlled by sending messages to the Arduino via a Bluetooth serial link. This setup allowed us easy prototyping of interactions with one lamp and works without additional home automation infrastructure (such as a *Philips hue bridge*).

4.5.3 Input Methods for the Light Control Bracelet

We designed the bracelet with multiple input channels (see Figure 4.19 for an overview) in order to offer several means of control. Users can perform easy control tasks with less effort than harder control tasks. Task difficulty is defined by the inherent informational complexity of the task. For example, switching a light on or off only requires 1 bit of information, whereas setting a specific color requires 24 bits.

The task of changing a light can be tackled at varying levels of complexity (as, e.g., also evident by the different control modes of the *Philips hue* app). From low to high complexity, we chose to provide five distinct control modes:

- Complexity ↓
- Switching a lamp on or off
 - Adjusting only brightness
 - Adjusting the *mood* of the light
 - Change light settings to one of several templates
 - Change the light to a specific color

With increasing complexity, users have to provide more data to the system, but also gain more control over the outcome. Where a model fills in what to do when changing the mood (i.e., picking an actual color for an abstract “*make the light warmer*” command), the user has complete control over the lights’ color when picking a color directly.

4.5.3.1 Switching a Light On or Off

For this basic task, we chose a similarly basic interaction. Users grab the bracelet with their hand, covering the touch slider, and hold steady for about 1.5 seconds. This requires little effort and can be a peripheral interaction [9]. Note that we have not included light selection for this prototype. The bracelet controls the nearest lamp, making light selection implicit. Changing the light in another room is a more complex task and is best handled with a smartphone fallback.

4.5.3.2 Adjusting Light Brightness

Adjusting the brightness of a light is modeled to be similar to using a dimmer switch. While covering the touch slider (as in the of/off interaction), user can turn their wrist to change the brightness value. The wrist's rotation angle is sensed via the accelerometer, assuming the arm is held mostly horizontal.

4.5.3.3 Adjusting Light Mood

Adjusting the mood of a light is a higher level control task, where users can make a light *warmer* or *colder*. These higher level commands are translated into actual color changes by a mood model. Users can make the light *cooler* by holding down on the two leftmost segments of the touch slider, while making it *warmer* via the two rightmost ones. While held down, the mood is continuously changing until release.

4.5.3.4 Switch to a Light Template

Via gestures, users can switch to a predefined light template. Light templates are equivalent to the *light scenes* in the *Philips hue* app. Gesture detection is activated by double tapping on the back of the bracelet (over the accelerometer). After activation, 150 accelerometer samples are collected and then fed to the 3\$ *recognizer* [158] algorithm. We define three light templates and corresponding gestures: (1) *work* activated by a Z-like gesture, (2) *relax* activated by a *circle* gesture, and (3) *lounge* activated by a *pigtail* gesture.

4.5.3.5 Setting a Specific Light Color

When users want a specific color, they can use a complex mode. This mode is activated by holding down on the slider for 100 ms. Sliding up and down on the touch slider then changes *hue*, *saturation*, and *lightness* of the light individually. Slider modes are switched by tapping on the back of the bracelet, which prevents accidental changes during switching.

4.5.4 Design Testing

We tested the bracelet prototype with three participants to evaluate our design. They found interaction with the bracelet to be generally enjoyable. There was some initial confusion as to where the bracelet is sensitive to tapping. This is due to the accelerometer not readily visible from the perspective of a user. After a brief hint, this worked for all of them.

4.5.5 Summary

We have presented the concept of a light control bracelet with multiple control modes. It enable users to avoid the cost of interaction with a phone for intermittent interactions. Moving interaction off the phone is a way to lessen the need for close interactions. By bundling multiple interactions into one device, users are given the choice of how much control they want to have and how complicated they are allowing the interaction to be. Quick and casual tasks like switching off the lights can be done with little effort. Only when desiring precise control do users need to switch to a high control mode.

We also gained insights into the design of bracelet devices. Using hard 3D-printed materials can result in a design that is uncomfortable to wear. This issue could be solved by increased print resolution, polishing the print, or covering it with paint (to smooth out the surface). However, any surface finishing can be tricky for more complex geometries and requires additional effort. Silicone-cast bracelets, however, were well received, are comfortable, and fit a wider range of wrist sizes. Electronics can be embedded into cavities, but cast prototypes can also be cut and changed easily afterwards. Bracelets itself are interesting, as they can move between the focus and the periphery of a user as the arm moves. They are also less intrusive than, e.g., headsets, and are acceptable in many settings (see, e.g., [10]).

For feedback, our prototype included an indicator LED that was used to signal state changes. However, the main feedback channel is the actual controlled light. As users interact with the bracelet, the light color changes and hence provides state information for the interaction. In the future, it would be interesting to add a display to the bracelet itself. A flexible display could, e.g., be embedded over the touch slider and would enable more complex interactions directly on the bracelet. For example, users could then also manipulate lights not in the same room (where no direct feedback is available). Such a wrist-worn flexible display was recently presented by Burstyn et al. [30].

4.6 Chapter Summary

Motivated by the general problems of control in smart homes, we have discussed three different projects. They share a common thread—they each strive to offer less engaged ways to control the house. In the case of the one-button recognizer, user identification fades into the background and is directly integrated with other interactions. The coffee maker now can tell who got a coffee and the light switch knows who entered the room. This clearly relaxes constraints on the certainty of a user's identity. Should we require stronger evidence that the person who just got a coffee is indeed who she pretends to be, we need to switch to a different method. However, this would often inevitably introduce additional effort for the user, having to enter a passcode or swipe a keycard.

Like the one-button recognizer, the capacitive couch focuses on a technological problem of the smart home: sensing what is happening on the couch. But here we put more focus on the actual scenario and how augmented furniture would fit into smart home control. Of course, a “smart” couch would not be the only way to control the room. After all, would we want to shift around in a certain way every time we want to switch to a new television channel? But for some scenarios, augmented furniture can offer a less-engaged way to trigger a change. Especially when trying to signal non-engagement to ones devices, this can fill an important gap in the interaction space.

Finally, we turn to light control and look at how to combine multiple ways to effect lighting change into one wearable device. Compared to the previous two projects, this one is more design-driven and conceptual. Yet, it provides a new perspective to an application space that is dominated by smartphone applications that allow users to concoct complex light settings, but can be too demanding at other times. This also connects back to the work we looked at already in Chapter 3, where we looked at ways to move interaction away from the phone.

5 Continuous Casual Control for Autocorrection

My spelling is Wobbly. It's good spelling but it Wobbles, and the letters get in the wrong places.

— A. A. Milne, *Winnie-the-Pooh*

As described in Chapter 2, casual interactions ideally allow users to pick their desired level of control/engagement with a system on a continuum. Yet, this is not possible with every type of input (e.g., pushbuttons only have two states). Pressure is an example of a modality that allows for continuous input. This makes it a good option for exploring casual interaction systems with continuous change of control. Specifically, we explore how pressure during typing on a soft keyboard can be used to yield or take control over how much autocorrection is applied to the entered text.

This ties into the larger problem of text entry on soft keyboards. Because of the absence of physical keys, and thus any form of tactile feedback, entering text on such keyboards can be challenging. By modeling the uncertainty of a user's touch and combining this with a language model, we can disambiguate user input and help them type faster. In this chapter the model we focus on is ForceType: A touch model that allows users control over the uncertainty of an input event by varying their typing pressure. Providing users with control over input certainty reduces the amount of text users have to correct manually and increases the text entry rate.

This chapter is based on a paper published at *CHI 2014*:

- Daryl Weir, Henning Pohl, Simon Rogers, Keith Vertanen, and Per Ola Kristensson. “Uncertain Text Entry on Mobile Devices.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '14*. New York, New York, USA: ACM Press, 2014, pp. 2307–2316. DOI: 10.1145/2556288.2557412

Statement of Original Work

To this chapter, Keith Vertanen and Per Ola Kristensson contributed the language model, while Daryl Weir and Simon Rogers contributed the touch model. Daryl Weir combined both models and ran the data analysis. I built the ForceType prototype and implemented the study. Daryl and I collaborated closely on other aspects while I interned in Glasgow and during a visit of Daryl in Hannover.

5.1 Introduction

Entering text on mobile touchscreen keyboards is a very common task. However, compared to physical keyboards, touchscreen keyboards are much more prone to error. Where the individual keys of the physical keyboard provide “tactile guidance” (i.e., users can *feel* if their finger is slightly off and correct for this error), such feedback is absent on touchscreens. Users can also not observe where they touch, due to obstruction by their finger (the *fat finger problem* [275]). Combined with noise introduced by the touch sensing or processing, this can lead to substantial errors in the entered text. In addition, users might just not know how to spell a word correctly. All this combined makes text entry on touchscreen not just a common but also a error-prone task.

Touchscreen keyboards can apply sophisticated error correction to both, the touch locations and the entered character stream. This allows to, e.g., correct “hellp” to the likely intended “hello”, or to slightly miss the keys. For this purpose we have built GPTYPE, which automatically corrects for touch and spelling errors.

However, such an automatic correction of the entered text is not always desirable. For example, when mixing languages, entering names, or using slang words, text may be autocorrected to something unintended. With GPTYPE and other correction systems, users then need to go back and manually correct these changes. We hence also built ForceType, which gives users a choice in how much correction they want applied to the text they enter. They do so by varying the pressure exerted on the keyboard during text entry. Higher pressure levels tell the system to take the input as-is, while lower pressure yields some control over the entered text to the system instead. While users can be casual and allow their phones to correct their mistakes for most text, they can elevate their level of control for phrases where they feel the autocorrection algorithm may go astray. So while GPTYPE performs automatic *implicit* correction, ForceType allows users to *explicitly* control the correction.

This chapter focuses on our second system, ForceType. We thus only look at the general structure of GPTYPE, but do not dive into the full evaluation of GPTYPE in this chapter. Please refer to our CHI paper [308] for additional details on GPTYPE.

5.2 Related Work

The underlying correction model for both GPTYPE and ForceType is the same. It builds on previous work on both touch and language models for text entry. ForceType in addition also connects to previous work on pressure input and autonomy handover.

5.2.1 Text Entry Correction Approaches

Several on-screen keyboard correction methods have been explored in the literature [162]. Goodman et al. were the first to combine a language model and a touch model to increase accuracy on soft keyboards [87]. They determined the best key sequence for a touch sequence by maximizing the likelihood, significantly reducing error rates. Kristensson and Zhai presented an alternative approach that used pattern matching to identify the words that most likely corresponded to users' tap sequences [164]. Gunawardana et al. presented an approach with dynamic key-target resizing which defined a minimum target size, preventing key targets from shrinking too much [94]. Rudchenko et al., with their *Text Text Revolution* game, showed that per-user touch models and personalized key-target resizing can further improve typing accuracy [264]. Kristensson and Zhai presented the gesture keyboard *Shark²*, which combined a shorthand writing system with a language model [165]. Goel et al. showed how accelerometer data can be used to correct typing errors resulting from inaccuracies due to walking [83].

5.2.2 Offset Modelling

On any touchscreen device, the location a user touches and the location the user *intended* to touch are offset. This has been studied as a perception problem based on how users target things hidden by their fingers [124]. This is also caused in part by the softness of the finger causing a large touch area—the so-called *Fat Finger Problem* [275].

A body of research exists based on modelling and correcting touch offsets. Henze et al., e.g., collected millions of touches in an Android typing game and modeled offsets using polynomials to improve text entry accuracy [112]. Weir et al. demonstrated offset modelling using a Gaussian process regression model, and in particular showed the importance of user specificity in offset models [309]. Bi and Zhai extend Fitts' law to finger touch using a dual Gaussian distribution approach [17]. Bi and Zhai further extend this to handle target selection on a touchscreen using their *Bayesian touch criterion*, demonstrating a significant increase in accuracy [19].

5.2.3 Pressure Input

Pressure has been used in interactive systems for a wide range of applications. Ramos et al.'s *pressure widgets*, showed how stylus pressure can be used in selection tasks and how many pressure levels can be discriminated [252]. *Force Gestures* by Heo et al. augmented tapping and dragging operations with pressure to extend the available gesture set [113]. A more general investigation of pressure-based interaction was done by Stewart et al. [284] who looked at how holding a device influences target acquisition times.

There is some work on combining typing with pressure data. For keypads, McCallum et al. showed how pressure-based disambiguation allows faster text entry than multi-tap [191]. Shi et al. showed that using a fisheye function for pressure-based selection reduces error rates [273]. Brewster and Hughes used pressure to allow the selection of letter case when typing [28]. Clarkson et al. presented a way of marking messages as urgent when typed more forcefully which is similar to our approach of giving more weight to more forceful typing [51]. In *One-press control*, pressure is used to create a new set of modifier keys [139]. *TypeRight* is a physical keyboard that prevents errors by increasing the resistance of keys for out of vocabulary words [121]. Arif and Stuerzlinger used pressure to allow users to more easily dismiss text entry predictions [4].

5.2.4 Autonomy Handover

ForceType relates directly to the concept of casual interaction, as discussed in Chapter 2. Furthermore, Williamson looked at interactions as a control process, where the level of uncertainty influences how much control is exerted and what kind of feedback has to be provided [314]. Uncertain input for GUIs has been investigated by Schwarz et al., who tried to enable vague input handling for standard widgets [270].

5.3 Implicit Text Correction: GPType

Before we can explore how to give users control over the level of correction, we first need to establish how correction works in principle. As mentioned above, errors occur either because of noisy touch data or because words are misspelled. One approach to remedy touch errors is offset modeling. The goal there is to predict intended touch locations for incoming touch points. But only correcting touch errors does not make use of the properties of the text entry task. After all, offset modeling only predicts likely intended touch locations, but does not take into account which letter these would result in at all. Yet, after a user has entered “t” and “h”, we know that it is very likely the next character is going to be an “e”. Language modeling is an approach that uses such statistical properties of text to determine which sequence of key presses is most likely. In GPType, we combine both, a state-of-the-art probabilistic touch model and a long-span language model for a novel decoder. Our decoder searches for the most likely text given the uncertainty of the tap locations, and possible missing or extra key presses.

Overall, GPType hence is a correction system consisting of three parts: (1) a language model which assigns probabilities to sequences of text, (2) a touch model, which assigns probabilities to each key on a keyboard given a touch location, and (3) a decoder which combines the language model and touch model probabilities and decodes what the user was trying to type. We take a look at each of these component independently below.

5.3.1 Language Model

We adapted a language model that has previously been used for a thumb-typing touch-screen keyboard [220]. The Twitter-based language model was trained based on 778 million tweets sent between December 2010 and June 2012. Duplicate tweets, retweets, and non-English-language tweets were eliminated via a language-identification module [182] (with a confidence of 95 %). Based on a tweet’s associated metadata we removed all tweets that did not originate from a mobile device. Tweets were split into sentences and we only kept sentences where all words existed in a list of 330 thousand words drawn from Wiktionary, Webster’s dictionary, the CMU pronouncing dictionary, and GNU Aspell. The final dataset consisted of 94.6 thousand sentences, 626 million words, and 2.56 billion characters.

The language model was built using the SRILM toolkit using a vocabulary of A-Z, space, apostrophe, comma, period, exclamation point and question mark. The character-based 7-gram language model was smoothed using Witten-Bell and no count cutoffs. The model was then entropy-pruned to reduce the memory footprint in anticipation of using the model on a mobile device. The final model had 225 thousand n-grams and a compressed disk size of 2.0 MB.

5.3.2 Touch Model

To generate input probabilities for the decoder, we use Gaussian process regression to model each user’s touch offset function. Gaussian processes are flexible, non-parametric statistical tools commonly used for regression and classification. Given 2D touch locations $\mathbf{s} = (x, y)$, the Gaussian process learns a function which maps to the offsets (Δ_x, Δ_y) . The Gaussian process is defined in terms of its mean function $\mu(\mathbf{s})$ and its covariance function $C(\mathbf{s}_n, \mathbf{s}_m)$. We chose the mean function to be zero, since in the absence of data we wish to predict no offset. The covariance function defines how similar the n-th and m-th outputs should be given the corresponding inputs. This approach builds upon previous work on using a Gaussian process for touch offset modeling [309].

Following Weir et al. [309], we choose a linear combination of a linear and a Gaussian covariance function:

$$C(\mathbf{s}_n, \mathbf{s}_m) = a\mathbf{s}_n^T \mathbf{s}_m + (1 - a) \exp\{-\gamma\|\mathbf{s}_n - \mathbf{s}_m\|_2^2\} \quad (5.1)$$

where a controls the relative influence of the terms, and γ is the scale parameter of the Gaussian function. As reported by Weir et al., this mapping from device touch location (or raw touch sensor values) to the intended touch location is mostly non-linear. We determined optimal values for the covariance function parameters via 10-fold cross-validation per user.

The advantage of the Gaussian process over a parametric method such as a polynomial (used, e.g., in [112]) is that the prediction is a *distribution* over possible offset values instead of the prediction of just a single outcome. For text entry, we use this to obtain probabilities over the different keys on the keyboard for a given touch. Intuitively, keys close to the mean of the Gaussian are likely targets, while keys further away are deemed less likely. This probability of a given key is found by integrating the density function of the predictive Gaussian over the rectangular area of the key. In practice this is not possible analytically. Instead, we approximate the probability by sampling from the Gaussian, counting which keys the samples fall into, and then normalizing these counts to obtain probabilities.

The sizes of the predictive Gaussians, and consequently the probabilities produced by the model, are learned from training data. Thus, in areas of the screen where the offsets are more variable, the Gaussians are larger and the probabilities more diffuse. This gives the language model more latitude to correct inputs in these areas. The Gaussian process models the uncertainty in the touch interaction and uses information about that uncertainty to make predictions. This is an advantage over other probabilistic approaches, such as that proposed by Bi and Zhai [19], which model the touch uncertainty with fixed parameters for all users. The trade-off is that users must provide calibration touches before using the system.

5.3.3 Decoder

We created a decoder which searches for the most probable character sequence given a sequence of taps. Each “tap” is in reality a probability distribution over every keyboard character. These tap distributions were set according to the Gaussian process tap model as well as the ForceType model. Both models incorporate a measure of the uncertainty of a given tap. For example, the Gaussian process tap model, as described above, might learn that taps in the upper left corner are less accurate than in the center. The probability distribution in the corner would then also be wider than one in the center.

As we will describe shortly, we explored four different correction strategies in our interface. Each strategy involved changing how much of the tap sequence the decoder was allowed to change. To facilitate this, the decoder allows some taps to be marked as fixed. Fixed taps served as language model context, but were not eligible for change during the decoder’s search.

The decoder searches in the space of all possible character sequences for the non-fixed taps in an observed sequence. During this search, a tap would most likely generate the actual key hit. But the decoder also allows substitution with all other possible keys according to the provided tap distribution.

Our decoder allows an observed tap to be deleted without generating a character. Similarly, it explores inserting all possible characters without consuming an observation. Both insertion and deletion hypotheses incur a configurable penalty.

During its search, the decoder makes use of a character language model. As each output character is generated, we combine the tap probability with the probability of the character given the language model by taking a weighted product. The relative contribution of the tap distribution and the language model is controlled by a configurable scale factor.




The search over all possible character sequences is exponential in the length of the sequence. Pruning is thus critical to ensure real-time performance. During its search, any hypothesis that becomes less probable than the current best answer is pruned. Additionally, we employed beam width pruning. In beam width pruning, the decoder tracks the best hypothesis found thus far for each position in the tap sequence. Hypotheses that are too improbable (i.e., outside the beam width) compared to the best one at a particular position are pruned. By varying the beam width, we can control the speed accuracy trade-off during recognition. The free parameters of the decoder were optimized with respect to the data we used to learn the Gaussian process probabilities.

5.4 The Autocorrect Trap

So far we have explored a novel approach of error correction for touchscreen keyboards. Current touchscreen keyboards use a wide range of other autocorrection techniques (e.g., adaptive but clamped target resizing [94]). These techniques, as well as our GPTyping method presented here, implicitly model uncertainty: the user has little influence on the way autocorrection works. In any such system, users might be able to reject a proposed correction, delete a character and retype with autocorrection switched off, or select the corrected word and pick from the originally typed version, or other suggestions. While this theoretically allows users to control text correction according to their needs, there are still widespread frustrations and situations where text prediction falls short. When typing unknown words, using regional dialects, or mixing languages, autocorrection is often not flexible enough.

We thus propose giving users control over how their phones correct text. Empowering users to vary the level of error correction per word allows users to fall back to automatic error correction for phrases they deem correctable while being able to tighten the reins during phrases they feel their phones cannot handle. This assumes that users have a sense of whether a phrase is hard for the text prediction on their phone or not. To get an idea of how the capabilities of auto-correction are viewed by users, we conducted a study.

Table 5.1: For a set of 20 phrases, we compared actual occurrence of autocorrection (based on behavior of three sample devices: an iPhone 5, an LG E700, and a Samsung Galaxy S3 Mini) with expected autocorrection (as rated by 28 participants).

Sentence	Triggers Autocorrect				Average Rating*
				overall	
The quick brown fox jumps over the lazy dog	no	no	no	no	4.46
Yesterday we went hiking up the Kanakou	no	no	no	no	2.46
Man! I just PAWND sum dum n0085	yes	yes	yes	yes	1.18
fo' shizzle mah nizzle	yes	yes	yes	yes	1.21
once i saw his face, i was a belieber	yes	yes	no	yes	1.25
To thine own self be true, and it must follow, as the night the day, thou canst not then be false to any man.	yes	yes	yes	yes	2.21
Fair fa' your honest, sonsie face, Great chieftain o' the puddin'-racer!	yes	yes	no	yes	1.29
The medications — primarily Adderall, Ritalin, Concerta and Vyvanse — often afford those with severe A.D.H.D. the concentration and impulse control to lead relatively normal lives.	yes	yes	yes	yes	2.25
Welschmerz in this meaning can cause depression, resignation and escapism	no	no	no	no	2.43
That took a lot of chutzpa	yes	yes	no	yes	2.82
Open the crate but don't break the glass.	no	no	no	no	4.75
Suikoden is the hardcore gamer's Final Fantasy.	yes	no	no	no	2.43
He was an agent provocateur.	no	no	no	no	4.18
She flopped trip nines and made a full house on the turn.	no	no	no	no	4.11
He graduated summa cum laude from Harvard.	yes	no	no	no	2.86
I sense Moriarty's hand in this.	yes	no	no	no	3.50
The swan dive was far short of perfect.	no	no	no	no	4.36
Leet speak is incredibly naff.	no	no	yes	no	2.07
Shinobi vs Dragon Ninja was a popular song at the time.	yes	yes	yes	yes	2.89
Yolo is just Carpe Diem for stupid people.	yes	yes	no	yes	2.50

* As rated by 28 participants on a 5-point Likert scale where low ratings indicate autocorrection is expected to occur and high ratings indicate autocorrection would leave the sentence as-is.

We designed a questionnaire listing 20 phrases (see Table 5.1) of varying levels of difficulty—some consisting only of common English words and others with proper nouns, slang, and/or words or phrases borrowed from other languages (e.g., *summa cum laude*). Participants were recruited using mailing lists and social media. For each phrase, participants were asked to rate on a 5-point Likert scale whether they thought autocorrect would change it or not when entered on a smartphone.

We received 28 responses (8 female, age 17–44, $\bar{x} = 29.0$, $SD = 7.5$) to the questionnaire. Asked to rate their English language skills, 86% of participants rated themselves as functionally native speakers. The rest of the participants still rated themselves at a near native level.

To establish a ground truth on whether our phrases would be autocorrected or not, we entered them into three smartphones—an iPhone 5 running iOS 6.1, an LG E700 running Windows Phone 7.5, and a Samsung Galaxy S3 Mini using the SwiftKey keyboard on Android 4.1. We noted for each phone which phrases were autocorrected (see Table 5.1 for results). In cases of disagreement, where some phones corrected and others did not, we used majority voting to produce a single binary value for each phrase.

We then split the user ratings for corrected and uncorrected phrases and produced histograms showing the counts for each rating. By normalizing these histograms we obtain discrete probability distributions over the ratings for the two groups of phrases. Our results are shown in Figure 5.1. For phrases which would actually be autocorrected, the most probable rating is 1 (certain correction) and the least probable is 5 (no correction). The monotonic decrease across the other ratings shows a clear trend, indicating our respondents had a good sense of phrases that would be autocorrected. For phrases that were not autocorrected, there is an inverse (yet more fuzzy) trend.

For phrases that were not corrected, the trend is less pronounced. The most likely rating is for no correction and there is a remarkably high probability that users rate these uncorrectable sentences as certain or near certain corrections. In essence, users overestimate how likely autocorrect systems are to take action. This is not necessarily an issue—if a user takes action to prevent autocorrection when none would have occurred, there is no difference to their final input. These results motivate the need for a system with finer control over automatic error correction behavior. The distributions over ratings are significantly different (Wilcoxon test, $p \ll 0.01$) for the two groups of phrases. This suggests that users have a functional mental model of the way autocorrect operates and would be able to identify situations where preventing autocorrection is desirable.

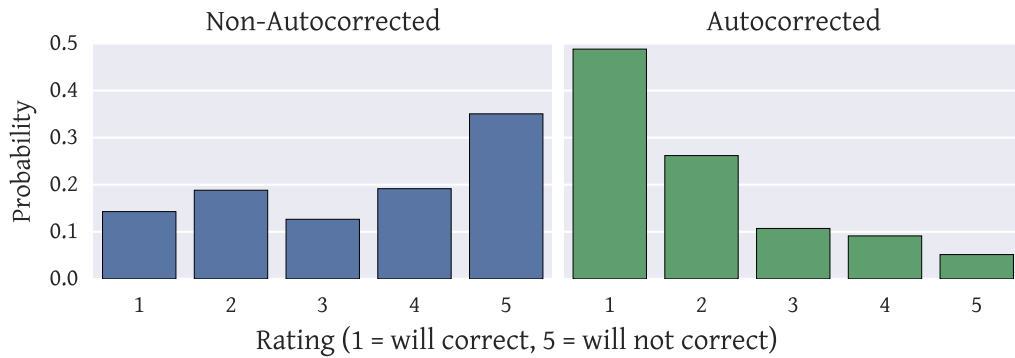


Figure 5.1: For a set of 20 phrases, we asked 28 people whether they thought their phone’s autocorrect would change it when entered, or leave it unchanged. Participants gave a rating between 1 (will definitely be changed) to 5 (will definitely not be changed). We classified each phrase by testing it for autocorrection on several different phones. This figure shows the probability of each point on the rating scale for both classes (changed and unchanged).

5.5 ForceType: Pressure as Certainty

Given that users already have a mental model of which words are likely to be autocorrected, we desire an input modality which enables them to express certainty about words they do not want corrected. In essence, we wish to allow users to negotiate control between themselves and the language model. For normal typing, the autocorrect functions as it does on current phones, but when the user suspects the word they are entering is unknown to the phone’s dictionary they can smoothly limit the autocorrect behavior. This essentially allows them to decide how much control they are willing to yield to the autocorrection. We choose pressure as the factor to control this negotiation. When the user wishes to indicate certainty, they simply press harder on the keys. Srinivasan and Chen [280] have shown pressure to be controllable by users and the idea of pressing hard for a different input behavior is simple to grasp.

Other research has considered negotiated uncertainty through user input. Rogers et al. [259] use the height of the finger above the screen as a proxy for uncertainty when browsing a map. However, although “hover” input of (albeit using a different, less accurate, technology) has been introduced on commercial phones, such as the *Samsung Galaxy S4*, it is not an appropriate choice for a typing application. Subramanian et al. [287] showed that users display unintentional drift when varying hover height, so controlling uncertainty via this modality may be suboptimal—hence our choice of the pressure modality. Hover also does not integrate as well with typing. Having users press down on the screen clearly delineates key presses.

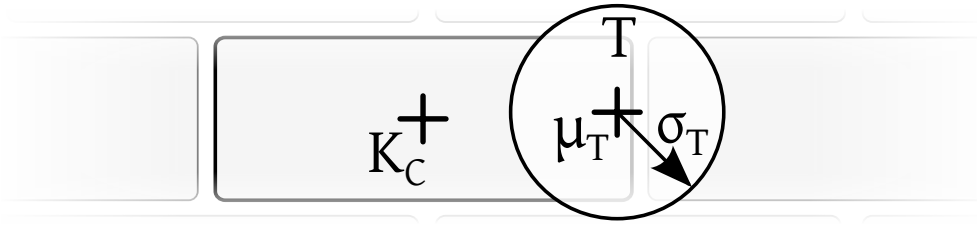


Figure 5.2: The touch model used by our correction system ForceType. For each key K , we evaluate the likelihood of its center K_C under a Gaussian on the touch point μ_T . The standard deviation σ_T is controlled by pressure. Higher pressure causes a narrower distribution.

5.5.1 Correction Model

We use the same language model and decoding algorithm as for GPTtype, but rather than generating key press probabilities from a GP we use a pressure dependent touch model. The likelihood of a key given a touch is computed as a Gaussian with a standard deviation that changes dependent on the pressure. In particular, we take the standard deviation as: $\sigma_T = C/\omega_T$, where C is a constant and ω_T is the pressure for touch T . Thus for high pressure touches the variance is small and the probability mass is concentrated in the pressed key, whereas for lower pressure touches the probability mass is spread over the keyboard, allowing correction to take place. The touch model is illustrated in Figure 5.2.

The constant C needs to be calibrated appropriately. We chose a value such that the distribution for a ‘typical’ touch had a standard deviation equal to half a key width. To determine what defined ‘typical’, we asked 10 participants to repeatedly write *the quick brown fox jumps over the lazy dog*, an English pangram, on our input device, a pressure sensitive touch pad called the ForcePad (more information on this hardware is given in the next section). This was done without presenting any feedback about what was typed, and in the absence of any correction scheme. This provided us with pressure information for typing with every key on the keyboard. From a histogram of the logged pressure information, it was clear the data were not normally distributed and had high positive skew. We chose to fit a gamma distribution to the data. Next, from the 2333 collected datapoints we removed 144 outliers (>2 SD away from the mean – these were primarily points where the hardware reached its upper sensing limit). We then refitted a gamma distribution to the remaining data. The mean of this distribution was 168.28 g/cm^2 (SD = 106.20 g/cm^2). We then chose C such that the Gaussian distribution for a touch with this mean pressure had a standard deviation equal to half a key width.

5.6 Evaluation

We conducted a user study to determine the effects of pressure-based scaling of text correction. We hypothesize that:

- H1** Pressure-adaptive error correction requires fewer word corrections by the user than constant-scale text correction.
- H2** Users are faster when typing words unknown to the error correction algorithm when utilizing pressure adaptation.

To test these hypotheses, we had participants type a series of phrases in two conditions: with and without pressure adaptation. For the latter, we use the same correction model but rather than adapting the variance of the touch Gaussian based on pressure, we used a fixed value, chosen such that the standard deviation was equal to one key width. This corresponds to the standard deviation in the pressure sensitive model when the touch has mean pressure and approximates the behavior of autocorrect on modern smartphones.

We used a between-subjects design for the study, so that each participant used only one correction model. This decision was made due to the difficulty of having a participant learn the pressure sensitive model and then asking them to “type normally” for the other condition. We suspected the learned changes in typing behavior might carry over to the non-adaptive keyboard condition. Proper counterbalancing of the conditions in a within-subjects design is thus not possible, restricting us to the between-subjects design.

In order to assess the impact of using the model on the text entry speed for the pressure group, the phrase set used needs to contain both *correctable* phrases, containing only words known to the correction model, and *uncorrectable* phrases, containing at least one word unknown to the model. The former should not require pressure typing invocations if the user has a good understanding of the language model behavior, while the latter set will require one or more invocations.

5.6.1 Participants

We recruited 16 participants (5 female, age 21–39, $\bar{x} = 25.69$, $SD = 4.48$), where all but three were smartphone owners. On average, participants had over 3 years of smartphone experience. On a 1–5 scale (1 = beginner, 5 = functionally native), participants rated their English language skills at 3.06. Each participant was randomly assigned to one of the two conditions—text entry with pressure-adaptive autocorrect and text entry without. After the experiment, participants were given a small non-monetary gratuity.

5.6.2 Apparatus

For pressure sensing and finger tracking we use a *Synaptics ForcePad* sensor. The device weighs 526 g and measures $14.2 \times 12.2 \times 1.1$ cm. The touch-sensitive area covers 10.9×6.9 cm on the device's surface. The sensor's diagonal thus is 12.9 cm—within roughly 6% of a *Samsung Galaxy S III*'s 12.19 cm. Overall, the Forcepad is slightly larger but considerably heavier (an S III only weighs 133 g) than current mobile devices. However, it does provide a comparable experience when holding the device in landscape mode and typing. Thus, we feel confident the Forcepad is a valid placeholder device for evaluating how future mobile devices incorporating pressure sensing touch could be used.

The ForcePad provides capacitive tracking with pressure information for up to 5 fingers. Force readings are provided with 6 bit resolution at 67 Hz. We take the maximum value in the last 10 frames as the pressure value for a touch. The force sensitivity of the ForcePad made us choose this device instead of for example approaches using accelerometers to infer typing pressure [132]. Such approaches are typically limited or inaccurate, and in the first instance we wanted accurate values to assess how users controlled the system. However, we consider an exploration of using these techniques to bring ForceType to modern smartphones as an important avenue for future work.

The ForcePad is an input-only device and cannot display visual feedback or instructions directly on its surface. Using an external monitor for feedback, however, would not allow us to evaluate normal typing behavior. We thus modified the ForcePad by attaching a 132×32 px LCD to the device on which we can display three lines of text in a medium-sized font. The LCD was glued to the top of the ForcePad increasing the overall weight to 651 g without impeding hands holding the device. The modified device is shown in Figure 5.3. An Arduino is used for control and allows the connected PC to set display content via the serial port. This combination of ForcePad and LCD enables us to simulate future mobile devices with pressure-sensitive touch. One limitation of this hardware is there is no visual feedback on corrections before they happen—this is another reason that exploring pressure proxies on current smartphones is a focus for future work.

To simulate an on-screen keyboard, we glued a keyboard overlay on the ForcePad. We used a modified version of the iOS landscape keyboard with all buttons triggering mode switches removed (so as not to confuse participants). The glued-on keyboard does not introduce new haptic cues and thus provides an experience comparable to current smartphone keyboards.



Figure 5.3: We modified a Synaptics ForcePad to allow for graphical feedback to users while typing by attaching an LCD display on top of the device. The display is controlled via an Arduino (not shown). A modified version of the iOS landscape keyboard is glued to the device. Shift, voice-recognition, and mode switch buttons were removed and the space key extended to encompass the @-key.

5.6.3 Procedure

We used a subset of the English language *NUS SMS Corpus* (version 2012.04.30) [40] as phrase set. This dataset contains 41 537 text messages, sent primarily by users in Singapore, India, and the USA. Text messages often contain slang and shorthand, making them a good example of the a text where autocorrect commonly fails. However, not all messages are equally appropriate for our evaluation and we removed all messages that:

- are shorter than 15 or longer than 50 characters
- contain any character not in the set given by the ISO basic Latin alphabet plus the space and period characters
- are shorter than three words
- contain unknown one-letter words

This filtering leaves us with set of 5733 phrases. Our main concern is our requirement for unknown words in the messages—words that can not be found in a standard dictionary known to a language model. To determine such unknown words, we make use of the built-in Android `en-us` dictionary. Words not found in the dictionary are ones that hence are also not handled by autocorrect. There hence is a high likelihood that they get corrected to a word that actually *is* in the dictionary.

We now split our phrase set into two parts:

Correctable Phrases that only contain words found in the dictionary ($n = 1272$).

Uncorrectable Phrases that contain at least one word not found in the dictionary ($n = 4461$).

For every participant, 20 correctable and 20 uncorrectable phrases were picked at random. While phrases were chosen randomly, a post hoc analysis showed that there was no significant difference across users in the number of out of vocabulary words per sentence ($p > 0.4$). Participants were not provided with an indication whether a phrase was correctable or not.

At the beginning of the study, we gave participants the chance to familiarize themselves with our prototype. Participants then had to complete the 40 trials. In each trial, they were shown the complete phrase on a screen in front of them and then had to copy that phrase. During text input, the phrase (clipped to the display size) was shown alongside the response text on the device. We asked participants to copy the phrases accurately—if they noticed a mistake, whether from a language model correction or their own typing, they were instructed to correct it. After using backspace to delete part of a word, autocorrect was disabled until the next word. This is equivalent to the behavior on phones and prevents infinite correction loops. Participants had to submit each phrase with the return key.

5.6.4 Results

One participant was excluded from the analysis, as the touch offsets were so large the touches rarely hit the correct key. For the remaining participants we analyze performance both as the number of errors and as text entry speed. Hence, **H1** relates directly to the former and **H2** to the later.

5.6.4.1 Errors

As our performance measure, we use the *active correction rate*, defined as the proportion of words which the user has to actively correct by backspacing and retyping. This is a similar approach to Hoffman et al. [121], where they looked at the number of times backspace needed to be pressed. If **H1** is accepted, we should see a decrease in active correction rate since users can use pressure to prevent autocorrection of non-standard words. Note that the more standard character error rate is not a suitable choice as we are not interested in how many *errors* users make, but how often they have to *intervene* with the autocorrection. Our choice is supported by the results—the mean character error rates for the two study groups were not significantly different ($\sim 3\%$ in each case).

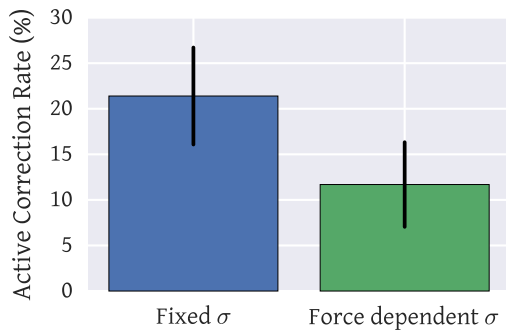


Figure 5.4: ForceType requires significantly fewer active corrections from users when entering text. Required corrections dropped by ≈ 10 percentage points. Error bars show standard deviation.

Figure 5.4 shows the active correction rate. Using pressure adaptation, the average user had an active correction rate of 10.86 %, compared to a value of 19.48 % for users without pressure adaptation. This is a significant decrease compared to the baseline condition (independent two-sample t -test: $t(12) = -3.48, p < 0.005$). We can thus reject the null hypothesis and accept **H1**—pressure-adaptive error correction required fewer corrections than constant text correction.

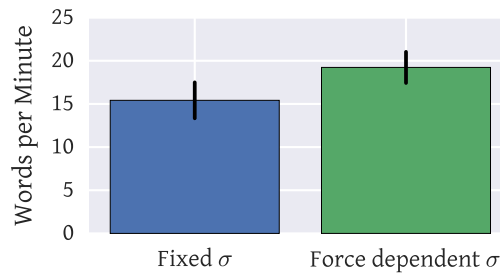
The active correction rates for both groups, ForceType user and users of the non-adaptive keyboard, are quite high. Since the phrases are from an SMS corpus, many of the words are not found in common English. Unfamiliarity with the form factor of our apparatus may also have increased active correction rate.

5.6.4.2 Entry Rate

Entry rate was measured using words per minute, with a word being defined as five consecutive characters, including spaces. With pressure adaptation active, users typed 19.23 words per minute, while without they only typed 15.42 words per minute (Figure 5.5). This is a significant increase in typing speed (independent two-sample t -test: $t(12) = 3.5002, p < 0.005$). We can reject the null hypothesis and also accept **H2**—pressure adaptation allows for faster text entry in the presence of out of vocabulary words.

We also looked at the impact of uncorrectable phrases on typing speed. A small drop in speed could be expected, as users have to invest more mental effort in processing those phrases. We saw entry speed in words per minute go down by 3.97 (ForceType) and 2.61 (baseline). Both changes are significant ($p < 0.05$), while the difference between the two changes is not ($p > 0.13$). Thus, ForceType and the control are both equally affected by uncorrectable phrases, resulting in a small performance drop. Overall, ForceType resulted in a faster text entry rate.

Figure 5.5: ForceType enabled users to enter phrases more than 20 % faster. A significant increase over the baseline. Errors bars show standard deviation.



5.7 Discussion

By combining a probabilistic touch model with a language model and decoding algorithm, GPType can increase text entry accuracy (for details see [308]). Building on this work, ForceType then tackled cases where traditional autocorrect systems failed. By allowing users to convey certainty in their input using pressure, autocorrect can be dynamically turned off to allow entry of words not in standard word lists, such as acronyms or words from local dialects. ForceType was successful in increasing typing speed and reducing the number of cases where users had to manually correct their entered text.

These two systems represent different approaches to the uncertainty inherent in text entry. In GPType, we model this uncertainty in a manner that is hidden from users, allowing the system to account for and correct many commonplace errors. In ForceType, the uncertainty is *explicitly* controlled by the user, allowing finer control over the system when the user can anticipate undesired autocorrection behavior.

It is likely that neither of these approaches is strictly better than the other. In future work, it would be interesting to combine offset modeling with explicit pressure control and see if input accuracy can be further increased.

5.8 Summary

Text entry is a ubiquitous activity on modern mobile devices, but continues to present challenges. The process is naturally very uncertain, both because of physical effects (e.g., the Fat Finger Problem) and due to uncertainty of intent—existing correction systems often assume users are trying to type words known to a dictionary, but this is certainly not always the case. With our novel combination of language and touch models with a state-of-the-art decoder in GPType, we can better capture this uncertainty and correct for resulting errors. This also allowed us to built ForceType, which allows users to explicitly control uncertainty in text entry—specifically for improving handling of autocorrection.

Autocorrect on current mobile devices provides valuable assistance when typing. However, the lack of control can be frustrating to users, who can feel overruled by the way autocorrect works. ForceType gives back some control to the users who can now decide for themselves how much power they are willing to give to autocorrect. We found that people have good models of words which need correction, but are often overly concerned about the potential for interference from autocorrect. In situations where they are unsure, increased pressure can give them peace of mind that they are in control. When typing normally, nothing changes for them, but if they see a need they can step in and retain control. Evaluating ForceType, we showed that users could successfully prevent autocorrection of words they did not want to be changed by varying their input pressure as they typed, resulting in faster overall text entry rates.

While our apparatus is prototypical in nature, we believe ForceType could easily be integrated in mobile devices. There are already devices on the market that are able to report pressure readings. The *Blackberry Storm 2*, e.g., has a *SurePress* touchscreen, where four piezoelectric sensors under the screen, are used to discern click actions from touches. Similar technology has been developed by Motorola [63] and Sony [329]. Since the *iPhone 6s*, Apple also includes pressure sensitive multi-touch sensors into many of their phones and devices.

Pressure can also be inferred from other sensor information. In *ForceTab*, e.g., Heo and Lee show how accelerometers can be used to determine the intensity of a touch and how this could be used to make touch more expressive [114]. Iwasaki et al. have used the accelerometer to determine typing pressure, which they use to enable, e.g., expressive typing (font changes with pressure) [132]. *GripSense* infers pressure information from observing vibration dampening using gyroscopes [84]. Arif and Stuerzlinger estimate pressure directly from touch screen data, combining previous work into a hybrid approach that takes into account touch point movement and timing [4]. We are thus confident that ForceType could be brought to mobile devices in the near future. ForceType could also inspire new research on methods for giving users more control in how automation works and when it is active.

The way control is yielded or taken from the system is more opaque in ForceType than in other casual interaction systems. For example, in the steering task study discussed in Chapter 2, users could pick between three techniques with different respective levels of engagement. Having a discrete set of techniques to pick from makes for a more obvious impression of choice than the slight change between control levels in ForceType. As the control trade-off is also implicit in ForceType (it is integrated with keypresses) it might be even less noticeable. But there is no requirement in casual interaction that yielding of control needs to be a conscious process.

6 Casual Communication

Words do not express thoughts very well. They always become a little different immediately after they are expressed, a little distorted, a little foolish.

— Hermann Hesse, *Siddhartha*

So far, we have looked at casual interactions in which some control is delegated to the interactive system. Users can then fall back to less precise or more ambiguous input, when they do not need full control over the outcome of the interaction. As described earlier, this might, e.g., be the case when they just want to change the *light mood* but do not care about the precise *light color*.

In this chapter, we explore how this leniency in expression is not just limited to control, but also applies to communication with others. Traditional mediums such as letters or emails allow for complex and precise expression (within the bounds of language itself). However, writing a letter or composing an email can be too *heavy* a task in many situations that call for a lighter form of communication. An example of such a reduced channel is *texting*, where limitations on the message length have resulted in a more concise and informal communication.

Within casual communication, we later focus specifically on emoji. These parts of the chapter are based on a paper¹ published at *MobileHCI 2016* and an article published in the *ACM Transactions on Computer-Human Interaction*:

- Henning Pohl, Dennis Stanke, and Michael Rohs. “EmojiZoom: Emoji Entry via Large Overview Maps 🗺️👁️.” In: *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services companion - MobileHCI '16*. 2016. DOI: 10.1145/2935334.2935382
- Henning Pohl, Christian Domin, and Michael Rohs. “Beyond Just Text: Semantic Emoji Similarity Modeling to Support Expressive Communication 🧑👤📱😊.” In: *ACM Transactions on Computer-Human Interaction* 24.1 (2017). DOI: 10.1145/3039685

¹Which in turn is based on work done by Dennis Stanke’s for his Bachelor thesis [281]

6.1 A Definition of Casual Communication

In Chapter 2 we already saw how *casual* has several meanings. So far, we have concentrated on casual interaction that caters to cursory or unconcerned use. But in the context of communication, casual is commonly taken as antonymous to formal communication. In fact, as also described in Chapter 2, previous work has used the term casual interaction to refer to informal communication scenarios, such as quick chats around the water fountain. Hence, casual communication in that respect primarily refers to the *situation* in which the communication is taking place, but also other contextual aspects such as *who* is communication or *what* the communication is about.

While informality is a major aspect of casual communication, the meaning of *casual* can also be related back to the control aspects underlying the notion of casual used in this dissertation. This comes into play when different ways to communicate come with different levels of control. As we will see in Section 6.2, this could be making use of automatically generated replies in order to avoid typing out a complete message. In this setup, typing the complete message offers more control over the content, but requires closer engagement as well. With the adaptive autocorrection system presented in Chapter 5, we already saw another example of varying control in typing. In this chapter, we revisit text entry, but instead of control of autocorrection focus on supporting less formal communication.

As with general casual interaction, which typing method is most appropriate in a given situation will depend on the specific contextual constraints (see Chapter 2). For example, consider the difference between sitting down at a desk to write an email and sending a quick text while on public transport. In the former scenario, focused interaction is much more appropriate while the latter scenario can benefit from interaction that is less demanding. Similar to those physical constraints, users might also opt for lower effort ways to communicate when they are lazy or tired. If we take into account social aspects of interaction then these play a particularly large role in casual communication scenarios. After all, communication is inherently a social endeavor and apart from the content each message also always contains social signaling as well. As Marshall McLuhan has pointed out: “*the medium is the message*” [194]. This means that a message’s content is not separable from its form: a hand-written love letter differs in meaning from the same text when assembled from newspaper cutouts. Hence, choosing a messaging format that is more casual does also carry an additional social signal to the recipient. For example, heavy use of emoji² (such as 😊) in a message might signal that the sender considers the relationship between the two participants to be more intimate and less formal than, e.g., a work relationship.

²A set of pictographic Unicode characters: <http://www.unicode.org/emoji>

Figure 6.1: The Casio Databank series is an early example of digital watches with data processing capabilities for note taking and calculating. Shown here is the *Casio DBC310-1* from 1997, which comes with a 16-key physical keyboard for both number and character entry. Yet, even with a physical keyboard, text entry comfort still falls short compared to a computer with a full-size keyboard, limiting how well users can communicate with such a device. Image © Casio America, Inc.



6.2 Case Study: Apple Watch

So far our investigation of casual communication has been rather abstract. However, to develop a clearer understanding of the concept, it is helpful to take a closer look at an example. Here, we will do so with the Apple Watch³, one current device that includes several mechanisms for casual communication.

Casual communication is facilitated in the Apple Watch due to a basic problem of smart-watches: intricate input is very hard on space-constrained devices. While there has been work on full QWERTY keyboard input for touchscreens in devices with very small form factors (e.g., [125, 218, 302]), text input is still comparatively slow and cumbersome on these devices. While it is possible to include physical keyboards in devices the size of the Apple Watch (see, e.g., the Casio Databank watch in Figure 6.1), those keyboards are far from the efficiency of a full-sized one and thus also do not offer a better way to communicate. However, the Apple Watch itself does not include any hardware or even software keyboard. Hence, the Apple Watch is ill-equipped for writing a formal response to an email or a long blog post. While we draw upon the Apple Watch as an example here, the same restrictions apply to a wide range of wearables with similarly small designs as the Apple Watch.

The Apple Watch, instead of requiring users to type a message, allows users to compose a range of informal messages (see Figure 6.2) using a feature Apple calls *Digital Touch*. When using *Sketch*, the screen becomes an input canvas for small doodles. Users can draw one or multiple lines to, e.g., sketch a heart or a quick “Hi”. The recipient not only sees the final drawing, but receives a playback of how the drawing was made. Haptic messages are sent when using either *Tap* or *Heartbeat*. While the former allows users to enter a custom tap pattern that is then played back at the recipients wrist, the later just uses the senders current heartbeat for the pattern. A linear actuator is used to generate these tactile effects.

³The Apple Watch was introduced in September 2014. For additional information see Apple’s product page: <http://www.apple.com/watch/>.



Figure 6.2: The Apple Watch supports three special casual communication modes (left to right): *Sketch* allows to send an animated colorful drawing (sketched on the watch by the user), *Tap* sends a haptic stimulus (or pattern) that is felt on the wrist, and *Heartbeat* records a short sample of the user’s heartbeat for playback on the recipients watch graphically and haptically. All images © Apple Inc.

Each of those three modes offers a different level of control over the content of the message. Where sketching allows for more complex messages, tapping only allows for basic patterns. When transmitting their heartbeat, users have to put no effort into the composition, but also have the least control over the message (short of, e.g., a quick jog before sending the message to bring up the heart rate). But while the modes differ in the level of control they provide, they also require different levels of effort from users. For example, sending ones heartbeat only requires resting two fingers on the screen (after activating “digital touch” mode). Sketching potentially requires the most effort as users’ input motion is directly used for the message. However, users do also have the option to send more *uncontrolled* messages by scribbling on the screen without devoting attention to it. Yet, such dashed off messages again necessitate relinquishing control over the actual message content. Generally, sending precise drawings is a higher bandwidth channel than coarse sketches, but hence also requires corresponding quality input. Hence, our overarching theme of trading control for reduced effort is also visible in this choice of message modes.

Apart from sending haptic or sketched messages, Apple Watch users can also receive notifications for incoming messages (e.g., text messages or emails). Yet, as mentioned earlier, there is no keyboard available to users for replying to those messages. However, if a message can be answered with a short response (e.g., if asked “are you on your way?”) having to take out the phone to reply would be too heavy an interaction. Instead, the Apple Watch offers three different methods (see Figure 6.3) for responding that do not require text entry, each with a different level of control. As with sending “digital touch” messages, replying to messages requires users to make a choice in how much control they want over the outgoing message. Each of the available methods also requires different amounts of engagement by the user.



Figure 6.3: The Apple Watch offers different ways to reply to a message. Based on the received message, it generates a set of canned replies that can be send with only one tap. If the predetermined choices are not to a user’s liking, she can also send an emoticon, or dictate a response and send this as either a voice recording or transcript. Dictating the response takes more effort than selecting from canned responses, but also allows more control over the response. All images © Apple Inc.

Examples for all these three reply-modes are shown in Figure 6.3. In the depicted scenario, the user received a message asking her whether she would prefer sushi or pizza for dinner. From this received message a set of canned replies are automatically generated, based on an analysis of the message for keywords and questions. In addition to these “smart replies”, users can also define custom default replies via a companion app on their phone. Users can then select one of these options (e.g., texting back “Sushi”, “Pizza”, or “Not sure”). However, while this is based on the content of the received message, those replies do not necessarily include what the users want to reply. For example, there will not be an automatically generated “You know what, I’d rather have a burger” response. Responses with these automatic replies are also comparably brief, and do not include custom embellishments, such as a 😊 or 🍕 emoji. Hence, our user in this scenario would be giving up some expressiveness in order to not have to enter a message on her own.

If none of these responses are to the liking of the user, she can instead opt to (1) send a graphical emoticon, or (2) dictate a message. Emoticons are selected from four categories: smileys, hearts, hand gestures, and emoji. In the three former, users can turn the watch’s crown to change between emoticon states. They can, e.g., switch between a smiley with its tongue out and a smiley with hearts for eyes. Emoticons from these three categories can also be subtly animated (e.g., showing little hearts floating around the big heart) or change their color. In the fourth category, emoji are selectable from a very long list. When dictating a response, it can then be send either as a voice message or as a transcribed text message. This does allow for custom replies, but also requires users to define the whole message and thus demands more effort from them than picking from canned replies.

The plethora of response options on the Apple Watch serves two main purposes: (1) allowing for more playful/personal responses (scribbles, heart beat, emoticons), (2) allowing communication with a range of different engagement levels. While there is a connection to the informal aspects, the second point is of primary interest to this dissertation. In the Apple Watch these methods are a product of the form factor's constraints. Not willing or able to fit a full keyboard on the watch, Apple had to design a range of replacement techniques. Speech input here fulfills the role of the full control input method (users can enter any text). However, as speech input can be cumbersome or awkward for users, the other techniques can be used in situations where it is not the good choice (e.g., on a busy train or during a loud concert). But even if speech input is a feasible option, users might not always *want* to dictate a custom reply. For many messages a simple “yes” or “no” answer can be sufficient and a canned reply can thus be an appropriate choice.

6.3 Casual Messaging

As we have seen with the Apple Watch, casual communication enables users to respond in ways that are less formal, but also ones that require less effort or precision. However, messaging around haptic and sketching interactions is comparably more exotic and not available on the majority of devices used for communicating. For the remainder of the chapter, we will thus move to a more ubiquitous communication channel: text and images as used in messaging. As we will see, text also offers means for more casual expression which users can opt to use in less formal situations. This is in addition to just writing in a more informal or formal style. But apart of enabling informality (and thus aligning with the respective definition of *casual*), such methods play into questions of control as well. As we saw in the Apple Watch example, enabling responses with, e.g., just emoticons or emoji, serves as a lower control side channel for situations where users do not want or are not able to type out a full message.

Text input is required for the messaging and social networking applications that are the most used class of applications on mobile devices [48]. This shows that connecting to others and expressing themselves are important aspects of users' computing experience. As such, text input has been a research focus for many years and researchers, e.g., have designed a large number of input methods to optimize text entry speeds (for a comparison see, e.g., [163]). But text input is not necessarily restricted to actual text. Instead of using characters to compose words, they can also be repurposed in *emoticons*, such as :), or <3. Here, characters are put together in a way that disposes of their actual meaning and makes use of their *look* to assemble larger shapes. Hence, the colon turns into a set of eyes and the parenthesis becomes a mouth.

In the Western world, emoticons traditionally mostly use a small number of punctuation characters. But non-Western writing systems offer many characters that enable more complex and expressive emoticons, such as `^-_ (ヾ) _/-`. While this potentially allows users to assemble intricate messages, this is not straightforward on current mobile phone keyboards which do not generally provide access to arbitrary Unicode characters. Emoticons have filled a need for a more casual and playful form of communication (e.g., adding a ;) to a text), but can sometimes be hard to use, especially on mobiles.

As Janssen et al. have shown, emoticon use increases *perceived intimacy* between chat partners [137]. This validates suitability for expression beyond just text and is likely what has also been driving adoption and use of emoji. Where emoticons assemble “pictures” from characters, with emoji, each character is itself pictographic. Instead of sending :D, users can then send 😊. While still just text, this character is usually rendered as a colorful visual icon. Emoji not just allow for expression of many emotional states (e.g., 😊, 😞, or 😱), but also enable users to decorate messages (e.g., 🌟, 🎯, or 💎), or replace words with visual stand-ins (e.g., 🍷, 🐱, or 🚗). Using a visual icon instead of a word enables users to introduce ambiguity and playfulness where they see fit.

6.4 Introduction to Emoji

So far we have looked at general patterns in casual communication. We have also identified emoji as one way for users to communicate in a more relaxed and casual manner. We now take a closer look at emoji, and in particular how to improve entry of emoji. By supporting emoji entry, we can aid users’ ability to express themselves. Apart from more casual communication, this is also a useful means for a more playful and visual type of messaging.

Emoji have been growing both in popularity and number over the recent years. As shown in Figure 6.4, new emoji continue to be introduced⁴. Current mobile devices all make a large number of these characters available for users to enter with dedicated emoji keyboards. This does enable a wide range of expressions for users and allows them to communicate many moods or situations without typing out a message. For example, sending a 😴 instead of describing that they are tired. As we will see soon, these possibilities of expression are currently limited by the available emoji keyboards, which make it too cumbersome to enter emoji. Apart from investigating emoji in general, we thus also look at an input method that makes it easier (i.e., require less effort) for users to enter them.

⁴For a full list, see <http://www.unicode.org/emoji/charts/full-emoji-list.html>

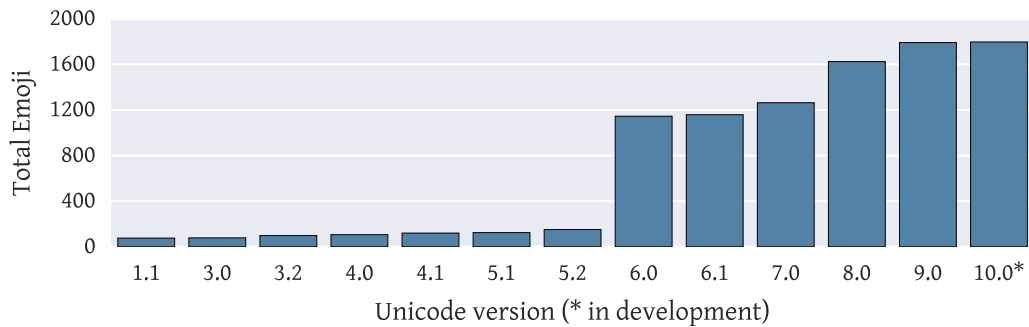


Figure 6.4: Emoji were first specified in version 6.0 of the Unicode standard (with some characters retroactively promoted to emoji), in October 2012. Since then their number has continuously grown and more emoji are added with every new version. For a list of included emoji per version see <http://www.unicode.org/emoji/charts/emoji-versions.html>.

6.4.1 The History of Emoji

In the late 90s, the first emoji were created in Japan for *NTT DoCoMo*. Emoji allowed sending small pictograms to other phones by only transmitting two bytes—the corresponding character code⁵. The Japanese origin of emoji still manifests itself in emoji such as 🏠 (Japanese post office), 🇯🇵 (outline of Japan), 🎏 (*Koinobori* wind socks—flown during *Children’s Day* celebrations in Japan), or 🎍 (*Kadomatsu* decoration used for Japanese New Year). Other Japanese carriers followed suit and several partially overlapping sets of characters were in use for a while. However, this situation posed problems when designing for interoperability beyond Japan, e.g., with email systems.

To resolve this situation, the Unicode consortium in 2010 standardized 722 emoji in version 6.0 of the Unicode standard, while also promoting many earlier characters⁶ to the status of emoji as well. In addition to symbols from the Japanese carriers, this included characters from *Zapf Dingbats* (e.g., ♠, or 🍷), *Microsoft’s Wingdings* font (e.g., 🐾, or 🏠), and Japanese TV symbols (*Association of Radio Industries and Businesses* [ARIB] set, e.g., 📺, or 📺). In fact, many symbols were part of multiple original sources and were merged to one Unicode code point. With Unicode standardization, interoperability between systems was secured—a necessary prerequisite for the rise of emoji to broad popularity. Unicode allows proposals for additional emoji and hence the set has grown over the years. For example, the 9.0 version of Unicode introduced emoji such as 🤳 (*Selfie*), 🦆 (*Duck*), and 🚣 (*Canoe*).

⁵In fact, this was not completely unified and there were different encodings in use by different carriers. See, e.g., the background information section of UTC document L2/08-081: <http://www.unicode.org/L2/L2008/08081-emoji-wd.html>.

⁶For an overview of emoji sources see <http://www.unicode.org/emoji/charts/emoji-versions-sources.html>

How emoji use has risen over time is difficult to show in general. However, data on emoji uptake is available from Instagram and Twitter. While not an account of overall use in texts, this data is representative for social networking and thus much of general use. Instagram saw a sharp rise in emoji usage from 0 % of texts using emoji to 20 % of them within less than half a year of the introduction of the iOS emoji keyboard⁷. Currently, about 40 % of Instagram messages contain emoji, and this number is even higher in some markets (e.g., more than 60 % in Finland). Twitter has reported data on the usage of emoji in TV-related tweets between April 2014 (when emoji were introduced on Twitter) and July 2015⁸. In that timeframe, the share of TV-related tweets containing emoji grew from 9.8 % to 14 %. This percentage highly varies by genre and, e.g., up to 22 % of tweets on music programming contain emoji. Twitter also found that younger and female users are more likely to include emoji in their tweets—hinting at why tweets on sport talk shows are least likely to include emoji (only 4 % do). Of course, uptake of emoji in social media and messaging is not representative of overall frequency of emoji in other forms of writing. Likely, much fewer emoji make their way into essays, annual reports, or news articles. Yet, personal communication is an important area and, as shown above, supporting emoji use here is an aspect of growing importance.

Growth of emoji popularity also shows in how much attention they receive. For example, *Oxford Dictionaries* prominently made an emoji, “😄”, their *Oxford Dictionaries Word of the Year 2015*⁹. While emoji allow users more visual expression in their messaging, that same quality is also attracting advertisers. One mobile marketing company, e.g., saw an 777 % increase of emoji usage in campaigns running on their platform¹⁰.

As we already saw, the Apple Watch also included emoji as one of their casual communication options, along other emoticons and speech input. Overall, emoji have seen strong adoption and popularity. They added a powerful graphical component to texts and are hence very suitable for informal communication. This suitability is also indicated by where emoji have seen their strongest response: in social media and messaging (however, emoji have made some inroads in the business world¹¹). We further explore the specific characteristics of emoji in the following sections.

⁷<http://instagram-engineering.tumblr.com/post/117889701472/emojineering-part-1-machine-learning-for-emoji>

⁸<https://blog.twitter.com/2015/emoji-usage-in-tv-conversation>

⁹<http://blog.oxforddictionaries.com/2015/11/word-of-the-year-2015-emoji/>

¹⁰<http://venturebeat.com/2016/03/24/marketers-might-over-doing-it-with-all-of-the-emojis/>

¹¹<http://www.theatlantic.com/business/archive/2015/05/why-emoji-are-suddenly-acceptable-at-work/393191/>

6.4.2 The Nature of Emoji

Though in some instances they supplant words entirely, they also open up new vistas of exchange and creativity. They are in a sense, the words that got away, and then returned. As smiles and frowns and jetliners. — Lisa Lebduska [171]

What makes emoji special as a means of adding visuals to texts is that they *are* text. Instead of sending images of smileys or airplanes, characters representing them are transmitted (they form a logographic writing system, i.e., each character represents a word or phrase). Hence, in contrast to images, they can be used in places such as URLs, email subjects, usernames, or passwords. The Unicode standardization only defines a mapping between a character code and an abstract emoji description. It is up to individual platforms to provide fonts that render the individual emoji as a graphical representation (note that some emoji, such as **!!**, can also optionally be rendered as text: **!!**). Hence, emoji can look different on different systems and even between versions of the same system. This can be problematic where graphical representations strongly differ, a problem just recently investigated by Miller et al. [198]. Table 6.1 shows several such examples where emoji vary so much in appearance that there could be misunderstandings between users of different platforms.

But the fact that they are text also allowed them to spread at the speed that they did. After all: emoji are not the first instance of visually augmented texts. Instant messengers like *Yahoo! Messenger* have for a long time supported inline smileys. Similarly, common forum software, like *phpBB*, have their own smileys¹². With no standard encoding for those smileys though, there was no interoperability. Forwarding a forum post via instant message would mean losing the embedded smileys. Emoji, however, now allow copying text with smileys freely between systems. This does not necessarily mean the emoji will show up on each system, though (users browsing the web on an older phone without emoji support would, e.g., just see empty boxes).

The textual nature of emoji, in a sense, allows them to “*sneak into*” applications. If an application renders text through an API like Microsoft’s *DirectWrite*¹³, emoji characters are automatically rendered correctly (if supported on the specific platform). Thus, while supporting custom smileys adds additional work, applications that can handle text essentially get emoji capabilities for free. In this case, no app-specific input method needs to be designed, but emoji entry is done via the system’s keyboard.

¹²https://www.phpbb.com/support/docs/en/3.1/ug/userguide/posting_smilies/

¹³[https://msdn.microsoft.com/en-us/library/windows/desktop/hh802480\(v=vs.85\).aspx](https://msdn.microsoft.com/en-us/library/windows/desktop/hh802480(v=vs.85).aspx)

Table 6.1: While most emoji look similar on all major platforms, they each have their own unique style. Some emoji, as shown here, differ considerably between platforms which could lead to unintended interpretations in cross-platform messaging. For example, some vendors chose to represent the *nail polish* emoji not as the *object*, but instead as the *action* of applying it. This could introduce misunderstandings in messages such as “Went to get some 🖍️.” More severe inconsistencies are, e.g., exhibited by some vendors representing *alien monsters* with the classic video game sprite 🐉, or showing two females dancing 🕺 where the standard calls for one “*woman with bunny ears*.” But vendors also use the chance to leave their personal mark. The mobile phone emoji, for example, shows phone designs by the respective companies. Note that we show two versions of the Microsoft emoji to highlight changes even within one vendor’s designs.

Official name	Apple	Google	Microsoft	Twitter	Facebook	Samsung	LG
Cyclone							
Flushed face							
Nail polish							
Vibration mode							
Woman with bunny ears							
Alien monster							
Mobile Phone							

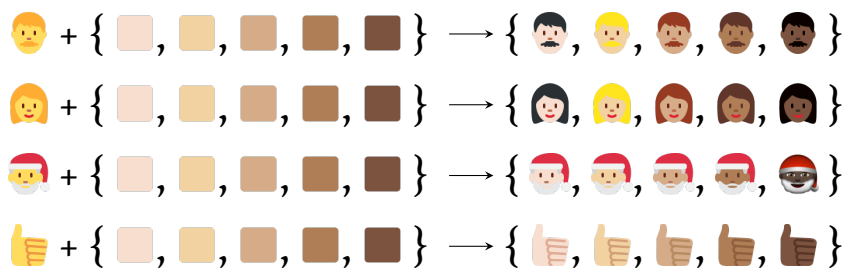


Figure 6.5: Since Unicode Version 8.0, the standard specifies that emoji showing people should have a generic color (such as yellow, blue, or gray). Those emoji can then be combined with one of five skin tone modifiers to produce a more diverse set of emoji.

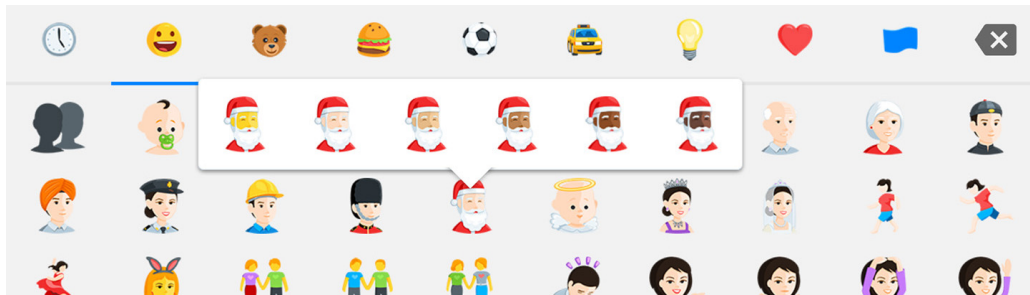


Figure 6.6: To accommodate the large number of skin tone variant emoji, emoji keyboards (shown here is the *Facebook Messenger* keyboard) have changed to pop up selection menus for corresponding emoji when touched for a short while. Users can then pick a skin color from the overlaid menu.

6.4.3 A Combinatorial Explosion of Emoji

While many emoji are defined as a 1 : 1 character code to pictogram relations, others break this pattern. For example, flag emoji are not each encoded as distinct characters. Instead, regional indicator letters are combined to spell out country codes which are then supposed to be rendered as flags: **U** + **S** ⇒ 🇺🇸. This approach makes the standard more flexible, as no list of flags needs to be updated with changing geopolitics. Which codes define a flag emoji is delegated to *ISO 3166*, specifically the 2-letter codes defined in *ISO 3166-1 alpha-2*. There are currently 249 officially assigned country codes, yet support varies a lot between devices. While 🇨🇳, 🇩🇪, 🇪🇸, 🇫🇷, 🇬🇧, 🇮🇹, 🇯🇵, 🇰🇷, 🇷🇺, and 🇺🇸 are commonly supported, some systems show no flags at all (e.g., Windows Phones), while others support many more (e.g., current versions of iOS). This includes non-country flags (e.g., 🇪🇺) and proposed regional flags¹⁴ (such as a flag for Wales: 🇬🇸). The large number of flags results in long blocks of visually similar and thus hard to distinguish emoji.

In an approach similar to regional indicator letters, Unicode version 8.0 brought the introduction of skin tone modifiers. While the emoji standard did initially not specify skin color, most platforms rendered people emoji (such as 🧑) with white skin color. Skin tone modifiers now allow (depending on platform availability) changing the appearance of emoji to one of five levels (Type I and Type II are combined in one level) on the Fitzpatrick scale [78] (see Figure 6.5). Along with this change, all major platforms have moved to neutral color (e.g., yellow) people emoji when no modifier is used (such as 🧑). This inclusion of skin tone brought a large increase in available emoji. However, instead of including those emoji directly in the overall emoji list (as with flags), all current platforms chose to make skin tone selection a separate interaction (hold down on emoji and select variant from popup, see Figure 6.6). While this somewhat limits growth of the emoji list, it also makes skin tone variants less discoverable and take more time to access.

¹⁴<http://www.unicode.org/review/pri299/pri299-additional-flags-background.html>

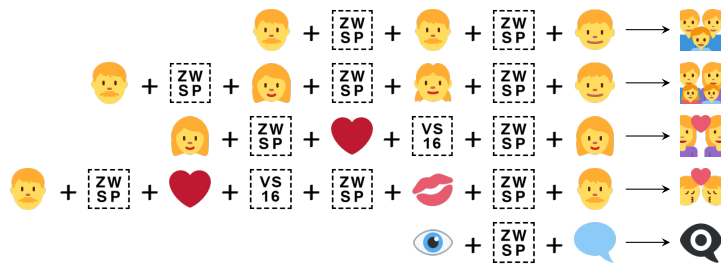


Figure 6.7: Some emoji can be combined into groupings. Joining multiple person emoji with ZW SP (U+200D: zero width joiner), e.g., results in family emoji. The same approach is used to generate couple and kissing emoji for all gender pairings. Here a VS 16 (U+FE0F: variation selector-16) character is used to force emoji style for the heart character (hearts can also be rendered as monochrome text). All this can potentially be combined with skin tone modifiers. This mechanism has also been exploited to create non-standard emoji. The last row, e.g., shows an emoji Apple included for the “I Am A Witness” anti-bullying campaign¹⁵.

In addition to skin tone variants, Unicode also allows assembling couple and family emoji to, e.g., send an emoji of a gay couple with two daughters (see Figure 6.7). This can be combined with skin tone modifiers to represent interracial families. While Unicode thus allows to specify arbitrarily complex family groupings, there is no guarantee these characters would be rendered as one emoji. Those changes to the standard brought some diversity to emoji, but also further increased the number of emoji. The general approach—to define emoji as a combination of several other characters—however, is recurring in the standard (e.g., **5** is a combination of the code points for 5 and that of a box). While this allows for many combinations, it also means the number of emoji to choose from is potentially very large.

6.4.4 Common Usage of Emoji

As we have seen, emoji use in textual communication is growing. Yet, at the same time, the number of emoji itself is expanding rapidly through inclusion of additional symbols, but also through the introduction of emoji modifiers. What is available to users thus is currently a moving target. It remains to be seen how adoption and user behavior change once the larger set of emoji sees wider availability. However, we can make some observations on how emoji are currently used, based on the many messages we inspected during work on this chapter (for a more quantitative view, see Section 6.5). Generally, we found emoji use to fall into one of five different patterns.

¹⁵<http://www.wired.com/2015/10/i-am-a-witness-emoji-ios-9/>

These five emoji usage patterns were:

Decorative use Here emoji are used as a sort of flourish, or decoration, for accompanying text, yet are not an integral part of it. For example, emoji can be used when congratulating: “Happy birthday! 🎂🎁”.

Stand-in use Here an emoji replaces an actual word, such as in: “Out for a 🚶”.

Emotional use Instead of just decorating a message, emoji can be used to change the tone or meaning of a message. One example is sarcasm, such as in: “Sure, go ahead 😏”. Another example is communicating feelings about something, such as in: “Got my test results 😊”, which would be a very different message when ending with a 😞.

Reaction use Here the emoji stands on its own and communicates a direct reaction to a previous statement, such as: “👍” (“alright”). This is mostly used in chat conversations.

Stand-alone use This is a generalization of *reaction use* for messages that contain only emoji. This presumes either familiarity of the recipient with this kind of use (similar to use of texting abbreviations) and/or context. In mid-December, a user might, e.g., send: “😓🎄🎅🎁” (~“I’m stressed out by Christmas shopping”).

A more in-depth investigation of emoji use was conducted by Cramer et al. [60]. In a crowdsourced study, they asked 276 participants for their last text message with emoji. From the 228 valid responses, they derive emoji distribution statistics, but also looked at the “intended function” of the used emoji. They identified three major categories: (1) providing additional emotional or situational information, (2) changing meaning through tone modification, and (3) building and maintaining engagement and relationships. They also point out linguistic and placement categories of emoji (e.g., emoji as text replacement), which is very similar to the usage patterns we found.

6.5 Quantifying Emoji Usage

So far, we have given a general overview on emoji and their use. However, ultimately we would like to build models for emoji and use computational methods to work with them. We thus set out to collect a large amount of real world data on emoji and how they are used. An ideal candidate would be instant messaging logs, but those pose privacy problems and are also not generally available. Instead, we turned to publicly available data and collected a large number of tweets containing emoji. Using data from Twitter has several advantages over other data sources: (1) it is available via an easy-to-access API, (2) in contrast to forums, which usually concentrate on one topic, tweets cover a much wider range of topics and use, from casual communication between friends, to curated marketing by social media experts, and (3) there is a large amount of data, with roughly several thousand tweets sent out per second.

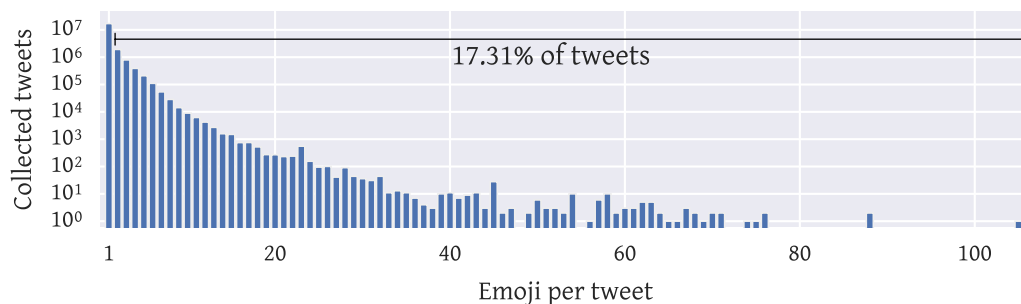


Figure 6.8: In our collected tweets (excluding spam) about 83 % (~17 million) only contain one emoji (we do not collect tweets without emoji). At the other extreme, we scraped one tweet containing 105 emoji.

As the amount of emoji in use is constantly changing (device updates bring new emoji), we decided to limit our data collection to a well defined set of emoji. We hence only collected tweets containing one of the 845 “level 1” emoji, i.e., those that are “commonly supported as emoji by vendors at present”¹⁶ (note that this definition has just recently been removed from the report). This is a subset of the emoji specified by the Unicode 7.0 standard and thus an older and more established set of emoji than, e.g., those specified in Unicode 9.0. While we limit ourselves to this well-defined subset, there are no technical roadblocks to extending this methodology to more emoji. However, including very new emoji could introduce a bias as only very few devices support them and their use would thus mostly be restricted to early adopters.

In total, we collected almost 21 million tweets over a period of about 29 days in July/August 2015 via the Twitter public streaming API¹⁷. To limit the number of tweets, we only collected tweets for 3 minutes at a time, or until 10000 tweets were gathered—whichever came first. We use Twitter’s keyword filtering to only gather tweets that contain emoji. However, as Twitter only allows specification of up to 400 keywords per request (each emoji being one keyword), we had to split the set of emoji into three equal-sized blocks. For each run of the scraper, we randomized block order and then scraped each of these blocks in sequence. A new scraping run was started every 15 minutes, starting at five past the hour. Hence, we had a large number of individual scraping sessions (96 per day), which were limited in duration though. At most, this would allow for 2,880,000 collected tweets per day, yet we collected just slightly more than 720,000 tweets on average. Ideally, one would want all tweets¹⁸. However, this requires special permissions and also resources set aside to handle the amount of incoming tweets¹⁹.

¹⁶http://www.unicode.org/reports/tr51/#def_level1_emoji

¹⁷<https://dev.twitter.com/streaming/public>

¹⁸Via the Twitter “firehose” endpoint <https://dev.twitter.com/streaming/firehose>

¹⁹About 6000 tweets are send out per second <http://www.internetlivestats.com/twitter-statistics/>

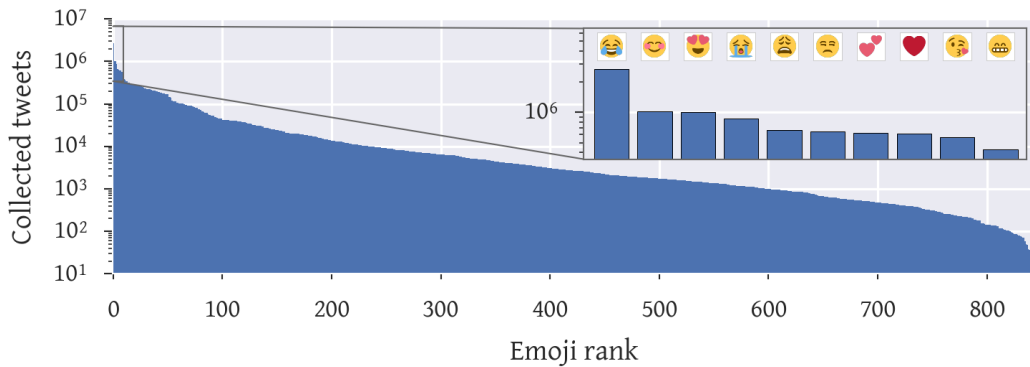


Figure 6.9: Emoji frequency in our collected tweets (excluding spam) follows a power-law distribution. While we saw over 2.6 million instances of 😂, we only collected 27 instances for each of 🍷 and 🏠.

For each tweet we only retain some data (primarily id, text, date, and username) and save it to a database. We only collect tweets in English (as identified by Twitter) and reject retweets (i.e., all tweets starting with “RT @”). While most tweets we collected appear genuine we noticed some spam. We define a tweet as spam if the tweet (or slight variations of it) reappear a large number of times. Such tweets often contain a running number but otherwise repeat the same text as earlier tweets. We set a conservative threshold of labeling tweets as spam if we can find more than 200 matching other tweets (we do not remove very short tweets such as “Goodnight 🌙”). For example, we collected 1336 tweets like:

[#] @justinbieber HEY JUSTIN! 🙏 PLEASE DO NOT IGNORE THIS! It is very important that you read. 🙏👉 [URL]

While the share of such tweets of the overall dataset is less than 2 %, we do remove them from further analysis. For many emoji this does not make a difference, but for a small subset of emoji, those tweets do skew the data as there is not a large number of tweets containing those emoji to begin with.

After initial tweet collection, we had 243 emoji occur less than 1000 times. As we intend to reason about the context of emoji, having only few samples per emoji would limit this capability. To gather more data for those emoji, we collected an additional 106,618 supplemental tweets (including 3083 we later removed as spam) between early August and mid September 2015. We dropped emoji from this scraping once we had reached 1000 samples for them. While this ran for almost 6 weeks, we were not able to gather 1000 samples for the 76 least frequently used emoji. However, our least represented emoji, 🍷, now has 211 samples (compared to 27 samples before supplemental scraping). Our final dataset then contains 20.6 million tweets.

We used the opportunity to investigate the gender distribution of users tweeting with emoji. For this, we randomly sampled 1000 usernames (not twitter handles, which are a unique user identifier). Note that this assumes people use their real name (or at least first name) on Twitter (this is not enforced). We parsed this data to extract first names and fed those names to the genderize.io²⁰ service, which, based on a database of more than 200,000 distinct names, tries to assign gender. In our sample, 27 % of the names were labeled as male and 34 % as female. However, 39 % of names (such as, *labrena*, *Utter Amateur*, and *Minnion*) could not be assigned a gender. While this hints at a larger share of females using emoji (as indicated in previous research), more research is necessary to confirm.

We also took a closer look at usage patterns in the collected tweets by randomly sampling 200 tweets and hand-annotating them as using emoji either *decoratively*, as *stand-in*, for *emotional* expression, or *stand-alone*. We found some overlap between categories (some tweets, e.g., make use of both decorative and emotional emoji). However, with 66 % of tweets, emoji were used to add emotional expression. While 34 % of tweets made use of emoji as decoration, only 8 % of tweets used emoji to replace a word. Surprisingly, we found no tweets containing only emoji and no text in our dataset. We discovered this is due to the Twitter API filtering out such tweets. For example, we would receive the tweet “Cute 💎” but not the tweets “💎” or “💎💎”. Unfortunately, this behavior is undocumented and we can only speculate why we observe this. For our use of the Twitter data this is not a large problem, as single emoji tweets do not provide context information. However, we do miss out on tweets with multiple emoji (yet no Latin characters). We believe our dataset is sufficiently large to include a sufficient amount of context data anyway.

From our initial, unbiased, tweet sample (excluding supplemental tweets), we can estimate how many and which emoji are used. We observed that 83 % of tweets only contain one emoji (see Figure 6.8). Hence it is necessary to draw on the surrounding word context in establishing emoji similarity across tweets. Considering only tweets in which multiple emoji occur side-by-side would severely limit the usable data. While there are some outliers with larger numbers of emoji, these are negligible overall. We also see a small set of emoji dominating in actual use (see Figure 6.9). The top ten emoji together appear about 9 million times, while the bottom ten only have 379 occurrences overall. This imbalance might be because some emoji are just not very appealing, or because they are too hard to discover. Emoji keyboards could take this skewed distribution of emoji into account and place emphasis on supporting common emoji especially well. Yet this should not be read as a reason to omit some emoji from the keyboard. The set of used emoji likely varies by person, situation, topic, or chat partner. Exclusion from a keyboard should thus not just be based on frequency of use. The relative frequency of the ‘z’ character in English texts, e.g., is only 0.096 %²¹, yet all English keyboards include it.

²⁰<https://genderize.io/>

²¹Computed from the Brown corpus [166]

Comparative data on emoji frequency and use is, e.g., available from the *SwiftKey Emoji Report*²². Their analysis is based on typing data gathered between October 2014 and January 2015 via their iOS/Android keyboard. For example, they note that 44.8 % of emoji use is only *happy faces*, something we also see in our data. They also find a large amount of differences according to users' location. Russian users, e.g., are twice as likely to use the ❄️ emoji compared to the average user. On the other hand, Australian users are 66 % more likely to use ☀️. We also compared our emoji ranking to the one of *emojitracker*²³ and found a strong correlation; Spearman's rank correlation coefficient $r_s = 0.95, p < 0.0001$. This shows that emoji use in our sample aligns well with overall emoji use (*emojitracker* has been running for much longer than our scraper).

Where we have looked at general emoji statistics, our data could also be used for sentiment analysis. This has been explored by Suttles and Ide [289], but also by Vidal et al., who focused on food-related tweets [303]. Instead of assessing the sentiment of a complete message, Novak et al. try to quantify the sentiment of individual emoji and other symbols [213]. They also find that more frequently used emoji have significantly more positive sentiment than less frequently used ones.

6.6 Evaluating the State of the Art of Emoji Entry

As we have seen, use of emoji has grown rapidly over the last couple of years. However, design of emoji keyboards has so far mostly stuck with long lists of emoji. The sheer number of emoji has made it impossible to show them all at once at a selectable size—a problem shared with other scripts containing many characters. While we will later look at other approaches to entering characters from such large volume scripts, here we first take a look at the common approach. With lists of emoji being used everywhere, an analysis of such keyboards aids in identifying problems and setting a baseline for any future improvements to emoji entry.

In this section, we will investigate how the default Google keyboard on a Nexus 5, running Android 5.1.1, fares when entering emoji. The version of the Google keyboard we tested offers 822 emoji (shown in Figure 6.10), split into 5 categories. All emoji are arranged in grids and span multiple pages, which users can swipe through horizontally. Each page is associated with a category and selecting a category jumps to a page belonging to it. However, users can also transition between categories by continuing swiping on the last page of a category. This is different than many other emoji keyboards where emoji in each category are shown in a list of their own and no continuous swipe-through is possible. Thus, the Google keyboard actually has an advantage when used for exploration where users want to quickly scroll through all emoji.

²²<https://blog.swiftkey.com/americans-love-skulls-brazilians-love-cats-swiftkey-emoji-meanings-report/>

²³A site tracking emoji use on Twitter since July 2013: <http://www.emojitracker.com/>

Faces



Objects



Nature



Places



Symbols



Figure 6.10: An overview of all emoji available in the Google keyboard on the Nexus 5 (running Android 5.1.1) and thus included in our evaluation. The emoji are shown in the same category and order as they appear on the keyboard.

The Google keyboard uses two mechanisms to facilitate entry of common emoji: (1) it maintains a list of recently used emoji, and (2) remembers the last used page per category. If users only enter emoji from a small number of pages, this approach often presents them with the target page when opening the keyboard or switching categories. However, this mechanism can be disorienting when users jump to the middle of a category on category selection, but are unclear whether the emoji they are looking for is in a previous or following page. The worst case occurs, when the emoji is on the first page of the category, yet after jumping to the middle users search for it in the other direction. Now users traverse half the pages to reach the wrong end and then need to backtrack all the pages of the category to find the actual emoji. In this situation, always jumping to the start of a category (as in most other keyboards) would have probably fared better. This issue likely resolves though, once users have a better mental model of emoji ordering.

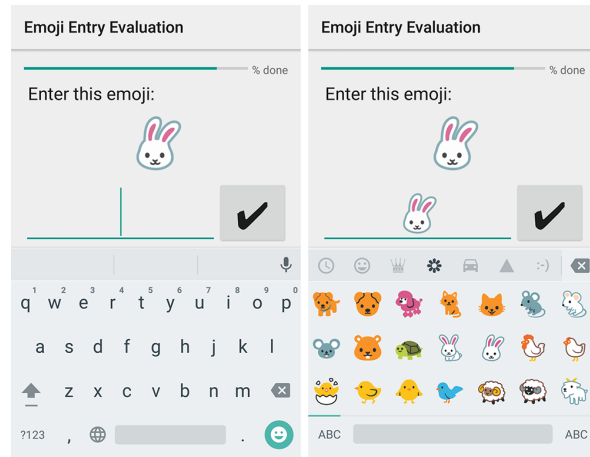
Each page of the Google emoji keyboard shows 7 columns by 3 rows of emoji. This is a common arrangement, e.g., also found in the Samsung keyboard on a Galaxy S4 and the WhatsApp keyboard on a Lumia 920. Many newer devices even show 4 rows of emoji or more. The number of usual emoji per screen thus is roughly between 18 and 50 (21 in the case of the Google keyboard).

6.6.1 Challenges When Testing Emoji Keyboards

Evaluating emoji keyboards comes with several challenges. With the set of possible entries commonly as big as 845 emoji (level 1 emoji), or even bigger (e.g., all Unicode 9.0 emoji), exhaustively testing the entry of every emoji in a lab-setting is prohibitively costly. However, only testing some, e.g., the most common ones, can bias the results. Drawing a limited random test set of emoji is one way to approach this. But even then, the question of how much coverage of the full set is needed for representative results remains open.

The bigger challenge, however, is picking an appropriate testing procedure. If one were interested in natural user behavior, a chat study where two participants exchange messages (such as in [100]) would be an appropriate choice. This could also be designed as a longitudinal study that monitors users' chatting behavior (such as in [296]). However, this approach risks that only a small number of emoji are actually typed and does not generate a lot of data as much of the time is spent not entering emoji. A longitudinal study also raises privacy concerns, as recording typed emoji can capture user mood. Larger amounts of data can be generated by deploying prototypes to an app store (such as in [20]). However, this still would not give control over the share of emoji in the data. One way around this is to find a game format that allows control of the task, while still engaging users, such as in [112]. Most commonly, though, text entry methods are tested with a task where participants have to copy text verbatim. That is also the approach we chose for our investigation, as this allows for control of which emoji are to be typed.

Figure 6.11: Layout of our evaluation application while testing the Google keyboard. Participants are shown the emoji to enter at the top of the app. In each trial, they need to activate emoji mode, find the respective emoji and click the commit button. Committing a selection also switches the keyboard back to QW-ERTY mode.



We adopt an approach where we test with an emoji test set sampled from Twitter. To generate this test set, we scrape an additional 10,000 tweets containing emoji present on the Google keyboard (822 different emoji). From the scraped tweets we only keep the emoji and store how often they occurred. This set includes 502 different emoji (as some emoji never occurred in the tweets scraped for testing). During testing, we sample emoji with replacement from this dataset. The dataset is hence heavily skewed towards the most common emoji. Thus, in order to broaden the range of emoji occurring during the evaluation, we log transform the emoji frequencies to boost the likelihood of rarer emoji appearing. When computing final keyboard performance we reverse that transformation. Thus, while we favor rarer emoji during selection, we make sure they only influence scores per their original likelihood.

6.6.2 Participants

For the study, we recruited 12 participants (3 female, age 21–41, $\bar{x} = 27.4$, $SD = 6.3$) from around our institution. The study took ≈ 30 minutes and after completion, participants received a small non-monetary gratuity. All participants owned a smartphone, however none of the participants owned a Nexus 5 as used in the study. Only two participants stated they had the same emoji design on their phone. Most participants had phones by vendors with custom UIs (e.g., the Samsung keyboard), or with older Android versions (which used a different design). However, while they were not intimately familiar with the Google one, default keyboards do not differ much currently. We asked participants to indicate whether they often use emoji on their phones on a 5-point Likert scale. While one participant strongly disagreed, four agreed, three strongly agreed, and four did not lean either way.

6.6.3 Procedure

Before the study started, participants could try out the Google keyboard. In this phase, they were only given emoji to enter that were not part of the test set. While this prevents participants searching for emoji which are later to be tested, they still gain an initial overview of the keyboard. We chose the training emoji such that they *reset* the Google keyboard's category state. As mentioned earlier, the keyboard remembers the last entered emoji for each category and jumps back to that position when reopened. By always having an emoji from the first page of each category last in the training phase, all category jump targets are reset to the respective first page of the category. While this ensures a consistent starting state for all participants, note that once they entered emoji in the testing phase, the category jump targets differ between them.

After entering 10 emoji we considered a participant to be sufficiently familiar with the interface and move on to the main study. The interface used here (see Figure 6.11) is the same as in the training phase, but shows emoji from the test set. If participants took more than one minute to find an emoji in this phase, we aborted the trial as pilots showed that very long search times frustrated participants.

During emoji entry, we draw emoji to enter by sampling with replacement from the log-transformed emoji set. Overall, 100 emoji are drawn from this set, resulting in 100 trials per participant. We only did 100 trials, as this allowed testing in a reasonable amount of time, after piloting indicated long trial times. In each trial, we start taking the time once the emoji keyboard is activated via the emoji button in the lower right of the Google keyboard. This allows participants to take a look at the target emoji for as long as they want before starting a trial. During a trial, we record any page transition, activated either by swiping left or right or by selecting one of the categories and jumping to a page. A trial is complete once the user commits the emoji with the button next to the text field. Upon completion of a trial the keyboard resets to the QWERTY view.

6.6.4 Results

As described above, we favor rare emoji in the evaluation to have a broader test set. However, in the analysis, we reverse that log-transform and give weights to emoji equivalent to their observed frequency in the test set. Thus, all time and error results reported here come in a raw and a corrected version, where the corrected version accurately captures expected performance for emoji use in the wild. Note that the difference between the two turns out to be minimal. This indicates that the emoji entry performance is not dependent on how frequently used an emoji is. We will revisit this aspect later, when we relate the results back to our observed emoji usage data.

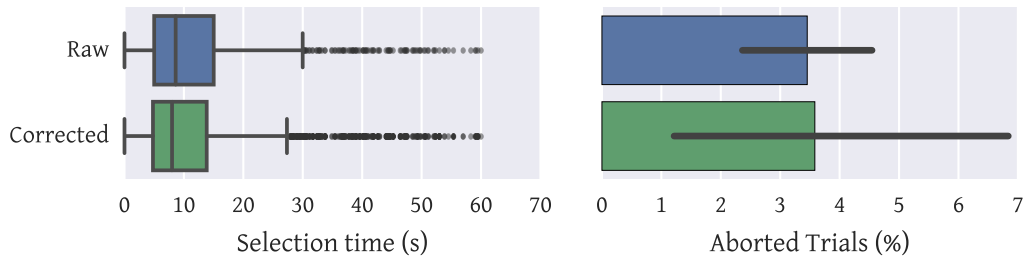


Figure 6.12: This figure shows (left) how fast users were able to enter emoji, and (right) how many trials were aborted because they took longer than one minute (error bars show 95 % confidence intervals). As we log-transform emoji frequency during evaluation, the raw results do not accurately reflect expected performance. We remove this bias towards rare emoji for the corrected values, which only slightly changes the results.

Figure 6.12 shows the outcome of the evaluation with respect to selection time and success rate. The raw selection times over all successful trials was 8.8 s (median) and 12.5 s (mean) respectively. Once we correct for the skewed frequency, this changes to 8.2 s (median) and 11.4 s (mean). If we consider each emoji a word, this performance equals about 5–7 words per minute. Note that those times only take into account successful trials, the average would have been higher, had we not stopped trials after one minute. In our fastest trial, the participant selected 🍷 in 864 ms. This was possible because no page change was necessary for the selection (the emoji was already on the initial page). In fact, 64 % of the 100 fastest trials did not require a page change at all. Correspondingly, there is a strong linear relationship between the number of page transitions and the resulting selection time of a trial; $p < 0.0001$. About 3.2 % of trials took longer than that one minute and were aborted (one participant entered all emoji within a minute, while all other participants exhibited failure rates between 1 % and 6 %). If we take into account the emoji frequency, the expected average failure rate corrects only slightly to 3.3 %. These results show that, for most trials, selection time is within a reasonable range. At the same time, there remains a small number of trials where emoji entry could not be completed within one minute. We will look at those a bit closer below.

To further investigate differences between trials, we compare selection times for all emoji with at least five trials (65 different emoji). This shows a large difference between the fastest and slowest emoji with 😊 entered in an average 4.3 s (5 trials), while 🌞 (also 5 trials) took an average of 35.5 s. In fact, the ten fastest emoji (😄, 😊, 😋, 🍷, 🍕, 🍎, 🍌, 🍓, 🍑, and 🍇) were entered more than 5 times faster than the ten slowest emoji (📷, 😊, 🍌, 🍌, 🍌, 🍌, 🍌, 🍌, 🍌, and 🍌). Interestingly, the faster emoji are in general use more than twice as much (per our sampled frequencies on Twitter) than the slower ones. However, a linear regression on mean selection time and log-transformed emoji frequency shows no significant relationship, $p = 0.30$. Hence, frequency of use is not a good criteria for clearly identifying a subset of emoji that are faster to enter than others.

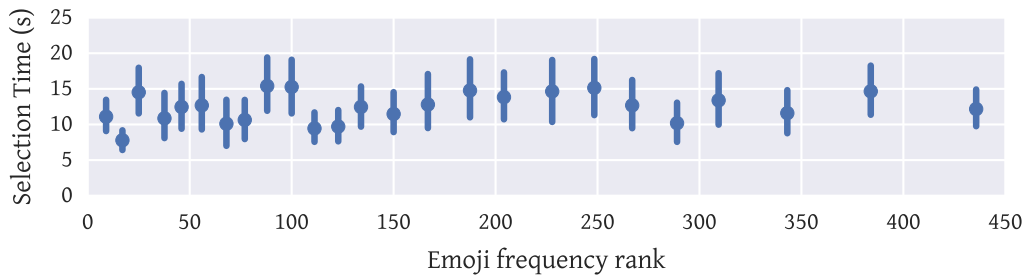


Figure 6.13: Selection time for entered emoji, ordered by rank. Error bars show 95 % confidence intervals for each of the 25 aggregated bins. There is no clear pattern of often-used emoji being selected faster than rarer emoji. Note that this plot only includes data for successfully selected emoji, excluding failed trials.

Failure rate also varies by emoji. For example, no trial of 🦋, 📊, 🚫, 🐱, 🍀, or 📈 was successfully completed. As we randomly sample from a large set of emoji, those six emoji only account for a total of eight trials though. These trials are thus not representative for general performance. For a closer look, we only consider emoji for which we have collected data from at least five trials: a smaller set of 70 emoji. From those 70 emoji, 61 were always entered successfully (385 trials). The remaining 9 emoji (🤔, 😂, 🍃, 🙌, 📖, 🍀, 🎯, 🌟, and 🌊) each only led to one failure (in a total of 58 trials). There is no clear pattern in this set of emoji to suggest that some specific subset of emoji are more likely to fail than others.

One might assume that frequently used emoji would be easier to enter. However, when plotting selection time as a function of emoji rank (see Figure 6.13), no effect of rank is visible. To confirm, we ran an independent-samples t-test, comparing selection time between the 50 lowest ranked (i.e., the emoji is not frequently used) samples and the 50 highest ranked samples. There is no significant difference in the selection times for lower ranked ($M=12.1$ s, $SD=11.3$ s) and higher ranked ($M=10.2$ s, $SD=8.1$ s) emoji; $t(98)=0.95$, $p=0.35$.

Finally, we checked how users progressed towards the target emoji. Figure 6.14 shows an overview of all trajectories towards the target emoji’s page. As their first action after opening the emoji keyboard, most users immediately jump to a different category. This was the case in 640 of the trials (~57 %) and is visible as the large share of category jumps (shown in red) at the left of Figure 6.14. But that initial jump was not always the right one and in 14 % of successful trials participants jumped to a category more than once (up to six times). Overall, though, there was fast progression towards the target page. However, there is a long tail, as some trials took much longer. This is also visible in Figure 6.15, that highlights how the majority of trials already reach the target emoji after about ten page transitions.

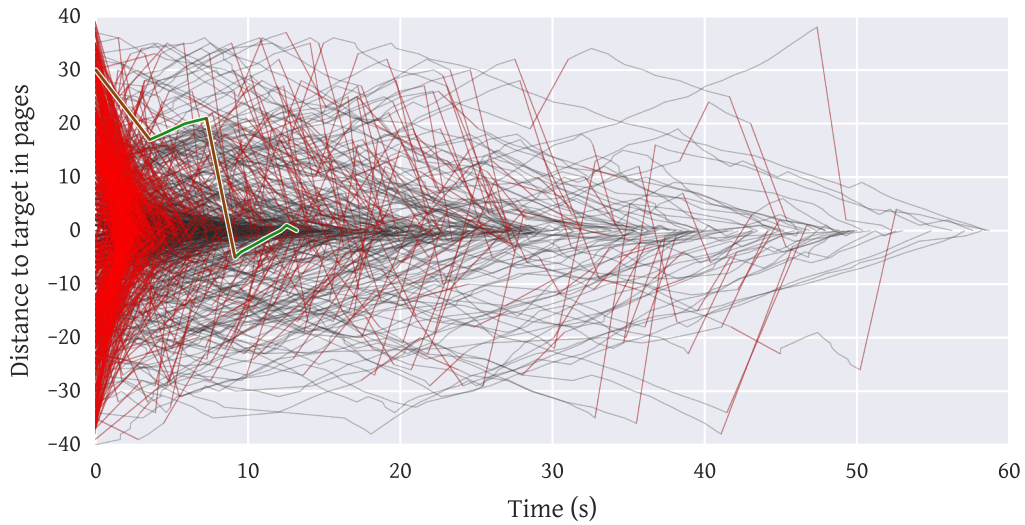


Figure 6.14: Here we show how participants progressed towards the target emoji. In each trial, users start at the last active page of the keyboard. From there, they need to navigate to the target page (here shown in the middle). Ideally, users would jump to the proper category (category jumps are shown in red) and then swipe through some pages (page transitions shown in black) until finding the target emoji. Here, we see that there is indeed a fast approach of the target (visible as the high number of red lines at the left), yet users in many trials get lost. The plot also shows page transitions from one trial highlighted in green. That participant immediately selected a wrong category, searched that category for a while, jumped to the right category, missed the target page while searching within that category, but finally selected the target emoji after about 13 s.

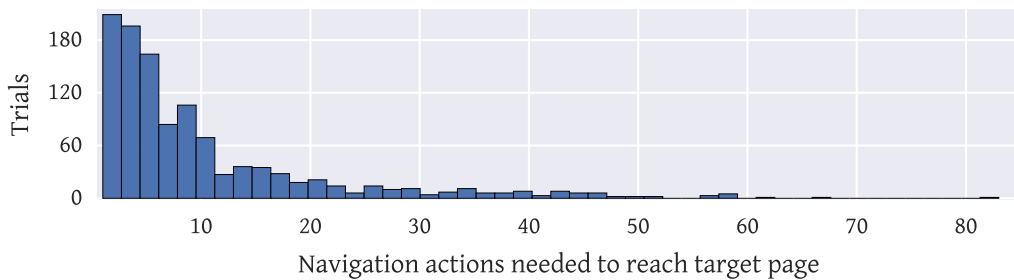


Figure 6.15: Most trials end after about ten navigation actions (category jumps and page swipes). However, there is a long tail where users navigate through many more pages before finally arriving at the desired one.

In 230 trials (~19%) users missed the target emoji. Users would often quickly swipe through the pages, miss the target emoji, and then backtrack to it. One example of this is shown as a highlighted trajectory in Figure 6.14. That participant jumped to and checked two different categories, also overshooting the target page in the process.

We find that in 802 (66.8 %) of the trials, participants used the category buttons to jump to a different page. In the rest of the trials, they either swiped through many pages or were already close to the target emoji. However, we also find that participants often chose the wrong category and had to pick another one. In 191 trials (15.9 %), participants picked a category at least twice. The number of jumps quickly declines though (6.6 % trials contain more than two jumps, 3.3 % more than three, and only 1.2 % more than four). The most extreme trial we recorded had the participant select a different category eight times—rechecking already visited ones as well—and visiting 83 pages in the process.

We can also take a look at the 38 failed trials to see how those searches progressed. Most surprisingly, we find that in 66 % of failed trials, participants actually visited the page with the target emoji, missing it and continuing elsewhere. Participants also visited many more pages in failed trials than in successful ones. On average, 53 pages were visited (50 median) in failed trials, while participants in successful trials visited only 10 pages (6 median). However, the worst recorded failed search spanned a total of 98 page visits, even though the keyboard only has 42 pages.

6.6.5 Discussion

Overall, emoji entry performance with the Google keyboard is adequate. Selection time can be slow, but only becomes very large in a limited number of cases. But in some situations, the Google keyboard exhibits problems. Especially the fact that users miss the emoji 19 % of the time is concerning. While the Google keyboard shows only 21 emoji per page, the emoji keyboard on the iPhone 6+ shows up to 50. A higher density of emoji makes it easier to miss one, or at least increases the time needed to scan a page of emoji. With the number of emoji growing, this search problem is bound to intensify.

Users often pick the wrong category for an emoji. In almost 16 % of the trials, they jumped to a different category more than twice. This indicates that there might be room for improvement in the category assignment, as users' model of emoji location does not always match the actual location. Presently, there is little data that can be used to inform category assignment. Later on in this chapter, we will look at using large amounts of tweets to inform which emoji should be close to each other. In actual use most users would probably not spend the time to find one specific emoji, though. Instead, they would settle on a different one or eschew emoji use altogether. However, as users should be able to enter any text, they should also be able to properly enter any emoji.

The fact that users sometimes failed at finding emoji in a reasonable amount of time also made us wonder about the base cost of using the Google keyboard. In our tested version, the 822 emoji are split over 42 pages. As an animation is shown for each page transition, just visiting a page already takes some time. The overall large number of pages then makes exploration and search of emoji cumbersome. To quantify this, we ran a quick informal study with 7 participants (1 female, age 22-34). We had participants start on the first page of the Google keyboard and asked them just to swipe through all the pages. This already took them about 21 seconds. If additional visual search effort is necessary, it is clear that exploring the available emoji can take a long time.

The current Google emoji keyboard does not favor more common emoji over less common ones. Hence, we could not observe faster selection for more frequently used emoji. Depending on the specific design goals, this can be perfectly acceptable behavior. If we consider all emoji as equally important then we would indeed not want to favor any of them. In fact, favoring the already more popular ones would only reinforce this difference. However, novel emoji designs could decide to bias selection efficiency slightly towards more common emoji. The critical aspect in such an endeavor would be how to find the right balance of bias to support.

Compared to traditional text entry, the 5–7 emoji per minute rate currently achievable with the Google keyboard seems slow. But even compared to other large character set methods, like pinyin (Chinese), emoji entry is slow, even though there are more Chinese characters than emoji. Pinyin can be entered much faster with performance varying between 15 and 35 (~25 on average) characters per minute, depending on the input method used [178]. With pinyin, users still enter Latin characters, yet those numbers show that other approaches to large character set text entry fare much better than the list selection one currently used for emoji.

6.7 Input Methods for Emoji & Large Character Sets

Having taken a closer look at the common list selection method for emoji entry, we now take a step back for a broader perspective. Even for emoji, while list selection is the predominant form of entry, other input methods do exist. But emoji is not the only text entry area where the set of characters to enter is large. Especially East Asian languages, such as Chinese, deal with similar problems of mapping characters to a format that allows for easy entry. With these scripts, there is not a clear best approach and thus there are, e.g., several different kinds of input methods for Chinese characters. Note that any character set can be considered *large* in some contexts. For example, in early phones methods such as T9 are used to input Latin characters via the smaller numeric keypad. Similarly, current research on text entry methods for smartwatches shows again how just Latin script can already be a lot of characters to support.

Table 6.2: Differences between traditional and emoji text entry

Traditional text entry	Emoji entry
<ul style="list-style-type: none">• Characters are entered from a small set of well-known symbols (the ISO basic Latin alphabet, e.g., has 26 characters).• The meaning of each character is well defined and there is no character-level ambiguity.• Input needs to be optimized for speed. Focuses on composition of words.• Designed for frequent and, sometimes, prolonged use.• A means for general purpose expression. Text is neither inherently playful, nor somber.	<ul style="list-style-type: none">• Characters are entered from a large set where users might be unfamiliar with many of the symbols.• Multiple interpretations per emoji are possible and ambiguity can allow for choice between several emoji (e.g., several emoji can connote enjoyment).• Speed and efficiency are less important. Instead, exploration of the available symbols needs to be easy.• Designed for intermittent and sporadic use.• Emoji are visual (and often playful and “cartoony”) in nature, making them immediately noticeable when embedded in text. In text, emoji can provide a sort of emotional annotation (e.g., pointing out intended sarcasm).

Compared to text entry for large character sets, methods for sets of up to ~30 characters are well researched. For those small sets, frequently each character is assigned to a button and the buttons are arranged in a “simple” layout (e.g., QWERTY). However, this approach does not work for entering emoji—there is no layout that shows all emoji at the same time at a selectable size. In fact, in addition to distinct layout problems, there are several general differences between entering text and entering emoji (see Table 6.2).

In this section, we present an overview of classes of text entry methods currently available for entering emoji. Furthermore, we take a look at existing input methods for other kinds of large character sets and how they could influence designs of emoji input methods. Particularly East Asian scripts have inspired a wide range of input methods, e.g., based on shape or phonetic of the characters to enter. While this section shows the breadth of methods available, we will then take a look at EmojiZoom—an input method specifically designed for the emoji entry characteristics outlined above.

6.7.1 Enumeration

Listing all possible characters and then allowing users to browse that list is the approach currently used by emoji keyboards. This approach has several advantages: (1) it does not require any ordering of the characters, (2) it allows users to browse all available characters and thus aids in discovery, (3) as users can see the available emoji, they do not need to remember exactly what it looks like, and (4) it is easy to implement and extend for new characters. However, as stated earlier, this approach also does not scale well to larger character sets. The absence of an ordering also means that it can be hard to remember the location of a character.














6.7.1.1 Categorization

To alleviate the problems of large lists, emoji keyboards split the emoji into different categories. In Android 5.0, e.g., those are: *faces*, *objects*, *nature*, *places*, and *symbols*. However, while this limits the number of pages per category, splitting introduces new problems. The assignment to categories is often arbitrary and a compromise. For example, 🐱 is part of the *faces* category, while 🐾 is part of the *nature* category with most other animals. Yet, while 🐾 is also in the *nature* category, 🐎 and 🏊 are found in the *places* category with other sports and activities. Such ambiguities are common. Should 🇺🇸 be with the other flags or near 🗳️? Why is 🔊 not next to 🎵 and 🎧 on the Google keyboard? Because emoji often allow multiple uses and interpretations, imposing a strict ordering is bound to produce such cases, where one would expect one emoji to be close to another, but it is found elsewhere. One way to work around this problem would be to put emoji in multiple categories. There is currently no keyboard that does this, and such an approach also comes with drawbacks as. Having emoji in multiple places would inflate categories and thus likely negatively impact search time (users would need to scroll through more pages per category).




6.7.2 Querying and Prediction

Instead of selecting characters from large lists, users can be enabled to query the set of characters to find the one they are looking for. While this allows for simple interfaces (e.g., only a text box) the main problem with this approach is that users have to know the character they are looking for. Thus, query systems fare poorly when users need to discover characters or cannot exactly remember the shape of a character. However, such systems do integrate well with existing Latin script keyboards as they require no mode switch to a dedicated emoji keyboard.

6.7.2.1 Handwriting/Drawing

Handwriting recognition [293] is an established method for entering characters. While it theoretically allows for very large character sets, it is effectively limited by how well users can remember and draw each character. Furthermore, recognizers need to be able to tell characters apart, limiting what kind of character sets can be used (e.g., two slightly differently sized squared could be hard to disambiguate). Earlier research already extended handwriting entry to pictogram retrieval [179]. Google has also been experimenting with using handwriting input for entering emoji²⁴. While their emoji mode works well for emoji with an easy to draw silhouette (e.g., , or ) or only few features (e.g., , or ), it fails with emoji where no clear handwriting equivalent exists or where the emoji is too complex (e.g., , , or ). Outline drawings are also problematic when distinguishing between emoji that only differ on how they are filled. For example, the flags of Germany , Estonia , and Hungary  only differ in the color of their stripes and are thus not distinguishable in outline sketches. Another example are balls, where distinguishing between , , and  would require drawing the internal lines as well. Handwriting detection could be useful for quickly narrowing down the set of emoji to choose from, e.g., drawing a circle to be presented with a list of all spherical emoji.

6.7.2.2 Textual Search and Replacement

When users know the name or description of a character (e.g.,  is called *oncoming bus*), they can search for it with text queries. While we are not aware of any emoji keyboard implementing this, some do offer a related method. For example, on *Windows Phone 8.1* users can select words and then pick an emoji for replacement (e.g., replacing “dragon” with ). This approach uses a curated subset of emoji annotations as given in the *Unicode CLDR*²⁵. In fact, this approach of linking emoji with their respective keywords is becoming more common. Apple, e.g., is planning to introduce the capability to replace words with emoji on tap to *iMessage* with *iOS 10*. Some systems such as the *Slack*²⁶ messaging platform, the *Discourse*²⁷ forum software, the *Github*²⁸ repository service, and many others²⁹, allow emoji entry via text codes. Here, entering `:thumbsup:` in a text field would, e.g., result in that text in the output to be mapped to the  emoji.

²⁴<https://play.google.com/store/apps/details?id=com.google.android.apps.handwriting.ime>

²⁵*Unicode Common Locale Data Repository*: <http://cldr.unicode.org/>

²⁶<https://get.slack.help/hc/en-us/articles/202931348-Emoji-and-emoticons>

²⁷<http://blog.discourse.org/2015/12/emoji-and-discourse/>

²⁸<https://github.com/blog/1289-emoji-autocomplete>

²⁹See, e.g., <http://www.emoji-cheat-sheet.com/>

6.7.2.3 Prediction

Instead of making emoji entry a dedicated task, some keyboards try to roll it into the autocorrection mechanism. An early example of this can be found on *Windows Phone 8.1* where, after entering “*birthday*”, the keyboard will suggest entering 🎂 next. Such emoji suggestions can also be found in some third party keyboards. The *Minuum* keyboard is one example, featuring *smart emoji prediction*³⁰. Instead of integrating into a keyboard, *Dango* floats on the screen, analyzing what is being typed to suggest emoji, GIFs and stickers to add to the current message³¹. Like our semantic emoji model, *Dango* uses a neural network to embed emoji and text in the same space. Instead of emoji, Urabe at el. analyzed affect in input text and then infer appropriate emoticons [299]. User thus implicitly control which emoticons are available with the text they enter. Our work in this chapter can support existing methods in this category. As we will show later, we are able to predict semantic similarity between emoji, which could directly support prediction.

One aspect of prediction-based methods that so far has seen little focus, though, is how they could support exploration. Currently, the existing systems only present users with a shortlist of matching emoji. However, going beyond this list is not supported yet. If similarity data for emoji is available, keyboards could be extended to allow users to grow a result set with other emoji, similar to the ones already shown. Similarly, systems could present an initial coarse but wider prediction and then only after users pick the rough direction they want to go in, present a more fine-grained list of predicted emoji.

6.7.2.4 Query by Picture

While not used as an actual text input method, *Image2Emoji* demonstrated how to select a set of emoji based on an input image [35]³². Input images are mapped to textual descriptions via a convolutional neural network, combined with any accompanying text (title, description, tags). Emoji are also mapped to text by representing each with its name. Which emoji are then similar to an image is determined by similarity of their textual descriptions in a semantic embedding. Such a method could be useful for generating a shortlist of emoji for use in captions when sharing images on social media or when commenting on media. However, for a general text entry method, requiring users to have an image of what they are trying to express handy is likely too cumbersome.

³⁰<http://minuum.com/exploring-emoji-the-quest-for-the-perfect-emoticon/>

³¹<http://getdango.com/emoji-and-deep-learning.html>

³²They have also shown the reverse: searching for video based on a set of input emoji [36].

6.7.3 Methods for East Asian Languages: Mapping to Latin Script

So far we have taken a look at methods already in use (even though only experimentally) for emoji entry. For East Asian scripts, there are several more methods in use which work by defining a mapping from each character to a sequence of Latin characters. Such a mapping is generally (1) phonetic, or (2) based on the shape of the character. Here we detail those two approaches and describe how those could relate to emoji entry.

6.7.3.1 Phonetic Mapping

Some of the most common methods here are based on *Pinyin*, the phonetic system for Mandarin. Instead of entering a character directly, users enter a phonetic representation (which is possible using an extended set of Latin characters). For example, to enter 你好, users can just input “*nihao*”. However, while such methods work well for, e.g., East Asian scripts, they are not directly applicable to entry of emoji or symbols where no clear phonetic representation exists. Instead, one could spell out the name of an emoji (e.g., “*tropical drink*” ⇔ 🍹). But many of these names are not obvious and there would need to be a way to indicate whether emoji or text entry is desired. A basic version of this is found on Windows Phone 8.1, where saying “*smiley*” inserts a :), while “*frowny*” instead puts a :(into the text. However, speech is generally poorly suited to the exploration required for entry from the larger emoji set. One possible approach here, though, would be to dictate text first and then insert emoji afterwards. Textual context could then limit the number of candidate emoji to a manageable set for speech input.

6.7.3.2 Graphological

In graphological mappings, characters are decomposed into parts and then represented as Latin characters. The Cangjie input method, e.g., represents all Chinese characters by 24 basic character components. For example, the 水 (water) radical is used in characters, such as 泰, 永, or 冰. Each of those components is mapped to one Latin character (the water radical, e.g., is represented by the letter E). To enter a Chinese character, users then need to input several radicals in a specific sequence. Users thus need to be familiar with the decomposition rules that define how to map characters to sequences of radicals.

This concept could potentially be used for emoji as well, if suitable decompositions are defined. For example, 😊, 😞, and 😄 all share a common visual component (the basic smiley face) with varying facial features. We can imagine entering emoji by chaining together descriptors (e.g., specifying the combination of *face* and *happy* already reduces the set of possible emoji to a manageable number, which could then be displayed for selection). Instead of showing a list of all emoji, emoji keyboards could then just show a much smaller list of emoji radicals. However, this requires manual or automated tagging of emoji radicals, where it is not clear what the set of radicals would be.

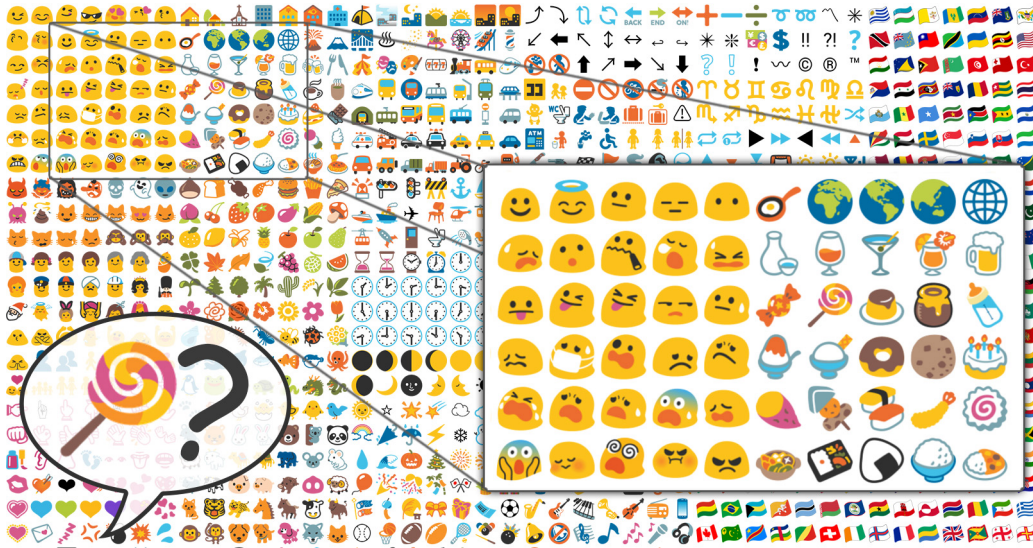


Figure 6.16: EmojiZoom is a novel keyboard for entry of emoji characters that is built around zooming. Instead of swiping through a list, users zoom to the area of the emoji and then select it from the zoomed-in view. Users can still easily explore all available emoji by panning the zoomed-in view.

6.8 EmojiZoom: A Novel Emoji Entry Method

In Section 6.6 we saw that emoji input with the Google keyboard can be a bit slow, with entry of one emoji taking 11.4 s on average (excluding trials not completed within a minute). We also identified that the search process over multiple categories and pages can sometimes be problematic and can have users missing the emoji they are looking for. The piecemeal nature of the emoji presentation also means that it is hard to get an overview of all available emoji. Furthermore, because category jump targets are dynamic it can be hard to build memory of how to get to any one specific emoji.

In Section 6.7 we then learned about the overall space of entry methods applicable to emoji. Here, we will discuss an attempt to address some of the shortcomings of the Google emoji keyboard, while adhering to enumeration of emoji as the underlying principle. Instead of showing emoji in many pages or large lists, the *EmojiZoom* (see Figure 6.16 for an illustration) input method shows all emoji at once. As this leads to very small emoji targets, users can zoom in and can then select emoji from this enlarged view. High resolution screens on current mobile devices enable users to still make out sufficient details, even at the zoomed out level. As the start view is always the same, this design also allows users to build spatial memory and, e.g., learn that smiley emoji are always found in the top left corner.

Overall, EmojiZoom has several advantages over selection from emoji lists:

- Showing all emoji at once, gives users an immediate idea of how many and which emoji are available.
- Supports building up spatial memory, as groups of related emoji are always found in the same region.
- Exploration is possible at multiple scales—users can zoom in slightly to explore the general overview or zoom in all the way to more closely explore a certain region.

6.8.1 Related Work

We are not the first to explore zooming as a modality for text input. Starting with *ZoomBoard* [218], there have been several zooming keyboards [41, 174]. The focus of those keyboards has so far been to allow entry of Latin characters on very small keyboards (such as on smartwatches). Instead of showing few characters on a small screen, we show many characters on a high-resolution screen. Both approaches solve a mapping problem where the number of symbols and the available space are mismatched.

6.8.2 Implementation

For our EmojiZoom implementation we forked the Google keyboard³³ (shown in Figure 6.18) and replaced the emoji components with our own. While this restricts the range of possible designs, it allows for a fair comparison as we retain the overall size. We also render emoji at the same size when zoomed in, as not to introduce a confounding factor in emoji selection. The QWERTY view stays the same, so users see an identical initial view when they activate the keyboard. In the emoji view, however, we remove the list control and categories and only render a grid of all available emoji (see Figure 6.17). We keep the lower bar, but add a *zoom out* button and the backspace button to it (see Figure 6.20). Users can zoom into the grid by either tapping on it or using two-finger gestures as in, e.g., map apps. Zooming out is possible via pinching, or by tapping on the *zoom out* button. While tapping zooms in a preset amount, the two-finger zoom allow users to pick any zoom factor.

When zoomed in, users can pan the view by dragging inside the emoji grid. If at or beyond 75 % zoom, a tap on an emoji selects it and returns the grid to the zoomed out view. We set 100 % zoom so that emoji are rendered at about the same size as on the Google keyboard. However, we reduce whitespace between emoji and can thus fit more emoji on the screen. Where the Google keyboard shows 7 emoji in a row, we can fit up to 9 emoji in a row.

³³<https://android.googlesource.com/platform/packages/inputmethods/LatinIME>

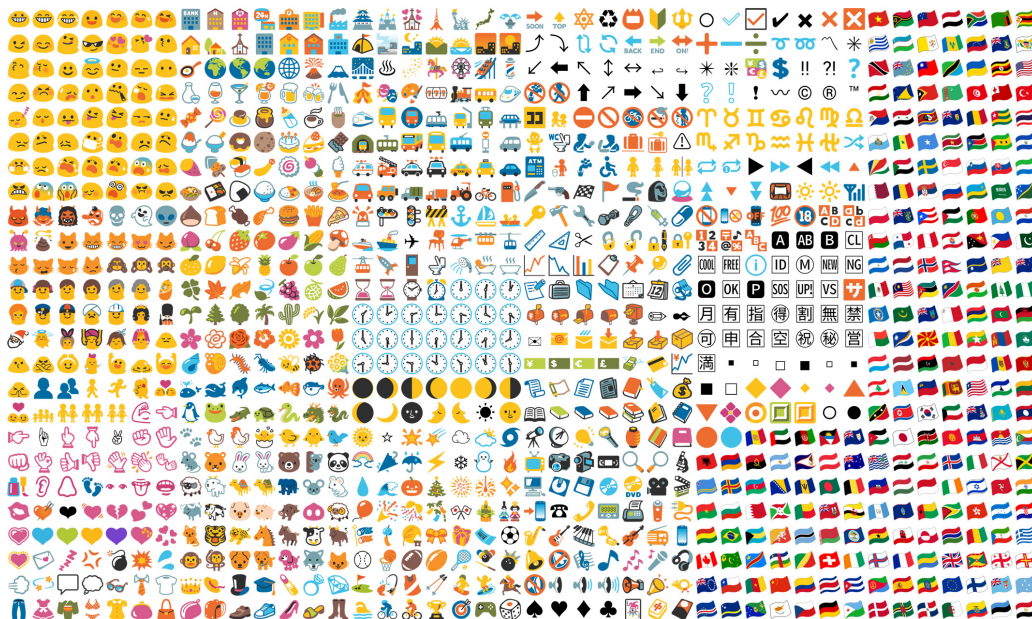


Figure 6.17: All emoji as shown in the zoomed out view of EmojiZoom. Emoji are arranged in a 42×25 grid along a snaking path, as illustrated in Figure 6.19. On the Nexus 5 we used in the study this whole grid is 6 cm wide: each emoji occupies about 1.4 mm. As the Nexus 5’s screen has 445 ppi this gives about 25 px per emoji—enough to identify many details.

When users tap to zoom in, we apply a zoom region interpolation as in *ZoomBoard* [218]. This tries to resolve the ambiguity of a zoom action: should the selected point after zooming in be under the finger or in the center of the screen? The applied interpolation is a compromise between both approaches and yields a center position between the two extremes. However, this also means that if users press exactly on an emoji in the zoomed-out view, that emoji is not under their finger in the zoomed-in view. They have to move their finger again to select the desired emoji. We believe this is not much of a trade-off though, as the large number of emoji makes it hard to precisely select one in the zoomed-out view anyway. Instead, the more viable strategy is to tap in the vicinity of the desired emoji. Users then need to reacquire the emoji in the zoomed-in view (where it should be close to the center) and touch it again.

One critical aspect in EmojiZoom is the arrangement of emoji inside the grid. The Unicode standard itself defines an ordering for emoji, yet this only prescribes a one-dimensional order. We hence chose to arrange emoji in columns according to this ordering. The grid is then filled in a snake-like pattern (see Figure 6.19), thus column direction is reversed every column. Inside each column, emoji are just ordered from left to right.

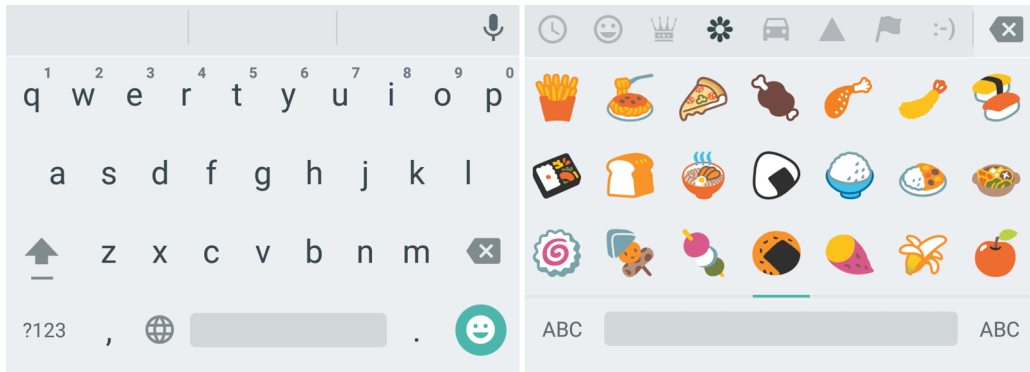


Figure 6.18: We compare against the *Google keyboard* on the *Nexus 5*. Here, emoji mode is activated with a button in the lower-right. Once in emoji mode, emoji are shown in a paging control where users swipe left/right to move to the next page. Tabs enable jumping to different categories. Note that this is a newer version of the keyboard with more emoji and categories than the one used in the previous study (and shown in Figure 6.11).

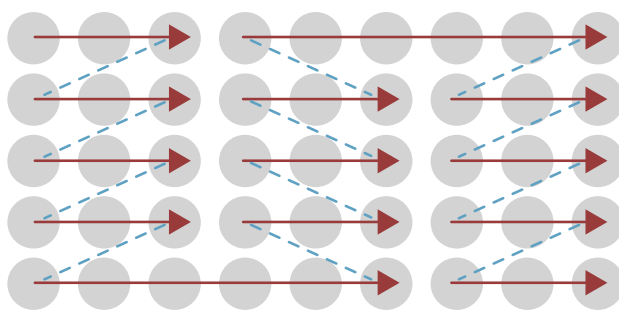


Figure 6.19: In order to map emoji to our grid, we use a snaking space-filling algorithm. Emoji are first ordered according to the Unicode standard’s definition. Columns are then filled either from top to bottom or from bottom to top, alternating every column.

6.8.3 Evaluation

With *EmojiZoom* implemented, we set out to determine whether emoji entry is faster with it. For comparison, we picked the default *Google keyboard* (version 4.1.23153.2501950) on a *Nexus 5*, running *Android 5.1.1*. The screen resolution on the *Nexus 5* is 1920×1080 px with 445 ppi. The default keyboard offers 1050 emoji, split into 6 categories. This is a larger number of emoji than in our previous study with the *Google keyboard*, as the keyboard was updated by *Google* in the meantime. Note that while our keyboard gives no preference to common or recent emoji, the *Google keyboard* employs a mechanism to facilitate entry of common emoji. For this purpose the *Google keyboard* maintains a list of recently used emoji and also remembers the last used page per category. If users only enter emoji from a small number of pages, this approach often presents them with the target page when opening the keyboard or switching categories.

6.8.3.1 Participants

We recruited 18 participants (4 female, age 21–26, $\bar{x} = 23.4$, $SD = 1.5$) for the study via social media. All but one participant owned a smartphone and 11 of them ran some version of Android on their phone. Participants using an iPhone all ran iOS version 9.2.1, while Android use was split between 4.x (5 participants), 5.x (5 participant), and 6.x (1 participant). No participant used a custom keyboard, thus all were most familiar with the default keyboard on their phone. However, note that some apps come with their own emoji selection mechanism and thus a participant’s default keyboard might not be the one she has the most emoji entry experience with. For example, *WhatsApp* copies the emoji panel from the iOS keyboard within their app. As we saw earlier, emoji design can vary between different phones, and between different versions of the same operating system. While we used the Google emoji style, all iOS users instead used the Apple style on their phones. In fact, the Google emoji style was only used on five of the participants’ phones. Four participants using Android instead used the Samsung design, one participant used the LG design, and one Android user had the Apple style emoji on his phone. Note that while this shows a wide range in different keyboards and designs between participants, the basic principle of selection from an emoji list is identical in all those devices.

Thirteen participants stated that they often or very often use emoji in their daily life. The primary use (according to 16 participants) is in chats and instant messaging. Two participants said they also use emoji in social media and one participant respectively indicated use in emails and forums. Overall, emoji were described as useful or very useful by 14 participants. Only two participants stated to see no use for emoji.

6.8.3.2 Procedure

As we already saw in Section 6.6, evaluation of emoji entry comes with some unique challenges. Here our test set with 1050 emoji is even larger and requiring each participant to enter each of them once, or even multiple times, is not feasible. While our earlier evaluation used a subset of emoji given by a Twitter scrape, here we just use random sampling. As described in Section 6.6.1, the choice of evaluation procedure is constrained. Here, we again resort to using a copy task or our evaluation.

We extended our earlier test application (as previously shown in Figure 6.11) to also work with *EmojiZoom*. As shown in Figure 6.20, it presents participants with an emoji and asks them to enter it. Before participants started a session, they were given time to try out the respective keyboard (we again consider them sufficiently familiar after 10 emoji). We also kept the one minute time allotted for each trial and abort trials accordingly.

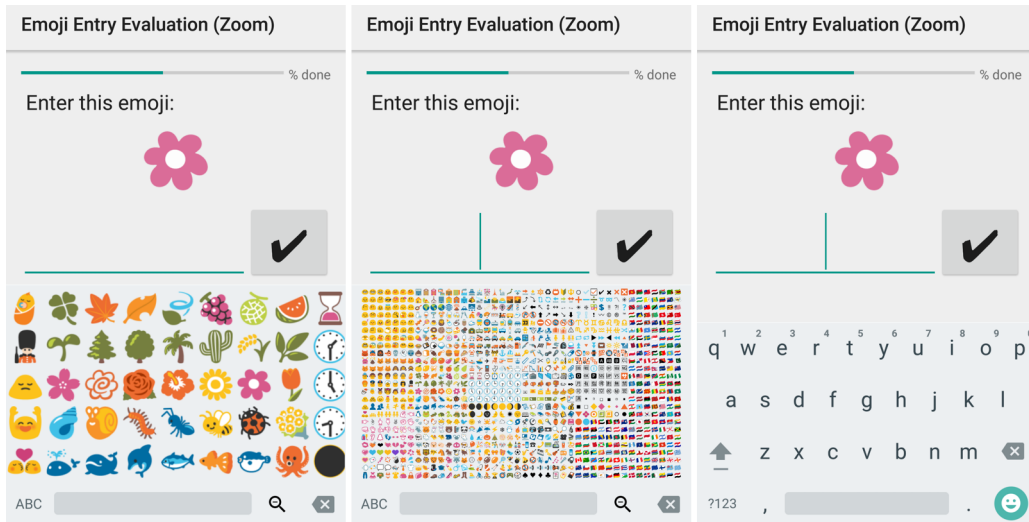


Figure 6.20: Layout of our evaluation application while testing EmojiZoom. Participants are shown the emoji to enter at the top of the app. In each trial, they need to activate emoji mode, find the respective emoji and click the commit button. Committing also switches the keyboard back to QWERTY mode.

6.8.3.3 Design

The study was a within-subjects design with keyboard as the only factor. We counterbalanced keyboard order between participants. For each participant, we draw a random set of 50 emoji to enter by sampling with replacement from the set of all emoji. This allows for paired comparisons for each participant as they enter each emoji twice: once with each keyboard. Each participant completed 100 trials, which took about 45 minutes.

6.8.3.4 Results

Overall, we saw that there is no difference between the two keyboards with respect to failure rate (trial was aborted due to taking too long). For both keyboards, about 4% of the trials were aborted because the participant could not find and enter the required emoji in one minute. We also count a trial as a failure if the wrong emoji is entered. A paired-samples t-test shows no significant difference in failure rate between the Google keyboard ($M=4.2\%$, $SD=3.3$) and EmojiZoom ($M=4.4\%$, $SD=2.7$); $t(17) = -0.24$, $p > 0.8$. However, EmojiZoom was faster than the Google keyboard (see Figure 6.21). A paired-samples t-test shows a significant difference in retrieval time between the Google keyboard ($M=15.6$ s, $SD=2.9$) and EmojiZoom ($M=12.7$ s, $SD=1.9$); $t(17) = 3.49$, $p < 0.01$. This is an 18% increase in emoji entry speed even though participants had no familiarity with methods like EmojiZoom. On the other hand, many had a lot of experience entering emoji with a standard category-based keyboard.

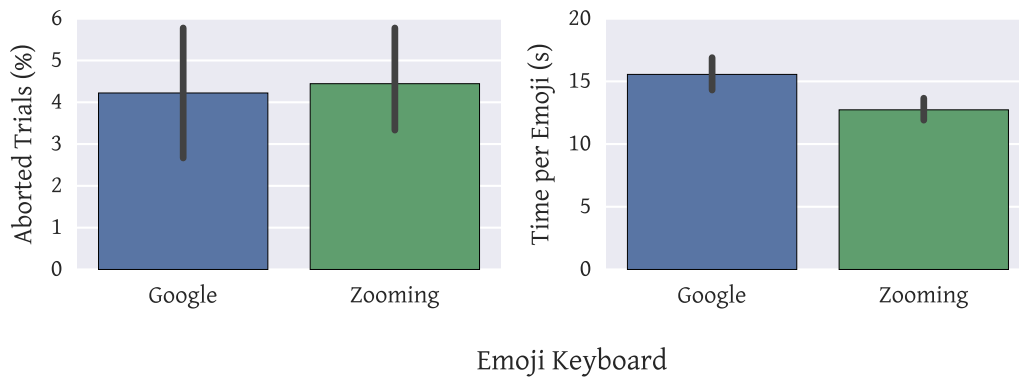


Figure 6.21: The Google keyboard and EmojiZoom do not differ with respect to the number of failed trials (4.2 % for the Google keyboard and 4.4 % for EmojiZoom). On average it took participants 15.6 s (10.7 s median) to enter an emoji with the Google keyboard. With EmojiZoom, participants only needed 12.7 s (8.1 s). This is significantly faster and about an 18 % increase in speed. Error bars show 95 % CI.

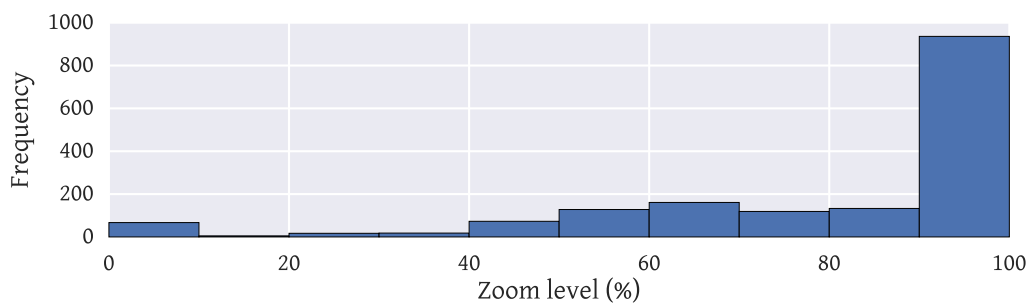


Figure 6.22: This histogram of zoom levels shows participants zooming behavior, excluding the initial zoomed-out view. Most of the time they chose to zoom in all the way, yet sometimes that chose a medium zoom level to aid in exploration of a smaller region.

When we look at how people used EmojiZoom, we can see some patterns emerge. An interesting question, e.g., is how users choose between zoom levels: do they only zoom in as much as possible and skim that view, or do they make use of multiple zoom levels? As shown in Figure 6.22, the maximum zoom level is dominating the statistics. Note that this view excludes the initial zoomed-out view, thus any occurrence of that zoom level in the histogram is due to users zooming out again. This could, e.g., be due to selecting the wrong initial region and backtracking to the start in order to jump elsewhere in the emoji grid. We can also see, though, that medium zoom levels were chosen quite regularly as well. This shows that users take up the chance to zoom in slightly, explore a region of emoji, and only then zoom in more to make the final selection.

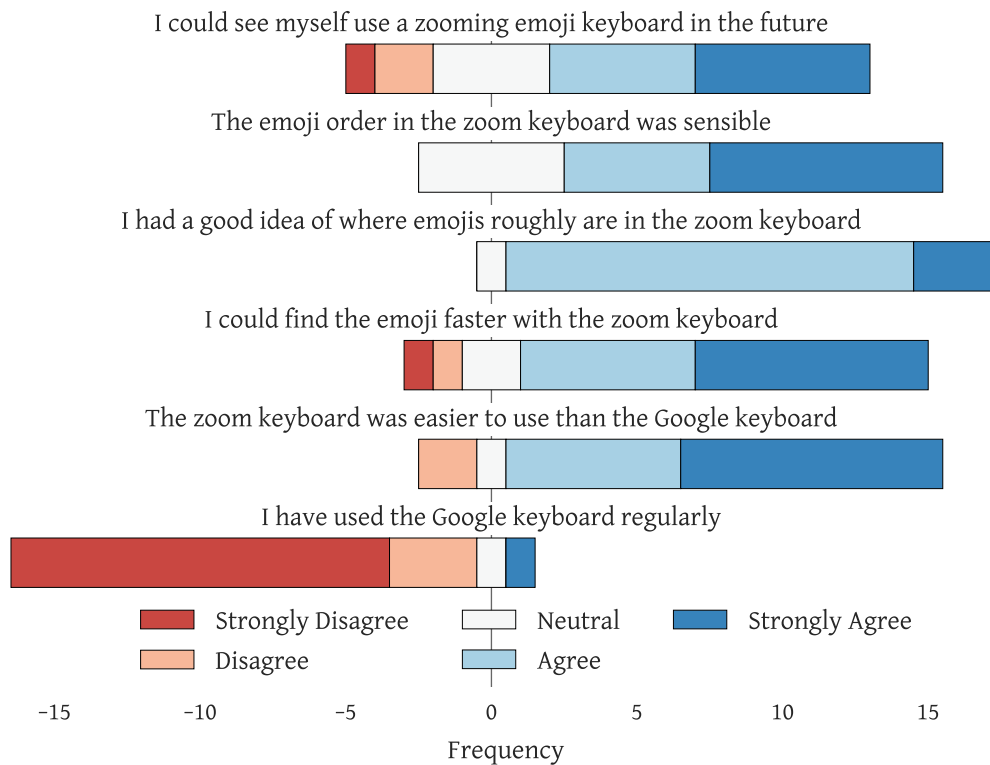


Figure 6.23: Participants rated six statements on a 5-point Likert scale in our exit interview. Results show preference for EmojiZoom over the Google emoji keyboard.

We also asked participants to rate six statements on a Likert scale after completion of the study (see Figure 6.23). Asked to rate the quality of EmojiZoom, all participants gave favorable ratings to the ordering. The majority of participants also preferred EmojiZoom to the Google keyboard. As the Google keyboard is very similar to most other emoji keyboards, this indicates EmojiZoom, even though unfamiliar, might make a better impression in general. In fact, most participants could imagine using a zooming keyboard for emoji entry in the future.

Interestingly, several participants stated that the ordering of emoji in EmojiZoom was better than the one in the Google keyboard. Instead of just using the order as specified in the Unicode standard, the Google keyboard moves some emoji around. For example, 🗑️, 🔊, 🔊, 🔊, 🔊, 🔊, and 🗑️ follow after one another in the Unicode ordering, yet are split over two different categories in the Google keyboard. Of course, any rigid ordering for emoji is flawed, as some emoji can have several interpretations (e.g., 🐶 could be put with other animal emoji or with other activity emoji, where 🏠 can be found).

6.8.3.5 Discussion

The results show that emoji entry with a zooming keyboard is a viable approach. In fact, EmojiZoom outperformed the Google keyboard by 18 %. Users also preferred using EmojiZoom and stated that the ordering was better than in the Google keyboard. By having all emoji visible at once, some of the problems of emoji categories are avoided. Where a search for an emoji in the Google keyboard carries a heavy penalty if a wrong category is picked, moving on to a different region is easier and faster in EmojiZoom. A limitation of our study is that the participants were all comparably young. We would expect performance to be lower for older users who might not be able to see the small emoji that well.

6.8.4 Longitudinal Evaluation

Our study showed that zooming emoji entry outperforms emoji entry with the Google keyboard. However, participants in that study only got to use our keyboard for a short time. To investigate whether performance could improve further, we ran a limited additional longitudinal evaluation.

We recruited three additional participant for this evaluation (all male, age 25, 28, and 42 years). One participant installed EmojiZoom on his own phone (Samsung Galaxy S6), while the other two used one of our Nexus 5 phones. The Samsung phone had a slightly higher resolution screen than the Nexus 5 at 2560×1440 px with 577 ppi and also ran Android 5.1.1. As this allows for slightly clearer rendering of emoji in the zoomed out view, its performance numbers (bottom plot in Figure 6.24) are thus not directly comparable. They each ran the evaluation several times over the following weeks, yielding 12, 19, and 20 runs respectively. This time, we only included our keyboard and did not gather longitudinal data for the Google keyboard. Each run, a random emoji test set was used to prevent repetition of a small subset of emoji over the course of the study.

6.8.4.1 Results

As shown in Figure 6.24, we observed steady improvement over the course of the study for all three participants. For example, while it took participant three 14.5 s per emoji in the first run, performance improved to 5.6 s in the last run. This is a speedup of almost 60 %. We can run a statistical test to check whether performance in the end is significantly better than performance in the beginning. An independent samples t-test shows a significant difference in retrieval time between the first two runs and the last two runs for all three participants with $p \leq 0.001$. Linear regression per participant also is significant with $p \leq 0.01$ for all three.

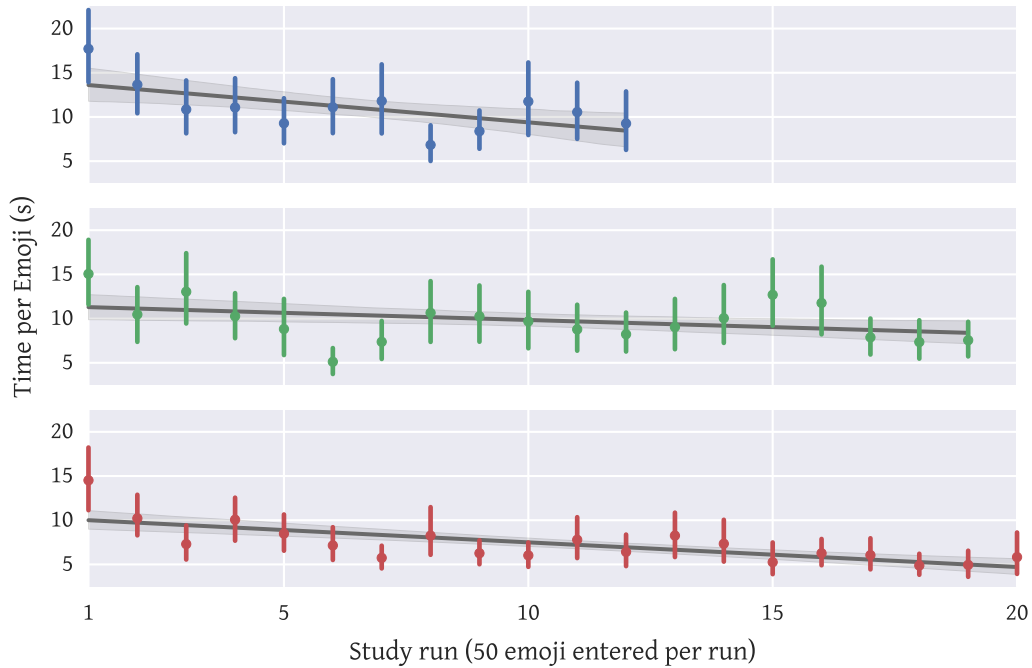


Figure 6.24: Three additional participants used EmojiZoom over the course of up to twelve days, completing the study 12/19/20 times (each time with a different emoji test set). Performance improved significantly (e.g., by ~60 % for the third participant). Error bars show 95 % CI.

6.8.4.2 Discussion

While we have only tested the impact of prolonged use with three participants, we saw clear improvements in a short time period. It seems like EmojiZoom indeed builds spatial memory and enables users to zoom into the rough vicinity of their desired emoji faster. However, additional studies are necessary to confirm this improvement, due to the small sample size of this study. We are working on making EmojiZoom available to a wider range of users and hope to gather in-situ data over a prolonged period of time.

6.8.5 Emoji-Level Analysis

So far we have only looked at overall selection time. But we were also curious whether there were any patterns in the data showing that EmojiZoom worked better for some emoji than for others. For this analysis we pool together all collected data for EmojiZoom: from the lab study and all trials from the longitudinal evaluation. This gives us a total of 3423 trials, of which 3235 were successful (95 %).

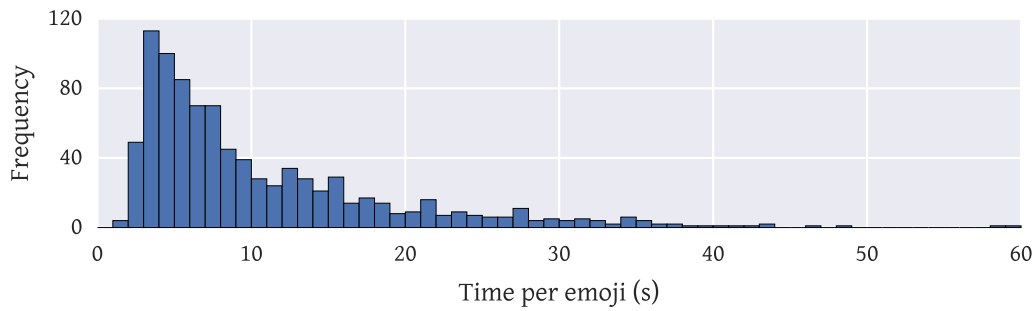


Figure 6.25: The distribution of selection times for EmojiZoom has a long tail. About 70 % of selections are done within 10 seconds, while some take much longer.

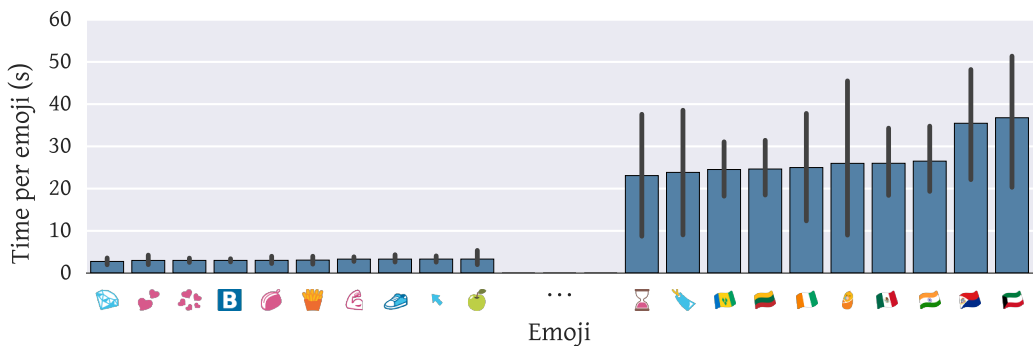


Figure 6.26: Best and worst performing emoji with respect to selection time. We only consider emoji which were part of at least 5 trials. Error bars show 95 % CI.

We first take a look at overall patterns with respect to selection times. As the distribution in Figure 6.25 shows, most selections are fast, but a long tail is visible as well. While 50 % of trials completed within 5.8 s, it took 34.8 s until almost all (95 %) selections finished. Note that we stopped trials after 60 seconds (these unsuccessful trials are not considered in this analysis). Had we allowed participants to continue, the tail would be longer. But as this data only makes up 5 % of the trials overall, the impact on the distribution would not be very strong. However, we observe that this distribution is not equal for all emoji, but varies for different ones.

As shown in Figure 6.26, some emoji were always selected quickly, while others tend to lead to slow selection times. For this comparison we only consider emoji with at least 5 recorded trials. Among the ten worst performing emoji we find 7 that are flag emoji. Checking back with the whole dataset, we find that while it took 22.1 s on average to enter a flag, it only took 9.6 s to enter other emoji. An independent samples t-test shows a significant difference between selection times of flags and other emoji; $t(3419) = 20.9$, $p < 0.0001$.

Table 6.3: Emoji with the highest share of failed trials overall. We only include emoji for which we collected data from at least two trials.

Emoji	Trials	Success	Emoji	Trials	Success	Emoji	Trials	Success
🟩	2	0.0 %	🇳🇱	3	33.3 %	🇳🇪	3	33.3 %
🔍	4	25.0 %	🇳🇮	3	33.3 %	🇳🇮	5	40.0 %
🌓	11	27.2 %	🇳🇮	6	33.3 %	🟩	5	40.0 %
📄	3	33.3 %	🇳🇮	3	33.3 %	📄	5	40.0 %

The likely reason for this strong difference is the visual similarity between all the flags. As could be seen in Figure 6.17 earlier, the flags form a large block on the right side of the grid. While some flags, such as 🇳🇮, stand out a bit, many flags are rather similar. An extreme example are flags from places with former ties to the British Empire:













- 🇳🇮 Ascension Island (AC)
- 🇳🇮 Australia (AU)
- 🇳🇮 Cook Islands (CK)
- 🇳🇮 South Georgia & South Sandwich Islands (GS)
- 🇳🇮 Heard & McDonald Islands (HM)
- 🇳🇮 Montserrat (MS)
- 🇳🇮 New Zealand (NZ)
- 🇳🇮 Pitcairn Islands (PN)
- 🇳🇮 St. Helena (SH)
- 🇳🇮 Tristan Da Cunha (TA)
- 🇳🇮 Turks & Caicos Islands (TC)
- 🇳🇮 British Virgin Islands (VG)

The ordering of flags is also not readily apparent as they are sorted by their corresponding region code. Hence, while they are ordered alphabetically, Germany (region code DE) is not found near other countries starting with G, but next to other countries with region codes starting with a D. However, countries' region codes are likely not familiar to users. Even if they were, region codes are not shown in the emoji and are thus not readily accessible.

The problem with flags also shows when we look at failed trials (those aborted after 60 s). As shown in Table 6.3, flags again make up a large share of those emoji. However, with a low share of failed trials overall and a large number of emoji to enter, there is only little data on failures for individual emoji. A larger scale deployment and analysis could uncover clearer patterns in which emoji are more likely to take a very long time and would thus benefit most from keyboard improvements.

While some trials failed, it is important to note that the majority of emoji were entered successfully each time. For 789 emoji (85 % of our tested emoji), we recorded no failed trial at all. In Table 6.4, we show the emoji with the largest number of successful trials. This does not take into account selection time, but only shows whether they were selected within the 60 s. Interestingly, this set also contains a flag, the one of Tuvalu. While participants always found that flag, this still took them 18.1 s on average. While this is fast for flags, it is slow compared to other emoji.

Table 6.4: Emoji with the lowest share of failed trials overall ranked by the number of trials recorded. This does not take into account the time it took to enter emoji.

Emoji	Trials	Success	Emoji	Trials	Success	Emoji	Trials	Success
	15	100 %		12	100 %		11	100 %
	12	100 %		12	100 %		11	100 %
	12	100 %		11	100 %		10	100 %
	12	100 %		11	100 %		10	100 %

6.8.6 Summary

We have introduced EmojiZoom, an input method for emoji that outperforms existing emoji keyboards built around selection from long lists. Our method shows a clear performance advantage with the potential for further improvement with additional training. Participants also preferred EmojiZoom and found the ordering of emoji superior to the one in the Google keyboard. However, the use of EmojiZoom necessitates a high-resolution screen. But as such screens have become more prevalent in current generation smartphones, methods like EmojiZoom have become viable.

Yet there are some possible improvements to EmojiZoom. In the current implementation flags take up a large share of the space. However, most users will only ever need a small subset of the available flags and, as shown, flags are hard to distinguish. Instead of showing all flags, flag selection could be relegated to a second level. Users would select a flag stand-in (which could be larger than regular emoji to make selection easier) which would switch the emoji selection to a dedicated flag mode. Here flags could either be displayed in a second grid or in any other arrangement. For example, it might be suitable to pick flags from a world map or from a list of country names. A similar approach might be possible for the clock or moon phase emoji. This would introduce a hierarchy to EmojiZoom, further strengthening its ability to cope with the growing number of emoji.

To implement these kind of adaptations, the quantitative emoji data collected earlier could be used. But a more critical aspect is the arrangement of emoji in the EmojiZoom grid itself. We went with the default Unicode ordering, but could only create the final grid by having a column-wise layout. Ideally, the emoji would be distributed in this grid in a way that makes sure related emoji are close to each other. With our method so far this is not the case over column borders and thus wildly different emoji can be next to each other. EmojiZoom could hence potentially be further improved by having an emoji similarity model available that can provide the necessary data to create the desired emoji grid. In the following sections we will thus explore such models more closely.

6.9 Motivating Emoji Similarity Modeling

Keyboards have always brought with them questions of optimization: after all, the whole point of the invention of the typewriter was to speed up writing. But optimization is particularly important, as there is a large potential within the nature of the device. As the letter-to-key assignment is arbitrary, in the sense that any such assignment is possible to build or implement, there is a large amount of leeway in the design. Common optimization goals for keyboards are, e.g., to make text entry faster, or to make disambiguation between neighboring keys easier.

Early keyboard layouts, such as the QWERTY layout, were hand-designed. But the number of possible layouts is very large and design intuition was eventually replaced by computational approaches. An early example is Zhai et al.'s Metropolis keyboard which tries to optimize for movement time between keys, weighted according to the respective bigram probabilities [327]. In fact, this formulation, based on Fitt's Law, is a common optimization target as it optimizes for the travel distance of the fingers and yields results where keys often used after each other are close to each other. Of course, speed is only one aspect to optimize for and thus there are other keyboards that optimize for multilingual input [18], touch input [308], or try to find the best layout for multiple optimization goals at once [70]. In addition to different objective functions, researchers are also applying more complex optimization methods, such as integer programming [143], to search for the best layout.

However, the optimization goals for text entry and emoji entry are completely different. For text entry a common goal to optimize for is speed of entry, or the related travel distance between keys. Hence, objective functions for this purpose generally include bigram probabilities as a weighting term. The underlying assumption here is that many characters are entered in series. If users only ever entered one key, such an optimization would not help at all. But while longer sequences are common in text entry, they are not a common case for emoji entry. Instead of entering many emoji after each other, users usually only want to add one or a small number of emoji to a message.

The most important aspect of emoji entry is thus the search for individual emoji and not the entry of emoji sequences. As such, we do not need to optimize travel distance between subsequently used emoji, but instead need to optimize for search time. This aspect of emoji entry connects more with work in visual search than work in text entry. Where text input methods have been less concerned with searching for the right key (users are expected to internalize the layout and only be restricted in their entry speed by travel distance at some point), research in menu or icon selections focuses specifically on this.

The benefit of pictorial presentation (scanning for icons is easier than for text) was shown by Niemelä and Saarinen [211]. They also saw a decrease in search time when related icons were grouped together. Grouping related emoji together could thus potentially also lead to reduced search times. There are also ongoing efforts to model visual search, similar to Fitt's Law capturing movement, in order to make performance predictions for interfaces. Such modeling generally falls into two categories: (a) trying to fit a mathematical model to experimental data, or (b) trying to model cognitive processes. An example of the former is Bailly et al.'s work, who fit a mathematical model to predict search time in menus [7]. Instead, Kieras and Hornof try to model the cognitive process of vision, in order to predict where people look and, in effect, how long it will take them to find a target [148]. Their model, e.g., captures that color is the strongest influence on how well targets can be distinguished. We could thus assume that pages of emoji very similar in color, (e.g., the smiley emoji: 😐, 😊, 😄, 😁, ...) make it harder to find a specific emoji than pages with stronger color differences (e.g., the hearts: ❤️, 💙, 💚, ...).

We can relate this back to the input methods presented in Section 6.7. For categorization style input methods, we need to optimize the arrangement of emoji so that related emoji are in the same category. But even within a category, we would expect more related emoji to be close to each other, in order to facilitate search within the category. For example, it is sensible to expect 🇺🇸, 🇩🇪, and 🇩🇪 to be not just in the same category, but also next to each other. A model for categorical emoji input methods would thus need to help with both category assignment and within-category ordering. If we only look at the more relaxed enumeration input methods, only the ordering aspect is necessary. This can be a one-dimensional ordering for linear lists, but could also mean a two-dimensional ordering for approaches like the *EmojiZoom* keyboard from Section 6.8.

Predictive methods also require an emoji similarity model. For example, if a user entered "Hitting the road", a model should be able to predict emoji like 🚗, or 🚙. Here, this would require capturing relationships between emoji and English language words. But even just modeling relationships between different emoji, excluding other text, is helpful. This could, e.g., be used to predict which emoji are likely to be added to a multi-emoji response, with one or more emoji already entered. After typing 🎁, it is likely users would also want to add 🎉 or 🍰 to the message. This can also facilitate exploration, where users can limit the space to a smaller set of candidates by providing a reference emoji.

Here, we concentrate on a between-emoji similarity model which can be used to guide emoji arrangement. The underlying motivation is that users would have an easier time finding an emoji if it is close to similar ones. Instead of looking at each individual emoji, they then only need to look at one to decide whether they are in the vicinity of the one they are looking for. For example, when searching for the 🏀 emoji, users can assume they are close once they see any one of 🏈, 🏐, or 🏉. It would also be fair to assume other sports related emoji are nearby, with 🏊, e.g., as close as possible to 🏀. On the other hand, if users see 🎮, 🎧, or 🎧, they should be able to safely assume that 🏀 is somewhere else. Building a similarity model of emoji is a necessary step to optimize for these criteria.

6.10 Towards a Model of Emoji Similarity

As the number of emoji is currently growing with every new version of the Unicode standard and more emoji make it into vendors' emoji keyboards, the current presentation of emoji in one or several lists is getting more and more problematic. The Google keyboard on the Nexus 5, e.g., already contains 42 pages of emoji for its 822 emoji (the latest version of iOS comes with about 1300 emoji). In our study in Section 6.6 we found that it is easy to miss an emoji. Furthermore, exploration of available emoji is expensive and a good ordering of them thus is necessary to achieve a reasonable average search time. While categories subdivide the space, which speeds up search somewhat, the assignment of emoji to categories, as shown earlier, can itself be problematic. In Section 6.8, we saw how this ordering problem also is relevant for two-dimensional grids of emoji where we would like to create layouts that place related emoji close to each other.

As we have described, keyboards for western languages already make use of models to inform the design of the keyboard. For example, when designing a keyboard for the English language, key arrangements can be chosen so that expected travel time between keys is minimized according to bigram probabilities for letter pairs. However, there is no clear equivalent of model-based optimization for emoji entry. Yet, as outlined in Section 6.9, a similarity model for emoji would help in guiding design of emoji input methods. In order to make model-driven improvements of emoji keyboards possible, we thus set out to build such a model.





















In this section, we will look at two ways to build emoji similarity models: (a) based on emoji annotations, as defined by the Unicode standard, and (b) based on semantic information derived from the large number of tweets we collected. Both similarity measures allow us to quantify which emoji are related and should be close together. We compare performance of both models in a crowdsourcing study and explore differences in what aspects of emoji they capture.

6.10.1 Deriving Emoji Similarity From Unicode Annotations

As mentioned earlier, all emoji are already tagged with some annotations³⁴. For example, 🐶 is tagged as *animal*, *pet*, *nature*, and *cat*. Those annotations are intended to help users winnow down the set of emoji by entering the tag and then selecting matching emoji.

³⁴Available at: <http://unicode.org/repos/cldr/trunk/common/annotations/>. Note that these annotations have recently been updated. The annotations we use in this chapter are per Summer 2015. Some tags were removed and 🐶, e.g., is now only tagged as *pet* and *cat*. The Unicode consortium also maintains a chart that shows all available annotations and the emoji belonging to them: <http://www.unicode.org/emoji/charts/emoji-annotations.html>.

Table 6.5: Jaccard similarity coefficient for a set of example emoji pairs.

Emoji _A	Emoji _B	Similarity	Emoji _A	Emoji _B	Similarity
		1.0			1.0
		0.75			0.75
		0.5			0.5
		0.25			0.25
		0.0			0.0

We initially hypothesized these annotations could be used to establish similarity between any two emoji: by comparing their annotations. Let T_{Emoji_A} be the set of all annotations for Emoji_A . Emoji are considered similar if they share many terms and disagree on few. We thus chose the Jaccard similarity coefficient J as quantifier, which is defined as:

$$J(T_{\text{Emoji}_A}, T_{\text{Emoji}_B}) = \frac{|T_{\text{Emoji}_A} \cap T_{\text{Emoji}_B}|}{|T_{\text{Emoji}_A} \cup T_{\text{Emoji}_B}|}. \quad (6.1)$$

Some examples of the resulting similarity values can be seen in Table 6.5. This similarity measure works well for clustering around concepts, such as *food*, *animal*, or *vehicle*. However, it does, e.g., not include a notion of two emoji coinciding in an activity. For example, the shown dissimilarity of 😊 and 🐶 only tells us that smiling and poodles are not the same *thing*. From a different perspective, though, one could certainly be happy about a new dog or a walk with a dog. This aspect is not captured by this categorical similarity, which does not take context into account. Hence, we set out to find an approach for establishing similarity that captures a wider notion of what emoji relatedness means.

6.10.2 Emoji Model Building from Tweets

To move beyond categorical similarity, we set out to build a similarity measure based on how emoji are actually used. By building on actual use, similarity is hence defined only by whether users enter emoji together or whether they enter emoji in similar contexts. We hypothesize that this would be able to capture connections between emoji that are not reflected in just their tags. For example, we would expect 😊 and 🐶 from our previous example to exhibit some connection once we take into account the context around emoji. We derive this contextual information from the large number of tweets we collected earlier.

Because most tweets only contain one emoji, we need to include all text to build a strong model of emoji similarity. For example, if we were to see both 🧑🏻 and 🐱 individually in tweets containing the word “besties”, we could conclude that they are probably related. We chose to use a word embedding approach to map each emoji (and every other token) to a word vector. Similarity between two emoji can then be determined through the similarity of their respective word vectors.

In a word embedding, individual tokens are mapped to n-dimensional vectors. For example, 😊 might map to $[0, 0]^T$, 😄 to $[0, 1]^T$, 😞 to $[1, 0]^T$, and 🐱 to $[1, 1]^T$. Here, one dimension encodes sentiment and one dimension encodes whether the emoji is a face or a cat. Other emoji could then also be represented in this space, e.g., 🐶 might be represented as $[0.5, 0]^T$. We can see how this basic embedding directly leads us to a similarity measure. If two emoji have similar numeric representations in this space, they are conceptually close. For example, an emoji at $[0.2, 0]^T$ and an emoji at $[0.3, 0]^T$ both would be rather cheery faces, quite far away from an emoji at $[0.9, 0.8]^T$ which would be some sad animal. In the word embedding we can compute the distance between two emoji 😊_a and 😊_b with any norm. We chose to use the Euclidean distance function:

$$\sqrt{\sum_{i=1}^n (\text{😊}_{a_i} - \text{😊}_{b_i})^2}. \quad (6.2)$$

Instead, one could use other norms, such as Manhattan distance, or similarity measures like cosine similarity. Note that we are not interested in the absolute distance between two emoji, but aim to rank emoji to find the n most related ones.

However, in our example embedding it would be hard to represent an emoji like 📺 as it does not align to the chosen dimensions. A television set is neither face nor animal while also not having a clear place in a sentiment dimension. To represent the diverse set of emoji, we need many more dimensions. If we were to hand-pick the dimensions we might, e.g., use concepts such as *sentiment*, *seriousness*, *edibility*, or *gendered*. But picking the right dimensions and then manually rating emoji would be a very complex and error-prone task. Instead, we use a neural network to learn appropriate dimensions for our word embedding. While those dimensions then do not individually map anymore to easily understood concepts, automating this process enables us to make use of much more data than humans could incorporate in a manual design.

To create the word embedding, we use *word2vec*³⁵ [197, 254]. Word2vec implements two versions of a neural network language model: continuous skip-grams and continuous bag-of-words. For both algorithms, there is only one hidden layer in addition to the input and output layers. We use skip-grams, where the vocabulary forms the input layer of the neural network and context words form the output layer. Thus, after learning, the hidden layer is tuned to predict the context (surrounding words) for an input word.

³⁵Actually, a Python implementation thereof: <http://radimrehurek.com/gensim/>.

During training of our model, input words and their context are taken from individual tweets. For example, consider the following tweet:

not Jony Ive @JonyIveParody


Happy birthday @iTunes Music Store! 13 years of flawless updates, perfect software, and pure UI/UX bliss. 🎂🎉🎈

3:56 AM - 29 Apr 2016





For the input token 🎂, the context is given by surrounding other emoji and words. Thus, when this tweet is used for training, it would strengthen connections in the hidden layer which result in predictions of tokens like *bliss*, *pure*, or 🎉. In this example, we would also train for a relationship between *UI* and 🎂. However, this pair is very unlikely to occur elsewhere in our dataset, while the tokens *birthday* and 🎂 likely occur together many more times.

After training, the neural network is tuned to predict the likely context for every input token. Hence, the output layer is a softmax regression classifier, outputting the probability for each output token. If we feed 🌧️ to the neural network it might, e.g., provide an output vector that assigns a high probability to the *rain* token. Keep in mind that the input is a one-hot vector: all values are set to zero, except the position representing the input token. While the hidden layer is a large matrix, a multiplication with the input vector essentially just selects one row from this matrix. If two input vectors result in the same output in the hidden layer, then their predicted context is also equal. From this follows that if two tokens share a similar context, their hidden layer weights must also be similar. We can thus use those weights directly to form the word vector for our word embedding.







As discussed above, we need a higher number of dimensions for our embedding to capture concepts of emoji. We chose a balanced 300-dimensional space for this: between the 100-dimensional word2vec default and the 500-dimensions used in the similar setup of Image2Emoji [35]. We experimented with different hidden layer sizes in the process, yet did not find noticeable differences. However, we only tested layer sizes between 200 and 400 dimensions. Very small layer sizes are likely not able to capture relationships between emoji that well. While we do not limit the vocabulary of the model (a word vector is trained for each word in the corpus), we ignore all words that occur less than 50 times. Given the large size of our dataset (after splitting all tweets, based on whitespace, we are working with 228 million tokens) this is very unlikely to affect proper words. However, this does limit the size of the input and output layers and thus helps with memory and time requirements of running word2vec.

One advantage of word2vec is that the amount of training data that can be used is not limited by memory. Thus, while our tweet database is 27 GB large, we can stream each tweet to the model during learning. We tokenize tweets by breaking them up at whitespace and emoji positions after removing all punctuation (excluding punctuation emoji such as ). This ensures, each emoji is an individual token, even when no whitespace separates it from a word or multiple emoji follow directly after each other. From the stream of tokens, we filter out tokens that are hyperlinks or Twitter handles (e.g., “@_CHINOSAUR”).

6.10.2.1 Exploring the Emoji Model

Once a word embedding for the tweet corpus is generated it can be used to query for emoji similarity. As stated earlier, we use the Euclidean norm to derive distances between pairs of emoji. In addition to basic ranking, word vectors also allow for more complex queries such as “*what is like*  *but not like* ” (answers according to our model, e.g., are , “*cakeday*”, or , i.e., other occasions for presents that are not Christmas). For more information on vector compositionality in word embeddings trained with word2vec, such as in the example above, see [197]. In this chapter, we do not further explore such multi-emoji relationships, but concentrate on pairwise similarity.

However, to get a more general idea of whether the model indeed learned relationships between emoji, a visualization of clusters is useful. As the model is high dimensional, it can not be plotted directly. We use an implementation [229] of *t-distributed Stochastic Neighbor Embedding* (t-SNE) [185] to reduce the number of dimensions from 300 down to 2 dimensions. Compared to other dimensionality reduction methods, t-SNE performs well at capturing local structure and making clusters in the data visually distinguishable.

The resulting visualization (see Figure 6.27), identifies several distinct clusters. On the right side, e.g., we can see all clock face emoji close to each other, as well as clusters of vehicles and writing-related objects. In the lower-left, food and animals form individual clusters. Here, subclusters around more specific themes can be seen as well. For example, animals living in the water form a distinct group, as do sweets or fruit. In the upper-left all smileys can be found. A closer look reveals child clusters for *loving*, *happy*, and *unhappy* faces. In general we see some emoji that cluster together tightly and others that are more loosely coupled to others. Examples of close clusters are astrological signs (e.g., ) , moon phases (e.g., ) , blood types (e.g., ) , or buildings (e.g., ). However, other emoji do not seem to connect as closely, e.g.,  is further away from others. But for some emoji this just shows that they are associated with multiple clusters or less tightly bound, compared to others in the cluster. For example, the  (dragon) is not as tightly bound to the animal cluster as other animal emoji. Note that while this visualization is useful in identifying groups of closely related emoji, it does not show any global similarity. Thus, two emoji that are on opposite ends of the visualization are not necessarily less similar than two emoji only half the width apart.

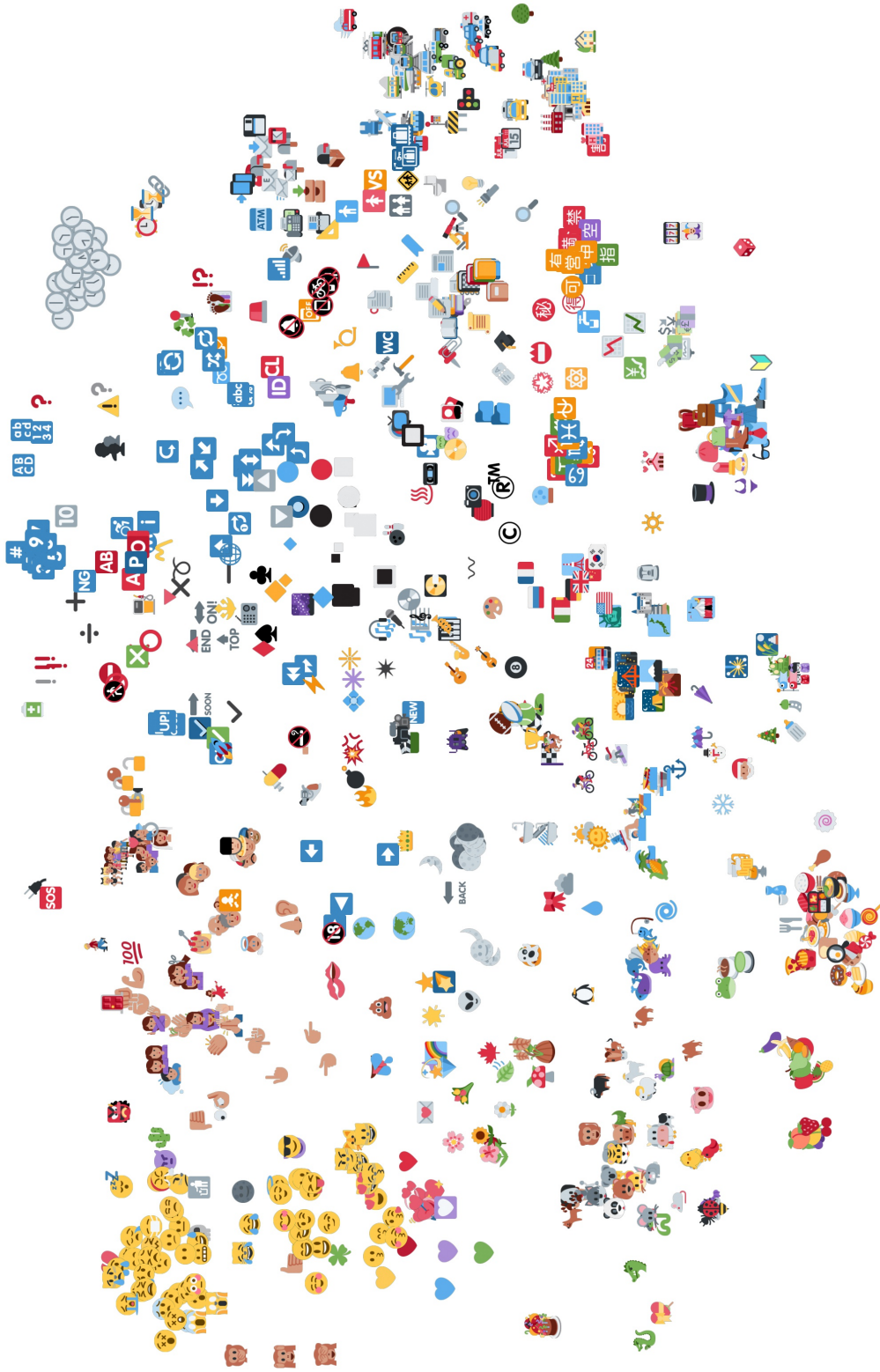


Figure 6.27: From tweet data we train a high-dimensional word vector model. However, this model cannot be directly visualized. Here, we show a two-dimensional embedding of the model, generated via *t-distributed Stochastic Neighbor Embedding* after an initial *principal component analysis* reduction. While the model contains vectors for every word in the training corpus, we show only the emoji here. Semantically similar emoji are clustered close together in this visualization.

Table 6.6: Top 10 most related emoji for several example emoji.

Emoji	1 st	2 nd	3 rd	4 th	5 th	6 th	7 th	8 th	9 th	10 th
(paw print)	(dog)	(dog face)	(cat)	(cat face)	(cat face)	(dog)	(rabbit)	(paw print)	(koala)	(tiger)
(heart with star)	(heart)	(two hearts)	(heart)	(two hearts)	(two hearts)	(kiss)	(heart with star)	(kiss)	(lips)	(angel)
(soccer ball)	(basketball)	(trophy)	(american football)	(baseball)	(golf)	(checkered flag)	(basketball)	(horse)	(United Kingdom)	(american football)
(Statue of Liberty)	(USA flag)	(night city)	(bridge at night)	(shopping bag)	(Japanese castle)	(night city)	(night city)	(stone face)	(Japan)	(train)
(tired face)	(tired face)	(tired face)	(tired face)	(broken heart)	(tired face)	(tired face)	(tired face)	(tired face)	(tired face)	(tired face)
(candy)	(cookie)	(wine glass)	(ice cream)	(magnifying glass)	(lollipop)	(cake)	(fried egg)	(ice cream)	(donut)	(rocket)
(party popper)	(fall leaves)	(balloon)	(gift)	(cake)	(fireworks)	(hot pepper)	(fireworks)	(beer)	(jack-o-lantern)	(clapping hands)
(octopus)	(fish)	(tropical fish)	(fish)	(octopus)	(dolphin)	(whale)	(whale)	(crocodile)	(turtle)	(rabbit)

Instead of showing relatedness for all emoji, we can also have a look at smaller groupings. Particularly interesting is which emoji are most related to a given one. As we saw earlier, the annotation-based similarity is limited in that it can only capture a category-centric notion of similarity. Instead, our emoji model allows to query for similarity due to related use of two emoji. In Table 6.6, we show a selection of emoji with their respective ten most related emoji. As can be seen, deeper connections are revealed. For example, the 🗽 is detected as being closely related to 🇺🇸, while they share no common Unicode annotations. Similarly, 😓 and 💔 also share no tags, yet are closely related in our model. This is a first indication, that our semantic model can indeed capture more detailed relationships between emoji. We confirm this later on in a crowdsourced experiment.

Another measure of model quality is whether it captures emoji diversity. It would, e.g., be problematic if all emoji were slightly similar, with no strong differences between pairs. However, Figure 6.28 shows that the similarity between emoji is actually heavily skewed. From all possible emoji pairings, only some are closely related, with the majority of pairings exhibiting medium similarity. This is a reassuring result as we indeed would expect the vast number of emoji pairs to only be slightly similar. For example, we would expect emoji such as 🌿 to only exhibit close relationships to a few other animal or ocean related emoji. With a larger range of other emoji, like 🏧, 🧑, or 🧒, we would not expect much of an overlap in actual use and thus no large similarity.

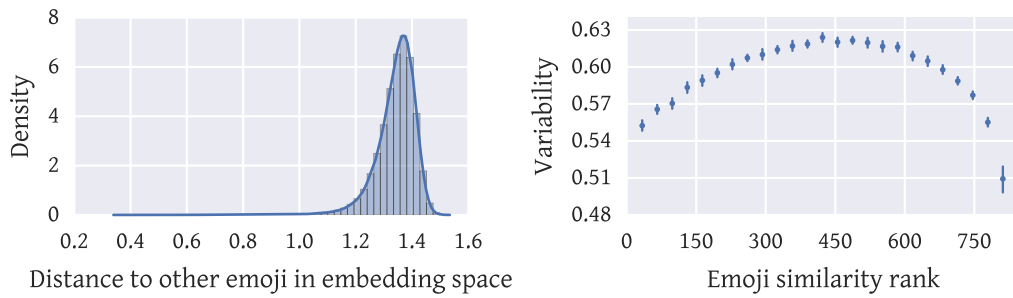
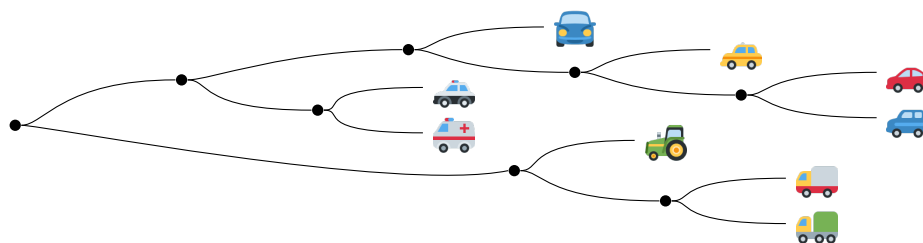


Figure 6.28: On the left, we can see that similarity between emoji (per our emoji similarity model) follows a skewed distribution. More closely related emoji pairs are found in the long left tail, while most pairs bunch together with medium distance. On the right, we explore whether there are some emoji that dominate the emoji similarity. For each of the 845 emoji, we sort the remaining 844 emoji according to the pairwise similarity. We can then, e.g., look at the set of all emoji ranked most similar. The variability (for each of the 844 possible list positions) describes how diverse such a set is. We aggregate several ranks here to better show the change in variability.

We also checked whether similarity is specific to the individual emoji. A naive approach could, e.g., always rank 😊 high, as this common emoji often occurs together with others. Hence, we looked at whether it was always the same emoji being highly ranked (e.g., 😊 again), or whether there was variability. Here variability is defined as the size of the rank set (e.g., the set of all emoji ranked second most similar) over the number of emoji (845 in our case). For example, if each emoji is ranked most similar once, the variability would be 1.0. Figure 6.28 shows how variability varies over the range of ranks. There is no strong bias and many different emoji are ranked first. For example, the most highly ranked emoji exhibit a variability of about 0.56, meaning that more than half the emoji appear at that rank (keep in mind this is always in relation to a second emoji, thus there are 845 individual rankings).

Finally, Figure 6.29 shows a hierarchical view of the emoji model. Where Figure 6.27 shows a global view of similarity, here related emoji are found in nearby nodes of the hierarchy. To generate this view, we use agglomerative clustering where nodes are successively merged (based on minimal distance) until a root node has been determined. We can look closer at one branch of the hierarchy:



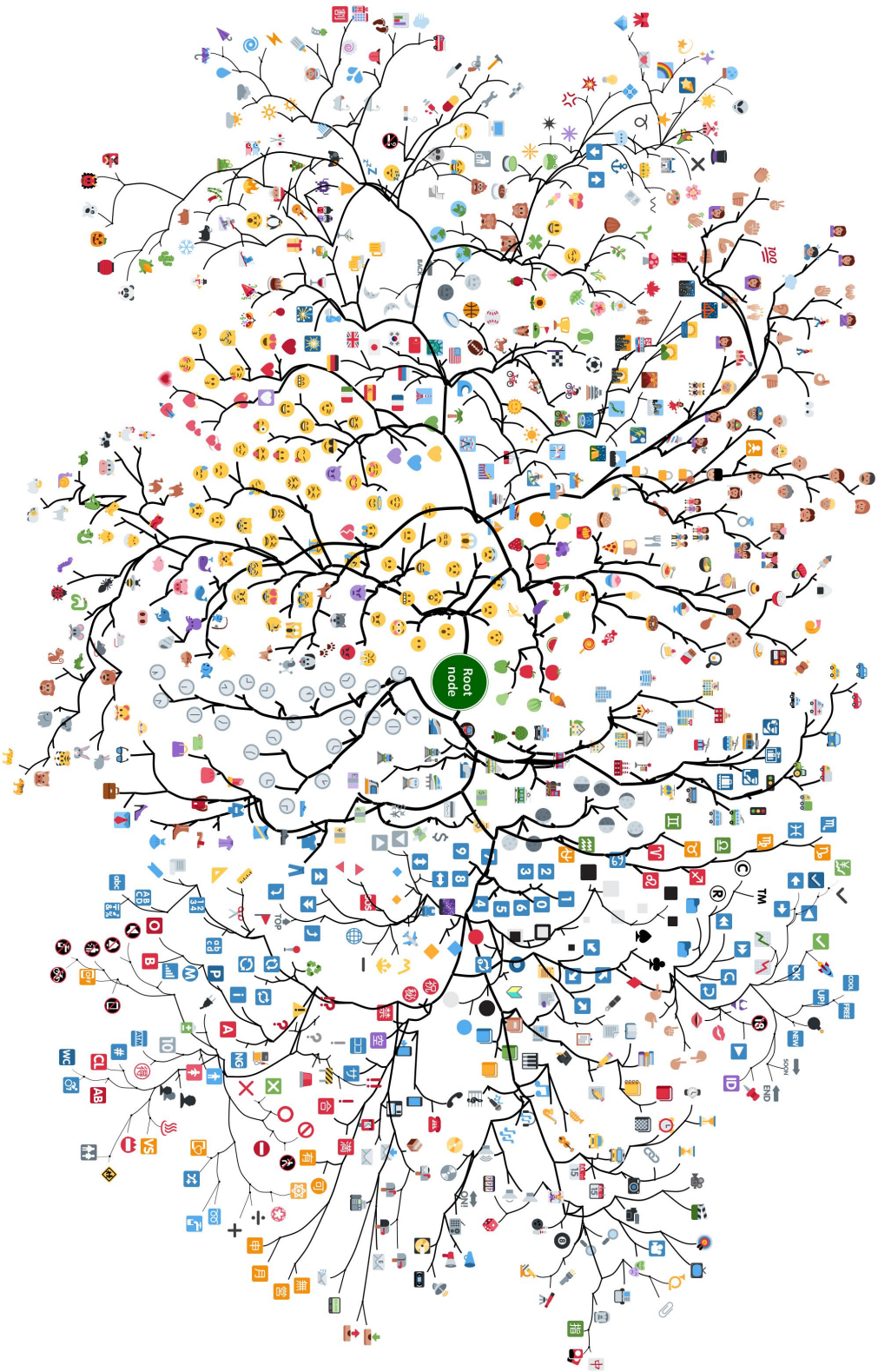


Figure 6.29: We use agglomerative clustering to organize emoji into a relatedness-hierarchy. Here we find related emoji close together in individual branches.

Here, we see that vehicles cluster together, with emergency and heavy vehicles forming their own subclusters. However, such a rigid clustering also creates artifacts due to the merge criteria (we, e.g., only find 🚑 in a nearby branch). We use *Ward's criterion* here, which tries to minimize the variance of merged clusters. A different approach would be to minimize the maximum distance in a cluster, more heavily penalizing outliers in a cluster.

However, not all odd locations are due to the clustering. For example, note that 🗼 (showing the *Tokyo Tower*) is shown as very related to 🇫🇷—more so than to 🇯🇵. Thus, people apparently interpret it erroneously as the *Eiffel Tower*. Such interpretations can be challenging: should keyboards arrange emoji as intended or as interpreted? And should the layout for Japan (where users probably recognize the tower) be different than elsewhere?

6.10.2.2 Exploring the Full Model

















































While we concentrate on the emoji-only model in this chapter, we actually have a more complete model for all tokens we used in training the emoji-only model. Here, we take a short look on how those other tokens relate to emoji and each other. Note that because we only trained on tweets, the word model has lower quality coverage than other models trained through word2vec. The usual approach is to include a larger secondary corpus to build a more in-depth model of word relationships. A common method, e.g., is to also use the English Wikipedia corpus during training.

Table 6.7 shows rankings for five example tokens. We show ranking for other word tokens and emoji separately, because other words are commonly much more related than emoji. Unsurprisingly, it can be seen there is strong semantic connection to different spellings of the same word. While we do not apply word stemming, this could be used to prune the vocabulary before building the model, in order to omit tokens such as “*birthdayyyyy*”, or “*loveeee*”. But we can also see that the closest matching emoji for the given query tokens capture the given token quite well. For example, all birthday-related emoji match the theme and could be used in birthday messages.

6.10.3 Evaluating Model Performance

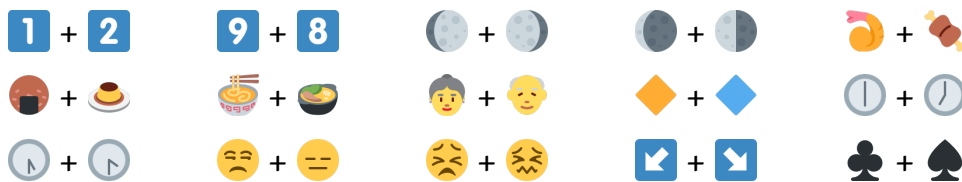
So far, we have shown overviews of the models and examples that demonstrate the quality of the results. However, we wanted to confirm the similarity predictions with data from a second source. Hence, we gathered human ratings of emoji similarity to compare with our computed similarity data from both the tag-based and the semantic model.

Table 6.7: Top 5 most related tokens and emoji for ten example tokens.

Token	1 st	2 nd	3 rd	4 th	5 th	1 st	2 nd	3 rd	4 th	5 th
Birthday	bday	birthdayyyy	birthdayyyy	18th	birfday					
Sad	upset	depressed	upsetting	saddening	heartbreaking					
Vacation	vacay	vacca	holiday	vacations	holidays					
Drunk	tipsy	stoned	intoxicated	sober	hammered					
Love	adore	looovoo	luv	loveee	loveeee					
Thesis	accounting	prelim	recitation	prelims	physics					
Dinner	lunch	brunch	supper	breakfast	meal					
Cute	adorable	cutee	cuteee	cuteeee	adorbs					
Angry	upset	frustrated	mad	grumpy	cranky					
Home	homeee	homeeee	homeeeee	homeeeee	homee					

We ran our crowdsourcing study on *CrowdFlower*³⁶, which, in turn, recruits contributors from a larger range of channels³⁷. In each task, we showed contributors ten emoji pairs and for each asked them: “How related are these emoji?” In the task instructions, we further specified that: “Two emoji could be related because they describe a similar concept, or because they are commonly used together. If you are unsure, think about how likely you are to use the two emoji together in a message.” Contributors gave their response on a 7-point Likert scale ranging from “unrelated” to “very related”.

To ensure rating quality, we designated a set of test questions for our task. We drew test questions from the emoji pairs where our semantic model indicates strongest similarity. After manually checking appropriateness and quality of those pairs, we selected the 15 most closely related ones as test questions. Hence, crowdworkers are expected to also judge these pairs as at least somewhat related and if they do not we can assume they are not properly working on the task. The selected pairs were:



While many of these pairs show the same thing (numbers, moons, arrows, ...), crowdworkers were also expected to recognize the relatedness of two different food items. In each task, one of the questions was automatically included from this test question set. If contributors did not answer this question correctly (indicating some level of relationship) they were excluded from the task as they are deemed unreliable.

The rest of the emoji pairs to rate were drawn from two different groups: (1) the 5% most related emoji pairs (per our semantic model), and (2) the remainder of emoji pairs. We excluded emoji pairs already used as test questions from this selection. The total number of emoji pairs is thus given as: $\binom{845}{2} - 15 = 356575$. From each of those groups, we randomly selected 45 pairs for inclusion in the crowdsourcing study. Hence, we use 45 out of the 17829 pairs in the top 5% and 45 out of the 338746 remaining pairs. For each of those 90 pairs, we collected 20 human ratings for a total of 1800 trusted judgments. Those 1800 judgments are trusted in the sense that CrowdFlower deems the raters reliable. Contributors are flagged as unreliable if they do not correctly responded to the test question in their task. Furthermore, before being served an actual tasks, contributors had to take a quiz composed of just test questions which, when failed, immediately flags them as unreliable.

³⁶<https://www.crowdfLOWER.com/>

³⁷<https://www.crowdfLOWER.com/labor-channels/>

Overall, 52 trusted contributors provided 2525 judgments (if we include the 98 rejected participants, who failed the quiz or are untrusted, the total number of judgments is 3265). However, 725 of those were test questions and only used to determine contributor reliability. This leaves us with a final 1800 judgments. Agreement of raters, per Krippendorff's alpha, is 0.39 (95 % bootstrapped confidence interval: 0.30 – 0.48). This shows rating relationships between emoji was no easy task for raters. We can thus expect some noise when relating this back to our model predictions.

6.10.3.1 Results

In our analysis we only consider ratings for the *top 5 %* and *remainder* groups, but exclude the test questions. Emoji pairs in the top 5 % percentile, per our semantic model, received a median rating of 4 (higher equals more related) in the 900 crowdsourced ratings. Yet, the 900 emoji pairs from the remainder of the dataset only received a median rating of 2. We used a Mann-Whitney's U test to compare the two groups and found a significant effect; $U = 544873.0, p < 0.0001$. Thus, emoji pairs considered the most related by the semantic model are also seen as significantly more related by human raters.

Contributors also had an easier time rating emoji pairs from the top 5 % group than from the remainder group. Krippendorff's alpha for the former is 0.42 (95 % bootstrapped confidence interval: 0.32 – 0.52), while the ratings for the remainder group only show an alpha value of 0.20 (95 % bootstrapped confidence interval: 0.07 – 0.35). This is likely due to the top 5 % group having more within-group similarity (per our semantic model), while the remainder group contains many emoji pairs with no clear connection. In such a case, some raters might see connections where others see none. For example, the relationship between 🎪 (barber pole) and 🇺🇸 is only apparent in some cultures and would likely lead to very different ratings by Americans and Germans.

So far, we have shown that our model's predictions are accurate on the level of distinguishing the top 5 % most related emoji from the rest. However, we also wanted to see whether similarity estimation holds on the individual emoji pair level. For this we check whether there is a significant correlation between the similarity scores given by the human raters and by the two models. All correlations in this section are given via the Spearman's rank correlation coefficient r_s . See Figure 6.30 for an overview of how correlation varies for different groups and comparisons. Figure 6.30 also shows bootstrapped 95 % confidence intervals for correlation, while we here only report averages. Over all emoji pairs with human ratings, we found that similarity predictions of the tag-based model and the human raters were correlated; $r_s = 0.50, p < 0.0001$. The distance predictions of the semantic model and the human similarity ratings showed similar correlation; $r_s = -0.37, p < 0.0001$. Correlation between random subsets of human raters is also at about the same level; $r_s = 0.39, p < 0.0001$.

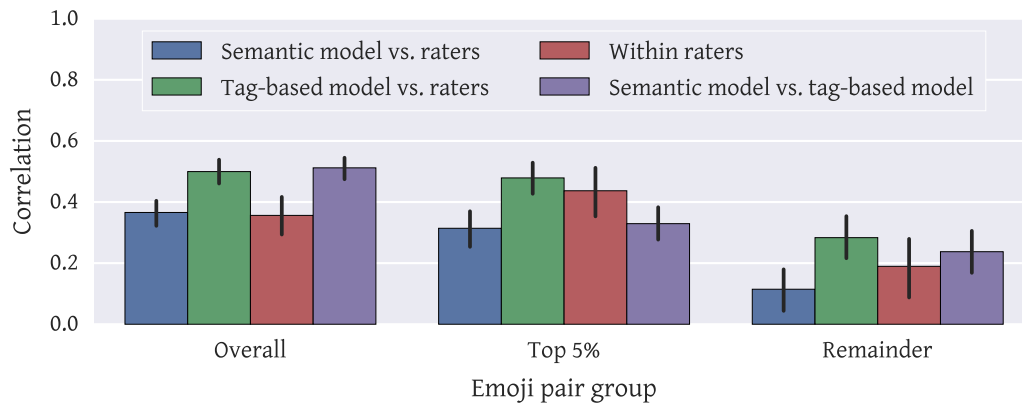


Figure 6.30: Spearman’s rank correlation coefficient for comparisons of human raters, the semantic and tag-based models. For comparison, we also show how human raters correlate among themselves (random split between workers). All correlations are significant. Error bars show bootstrapped 95 % confidence intervals.

Comparing human ratings and the tag-based model for the top 5 % group shows significant correlation; $r_s = 0.48, p < 0.0001$. The same holds for the semantic model; $r_s = -0.31, p < 0.0001$. Similarly, human ratings and tag-based similarity predictions for the remainder group are significantly correlated; $r_s = 0.28, p < 0.0001$. Finally, a significant correlation is also observable when comparing human raters and semantic model predictions; $r_s = -0.11, p < 0.001$. Within the human raters, the top 5 % group shows higher correlation than the remainder group; $r_s = 0.41, p < 0.0001$ and $r_s = 0.27, p < 0.0001$.

6.10.3.2 Discussion

In our crowdsourcing study, we saw significant correlation between human raters and both the tag-based and the semantic models. This shows that both models indeed capture differences in emoji similarity. However, there are differences between the two models. While both align with human raters, as we have seen, they sometimes disagree on the similarity of emoji. This is due to them focusing on different aspects of similarity: the tag-based model denoting whether two emoji show the same or a similar *thing*, while the semantic model captures a more fuzzy notion of relatedness, based on co-occurrence in or similar use. We hence need to further explore the differences between the two models.

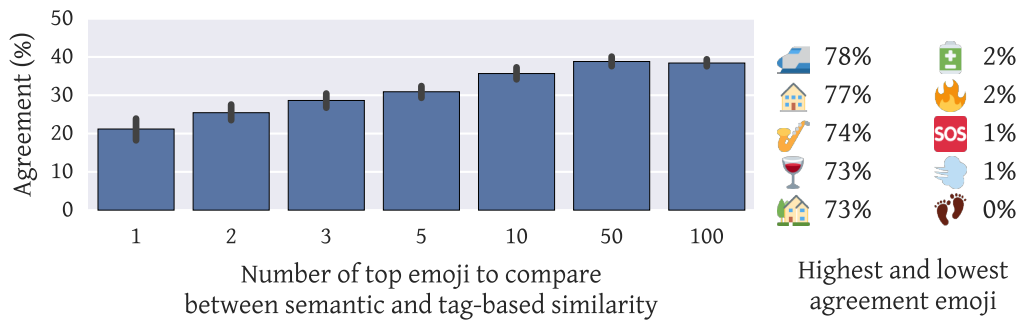


Figure 6.31: We derive agreement between the semantic and the tag-based model from the overlap in the top ranked emoji. For example, the two models have 30 % of the top five emoji in common. Agreement increases as more ranks are considered. Error bars show bootstrapped 95 % confidence intervals. On the right, we highlight the five emoji with the respective highest and lowest agreement (averaged over different rank sizes).

6.10.4 Comparing Tag-Based and Semantic Emoji Models

As we have seen, both models make similarity predictions that align with human raters. Yet, their predictions differ and they both capture different aspects of emoji similarity. Here, agreement between the two models shall be defined as the amount of overlap in the emoji similarity ranking they generate. For example, we could ask both models which ten emoji are most similar to 🧠, giving us two sets \mathcal{E}_A and \mathcal{E}_B . The agreement between them is then given as:

$$\frac{|\mathcal{E}_A \cap \mathcal{E}_B|}{|\mathcal{E}_A \cup \mathcal{E}_B|}, \quad (6.3)$$

which is equivalent to the Jaccard similarity coefficient from Equation 6.1. Note that we do not consider rank position in this definition of agreement, but only whether it is included in the other result set or not. After all, for reasonably small result set sizes it does not matter much which rank position an emoji holds. For example, whether an emoji is 3rd or 4th in a eight item result set has no strong impact on retrieval time for that emoji if all items are shown at once (e.g., in the autocorrect bar).

Figure 6.31 shows agreement between the two models for varying sizes of rankings. For example, when we only consider the top ranked emoji, the two models agree about 20 % of the time. Naturally, increasing the size of the respective result sets increases agreement (at size 844 every other emoji would be included and agreement would be a guaranteed 100 %). Overall, there is no strong agreement between the two methods. However, the level of agreement is not equal over all emoji, but differs depending on which emoji are considered. Figure 6.31 thus also shows the five emoji where the two models agree and disagree the most. We consider agreement over different ranking sizes when computing this average per-emoji agreement.

The two models have high agreement for emoji with a straightforward interpretation, but disagree where interpretation is more open. For example, 🚆 is likely to allow for less flexibility in interpretation than 🔥. The latter could be used to describe actual *fires*, trends that are *hot*, food that is *spicy*, people that are *attractive*, stores that are *busy*, and possibly many more scenarios. On the other hand, trains do not lend themselves to equal levels of ambiguity in interpretation.

We can take a closer look at these ten extreme examples of agreement and disagreement between the two emoji models. For each of these emoji, Table 6.8 shows the ten most similar ranked emoji by each of the models. As can be seen, where there is agreement the sets largely overlap. Even the ranking within the two results is close (the top two are always the same, for 🏠 even the top four are identical). However, when there is strong disagreement, the two result sets are very dissimilar. Yet, the way the results differ can tell us a lot about the strong suit of each of the two models.

A good example of model differences is their ranking for the 🔋 emoji. The semantic model makes several predictions that use *battery* as a stand-in for general *energy level*. For example, drinking a ☕ to wake up, or the quality of a wireless connection 📶. We can also see a direct link between the battery and the phone it is often in, with emoji like 📱 or 📴. Likewise, 📴 could be used to indicate a “dead” phone, but could also indicate that a user is very tired. While the presence of the 🐸 emoji might be puzzling at first, there is a strong connection between 🐸 and ☕. As an emoji combination, 🐸☕ is sometimes used to reference a meme on Twitter³⁸. On the other hand, the tag-based model can not make good predictions for the 🔋 emoji. This is primarily due to the fact that this emoji is only annotated with two tags: “battery” and “object”. Yet the “battery” tag is not shared with any other emoji and the ranking thus degenerates to a random selection of other objects. One could add additional tags to emoji, but another option might be to find connections between tags. For example, there is also the “electricity” tag, which is used by ⚡ and 🖱, but, surprisingly, not by 🔋 itself. The same holds for the “electric” tag, which is used for five emoji, yet also excludes 🔋.

Poor tag coverage is a fundamental problem of the tag-based model. Overall, 1175 different tags are used to describe the 845 emoji we investigated. Yet, while 🇬🇧 has 15 tags, 12 other emoji only are described by two tags (e.g., 😊, or 🌸). These are the extremes though and the median number of tags per emoji is 5. The different tags are very unevenly distributed though. While 274 emoji are tagged as “object”, 215 as “symbol”, 199 as “nature”, and 184 as “person”, use count quickly goes down and 719 tags are only used by a single emoji. For tag-based similarity, these single-use tags are detrimental and do not help with establishing relationships between emoji.

³⁸<http://blog.getemoji.com/post/134792876960/what-does-the-frog-and-teacup-emoji>

Table 6.8: The semantic and tag-based models differ in what they capture about emoji. Here we explore the differences for the five emoji where agreement (per Figure 6.31) between the two models is lowest/highest. Note that in this set only 🔥 is frequently used (23rd most common emoji in our Twitter dataset).

Emoji	Model	1 st	2 nd	3 rd	4 th	5 th	6 th	7 th	8 th	9 th	10 th	
Least agreement	👣	Semantic	📊	♻️	📱	👏	🚶	🚢	🚆	!?	3	🚗
		Tag-based	👏	👄	👂	🔔	👄	👉	👏	👔	👗	👕
	☁️	Semantic	🚒	🚶	🚑	📊	🚂	🚆	🚗	🚆	👉	🚂
		Tag-based	💧	💭	💬	💥	🎀	🌟	💧	💕	💢	💍
	🆘	Semantic	📉	😞	😞	🌟	🐾	🕒	🏠	😞	😞	😞
		Tag-based	🆘	🇯🇵	COOL	NEW	®	🚫	OK	©	NG	割
	🔥	Semantic	😵	🐷	💣	🌙	🔥	!!	💥	🔊	😾	合
		Tag-based	🔨	🔬	🔧	🚀	✂️	💉	🔍	🔍	🔪	🔦
	🏥	Semantic	🔌	😞	📊	🍲	💿	🚫	😞	🐸	😞	OFF
		Tag-based	📱	🖋️	🔗	🔬	🔧	💣	📖	🚰	🔔	🌸
Most agreement	🚗	Semantic	🚗	🚆	🚆	🚂	🚂	🚆	🚂	🚆	🚆	🚗
		Tag-based	🚗	🚆	🚂	🚆	🚂	🚆	🚆	🚆	🚆	🚗
	🏠	Semantic	🏠	🏛️	🏢	🏛️	🏛️	🏛️	🏢	🏢	🏢	🏢
		Tag-based	🏠	🏛️	🏢	🏛️	🏛️	🏛️	🏢	🏢	🏢	24
	🎷	Semantic	🎷	🎸	🎹	🎵	🎸	🎵	🎤	中	🎺	🎤
		Tag-based	🎷	🎸	🎸	🎹	🎵	🎵	🎵	🎧	🎤	🌟
	🍷	Semantic	🍷	🍹	🍺	🍺	🍺	🍴	💊	🍝	🍲	🍲
		Tag-based	🍷	🍹	🍺	🍺	🍺	🍲	💊	🍝	🍲	🍲
	🏠	Semantic	🏠	🏢	🏛️	🏛️	🏛️	🏛️	🏛️	🚗	🏢	👨
		Tag-based	🏠	🏢	🏛️	🏛️	🏛️	🏛️	🏛️	🏛️	🏛️	24

An emoji with better tag coverage is 👣, which is annotated as “body”, “clothing”, “footprint”, “person”, and “print”. Hence, the tag-based model matches this emoji with other body parts and clothing items. This is an emoji the semantic model does more poorly with. However, it does identify a connection between 👣 and walking/movement related emoji 🚶 (dash symbol) and 🚶. In fact, this connection to walking is not identified by the tag-based model. Checking back with the emoji clusters, as shown in Figure 6.27, we can see that 👣 does not connect well to larger agglomerations of related emoji. Additional training data might help strengthen some connections: we only collected 14609 tweets containing 👣.

Overall, the closer look at these emoji strengthens the impression that *object-related* emoji allow for more consensus in similarity estimation than emoji allowing for more abstract interpretations. However, it is precisely these emoji that add to the vibrancy of emoji use in messaging. Adding a 🏕️ when going camping makes the message more colorful and playful. Yet, the 🏕️ does not allow for a range of expression as wide as, e.g., the 🔥 emoji.







6.11 Summary

In this chapter, we have explored casual communication. In particular, the focus has been on emoji text entry. Starting from an introduction to emoji, we first looked at how emoji are used and how well current input methods support them. As we showed in a study of the state of the art, arrangement of emoji into categories is a weak spot of current emoji keyboards. Hence, we identify improvements to emoji ordering as an important direction for research on emoji.

The large number of emoji makes it hard to manually optimize emoji ordering though. We have motivated building an emoji similarity model as a way to allow automatic optimization of or reasoning about emoji layouts. This would also scale to different cultural context, e.g., by only considering German language tweets for a German emoji model. We have presented two candidate models to establish emoji similarity: (1) based on emoji annotations and (2) based on semantic information on emoji. In a crowdsourcing study, we have shown that both models correlate with human raters. Yet, what each model captures is quite different. Direct comparison of the two models indicates that semantic similarity is able to capture more nuanced relationships of emoji. However, this is prone to more noise than tag-based comparison. On the other hand, tag-based approaches have poor coverage for many emoji. As the semantic similarity approach does not rely on manual annotation, it scales better with the large number of possible emoji pairings.

Emoji have seen strong uptake and cultural influence. The fact that they are part of the Unicode standard also gives these characters a likely higher permanence than application-specific smileys or stickers. The *Unicode Character Encoding Stability Policies*³⁹ explicitly state that “Once a character is encoded, it will not be moved or removed.” Hence, emoji are here to stay.

But, as we discussed, entering emoji is quite different from entering regular text. Emoji have some unique characteristics, such as their lack of a clear phonetic interpretation and their visual nature. Existing keyboard layout optimization also does not translate well to emoji entry. Instead of a layout optimized for entering character sequences, emoji require input methods that optimize for search and exploration. Our emoji similarity model can inform this process. As we have shown, this model can be used to structure the emoji space and thus, e.g., inform category assignment. It could also be used to retrieve emoji fitting a current context (text or other emoji). Yet, while making emoji available via a retrieval method might work for a few users, it is likely inadequate for a large number of users. Exploration of available emoji is a critical aspect in emoji entry. While familiarity with all letters of Latin script is assumed in other keyboards, emoji keyboards cannot make the same assumption.

Entering emoji still means entering text, not uploading an image. Compared to entering Latin script though, appearance plays a crucial role. For example, while , , , , , and  all show closed books, they each have a distinct color. While there might be a clear mapping when users actually want to describe, e.g., a red book, most of the time the choice of book emoji comes down to taste. The word *book* itself does not pose the same trouble. It is only once the *book* becomes a *notebook* or *novel*, that the meaning changes. As the choice of book emoji thus depends on taste and might be different depending on mood, it is not sufficient to just show one of them. Exploring the available emoji and picking the right one for the current situation is a critical aspect of emoji entry.

This visual variability of emoji relates back to where we initially started: emoji can add playfulness to messaging and allow users to express themselves on a new level. The most frequently used emoji in our Twitter dataset all add to a text on the emotional level. Whether a message is followed by 😊 or 😞 can dramatically change the tone. For example, consider receiving either “Susan’s coming over later 😊”, or “Susan’s coming over later 😞”. Emoji are a first to make access to this kind of expression easy and ubiquitous.

Yet, the emoji entry we explored in this chapter is but one method to add such level of expression to messaging. As noted earlier, emoticons are the classical example of this kind of content. However, there are many possible extensions here and other means to give expressive power to users. We take a closer look at those methods, “beyond emoji entry”, that are possible paths to support the underlying goal: allow users to communicate in a playful and casual way.

³⁹http://unicode.org/policies/stability_policy.html

ⓅⓆⓇⓈⓉⓊⓃⓄ

Fraktur

Double

Handwriting

ZALGO

Figure 6.33: Unicode characters can be used to stylize text. The first four examples use characters from the *enclosed* and *mathematical alphanumeric* blocks while the zalgo text uses *combining marks*.

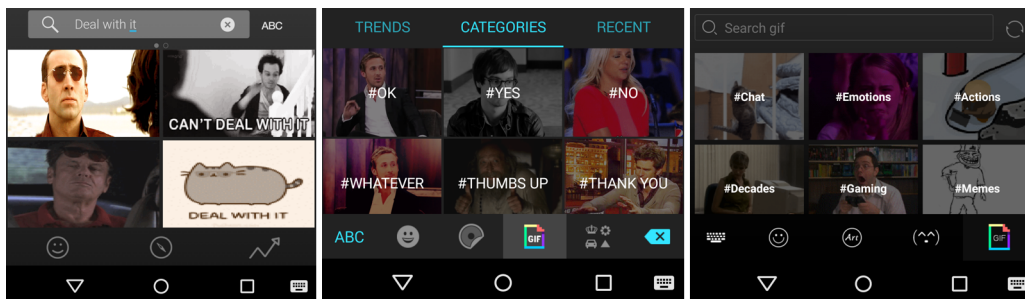


Figure 6.34: Some third-party keyboards also allow GIF entry in addition to text and emoji. Shown here are (1) Fleksy, (2) Ginger Keyboard, and (3) EmojiKeyboard Pro. GIF entry provides users another way to communicate emotion or mood. This is also apparent in the used GIF categories shown above, where users can choose GIFs that represent *#OK*, *#THANK YOU*, or *#WHATEVER*.

Unicode characters can also be used to stylize Latin script (see Figure 6.33). Here, instead of composition, appearance is changed by using stand-in characters. For example, instead of U+6B, one can use U+1D528 when entering the letter *k* to get the fraktur version *ƙ*. The Unicode *mathematical alphanumeric* block, e.g., contains several stylized variants of Latin letters, which allows to effectively control the font of messages where no font information can be transmitted. Finally, Unicode also contains combining marks that are designed to modify other characters. While this can, e.g., be used as a way to add accents to characters (such as in *ã*), it also allows to scramble text. One variant of this is *zalgo*⁴¹ text (Figure 6.33, right), which strives to give text an appearance alluding to insanity. As with Unicode emoticons, there is currently no convenient way to add combining marks to characters on mobile devices. However, some keyboards relegate characters with diacritics to menus attached to the respective keys (e.g., the German keyboard on Windows Phone 8.1 offers selection of “a” or “ä” from a popup, shown after long pressing the a key). Such a mechanism could potentially be extended to larger sets of alternative characters and diacritical marks.

⁴¹<http://knowyourmeme.com/memes/zalgo>

But Unicode characters, as emoji or emoticons, is only one way expression is added to messages. Another popular approach is to add images, e.g., in the form of memes or animated GIFs. Such content was just recently shown by Bakhshi et al. to be significantly more engaging than just text [8]. The Unicode consortium itself notes that, in the long run, applications should support arbitrary images in text:

*The longer-term goal for implementations should be to support embedded graphics, in addition to the emoji characters. Embedded graphics allow arbitrary emoji symbols, and are not dependent on additional Unicode encoding.*⁴²

Today, several third party keyboards such as *Flesky*⁴³, *Ginger*⁴⁴, or *EmojiKeyboard Pro*⁴⁵ already also allow “typing” images. Such images are either found by browsing tags or searching with a phrase (see Figure 6.34 for several interface examples). One common use of such images is in the form of a *reaction gif*⁴⁶, where an animated gif is used to represent approval, happiness, shrugging, or eye rolling.

With emoticons, emoji, and images, mobile text entry faces two challenges: (1) how to make the large set of existing content available, and (2) how to allow users to create their own content for personalized expression. For emoji we only need to solve (1), yet if arbitrary images and emoticons are allowed, creation becomes a crucial aspect. However, while there are several web-based tools and browser extensions for styling text or creating memes, many of those are not easily accessible on mobile devices. Yet being mobile need not mean that users are necessarily restricted to existing content. In fact, as demonstrated by *12Pixels*, custom-tailored applications are able to allow creative making on mobile device much more restricted than today's smartphones [316]. As text can often be insufficient as a means to communicate emotions, this kind of visual language can fill in the gaps and offer an additional means of expression. If we restrict input to just text or make it hard to express emotions, we cannot communicate with each other as effectively as possible.

⁴²http://www.unicode.org/reports/tr51/#Longer_Term

⁴³<http://fleksy.com>

⁴⁴<https://play.google.com/store/apps/details?id=com.gingersoftware.android.keyboard>

⁴⁵<https://play.google.com/store/apps/details?id=com.emoji.coolkeyboard>

⁴⁶See, e.g., <http://giphy.com/categories/reactions/>

7 Feedback in Casual Interaction

We perceive and are affected by changes too subtle to be described.

— Henry David Thoreau

So far, we have looked at casual interaction primarily from an input perspective. However, interaction is a two-way process and designing for lower engagement also means considering the feedback that is provided. In Chapter 3 we already saw an example of feedback that changes depending on the level of engagement with the device. However, there the focus was on the around-device interactions. In this chapter we will investigate those aspects of *casual feedback* more closely. Furthermore, this chapter also presents an overview of related work which explored more subtle feedback and/or engagement-dependent feedback. The following two chapters then describe concrete examples of casual feedback methods: Indirect light feedback in Chapter 8 and compression feedback in Chapter 9.

7.1 What Makes Feedback Casual?

Our definition of casual interaction is based on engagement and control. Feedback, though, is an inherent part of a closed-loop control system, at the foundation of interaction. Depending on the feedback a user receives, she updates her input and exerts control on the system. Hence, feedback with more relevant information to the task does theoretically enable users better choice of what they need to input to achieve their goals. Yet, there is a cost in processing feedback and more feedback thus not necessarily equals better interactions.

To explore the potential cost of higher fidelity feedback, consider a user engaged in a writing task. As she is writing her dissertation, she misuses a word, which is detected by her word processor. To notify her of this error, the word processor now puts a red squiggly line underneath the erroneous word. When she clicks on the word with her mouse, additional information becomes available, informing her which corrections she might want to consider. This kind of feedback tries to be non-intrusive and is initially very low-bandwidth (only communicating a binary error-present state). Users can register the red underlining, but continue with their writing unimpeded.

However, the word processor potentially has much more information available on the error than just its presence. Instead of underlining, it might instead, e.g., block out a part of the document and display a longer paragraph on why this is an error. For example, in case of a grammatical error, the corresponding rule could be displayed. Clearly, more available information in this example does not mean the user is able to better or faster respond to the mistake. Just parsing a lengthy paragraph of explanations imposes a cost on the interaction, when a basic choice of alternatives or fixes might have sufficed.

If we relate this back to casual interaction, then we can postulate that the level of feedback a user needs is equal to the one required for her desired level of control. In the above example, if a user cares more about grammar and has a wish to be precise and correct, having more information available might benefit her in her task. Yet, a user writing a short instant message to a friend likely cares less about this and might just want to complete the message as fast as possible, even if some writing errors remain. So on one hand we have a user desiring more control in the outcome of the writing, while the other user is willing to trade some of that control for a more convenient interaction. Hence, feedback that supports casual interaction would be adaptive to the current state of the control loop: providing more information to a user engaged in a tight loop, while lessening the cognitive load for a user in a looser control loop. If this state can not be automatically ascertained, then users might still be given a choice of feedback method or the ability to “escalate” the feedback if they see the need for more detailed information.

Yet, fidelity of the feedback is only one perspective one can take. Another aspect is *noticeability* of feedback. Feedback that is less noticeable can stay in the background for a while, not disturbing the user as much. For example, consider a phone ringing audibly or on mute. In the former case, a call is readily perceivable, even if the phone is not the current focus of attention. However, if a call is only shown as an on-screen status, it becomes much easier to miss it. At the very least, noticing such a call, on average, is going to take longer than when more disruptive feedback is chosen.

Indirectly, noticeability relates back to the control loop as well. As long as users do not perceive a feedback, they cannot react to it. Hence, lower noticeability results in looser control loops. But as with lower fidelity, users can gain something from lower noticeability as well. Where a ringing and vibrating phone is disruptive of the current context, a phone just pulsing the screen is more respectful of what a user is currently engaged in. Now, whether users want to be disrupted or not will change depending on the given situation. However, as with designing with casual input, designing more subtle and less disruptive output enables systems to support a wider range of engagement levels with the device.

7.2 Implementing Casual Feedback

Our general definition of casual feedback calls for a feedback mechanism that scales from low fidelity, subtle feedback all the way up to high bandwidth or intense feedback. Yet, this continuum is not equally well explored. There already exist a large number of high fidelity and intense types of feedback. For example, current mobile systems can always make a lot of noise or vibrate ferociously to grab our attention. On the desktop, the large and high-resolution screen similarly already covers the requirement for high-bandwidth feedback.

However, more subtle and lower fidelity feedback methods are less prominent. Yet, if a hand-bandwidth feedback channel is already available, we only need a secondary lower fidelity channel to achieve coverage of the feedback spectrum. Thus, in this dissertation there is a focus on feedback techniques that can support this approach. They do not need to stand completely on their own or scale all the way up to high-bandwidth and intense (as we will see though, compression feedback does cover a larger part of this range).

Designating different feedback mechanisms to different ranges of the casual interaction continuum can potentially aid users as well. Instead of needing to configure one feedback technique for changing requirements, they can just switch on or off individual mechanisms to constrain the feedback to a range suitable for their currently desired engagement. This behavior can be observed in mobile phone use where ringer, vibration, and screen can already be controlled individually. A user can thus, e.g., keep the screen on if she desires to be able to immediately read an incoming notification, or turn it off and only receive an auditory signal for it. Another example of this are wake-up lights, such as the *Philips HF3470*¹. For 30 minutes before the alarm time, they slowly increase light intensity to gently wake up the user. However, in the last 90 seconds an audible alarm also fades in. Thus, should the more subtle feedback method fail, the system automatically escalates to a more intrusive approach.

7.3 Related Work in Casual Output

There is a wide range of existing work relating to casual output. Here, we take a look at both (1) work investigating more subtle output, and (2) work on feedback with multiple levels of complexity. While this helps with describing the general concept of feedback for casual interaction, it also sets up the background for the following two chapters. In those, we investigate two concrete casual interaction feedback methods.

¹http://www.usa.philips.com/c-p/HF3470_60/wake-up-light/overview

7.3.1 Subtle Feedback

Human attention is a limited resource and how to do feedback that takes this into account has been an area of intense research. An early example is the *Reminder Bracelet*, which uses three LEDs on a wristband to enable subtle notifications [101]. Costanza et al.'s *eye-q* is a more refined version of the same principle [58]. Here, four LEDs are placed on the end of both arms of a pair of glasses—sitting in the corners of the lenses. In two studies, they evaluate the factors that influence the perception of these subtle, peripheral stimuli. They find that higher workload, dimmer illumination, and slower animation all lead to less noticeability of the feedback. In *NotifEye*, Lucero and Vetek display virtual butterflies in see-through interactive glasses [181]. While this version makes no attempt to display particularly realistic butterflies, such a system could be extended to have the notifications *blend in* more with the world by rendering more lifelike butterflies.

While the above works used visual subtle notifications, Pielot and de Oliveira instead applied vibrotactile feedback [232]. They set up their prototype to constantly vibrate, but tuned down vibration to be just barely noticeable. Users then had to detect the stimulus being *switched off*. Slow reaction by participants showed that the stimuli did not occupy the focus of attention and that it is a viable form of subtle feedback.

7.3.2 Feedback with Multiple Complexity Levels

A great example of a device with different feedback levels is Tanahashi and Ma's *Stock Lamp* [292]. They designed three versions of a stock analytics software: (1) active and focused interaction, (2) passive and focused interaction, and (3) passive and peripheral interaction. Each version is associated with an engagement mode, which is inferred from the level of attention (users are assumed to be attentive when they look at the screen), and users' interactions (i.e., use of mouse or touch). They compare these engagement-dependent designs against a standard dashboard design and find it is, e.g., better suited for multi-tasking.

We already encountered Vogel and Balakrishnan's work on interaction zones for public displays in Chapter 2. In their system, the interactions and content changes depending on how much users engage with the display [304]. For example, some labels are only revealed up close and stay out of view when users only walk by and glance at the display. Similarly, while public calendar information is always available, it only gets augmented with private calendar data once users engage with the display.

7.4 Towards Specific Casual Output Systems

So far in this Chapter, we have looked at the general aspects of output for casual interaction. The next two chapters detail two specific systems for casual output. Where Chapter 8 describes using indirect light for feedback, Chapter 9 is on compression feedback. The former method attempts to fill a gap for mobile notifications: more subtle feedback. The later strives to cover a wider range of attention capture, but also focuses primarily on notifications.

Both chapters hence are intimately connected to the principles discussed in this chapter. However, they are split over two distinct chapters as, while they share conceptual aspects, they both go into completely different direction with the actual feedback they provide. Where indirect light feedback is non-intrusive and only visual, compression feedback can be more forceful and, as a haptic feedback method, exerts actual forces on the user.

8 Casual Feedback: Indirect Light Feedback

*There are two kinds of light — the glow that illumines,
and the glare that obscures.*

— James Thurber, *Lanterns and Lances*

As described in Chapter 7, one way to have more casual feedback is to make it more subtle. In this chapter, we take a look at a novel type of feedback that does just that for smartwatches: *indirect light feedback*. For this kind of feedback, light is emitted into the skin and only the remittance can then be observed by the users—hence the naming as *indirect*. One major use case of smartwatches is display of notifications, but while not every notification is important, smartwatch feedback is equally disrupting for all of them. Smartwatches commonly use vibrotactile feedback, which can be annoying and does not allow for acceptable prolonged feedback. Our goal with indirect light feedback is to not disrupt the wearer in the same way as vibration feedback. Its use of light scattering in the user’s skin also keeps the display free as a primary feedback channel and also connects more naturally with the user’s body.

Use of indirect light feedback was explored using a custom prototype that fits into a watch form factor. While the prototype is presented here, the focus of this chapter is on the evaluation of the feedback. By applying an in-the-wild study approach, we were able to gather performance data in more realistic settings than a lab study would have allowed. We compare the results to similar work and show how indirect light feedback works well as a subtle feedback modality.

This chapter is based on a paper¹ published at *Mobile HCI 2016*:

- Henning Pohl, Justyna Medrek, and Michael Rohs. “ScatterWatch: Subtle Notifications via Indirect Illumination Scattered in the Skin.” In: *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services companion - MobileHCI '16*. 2016. DOI: 10.1145/2935334.2935351

¹Which in turn is based on work done by Justyna Medrek for her bachelor thesis [195].

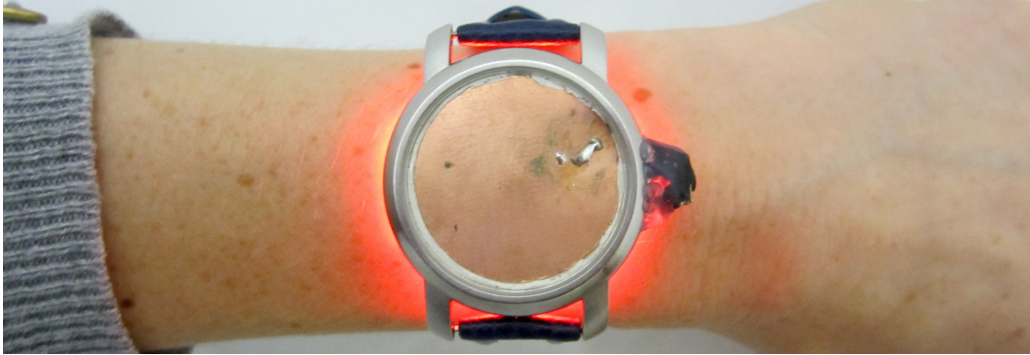


Figure 8.1: We added LEDs to the bottom of a watch case for use as a notification mechanism. The red light can pass through skin, scatters, and can be observed all around the watch. This enables a more subtle means of feedback—not as disruptive as vibration and more connected to the body than outward-facing illumination. This kind of indirect illumination feedback is still noticeable to users and visible through some clothing.

8.1 Introduction

Apart from telling the time, displaying notifications is the primary function of smartwatches [233]. They fit this role well as the watch form factor is uniquely suited for glanceable, quick, and convenient access [184]. However, interruptions due to many notifications can even induce inattention and hyperactivity in users [168]. But at the very least, undesired interruption is annoying, especially for non-critical notifications. We might be happy with our smartwatch vividly buzzing on our wrist when an important message comes in, yet would be less appreciative when this happens for every *like* on Instagram.

An approach for less disruptive notifications are subtle feedback methods, such as peripheral light indicators mounted to glasses [58]. However, a glass form factor is not suitable for everyone. Instead, we propose embedding LEDs into the back of smartwatches to create subtle light feedback on the wrist. Emitted light scatters in the skin and creates a subtle glow around the smartwatch (see Figure 8.1). This also keeps the smartwatch screen free for interactions and high-bandwidth feedback, or allows it to be switched off.

Smartwatch aesthetics have been identified as an important factor for their adoption [184]. Indirect light feedback forms an immediate connection between the watch and the arm. On an aesthetic level this extends the design space of future watch designs. But the visible nature of the feature also potentially makes such notifications more socially acceptable, as others can observe them as well [102].

8.2 Related Work

In addition to the related work already discussed in Chapter 7, indirect light feedback also relates to the larger space of LED feedback and specific feedback methods for smartwatches.

8.2.1 LEDs for Feedback

Almost every piece of consumer electronics contains one or more LEDs. However, their use for feedback is commonly limited to just a binary on/off state indicator. LEDs allow for a much richer output repertoire though, as explored by Harrison et al. [103]. We similarly play back several different patterns, e.g., fading the light intensity in a sinusoidal fashion.

Xu and Lyons explore smartwatch designs that replace the front screen with an LED-based design [323]. Their *Shimmering Smartwatches* show how such low fidelity feedback can still support common smartwatch interactions.

8.2.2 Novel Feedback on the Wrist

Current smartwatches and other wrist-worn wearables commonly include vibration feedback and visual feedback. However, just as we present a new form of visual feedback, several projects have explored feedback modalities beyond those two.

Instead of just one vibration motor, *BuzzWear* arranges three motors on the wrist [172]. Compared to basic vibration feedback, this also allows for playback of tactile patterns. They showed that 24 presented patterns could be well distinguished and were fast to detect even when engaged in a distractor task.

Baumann et al. explore haptic feedback systems that emulate human attention capturing [14]. They mount two different actuators on a “*watchface-like body*”, attached to a wristband: (1) a *squeezing* mechanism to constrict the strap, and (2) a *tapping* mechanism that actuates a foam “finger”. Depending on the actuation method, they find that this feedback can elicit relaxing or agitating sensations. Squeezing was also explored by Song et al. [277]. They also looked at thermo feedback on the wrist using a pair of peltier elements. However, users had a hard time distinguishing thermal stimuli. Thermal feedback also runs the risk of being uncomfortable if used outside of a small actuation range [319].

Pasquero et al. investigated haptic feedback on the wrist as well, but fit their actuator into a wristwatch form factor [224]. They embedded a piezoelectric transducer in the bottom of their prototype. When activated it changes to a dome shape and presses against the user’s wrist. This can provide a sensation users compared to a “*strong heartbeat pulse*”.

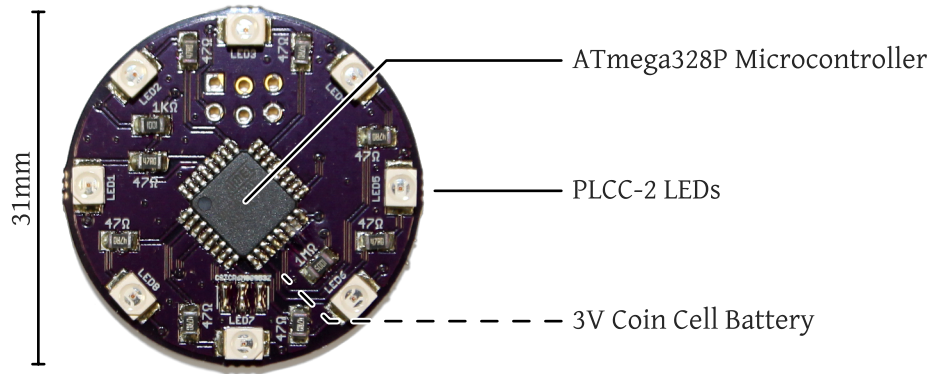


Figure 8.2: The central part of our indirect light feedback prototype is a custom PCB containing 8 LEDs. The LEDs are controlled with an ATmega328P microcontroller and powered from a coin cell battery mounted on the back of the PCB. This board is sized to fit snugly into an empty watch case.

A mechanical feedback mechanism is used in *Skin Drag Displays* [130]. Here, a pin inside a wrist-worn prototype moves over the skin, exerting tangential forces on it. This is used to draw shapes on the skin—a form of gesture output. Compared to vibrotactile arrays, such dragging feedback makes it significantly easier to distinguish different gestures.

8.3 Prototype

We designed our prototype to be as similar as possible to a traditional watch. For this purpose, we acquired an empty wrist watch case with attached strap and intact front glass, but without the back of the case, from a flea market. This circular watch case is 7 mm thick and has a diameter of 4 cm. From lug to lug it measures 44 mm. With all internal clock parts removed, this gave us the space to embed custom electronics inside the watch. As the clock face was missing, there is a problem of light leakage out the front of the prototype. We solve this by sealing the front glass with a glued in opaque cutout (made out of a thin copper sheet with plastic backing).

We designed a custom circular (31 mm diameter) PCB to fit inside the watch case (shown in Figure 8.2). It primarily holds an *ATmega328P* microcontroller, a coin cell battery holder, a 6-pin programming port, and eight LEDs along the perimeter. For the LEDs, we chose the *ASMT-URB4-YU802* from *Avago Technologies*. These red LEDs measure 2.8×3.2 mm and are 1.9 mm high (standard PLCC-2 package). When operating at the rated 20 mA, they provide 1.4 cd of luminous intensity. Light is emitted with a viewing angle of 120° and at a dominant wavelength of 623 nm.

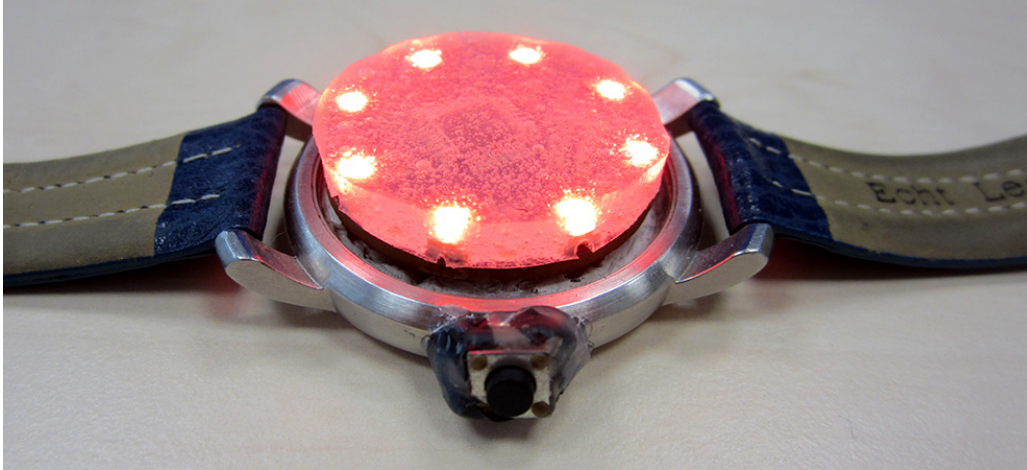


Figure 8.3: We cover the LEDs in a 4 mm thick layer of clear silicone for added comfort. This prevents the hard edges of the LED housings from irritating the skin, while still allowing light to pass through.

Because of the small form factor, we did not include wireless capabilities or additional permanent storage. This complicates use of the device for logging, though. While the *ATmega328P* has 32 kBytes of non-volatile Flash memory, it can only be written to during upload of a new program. Hence, it is not possible to write log data to Flash memory at runtime. There are also 2 kBytes of RAM, but as this is volatile memory it is easily lost if, e.g., the battery is accidentally disconnected before log data can be copied to a computer. We hence opted to use the built-in 1 kByte EEPROM to store log data. As this is non-volatile it can be read after a study has completed, without risking data loss. However, the small size of the EEPROM restricts how much data can be saved. We hence reduced the resolution of saved timing data to seconds, instead of the milli- or microsecond resolution available to us. Ideally, logging data would be transmitted via Bluetooth to a nearby phone for storage. However, in addition to required footprint on the board, this would also increase the power requirements of the prototype. With this version we hence opted for the safer on-device logging.

We cover the whole side carrying the LEDs with a 4 mm thick coat of translucent silicone (*SORTA-Clear 37*, see Figure 8.3). This provides added comfort by padding the sharp edges of the LED cases, while allowing light to pass through. When wearing the prototype, the silicone compresses and the prototype sits just above the wrist. If worn loosely, some light also shines through the lower gap at the side (see Figure 8.4 for a comparison). This is visually not distinguishable from light scattered from the skin. However, the overall illumination is slightly more intense due to this escaping light. In future smartwatches, LEDs could instead be put behind a custom glass watch back.



Figure 8.4: When wearing the watch directly on the arm (left) it sits tighter than when using the silicone padding (right). The slight gap due to the silicone also leads to additional light on top of the scattered one.

In the place reserved for the crown, we instead attach a small push button by gluing it to the frame. This is used for acknowledging any feedback from the watch. To limit the chance of accidental activation, we programmed the button to only trigger an event if held down. When reacting to a stimulus, the button needed to be pressed for 800 ms. When no stimulus is shown, holding down the button for 5 s powers down the watch, the same action starts it again. Both actions are acknowledged with an LED animation. This functionality is used to, e.g., turn off the watch during the night.

The prototype is powered by a CR1620 coin cell battery (see Figure 8.5). We use lithium batteries with 70 mAh capacity, that provide 3 V nominal voltage. As our prototype spends most of the time in power down mode, this battery lasts for more than the required day of operating time. In power down mode, only the watchdog timer remains active and the ATmega328P draws about $4.2 \mu\text{A}$. During LED operation, the power draw is much higher. Depending on active mode and PWM settings, the LEDs require 20–160 mA.

8.3.1 Choice of Light Color

Most of the light emitted by our prototype is not observed directly, as the LEDs are directed towards the skin. What is seen instead is the remitted light that passed through the skin. As illustrated in Figure 8.6, only about 5% of light is directly reflected by the skin. The majority of the light is either absorbed by the skin or scattered inside of it. Such scattering is caused by inhomogeneities inside the skin, which also leads to diffusion of the light as it passes through the skin [3]. This leads to a much larger area remitting light, compared to the area that received light in the first place.



Figure 8.5: The bottom of the PCB holds a coin cell battery with enough power to run the prototype for a day. We sealed the front of the watch to prevent light leakage to the front. The used putty also holds the PCB in place and prevents damage due to free movement inside the case.

However, different types of light behave differently inside the skin. Some light is absorbed more, while other light more easily passes through the skin. As shown in Figure 8.7, the amount of remittance depends on the wavelength of the light. Optimal remittance for fair Caucasian skin is achieved around 700 nm, while remittance for darker skin peaks at about 1100 nm. Unfortunately, this puts the optimal light well into the infrared spectrum—making it invisible for users. We hence settle on light with a slightly lower wavelength, in the red part of the spectrum. The 623 nm LEDs we picked are a compromise between size, wavelength, and luminosity.

While red light performs best for feedback through indirect illumination, this does not mean other light colors are impossible to use. However, the effect is much weaker and thus stronger LEDs need to be used, which comes at a cost to power efficiency. Optimizations to LED placement (more towards the edge) or orientation (angled outwards) could also help, but complicate construction and pose tighter design constraints. If one only wants the light effect without the glowing of the skin, another option also is to shine light on the skin from above. One example of this is Laput et al. using laser projectors for skin-projected buttons [169]. This does allow for more colors, but, on the other hand, does not allow the light source to disappear underneath the watch.

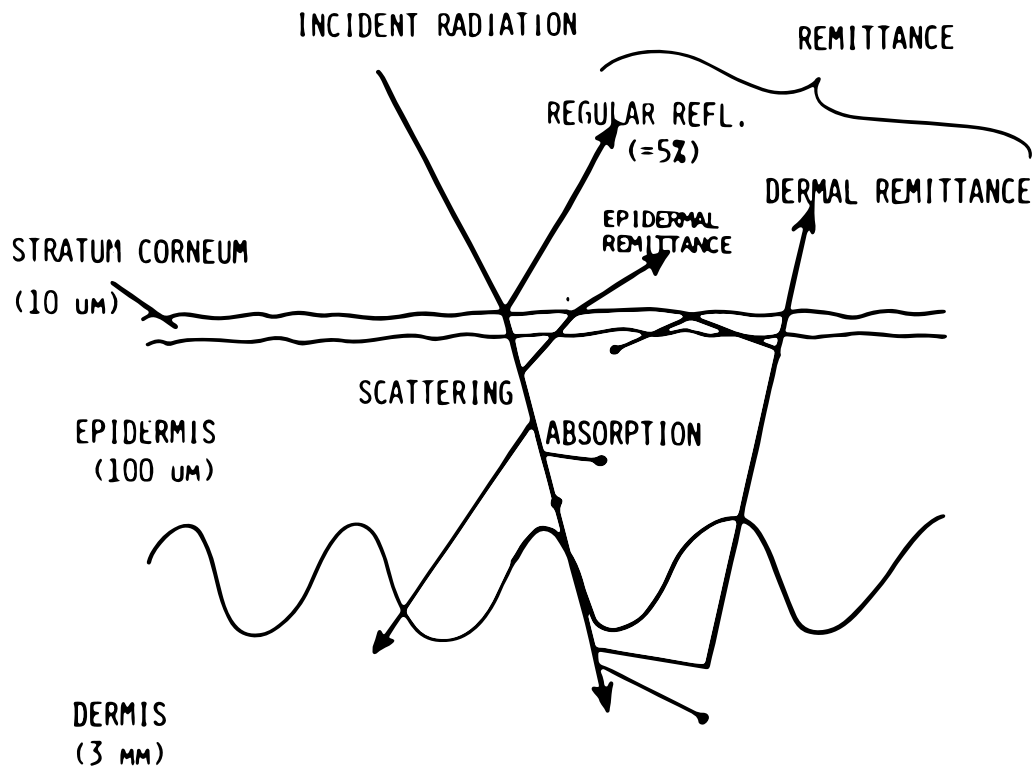


Figure 8.6: Light can be reflected by or transmitted through the skin in different ways. For indirect light feedback, we are interested in the light that is remitted from the skin. This remittance can happen at different levels of the skin. Light which scatters and is only remitted after a longer path through the skin is particularly interesting, as it is responsible for much of the glowing skin area of indirect light feedback. Reprinted from *Journal of Investigative Dermatology* 77(1), Anderson and Parrish, *The Optics of Human Skin*, pp. 13–19, Copyright 1981, with permission from Elsevier [3].

8.3.2 Feedback Modes

Instead of just running all the LEDs at full power constantly, we implemented several feedback modes that modulate the light output. Even with just one LED, the number of possible patterns is large, as demonstrated by Harrison et al. [103]. For example, LEDs can blink at different speeds, twinkle, flicker, or fade in and out. Having multiple LEDs opens up an even larger design space and allows for rich messages. For example, Campbell and Tarasewich experimented with three LEDs as an information display and found that up to 5 bit of information could be processed by users without a decline in performance [34]. Where one LED only allows for intensity patterns, multiple LEDs also allow design of animations. After piloting, we limited ourselves to a set of seven different modes (also illustrated in Figure 8.8).

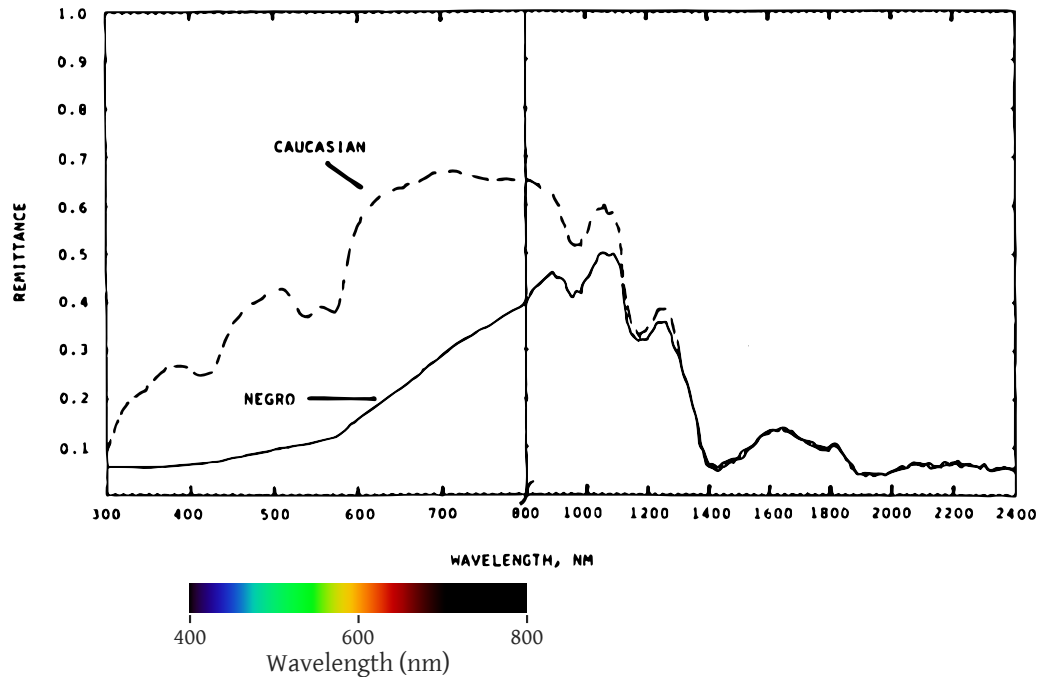


Figure 8.7: How well light can pass through the skin is dependent on its wavelength. The top plot shows cutaneous spectral remittance curves for light and dark skin (figure labels left as in original [3]). As can be seen, light between 600–1100 nm in wavelength best passes through the skin. However, only a small part of this range is inside the visible spectrum (shown below). This means that while infrared light would work much better for both skin types, this kind of feedback would no longer be visible to humans. Accordingly, red light is the clear choice for indirect light feedback. Top plot reprinted from *Journal of Investigative Dermatology* 77(1), Anderson and Parrish, *The Optics of Human Skin*, pp. 13–19, Copyright 1981, with permission from Elsevier [3].

The chosen LED patterns were:

- Illuminating all LEDs
- Only illuminating the LEDs on the left side of the watch
- Only illuminating the LEDs on the right side of the watch
- Illuminating all LEDs, but reducing the intensity
- Blinking all LEDs at a frequency of about 1 Hz
- Illuminating two LEDs, but moving which ones are lit up clockwise every 1/8 s. This gives a rotating animation running at about 1 Hz
- Oscillating all LEDs in a sine wave pattern (about 1/2 Hz) between full and no intensity

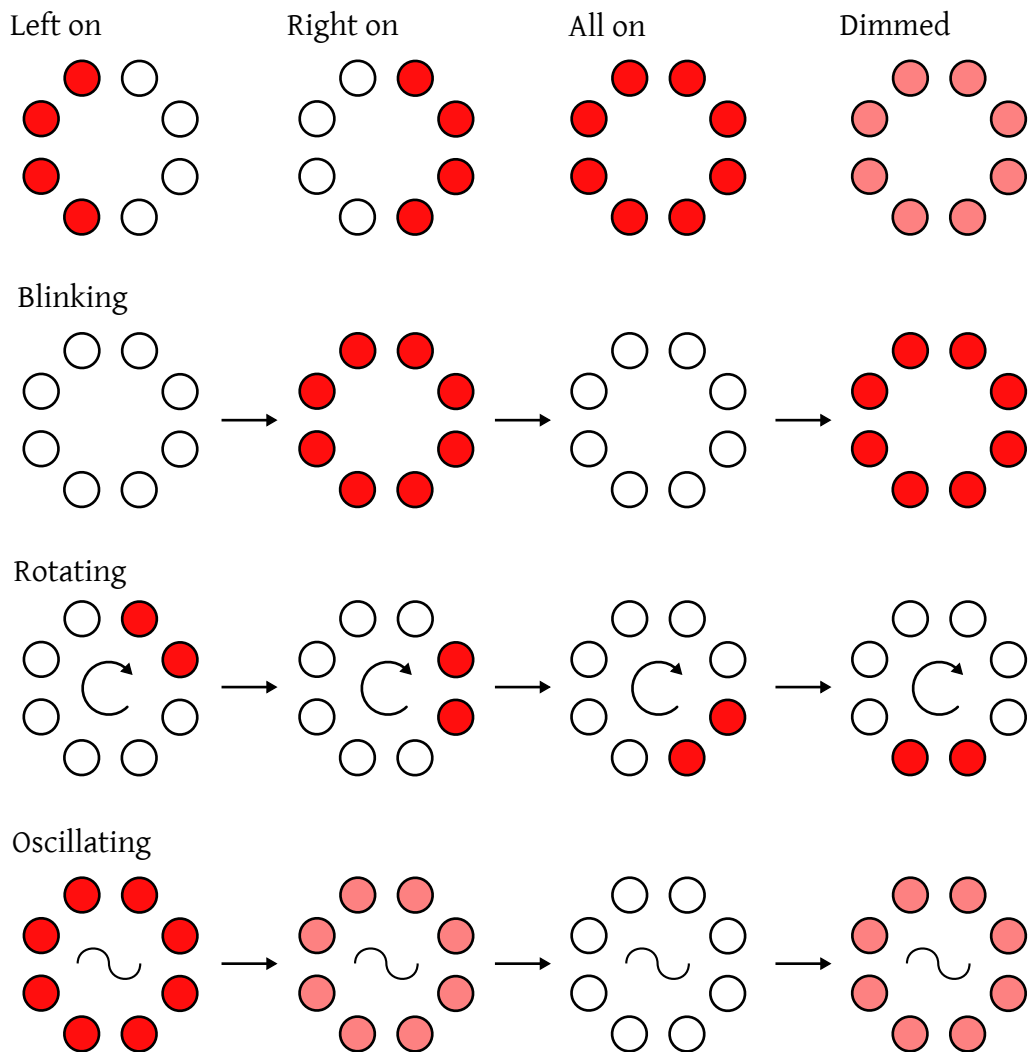


Figure 8.8: The LEDs in our prototype can display seven different patterns. There are four static patterns: only left/right side on, all on, and all on, but dimmed to a lower intensity level. Furthermore, we designed three animations. In the blinking animation, all LEDs toggle between on and off. In the rotating pattern only two LEDs are on at a time. Which two are on changes in a clockwise pattern with a full revolution at about 1 Hz. Finally, the oscillating pattern is similar to the blinking pattern, but instead of just toggling LEDs smoothly fades between on and off states.



Figure 8.9: Users in our study wore the prototype for one day while tending to their usual affairs. Thus, our prototype watch was used in a wide range of situations, such as at the office, at home, or outside. For each stimulus participants reacted to, they recorded their current activity in a diary.

8.4 Evaluation

While our prototype demonstrates, that indirect light feedback can be implemented in a smartwatch form factor, it is not yet clear whether it is also viable as a feedback mechanism. In particular, we worried that it might be too subtle and users might not be able to see it very well. We thus conducted a user study to determine how well users can perceive such stimuli.

A critical aspect when evaluating notification mechanisms is distraction. If users only need to concentrate on the feedback, reaction times will be very low, as the performance then only depends on how fast users can physically acknowledge a stimulus. Thus, lab studies commonly include distractor tasks to capture participants' attention elsewhere. However, the impact of such measures can vary significantly between different distractor tasks [58]. Another critical aspect with our device is lighting conditions, as the level of ambient light influences how well the feedback can be seen.

We thus opted for an in the wild approach, similar to the one taken by Cauchard et al. in their *ActiVibe* project [39]. Instead of tightly controlling all factors, we make use of the natural situational diversity (for a discussion of the benefits of field studies see, e.g., [152]). This enables us to gather more realistic data on the feedback and increases ecological validity. Participants wore the prototype for a whole day, going through their normal routine (see Figure 8.9 for some examples). In addition to evaluating the performance of the feedback, this also allows us to capture how using this kind of feedback feels to users—an aspect where running a study in the lab does not provide adequate data. For example, one might assume that having a glowing wrist might be uncomfortable for some. They might, e.g., feel like the feedback is drawing unwanted attention to them.

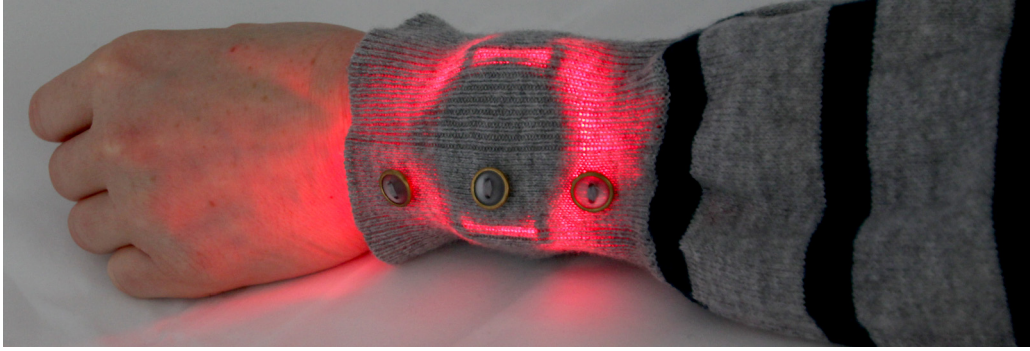


Figure 8.10: Some clothing (e.g., light sweaters) allow the light to pass through. However, thicker and/or darker clothing prevents the light from being seen. While this is less of an issue in the summer, winter clothing can prevent the use of this kind of feedback.

There are also drawbacks to our experimental approach. As we have no control over the situations participants encounter, we can not balance conditions. One participant, e.g., might use the watch only use at home, while another one goes outside for most of the day. The way participants report on their day likely also varies. What one considers dim light, the other might rate as regular light. As we depend on self reporting, this can make overall comparisons less clear than if we had accompanied every participant. However, monitoring participants (personally or with a recording device), can induce changes in their behavior. We opted for the more ecologically valid approach and accept that the resulting data is hence more noisy than data from lab-based evaluations.

The study was conducted in Hannover, Germany in the summer of 2015, between late August and mid September. Hence, many participants spent time outside and the ambient lighting was comparably high. During the days of the study, there were between 0 and 15 mm of rain and 0–11 sunshine hours (6 h of sunshine per day on average). Temperatures averaged a daily low of 12 °C and an average high of 20 °C. We would expect noticeability to increase in the winter, when ambient lighting is reduced. However, more and thicker clothing could also mean that the wrist is less visible and reaction times would go up. Short-sleeved summer clothing does not pose this issue. As can be seen in Figure 8.10, even light sweaters can allow sufficient light to pass through.

With our study, we set out to determine how well indirect light feedback performs. Taking into account what we expect our participants to encounter, we hypothesize that:

- H1** Indirect light feedback is as easy to detect as direct light feedback (as reported in previous work).
- H2** How fast users detect feedback is dependent on their current task, with some tasks requiring less intense focus or keeping the watch better in sight.
- H3** Indirect light feedback is noticed earlier in darker environment, while it is less noticeable in, e.g., direct sunlight.

8.4.1 Participants

We recruited 13 participants (4 female, age 23–34, $\bar{x} = 27.1$, $SD = 3.3$) for the study via social media. Six of those participants regularly wear a watch, two wear one sometimes, while the remaining five never wear a watch. None of the participants owned a smart-watch. All participants were right-handed, and they all wore the prototype on their left wrist.

8.4.2 Procedure

Each participant first signed a consent form and received instructions on how to control the prototype. After putting on and starting the watch, we walked participants through the first two stimuli (not included in the later evaluation). However, this gave us a chance to provide instructions on how to respond to stimuli. Participants were instructed to acknowledge the feedback by pressing the button on the side of the watch.

After each acknowledged stimulus, participants made a diary entry related to this situation either online, via a web form, or on paper. In this diary, we asked them to provide information on (1) which kind of feedback was shown, (2) the brightness of their current surroundings, (3) what they were doing when they noticed the feedback, (4) how comfortable the feedback was, (5) how bright the feedback was, and (6) how fast they think they reacted to the feedback. They could also enter additional remarks if necessary. After wearing the watch for one day, participants returned it and were interviewed by us.

8.4.3 Apparatus

Participants wore the prototype described above. We configured the system to display each stimulus for up to two minutes. If not acknowledged by the user in these two minutes, the trial is logged as not completed and the stimulus is stopped. We chose this threshold as previous work showed 99 % of direct light stimuli on the wrist could be detected within this time [107]. We thus expect indirect light stimuli to elicit a similar, or at least not drastically different, response.

Stimuli order was randomized and a new stimulus was shown every 8–12 minutes. Thus, on average, participants receive six stimuli per hour. While participants wore the watch for a full day, we cannot expect data for the full 24 h period, as participants were sleeping at night (and hence instructed to turn off the watch). Instead, we planned for roughly 12 h of actual use (participants, e.g., also had to turn off the watch while showering as it is not waterproof). This would still give us enough trials overall to investigate indirect light feedback.

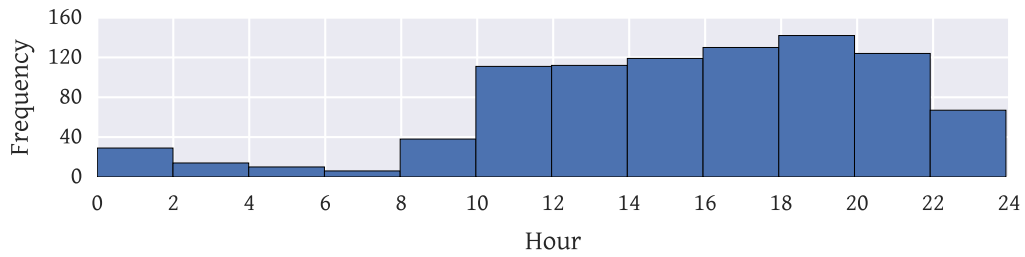


Figure 8.11: The majority of our trials took place between 10:00 and 22:00. However, some participants stayed up longer or woke up earlier.

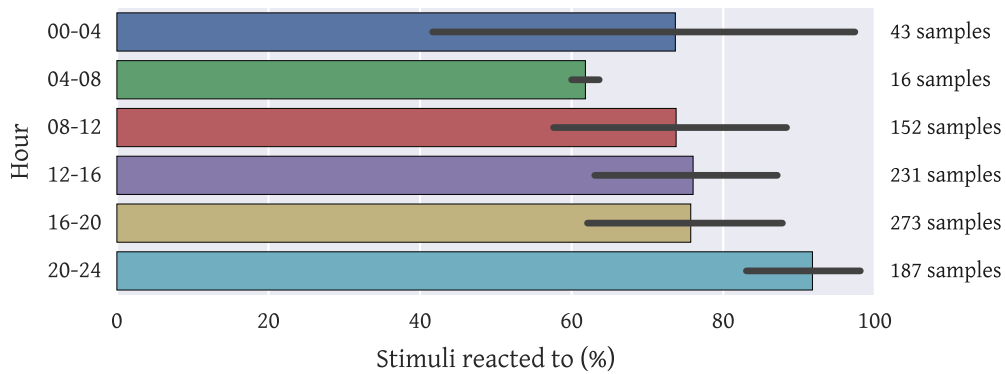


Figure 8.12: There is no clear pattern for how well participants reacted to stimuli on specific times of day. Error bars show bootstrapped 95 % CI.

8.5 Results

We logged between 52 and 90 trials per participant (70 trials on average) for 902 trials overall. As shown in Figure 8.11, the majority of trials occurred between 10:00 and 22:00. Thus, it was still light outside for most trials (in early September astronomical twilight is around 21:00). However, a few trials were also logged in the middle of the night. If we only look at trials between 23:00 and 6:00, six participants contributed 75 trials. Yet, only two participants were responsible for 69.3 % of these trials. Data on nightly trials is thus not very representative.

One might assume that participants are better at reacting to stimuli during the day and get worse as they are tiring, later in the evening. However, if we take a look at the percentages, there is no such pattern (see Figure 8.12). There is a small drop-off between 4:00 and 8:00, but we only have 16 samples for this time period. As mentioned above, there was generally low activity during the night. In fact, between 4:00 and 6:59 only one participant was awake—and reacted to 7 of the 11 shown stimuli. Further data is thus needed for conclusive answers on stimulus noticeability at night.



Figure 8.13: On average, participants reacted to the indirect light feedback in 80.3 % of the trials. Yet, there was large variation in reaction percentages between participants.

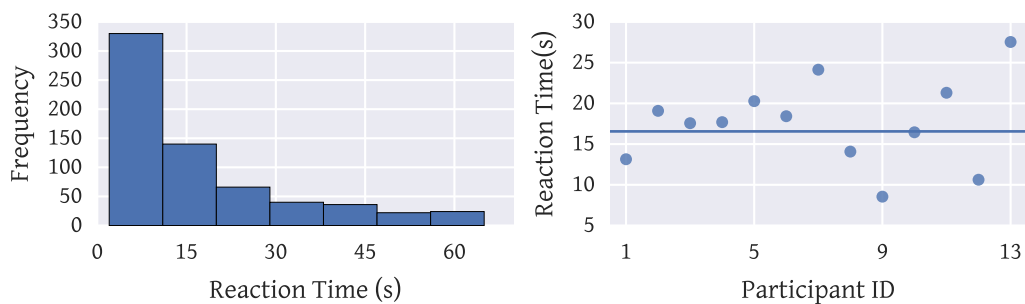


Figure 8.14: Reaction times follow a power-law distribution. On average, it took participants 16.6 s to react to a stimulus, however there is large variation between participants.

Overall, participants reacted to 80.3 % of the presented stimuli. As shown in Figure 8.13, there are large differences between participants. While one participant reacted to all stimuli (78 were presented), another participant only reacted to 42 % of the 73 presented ones.

If we only look at the trials in which participants reacted to the stimulus, we can see that the reaction time follows a power-law distribution (see Figure 8.14). The average reaction time was 16.6 s (acknowledgement took at least 600 ms as the button had to be pressed long to prevent accidental activation), yet we again see large variation between participants. As before, the same participant performs the worst and took 27.5 s on average to react to a stimulus.

With the reaction time distribution this skewed, it is useful to look at the response curve (see Figure 8.15). For example, 50 % of stimuli were acknowledged within 10.5 s. Performance trails off after this: 75 % were acknowledged within 22 s, 90 % within 41 s, and 95 % within 52 s. The two minute window we set for reacting to stimuli was thus much larger than required. Our data shows, that if users do not react within one minute it is very unlikely that they will react at all. For important notifications, systems could then switch to vibration feedback to increase the chance of the user reacting.

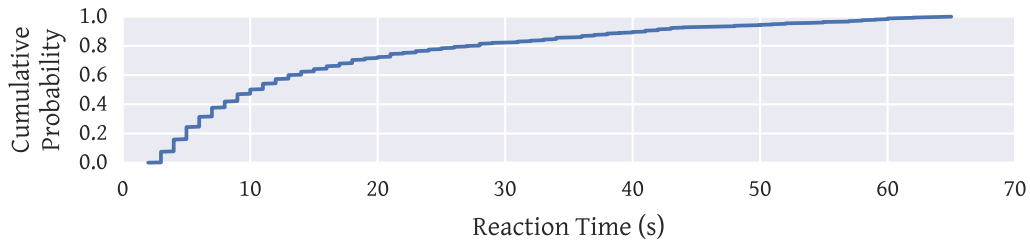


Figure 8.15: The response curve for indirect light feedback shows a fast rise in the likelihood of participants reacting to a stimulus. After 10.5 s, 50 % of stimuli were already acknowledged. However, reaction speed then fades slowly. Note that the cumulative probability here only takes into account successful trials.

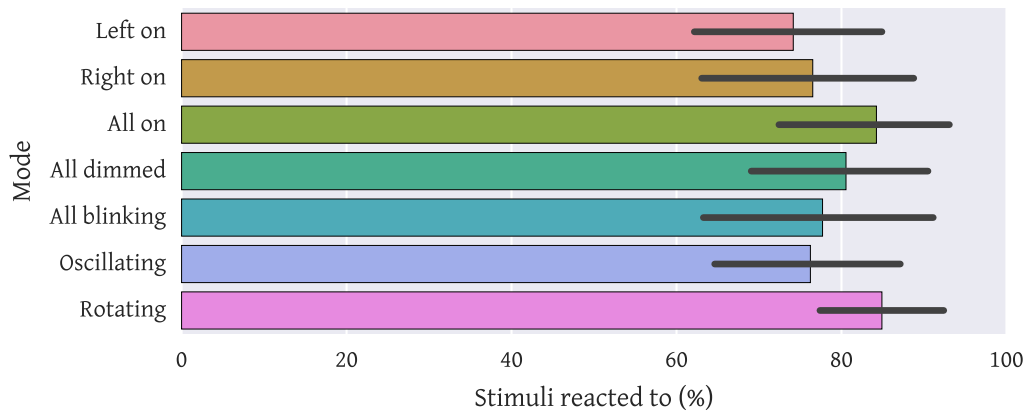


Figure 8.16: The different modes performed similarly with respect to whether participants reacted. Error bars show bootstrapped 95 % CI.

8.5.1 Influence of Illumination Mode

Between the different feedback modes, we did not see a difference in whether participants reacted to them (see Figure 8.16). At the extremes, *Left on* has a 76.0 % reaction rate, while *Rotating* has a 86.0 % one. However, variation between participants is much larger than variation between modes. We can take a closer look at the best and worst performing modes per participant. For this we only consider best and worst that are above/below the 80.3 % average reaction. This is done because we have participant that were consistently good or bad at reacting to stimuli and their worst/best modes are not representative. *Rotating* was the best mode for four participants, with *oscillating* a close second (three participants). On the other hand, *oscillating* was also the worst mode for three participants. Worst overall were the *left/right* modes with four participants having one of these as their worst performer.

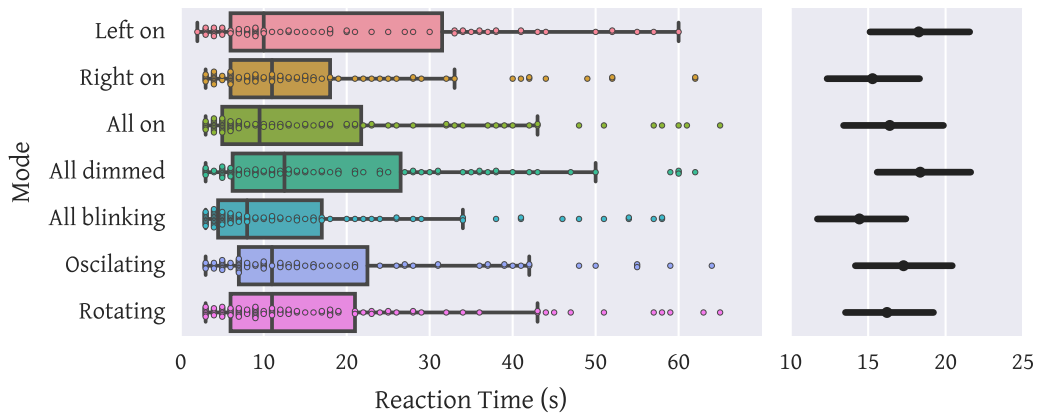


Figure 8.17: The distributions of reaction time for the different modes are similar. Error bars in the right inset show bootstrapped 95 % CI of the mean.

Reaction times for the different modes show a similar picture (see Figure 8.17). Compared to variation in reaction time per participant, variation per mode is much less pronounced. Hence, illumination mode does not seem to make a difference with respect to reaction time. There is a small trend for some modes to have sometimes slightly longer reaction times than others (e.g., there is a stronger reaction time tail for the *left* mode). But this is only a difference at the fringe and for most reactions there is no difference (this is also shown by the overlapping 95 % confidence intervals). However, different modes are useful to distinguish notification types or for aesthetic reasons.

When using different illumination modes for different notifications, it is important that they are distinguishable. We hence asked participants to note down the pattern they identified in their diary. To limit the size of the questionnaire, we grouped four of the modes together into two groups. The *left* and *right* mode were combined to one choice, as differentiating them depends on watch orientation and they could thus be easily confused. We also grouped the *blinking* and *oscillating* modes as they only differ in the animation profile. While this does limit the available comparisons, it slightly reduced the required effort for participants. We can then compare their responses with the actually played back feedback. Figure 8.18 shows a confusion matrix for each feedback mode.

Confusion here could be due to (1) participants not paying much attention to the feedback when reacting, or (2) two modes being visually similar. As can be seen, the rotating feedback was the easiest to tell apart (it is also the most distinct of the set). Confusion arises primarily between muted illumination and full illumination. This is understandable, as there is no reference and external lighting conditions have an impact on how bright the feedback is perceived. There was also some confusion between feedback that was only active on one side (left or right). Sometimes those stimuli were taken for all-around illumination. This is likely due to light leakage. While one side is much brighter when only half the LEDs are on, some light also scatters and shows on the other side.

Mode played back by watch	All on	70	5		15		17
	Blinking	6	82	1	2	1	10
	Fading	3	78			3	15
	Left	23		44	17	1	13
	Muted	50	5		40	1	9
	Right	42	1	35	9		10
	Rotating	1	1		1	103	10
		All on	Blinking/fading	Left/right	Muted	Rotating	Undecided
		Mode recognized by user					

Figure 8.18: This confusion matrix compares played back patterns to what participants recorded in their diaries. Participants noted down the correct pattern 62 % of the time. In 12 % of trials, participants could not decide on a pattern. While rotating is almost never confused, participants had a harder time distinguishing one-sided apart from all-around and muted from non-muted.

8.5.2 Influence of Activity and Surroundings

We also took a look at how users' activities and surroundings impacted their performance. We coded the participants' diary entries and labeled all activities as either (1) *walking*, (2) *waiting or relaxing*, (3) *using a laptop or phone (private)*, (4) *being at work or university*, (5) *driving*, or (6) *housework*. Figure 8.19 shows reaction time distributions for each of those activities. While the average reaction time when being on the laptop or phone was 11.9 s, it was 18.8 s while at work. However, variance is high in each case. We thus cannot reasonably conclude that any activity enables much better performance than any other. Answering **H2** thus requires additional investigation.

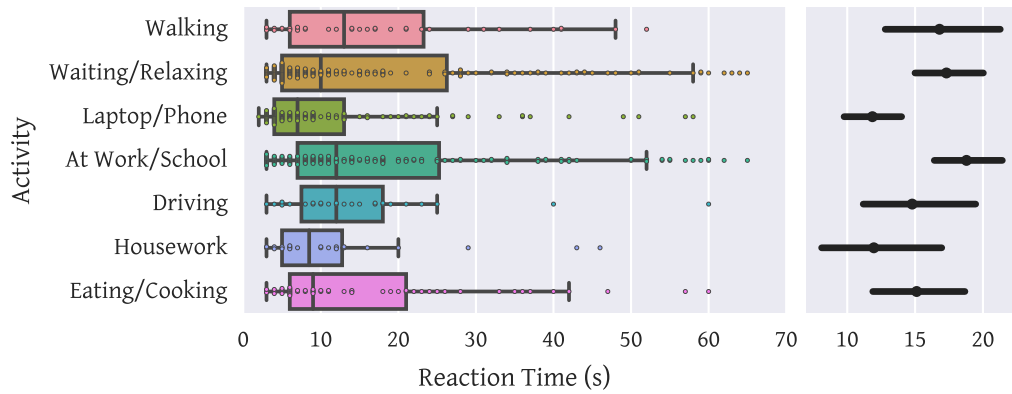


Figure 8.19: Participants recorded their current activity in their diary each time they reacted to a stimulus. We categorize activities into six groups. There are some differences between groups, but there is no activity clearly outperforming others. Error bars in the right inset show bootstrapped 95 % CI of the mean.

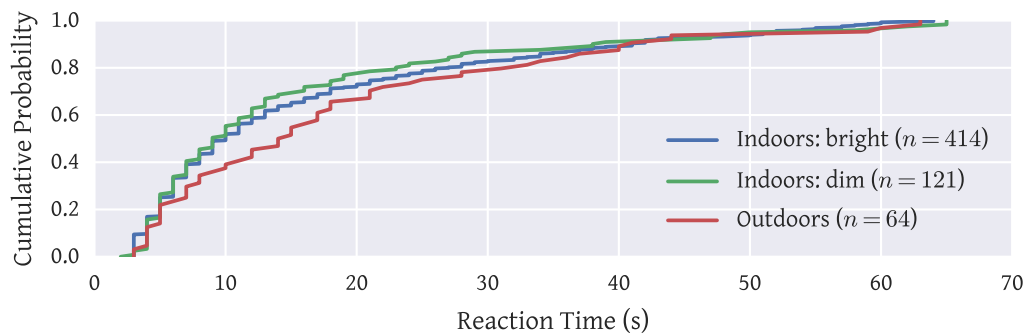


Figure 8.20: We group participants diary entries on their environment into three categories. The response curve shows that participants when indoors tend to react a bit faster than when outside.

Finally, we can compare how fast participants reacted, depending on their current environment. We coded their diary entries as either *outdoors*, *indoors + bright*, or *indoors + dim*. Figure 8.20 shows the response curves for those three conditions. As can be seen, participants tend to react a bit faster in the two indoor levels. However, this is a subtle difference and there is no large offset between the two conditions. The data is also rather unbalanced—participants spend much more time inside than outside. Hence, even though there is some support for it, we cannot yet give a conclusive answer to **H3**.

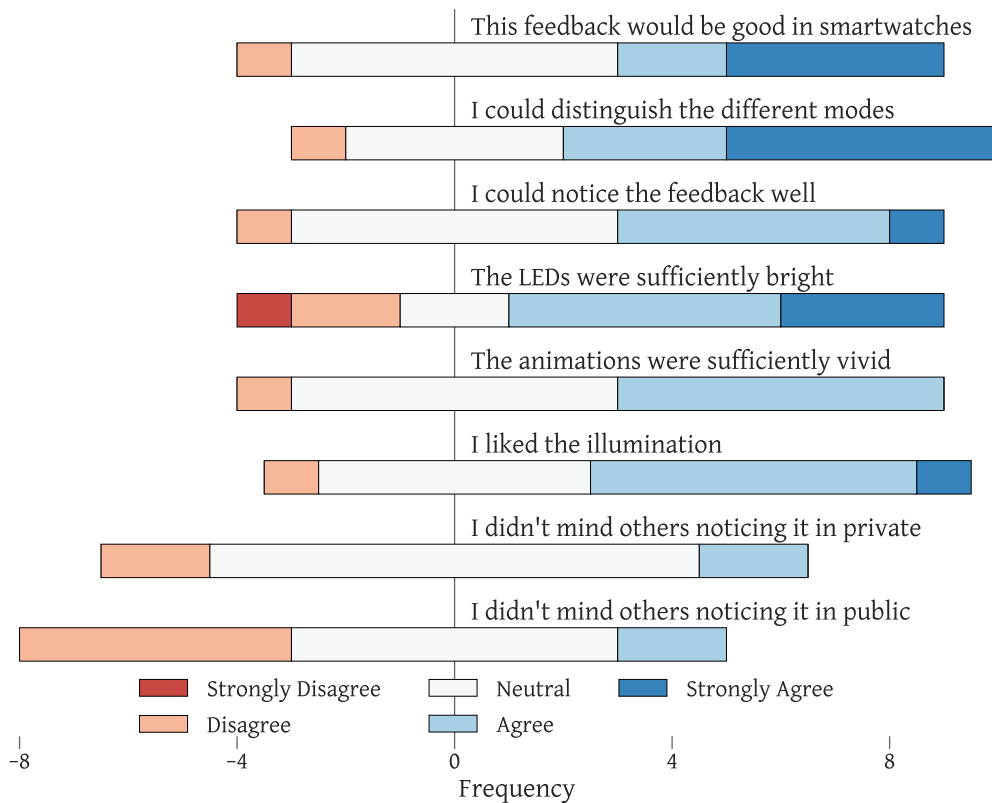


Figure 8.21: During our exit poll, participants rated several statements on a 5-point Likert scale. Ratings of our system were overall mostly positive. Participants could see themselves using it and liked it. They only had some reservations regarding drawing attention to them when using the system in public.

8.5.3 Qualitative Ratings

In addition to measuring participants’ reaction to stimuli on the watch prototype, we also asked for their qualitative feedback in an exit poll. For each question, participants indicated their level of agreement on a 5-point Likert scale. An overview of their answers is shown in Figure 8.21.

Asked to indicate whether they liked the feedback and whether this would be a good addition to future smartwatches, most participants agreed. We also asked participants how they felt about the feedback when in private or in public. Some participants did not appreciate use in public—likely because the feedback can draw attention to the user. While vibration feedback can generally be received without alerting others, indirect light feedback is equally noticeable to bystanders.

Asked to rate the quality of the feedback, most participants rated the animations to be sufficiently vivid (e.g., rotating fast enough). While some wished for brighter LEDs, most stated the illumination was also sufficiently bright (even though they wore the prototype in the summer and sometimes were also outside). We also asked participants which modes they liked/disliked the most. There was no clear consensus, but most liked the *all on* and *rotating* modes, with *fading* a close third. However, *rotating* evoked disagreement and was equally disliked as liked. The most disliked mode was *blinking*, though. This mode was also mentioned by participants when asked for general annoyances during the study.

We also asked participants about specific situations where they liked or disliked the feedback. Participants generally appreciated the feedback in private or calm situations, such as at home, when working on their laptop, or relaxing. Three participants stated the feedback was basically fine all the time. Several participants indicated a dislike of the feedback in social situations (particularly, conversations). This relates back to the questionnaire response and shows that others noticing the feedback is sometimes disliked. One participant specifically mentioned disliking it when people asked him about the device (which presupposes them noticing it in the first place).

Many participants expressed annoyance by the frequency of the feedback. While we limited stimuli frequency to about six per hour, this might still be too often. This is exacerbated by the fact that the feedback did not actually notify participants of anything. It might hence be a good idea to integrate feedback with each participant's personal notifications, e.g., illuminating the LEDs when an email is received. While this gives less control over the frequency and time of each trial, it could be less annoying for participants.

8.5.4 Discussion

The results show that indirect light feedback generally works well in a wide range of situations. However, we could not find strong differences for any one feedback mode or activity. We can thus not confidently confirm **H2** or **H3**. This is a drawback of the study design we used. While the good performance under variable conditions shows indirect light feedback is viable, the uncontrolled nature of the in the wild approach makes it hard to compare specific conditions. Hence, it will be necessary to run more constrained evaluations in the future. For example, it would be worthwhile to control for activity and, e.g., compare how well indirect light feedback works when in the office or when on public transport.

Our evaluation showed that participants did not react to 19.7 % of stimuli. We can take a look at those trials where participants did not react. In 56.6 % of the cases, participants only missed one trial before reacting again to the next stimulus. However, we also had one participant miss 10 trials in a row between 13:20 and 15:10. Such longer lapses are likely the result of taking off the watch or a short nap. The nature of the in the wild study design does not prevent such participant behavior. For short gaps, the diary entries for the preceding and following trial can provide clues regarding the likely setting and activity at the time of the missed trial. In 42 % of such gaps participants likely were indoors with bright lighting. However, this is also the most common setting overall (see Figure 8.20) and is actually underrepresented in gaps. More interestingly, 25 % of the gaps happen between a trial that occurred indoors and a trial that occurred outdoors (either direction). Stimuli here probably coincided with either arriving or leaving—times of more activity where participants are more likely to miss notifications. If leaving in a rush (e.g., to catch a train), it is also more likely participants ignored a stimuli.

We can compare the reaction times in our study to results from previous work. It took users almost 3 s to react to light stimuli when wearing the *NotiRing* [262]. This is an average over five levels of physical activity (users, e.g., were walking on a treadmill during part of the study). For their glass-mounted light feedback, Costanza et al. report about 1 s reaction times for a study where users were engaged in a reading task [58]. While walking around or editing text, the mean reaction time increases substantially (unfortunately no averages are reported). Those results from the lab are in stark contrast to results from Harrison et al. [107]. During their study participants were left to continue their normal daily routine while wearing a prototype for 2–3 hours. While they mention that almost all that time was spend working while sitting somewhere, this still creates a very different study situation. Correspondingly, reaction time to light stimuli on the wrist was about 19 s. Averaged over all trials, we found a similar reaction time to light stimuli on the wrist of 16.6 s. A similar study design was also used by Lyons, who investigate visual stimuli on smartwatch displays instead of through LEDs [183]. However, in his study it took participants an average 76 s to react to a stimulus (participants reacted more quickly to larger area or faster blinking stimuli). The different mode of illumination is likely the reason for the much slower reaction time, compared to the other studies.

The results from this set of studies show that study design has a strong impact on measured reaction times. One aspect here is that an in the wild setup provides for a much more diverse and natural set of distractions and primary tasks. But as participants wear a device for a longer period of time, they are also likely getting used to it—*forgetting* it to some extend. A study situation in a lab on the contrary creates artificial focus and hence deceivingly low results for reaction times. However, the similar performance of indirect light feedback and direct light feedback [107] in a similar study setup does support **H1**: that both stimuli are equally hard to detect.

While we look specifically at the use of indirect light feedback for notifications, this is not the only possible use. As illustrated by Qin et al., LEDs along the perimeter of a device can be used to visualize off-screen content [248]. Correspondingly, the eight LEDs in our prototype could also show navigation directions to the user. This can augment an application running on the screen (e.g., while running a maps application) or display directions while the screen is switched off to reduce power drain. When used as a secondary screen for the smartwatch, the indirect light *screen* is useful for communicating longer running background state. So while users read a text they received on the watch, the illumination could indicate the time left until the next appointment, increasing in intensity as the appointment approaches. But use of the illumination could also be just driven by aesthetic concerns, e.g., pulsating in the beat of the music currently playing.

8.6 Summary

This chapter presented indirect light feedback: a subtle way to, e.g., relay notifications. This kind of feedback can be added to future smartwatches or even analog watches. Compared to vibration feedback, indirect light feedback is less disruptive, yet remains noticeable. The subtle glow around the watch does not preclude other feedback on the watch face. In fact, the low-fidelity nature of indirect light feedback nicely complements high-fidelity information shown on smartwatch displays. By including different levels of feedback, devices could then pick feedback depending on the current engagement of the user, supporting the range from casual to focused interactions. This aspect was also pointed out by Lyons in his research on visual parameters for smartwatch notification [183]. As mentioned above, he investigated visual stimuli directly on smartwatch displays, varying size, color, and blinking frequency of a shown circle, where all but color had a significant impact on reaction times. He notes that once information on the reaction times for different stimuli is available, “a designer can chose a visual prompt that matches the urgency of the information to be shown on a smartwatch and correspondingly help better manage one of a wearables user’s most critical resources — their attention.”

In general, indirect light feedback is a novel addition to the ensemble of smartwatch feedback methods. But not only does it bring subtle notification capabilities to the form factor, it also opens up a new design space and a venue for aesthetic expression that connects with the wearer’s body. The light does not just show on the face of the watch. Instead, the arm around the watch glows and the boundaries between device and body break down to some extent. We believe this is an attractive addition and one participant in fact remarked:

I don’t own a smartwatch. However, I would buy one if it had this illumination.

In the future, we also plan to investigate other uses of indirect light feedback. While this chapter focuses on notifications, indirect light feedback enables a wider range of off-screen interactions on the wrist when coupled with an appropriate input technology (a system like *SkinTrack* would be one possible option for sensing the finger on the skin around the smartwatch [330]). A similar approach has already been used by *Skin Buttons*, albeit with a top-projection [169]. Where we explore personal use of indirect light feedback, it could also be used as a form of coarse public display [227].

Furthermore, we plan to iterate on the prototype and integrate indirect light feedback directly with a smartwatch. Pebble smartwatches, e.g., allow custom bands to connect with the watch. In such a version, indirect light feedback could also be extended along the strap. While feedback behind the watch face is more public, feedback at the middle of the strap is more private, as that part of the watch is commonly not facing outwards when worn. Integrating with a smartwatch would also allow us to provide indirect light feedback for a user's actual notifications.

9 Casual Feedback: Compression Feedback

*I felt like grabbin' you real tight,
Squeeze and squeeze with all my might.*

— Elvis Presley, *Paralyzed*

In this chapter, we take a look at compression feedback as a kind of feedback that scales well from subtle to intense. Compression feedback works by tightening straps around wrist, legs, or arms. This compression creates an overall sensation of pressure at that location, but at higher pressure levels also can be used to *inhibit* movement of the affected limbs. We specifically focus on pneumatic and wearable forms of compression feedback. While we also explore other uses of compression feedback, we primarily investigate the use as a means to communicate notifications.

This chapter draws mainly from a paper and an extended abstract, both published at *CHI 2017*; an early prototype of compression feedback was already shown as a demonstration at *CHI 2015* and thus was also published in the extended abstracts:

- Henning Pohl, Peter Brandes, Hung Ngo Quang, and Michael Rohs. “Squeezeback: Pneumatic Compression for Notifications.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '17*. New York, New York, USA: ACM Press, 2017. DOI: 10.1145/3025453.3025526
- Henning Pohl, Franziska Hoheisel, and Michael Rohs. “Inhibiting Freedom of Movement with Compression Feedback.” In: *CHI '17 Extended Abstracts on Human Factors in Computing Systems - CHI EA '17*. New York, New York, USA: ACM Press, 2017. DOI: 10.1145/3027063.3053081
- Henning Pohl, Dennis Becke, Eugen Wagner, Maximilian Schrapel, and Michael Rohs. “Wrist Compression Feedback by Pneumatic Actuation.” In: *CHI '15 Extended Abstracts on Human Factors in Computing Systems on - CHI EA '15*. 2015. DOI: 10.1145/2702613.2725427

Early exploration of compression feedback was conducted in a project that was part of our class on *physical computing* in Summer 2014. Furthermore, I supervised several bachelor and master theses in the area of pneumatic compression feedback¹, the results of which are also included here.

¹Namely, the bachelor thesis of Franziska Hoheisel [122], the bachelor thesis of Peter Brandes [24], and the master thesis of Hung Ngo Quang [249].

9.1 Introduction

Haptic feedback in current mobile devices is mostly vibrotactile. However, vibration feedback captures much of a user's attention and can be disruptive [99]. This limits its use to short periods and prominent notifications. It is also rather limited in the acceptable stimulus strength, i.e., too strong vibrations are uncomfortable to users [187, pp. 22–23] (however, the relationship between vibration parameters and perceived strength is a complicated one in general [203]). Pressure feedback, on the other hand, can support less attention-demanding feedback [335]. Hence, we investigate *compression feedback* as a form of pressure feedback. Instead of commonly used point pressure, we use inflatable straps to generate uniform pressure. We believe this kind of feedback would integrate well with workout armbands (used, e.g., when running, cycling, or in gyms), smart clothing, or smartwatches/jewelry.

Overall, we see a number of advantages of compression feedback:

- It works over a wide range of attention capture—from subtle to inhibiting and forceful. Strong constriction demands instant attention. This range of attention capture makes it a good feedback channel for *casual interaction*.
- It can provide constant background feedback (prolonged vibration would be too disturbing) which can ramp up to slowly bring something to the user's attention (e.g., an approaching appointment). This is similar to wake-up lights—a stimulus level slowly increases and is noticed at some point after the detection threshold is crossed.
- It can feel similar to human attention grabbing behavior—making it potentially useful for intimate communication scenarios. A slight compression on the wrist, e.g., can feel similar to the wrist being grabbed by another person.
- In the extreme, it can constrain a limb's movement (e.g., stopping a person from continuing weight lifting exercises). By inflating a cuff around and thus *locking* joints, the bending angle can be limited.
- Possible attachment positions (arms, legs, waist, fingers, hands) align well with good locations for wearables (i.e., many people already wear items of clothing there). The strap-like nature of the actuators also lends well to direct integration into clothing.

In this chapter, we explore technical and design aspects of compression feedback. Starting from the basic principles of compression feedback, we explore how to build compression feedback systems. To show how compression feedback can scale to higher, inhibiting, intensity, we look at a pervasive gaming prototype designed for this. But primarily, we report on several experiments, where we look at how well compression feedback works for notifications.

9.2 Related Work

The general aspects of feedback for casual interaction, as discussed in Chapter 7 also apply here. But compression feedback systems also build upon existing work in pneumatic actuation. There has also been previous work in using pressure as a feedback modality.

9.2.1 Pneumatic Actuation

There are a large number of projects that have used pneumatics for user interfaces. Pneumatics can be used to create dynamic buttons that can appear or disappear [106], but also to just recreate the *sensation* of pressing buttons [151]. Such feedback for the fingers was also used to provide haptic feedback to surgeons [61], or to drivers [75]. Another big area is shape, where pumping air into devices has been used to create shape changing controllers [150], or to actuate tangibles [80, 189, 325]. We use similar principles, but inflate straps around user's bodies.

9.2.2 Pressure Feedback

Point pressure actuators (pactors) [14, 334, 335] are a common means for pressure feedback and, compared to vibration feedback, have less attention capture and are less agitating. Instead of point pressure, *ServoSqueeze* tightens a band around the wrist [14]. In contrast to our system, *ServoSqueeze* is rigid and cuts into the arm instead of applying uniform constriction. *HapBand* [44] and *HaptiHug* [298] work similarly. Suhonen et al. used shape memory alloys for the same effect [288]. A set of balloon actuators around the leg was used for feedback by Fan et al. [77]. However, this was used for discrete point feedback instead of overall compression. He et al. constructed a bracelet with balloon actuators to provide pressure feedback [110]. As their device has discrete chambers, connected by valves, it can also do moving and tapping sensations.

Both Patterson and Katz [225] and Tejeiro et al. [294] use blood pressure cuffs to apply pressure feedback. In both their works, the pressure is used to replace missing tactile feedback when using prosthetics. The focus in their work is on the control aspect and sensory replacement, not on the perception or properties of pressure feedback. Also, only strap placement on the arm is explored, while we look at a larger set of locations.

Pressure via a blood pressure cuff on the forearm was also investigated by Mitsuda [200]. In his work, he investigates how well pressure feedback can be used to communicate forces to the wearer. He finds that putting 6 kPa pressure on the forearm results in a feeling equivalent to a force of ~10 N (as during holding a weight). We study similar aspects of psychophysics, but also explore several other use cases of pressure feedback apart of force displays.

Vaucelle et al. use inflatable straps with embedded *plastic teeth* in *Hurt Me*, to provide sensations “*akin to being bitten*” [300]. In this kind of system, the compression feedback is only used as a means to transport a different, second feedback: the sensation of the teeth pressing down on the skin.

9.2.3 Pneumatic Input

While we use air pressure sensors only to monitor the feedback, such sensors can also be used to detect input. Sudden pressure changes can, e.g., be interpreted as the user touching the wrist strap (or, e.g., resting the arm on a table). This is used by Vázquez et al. as one possible input channel for their controls with pneumatically actuated resistance to change [301]. By embedding air bubbles in puppets, Slyper and Hodgins could detect presses and bends [276]. This requires multiple air chambers with separate sensors.

9.3 Compression Feedback

Compression feedback uses inflatable straps to tighten around body parts. At low pressure levels the sensation is subtle and only a slight contraction can be felt. With increasing pressure, this grip tightens and users become easily aware of the feedback. At this level, the compression is comfortable and feels neither painful nor does it restrict normal body movement. Pressure can be further increased to levels at which movement is impaired (e.g., because a joint is locked) and discomfort sets in. At such extreme levels, users are effectively forced to react to the feedback.

Compression feedback tightens around the body—a sensation closely related to human interactions such as hugging. Hugging interfaces naturally facilitate intimate communication [68, 298]. We believe that this also translates to compression feedback. In fact, previous investigations in using inflatable vests for remote hugging were promising [206].

Compression feedback feels different than other pressure feedback methods. Instead of pressing an actuator *into* the body, pressure is applied towards the center of a larger circumferential area. Thus, the sensation feels uniform, not like something *pushing into you*, but more like something *tightening around you*. As illustrated in Figure 9.1, this difference in force is what sets compression feedback apart.

The difference between pressure and vibration feedback is primarily one of actuation frequency. However, this difference is so strong that they are regarded as distinct modalities [46]. Vibration feedback captures attention more effectively than pressure, but is also seen as less affective [334]. Vibration and compression feedback, however, are highly complementary—compression results in an inward force, while vibration provides tangential forces on the surface.

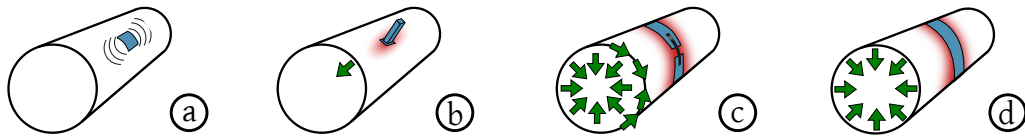


Figure 9.1: (a) Vibration does not exert a sustained inward force, while (b) pactors [334] provide pressure at a specific location. (c) Tightening bands [14] provide a more uniform squeezing sensation, but also exert shear forces due to the band movement. (d) Inflating straps provide uniform pressure but can do so without the amount of shear force created by tightening bands.

Traditional high-frequency vibration feedback is mainly detected by Pacinian corpuscles which have optimal sensitivity at 250 Hz. This receptor, however, is only one of four types of mechanoreceptors (responding to pressure and vibration) in the human skin [22, 154]. Meissner corpuscles (sensitive to pressure change at 10–50 Hz), or Merkel and Ruffini cells (responding to sustained pressure and stretching) are not stimulated strongly through vibration. Compressive feedback is able to stimulate these slow-adapting receptors.

Compared to individual pactors [77, 334], force in compression feedback is less localized and distributed over a wider area. However, with an increasing pactor density, or with larger pactor designs, both methods somewhat converge as the pressure is distributed over a larger area and not as localized. For example, *HapBand* [44] uses three plates to simulate the arm being grabbed by a hand. Similar to our approach, *ServoSqueeze* [14] and *HaptiHug* [298] constrict around an arm/body, exerting an inward force. However, here a strap is tightened by changing the length with a motor-adjustable clasp. Thus, in addition to the inward force, a non-negligible additional shear force is present.

9.4 Compression Feedback Systems

To evaluate properties of compression feedback, we built several prototypes. At the very least, such a device needs a way to push air into the system, a strap to inflate and a valve to release air from the system.

The first required component are inflatable straps. For the uniform sensation of compression feedback, strap need to wrap around the attachment position and fit comfortably. Such inflatable straps could be manufactured from silicone composites [189, 325] or by heat bonding plastic sheets [212]. We found that the critical part in custom made straps is embedding a connector. As compression feedback can reach higher pressure levels than, e.g., those necessary for actuating origami [212], a sturdier connection is required. We thus chose to repurpose already available straps that come with suitable connectors: blood pressure cuffs. These cuffs are already designed for the larger pressures necessary to stop blood flow during blood pressure measurements.



Figure 9.2: We used a wide range of blood pressure cuff sizes to test our system on several body locations. The smaller cuffs can be wrapped around single or multiple fingers, the 14-19.5 cm cuff fits comfortably around a wrist, while the larger cuffs fit around upper arms and legs.

Blood pressure cuffs commonly consist of an inflatable air bladder enclosed in a band adjustable to different sizes using velcro. Such cuffs have been used before to provide pressure feedback on the arm [200, 225, 294, 300]. However, we experimented with a wider selection of cuffs (see Figure 9.2) ranging from ones designed for circumferences as short as 3 cm to ones as long as 48 cm. Cuffs are manufactured for patients ranging from newborns to the heavily obese—necessitating such a wide range of sizes.

While blood pressure straps are readily available for use in compression feedback, a complete compression feedback system also needs a controller. It is responsible for inflation and deflation of the strap, as well as monitoring in-strap pressure levels, and thus the force exerted on a user. We have built several different controllers for this purpose (see also Figure 9.3). They all repurpose pumps and valves designed for blood pressure monitors, such as the *AEG BMG 5610*. This wrist-worn medical device comes with an inflatable belt (adjustable from 14–19.5 cm length, 7 cm width) and is designed for belt pressures of up to 300 mmHg² (~40 kPa). It also contains an air pump, a safety valve, an air pressure sensor, and a solenoid valve, allowing for controlled in- and deflation of the belt. We designed custom PCBs to attach to *Arduino Nano* and *Adafruit Feather* microcontrollers in order to connect them to those pumps and valves.

To measure internal strap pressure, we use *Freescale MPXV5010 series* sensors, which are designed for the 0–10 kPa range. While this is far from the belt’s maximum pressure, this lower pressure range is more appropriate for the notification studies we intended to run. Pressure sensors are connected to the attached microcontrollers’ ADCs. While we designed the prototype for wireless streaming of sensor data via Bluetooth, we used a USB connection during the lab studies.

²Normal blood pressure is in the range of 90–119 mmHg with pressures larger than 180 mmHg being a hypertensive emergency.



Figure 9.3: We built several compression feedback controller prototypes. They all reuse parts from off-the-shelf blood pressure monitors (shown in upper left image). We only keep the sensor and valves at the wrist for our first prototype (upper right), but move the pump to reduce its influence on the sensation. The second prototype (lower right) is designed for mobile evaluation, fits into a pouch, and is controlled from a phone via a Bluetooth module. The third prototype (lower left) miniaturizes this setup for even better mobility.

We use a PID controller algorithm to adjust pump power for setups requiring the capability to maintain specific pressure levels. Generating such a stimulus requires carefully approaching the desired pressure, leveling, and then maintaining pressure over the stimulus' duration. This is challenging due to loss of pressure in the system (e.g., in tubing), and back pressure influencing how the system reacts. For example, loss can require the pump to keep running at a low level to maintain pressure. We also found changes in atmospheric pressure having a noticeable impact on system behavior. Atmospheric pressure varies over the course of a day and can change in the order of several kPa with the weather. We tuned the PID parameters of our system for 13 target pressures and use cubic Hermite interpolation for in-between pressures.

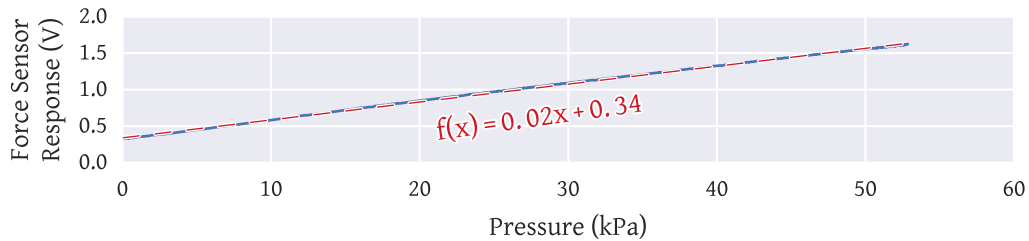


Figure 9.4: We measured on-arm force (raw sensor response in V) for increasing in-strap pressures (650 total samples). A strong linear relationship ($R^2 > 0.99$) shows that pressure predicts arm force.

9.4.1 How Air-Pressure Relates to On-Arm Force

While our prototypes measure strap pressure, we are actually interested in the force exerted on a user's arm. Adding a force sensor between the strap and the arm would change the feel of the feedback, though. Hence we confirmed dependency of the two measures in an experiment. For this, we put an *FX1901* compression load cell (50 lbf range) between the strap and a dummy arm and amplified the signal using a *INA125P* instrumentation amplifier. Air pressure is measured with a *Freescale MPXV5050 series* sensor (0–50 kPa range). We then recorded raw force sensor and air pressure values while inflating the strap. As shown in Figure 9.4, air pressure is indeed a very good predictor of force on the arm. For the remaining studies we hence only recorded air pressure.

9.5 Properties of Compression Feedback

So far, we have provided an overview on technical aspects of compression feedback. However, a crucial question for compression feedback is how well it actually works. In particular, we focus on use of compression feedback for notifications. We thus ran a series of studies to determine psychophysical properties of this feedback in the lab and evaluate how it fares in the wild.

Like other kinds of haptic feedback, compression feedback could be used for a wide range of scenarios. For example, virtual reality or pervasive games could also benefit from this kind of haptic feedback. However, here we focus on notifications as a common use case for feedback in mobile computing. Hence, we also designed our prototype around a wrist strap, as the wrist is a common location for wearable devices, such as smartwatches or bracelets. Another relevant location would be the upper arm, which is already a common location for straps worn during workouts which could readily include compression feedback.

9.5.1 Background Feedback

One of the properties of compression feedback is that it can be applied continuously, making it well-suited for feedback in the background. Instead of springing to the user's attention, the feedback persists and thus moves to the front- or background depending on how much the user concentrates on it. In this characteristic, compression feedback is similar to thermal feedback [319]. This is useful for communicating information not sufficiently urgent to warrant an interruption. For example, compression feedback could be used to represent weather when hiking. As long as the weather only changes slightly, the feedback stays in the background. It becomes noticeable only when paid attention to and does not inhibit the user. When a storm front nears, pressure increases bringing this to the immediate attention of the user.

We tested the appropriateness of such continuous feedback in a small study with 9 participants (all male, age 23–41, $\bar{x} = 30.0$, $SD = 6.3$). Participants wore a manual blood pressure meter around their upper arm which we inflated to 10 mmHg (1.3 kPa). This is only a slight inflation which is far from levels where the cuff cuts off blood flow (typical values are 120 mmHg and 80 mmHg for systolic and diastolic pressures, respectively). The pressure level can vary slightly, depending on arm movement, increasing when the strap is compacted. Participants wore the strap for 1 hour while continuing their daily routine (mostly working at desks). Afterwards, none of the participants reported feeling inhibited or annoyed by the device. This shows that compression feedback is indeed suitable for prolonged display of state information (something vibration feedback is less suited for). However, note that while we did not instruct our participants to engage in a specific task, there was little variability in their exhibited behavior. Additional studies are needed to investigate whether background compression feedback is also suitable in a wider range of other environments.

9.5.2 Absolute Detection Threshold: In the Lab

When using feedback for notifications, an important question is how strong the feedback needs to be. We hence set out to investigate the pressure threshold where users can first perceive a compression stimulus. By conducting this study in the lab, we can determine the lower range of pressure usable for compression feedback notifications. We used a standard two-down/one-up staircase design [176]. At the start of each trial the pump increased pressure till the stimulus level was reached, which was then held for 5 seconds, after which the outlet valve opened and the strap fully deflated. Starting at a stimulus level of 0.6 kPa (determined as a first estimate for the threshold during pilot studies), we increased pressure by 25% after a stimulus was not felt and decreased by 25% if a stimulus level was perceived twice in a row. The experiment ended after 7 reversals.

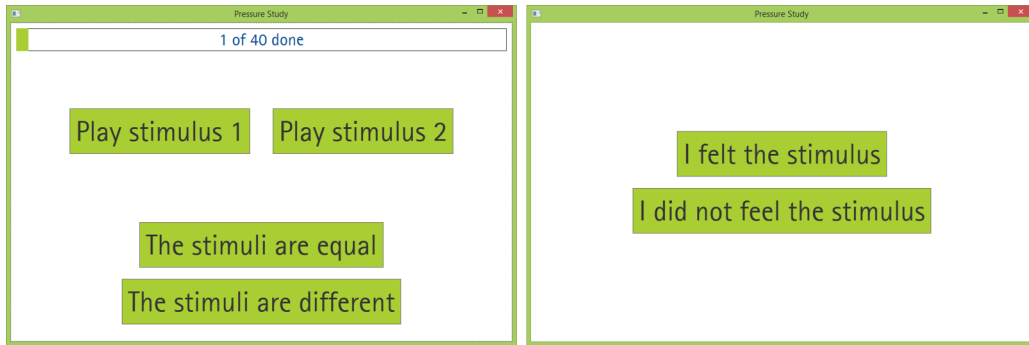


Figure 9.5: GUIs for participants in the (left) absolute detection threshold and (right) just-noticeable differences studies in the lab. During stimulus playback input was blocked.

In this study we use the stationary prototype. As noise and vibrations from the pump would provide additional cues to participants, we moved the pump away from the strap and connected it via a ~1 m long silicone tube. This makes the device non-mobile, but allows more accurate measurements of in-cuff pressure (the mobile prototypes have the sensor close to the pump, which can affect readings). Participants also wore noise canceling headphones, playing Brownian noise. During the study, participants were seated and wore the device on their left wrist, rested their hand on a table, and kept that arm stable. With their right hand they controlled a mouse and provided responses via a study interface on a screen in front of them (see Figure 9.5).

We recruited 14 participants (2 female, age 23–47, $\bar{x} = 28.6$, $SD = 6.6$) from around our institution. None of the participants had their blood pressure measured recently (one participant had so in the last month), so there was no recent familiarity with the provided sensations. Participants completed the study in ~5 minutes and were compensated with a small non-monetary gratuity.

9.5.2.1 Results

We computed the absolute detection threshold for each participant from the mean pressure between reversals (i.e., between points where pressure was felt and where it was not). As shown in Figure 9.6, participants' thresholds varied between ~0.22 kPa and ~1.59 kPa. The distribution of thresholds (also shown in the Figure) is skewed and has the mean at 0.72 kPa and the median at 0.66 kPa. The bootstrapped 95 % confidence interval for the absolute detection threshold ranges from 0.55 kPa to 0.94 kPa. We noticed that those participants with larger absolute detection threshold in general had larger-diameter arms, which might influence inflation characteristics or how well pressure can be perceived. Further studies are thus needed to investigate the relationship between body type and sensitivity to pressure stimuli. However, pressures above roughly 1 kPa are very likely to be detectable for most.

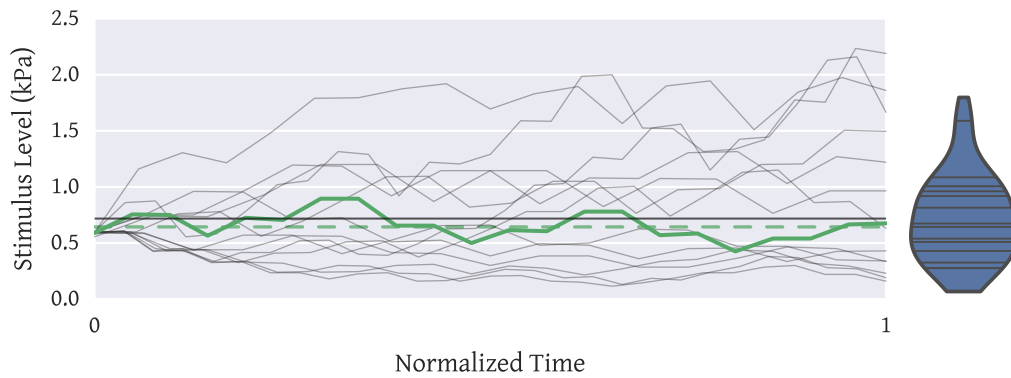


Figure 9.6: Each curve on the left shows the progression of one participant in the absolute detection threshold study. We highlight the curve and threshold of one example participant in green. The violin plot on the right shows the distribution of the resulting absolute detection thresholds, with each participant's thresholds represented by a horizontal line. The overall detection threshold for all participants, indicated by the stronger black line, is 0.7 kPa.

9.5.3 Absolute Detection Threshold: In the Wild

The previous study provided data on lab performance of compression feedback. However, the ideal conditions encountered here would not be found when using compression feedback in actual wearables. For example, there would likely be many more distractions than in the lab and users would be engaged in other tasks. We thus ran a second absolute detection study to determine absolute detection thresholds for a wider range of situations. While there is less control over the study conditions in this kind of setup, it allows for more ecologically valid data on compression feedback perception.

In this study, participants wore a mobile prototype in a pouch (see Figure 9.7) and used the same pressure cuffs on their wrist as in the lab study. Instead of presenting specific stimuli and recording participants' responses, here we chose a study design more closely aligned with real use. We hence had participants walk around and take public transport, simulating actual use. Participants listened to music on headphones for the entire study and played a mobile game while on public transport. We instructed participants to take note of their environment during the walking portions in order to prepare for subsequent questions. During the study, stimuli were then presented randomly every 1.5–3 minutes. For each stimulus, we programmed our system to slowly increase pressure in the cuff. We instructed participants to press down on the cuff to signal they noticed a stimulus. An experimenter followed participants during the study to record their current state (i.e., walking or on public transport) and their reactions.



Figure 9.7: During the in the wild study on absolute detection thresholds of compression feedback, participants wore the prototype controller in a pouch around their waist. The same setup was also used in the jogging game evaluation.

The absolute detection threshold is then given by the cuff pressure just before the participant pressed down on the cuff. Pressing on the cuff can be detected as a spike in the pressure readings, as external compression also increases internal pressure (the space available to the air is reduced). This yields a conservative (slightly higher) estimate for the threshold, as it does not account for participants' reaction time. Additionally, because of the study setup (i.e., much more distraction than in the lab) we would expect less sensitivity in general.

We chose to have participants press down on the cuff instead of reporting stimuli directly to the shadowing experimenter for three reasons: (1) this allows us to demonstrate how compression feedback devices can also provide input capabilities, (2) as participants were occupied elsewhere, addressing the experimenter would have necessitated additional effort by the participant (e.g., to turn around), but most importantly (3) the context (e.g., being in a subway) made it inappropriate for participants to just shout out notifications to the experimenter (wearing headphones also makes it hard for people to regulate the volume of their voice).

For this study, we recruited 12 participants (2 female, age 18–30, $\bar{x} = 24.7$, $SD = 3.0$). None of the participants were familiar with the feedback and none had their blood pressure taken recently. This study took about 40 minutes and was also rewarded with a non-monetary gratuity.

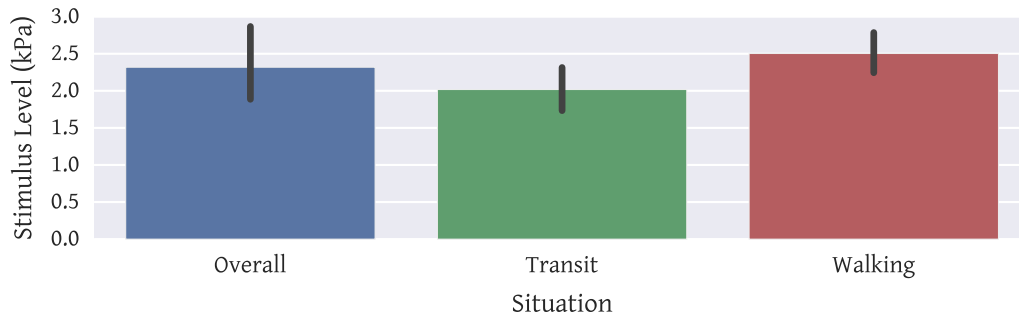


Figure 9.8: Absolute detection threshold as determined in an in the wild study. Participants were able to detect stimuli at about 2.3 kPa. There was no difference in performance between walking and being on public transport. Error bars show bootstrapped 95 % CI.

9.5.3.1 Results

As shown in Figure 9.8, there was no large difference in the absolute detection threshold for the different settings. A paired samples t-test, comparing transit and walking pressure thresholds, showed no significant difference between the two; $t(11) = -1.62, p > 0.05$. Participants were able to detect stimuli at about 2.3 kPa. It only took participants 7.9 s on average to react to a stimulus. However, one participant was able to react in as fast as 2 s. In this situation the cuff was already constrained a bit, resulting in an increased pressure at the start of the trial. The participant was thus able to notice the feedback much earlier than usual (only a slight inflation was needed to increase the pressure from the initial to a noticeable level). The longest it took a participant to react was 32 s.

The results thus show that in the wild performance, while slightly below lab performance, is still quite good. Participants are already able to perceive pressures of 2.3 kPa. There is some variability around this threshold, as in the lab study, hinting at participant specific thresholds. Taking into account that the final threshold is influenced by participants' reaction times, we can assume that slightly lower pressures might work as well, though.

9.5.4 Just-Noticeable-Differences

Our first two psychophysical studies have given us good data on the minimum pressure level needed for effective compression feedback notifications. However, we were also interested in how many different levels of notifications users could distinguish. We thus ran an additional experiment on the just-noticeable differences of compression feedback. Note that while we are the first to investigate absolute detection thresholds for compression feedback, the just-noticeable difference has been studied before by Mitsuda [200].

However, our setup differs from Mitsuda's work in two aspects: (1) we look at feedback given by a wearable prototype, while Mitsuda used a much larger and powerful electropneumatic regulator. We found that inflation behavior does provide subtle cues (e.g., users sensing the change in pressure, not the absolute pressure) and would expect some differences between these pump types. Secondly (2), in Mitsuda's study a staircase design (as with our first absolute detection threshold study) was used and stimuli playback alternated between the two pressure levels, without dropping to zero in between. Thus, user might pick up subtle cues from, e.g., pump activating, informing them a change is taking place. Instead, we fall back to zero pressure between stimuli, preventing participants to make observations of changing pressure between stimuli. This makes the task much harder, but also covers a different use case. For notifications, it is less relevant to detect the change from one notification to another and more important to identify one notification that is played back. Stimuli have to be detected standing on their own, instead of within a sequence of stimuli. Thus, while this makes our study not directly comparable (we would expect worse performance due to the more difficult task) the ecological validity for the scenario of notification feedback is stronger with our setup.

Thus, to determine just-noticeable differences, we gave participants stimuli pairs and asked them to determine whether the two are *equal* or *different*. Participants could play back each stimulus as often as they wanted, but had to play each one at least once to proceed. When participants played back a stimulus, the pump increased pressure up to that level which was then held for five seconds after which the air was fully released. During a stimulus, input was disabled and participants had to wait till after deflation to play back another stimulus or vote on stimuli equality. In this experiment we use four different base levels (at exponentially increasing levels of 0.5, 1.0, 2.0, and 4.0 kPa) in combination with five different offsets (0, 0.2, 0.4, 0.8, and 1.6 kPa) yielding 20 different conditions. Each condition was repeated once for a total of 40 stimuli. Stimuli order was randomized and we also randomly assigned base and offset stimuli to the two playback buttons (shown earlier in Figure 9.5). Thus, participants could not immediately tell which was the higher pressure and which was the lower pressure stimuli.

We recruited 12 participants (1 female, age 22–35, $\bar{x} = 26.8$, $SD = 4.5$) for this experiment. Seven of these participants had already participated in the first lab study, but for these repeat participants, the two lab studies were several days apart. In a post hoc check, we did not find a systematic effect for those repeat participants. We thus only report results for all participants at once and do not further distinguish repeat participants. Participants in this experiment also did not have their blood pressure taken recently. The study took about 20 minutes and was rewarded with a non-monetary gratuity.

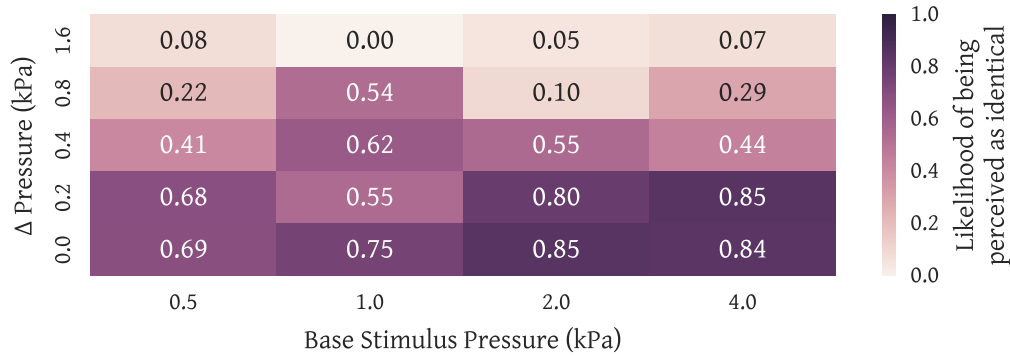


Figure 9.9: Table showing how often stimuli pairs were judged as being equal. For example, for a 4.0 kPa base pressure and a second stimulus 0.2 kPa higher, the two were rated as equal in 85 % of the respective trials. As can be seen, the higher the pressure differential, the more likely two stimuli were perceived as different.

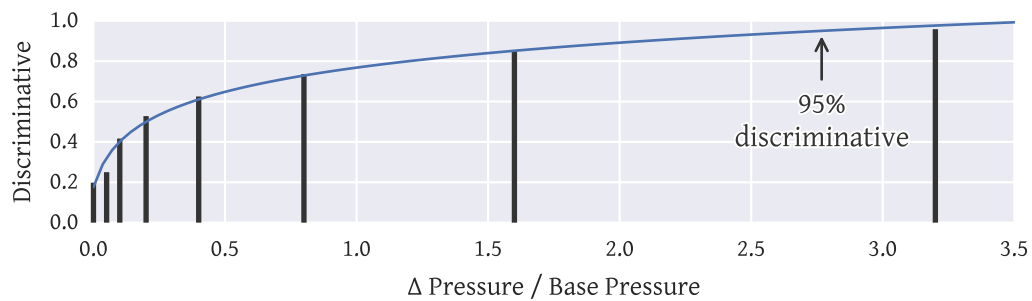


Figure 9.10: The larger the ratio of offset pressure over base pressure, the better users are able to discriminate the two stimuli. Bars are measured values, while the curve shows a least squares fitted logarithmic function.

9.5.4.1 Results

We define the just-noticeable difference as the pressure difference where 95 % of users were able to tell two stimuli apart. We chose this non-conventional definition as it is a better fit for what we try to find out: how much to space pressure stimuli for different notifications. By picking a 5 % error level, we find that difference where we can assume participants only confuse two different kinds of notifications 5 % of the time. Hence, this is also a conservative estimate and just-noticeable differences for individual users might be lower (as we saw already with the active detection threshold). In a first step, we aggregated a score (% of users who felt that the two stimuli are equal) for each base/offset pair (shown in Figure 9.9). We can see this was a hard task, especially due to the sequential playback of stimuli (participants played back 3.1 stimuli per trial—1.1 more plays than required).

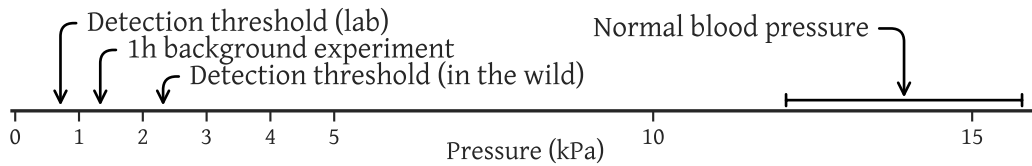


Figure 9.11: Overview of feedback pressures used in comparison to the normal blood pressure range. All pressure levels in our experiments so far are far away from pressures that would restrict blood flow.

According to *Weber’s law*, the just-noticeable difference should be constant proportional to the base level multiplied by the ratio between offset and base level. Hence two stimuli of 5 kPa and 6 kPa are supposed to be harder to distinguish than two stimuli of 1 kPa and 2 kPa. To investigate at what ratio the just-noticeable difference equals our desired 95%, we aggregated our data a second time. As shown in Figure 9.10, this aggregation is per pressure delta to base pressure ratio. We used non-linear least squares to fit a logarithmic function to the data ($R^2 = 0.99$). This model was then used to compute the value where 95% of participants would recognize the difference—at about a 2.77 offset to base pressure ratio. For example, at 0.8 kPa of pressure an increase in pressure of 2.2 kPa is needed to attain 95% chance of users being able to distinguish the two stimuli. Note that Figure 9.10 also shows the resulting ratios for other error levels. When we want to reduce the error level (make stimuli more distinguishable), we move to the right of the curve to a larger pressure delta to base pressure ratio.

Compared to Mitsuda’s results, we find a larger required offset between pressure levels. At the 4 kPa level, Mitsuda identified 0.4 kPa offset for the upper difference threshold (judged more intense in 75 % of trials). Our fitted model predicts 40 % discrimination for the same offset over the baseline. However, to reach the same 75 % level, our model predicts a need to increase pressure by 3.6 kPa—almost doubling the pressure.

9.5.5 Discussion

In the three psychophysical studies we have determined thresholds for the use of compression feedback on the wrist. However, we found that the results can be hard to put into perspective. As can be seen in Figure 9.11, the found thresholds are very low compared to the kind of pressures our straps are designed for. Similarly, the pressure we used when exploring background feedback also is quite low. While compression feedback systems will probably not use the whole pressure range shown (near blood pressure levels the strap would cut off blood supply and could not be used for longer durations), there is still much space to use for stimuli. As shown in the just-noticeable differences study, participants are able to tell stimuli apart once sufficiently spaced. This can be used to communicate different messages (two distinct levels give 1 bit of information).

The results from the absolute detection threshold and just-noticeable differences studies provide a starting point for the exploration of compression feedback. While the results provide design parameters for compression feedback's prime location—the wrist—further work is needed to determine similar parameters for other locations. In the lab setting, we also specifically investigated seated users (a common posture, e.g., for office workers) that were concentrating on the task. The in the wild study also provides data for one common scenario where notifications are used: walking around and taking public transport. Here we have also seen that mobile and distracted users are less perceptive of feedback nuances. In the comparison of our results to Mitsuda's work, we find that the used study setup has a strong impact on how well participants can tell stimuli apart. However, for our intended use case—feedback for individual notifications—our setup provides more ecologically valid results. Note though that we still ran a lab study and would expect users ability to tell stimuli apart to deteriorate in the wild. An alternative to having different pressure levels to distinguish notification types is incorporating inflation patterns. We explore this later on in this chapter.

As Figure 9.11 shows, standard blood pressure meters are actually over-dimensioned for typical application scenarios. These devices are able to generate pressure levels high enough to cut off blood flow. For safety reasons, interaction devices based on compression feedback should stay below these levels. We expect pressure levels below 5 kPa to be sufficient for many applications. Creating pressure in this range is possible with thinner inflatable straps and smaller pumps with lower energy requirements.

9.6 Reactive Compression Feedback

So far we have looked at *active* feedback—the system plays back a stimulus by inflating the strap to, e.g., signal a new notification. However, compression feedback can also be used *reactively*, i.e., giving feedback in reaction to a user's action. A user could, e.g., press down on an inflated strap to query for state (e.g., whether there is an email waiting to be read). Depending on the state, the strap would either deflate a bit or remain at the same pressure level. This allows for a different kind of notification mechanism: instead of signaling a user when a new notification comes in, users can decide for themselves when to check. The strap would inflate slowly to indicate that there is a message waiting (this is actually a kind of active notification), yet the full content is only revealed as needed.

The resistance of the strap to squeezing it is related to the internal pressure level. As illustrated in Figure 9.12, more complex deflation patterns can also be used. Furthermore, this user action does not need to be a hand pressing down on the strap, flexing the muscles underneath a strap could also trigger the same procedure (however, the amount of external force on the strap is more limited here). With *Shoogle*, Williamson et al. explored a vibrotactile/audio version of feedback mediated through exploration [315].

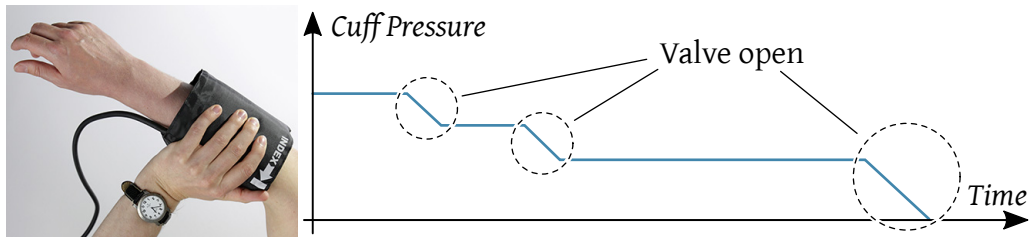


Figure 9.12: Compression feedback can also be used *reactive* to the user. As the user presses down on an inflated cuff, air is let out of the cuff in two short bursts and another longer burst slightly later. Users sense this deflation pattern by how the cuff *gives in* to the force they apply.

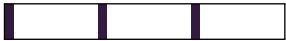









Detecting user’s manipulation of a strap can be achieved with the embedded pressure sensor. Outside force on the strap results in an increase of internal air pressure. A quick squeeze thus, e.g., can be detected as a sharp momentary peak in pressure levels. Detecting those pressure changes for use as an input modality has been explored in previous work (e.g., [150, 221, 276]). Internal pressure of controls has also been used to provide different sensations or resistance when those controls are used [86, 301]. However, using deflation patterns to provide additional information during the interaction yet needs to be investigated. As we have shown above, compression feedback is suitable for feedback lasting a longer period of time. Combined with reactive feedback, such systems can be used to communicate a persistent background information and only surface details (e.g., indicating one of three specific cases) when actively queried.

9.6.1 Evaluating Reactive Notifications

Just as we evaluated how well compression feedback can be used for active feedback, we also set out to evaluate how well it is suited to reactive notifications. For this we ran a lab study where participants were presented a range of reactive patterns and were asked to rate and distinguish them. This study makes use of the same prototype as the previous in the wild study. However, instead of wearing the device in a pouch, the system was instead placed on a table. As in the other lab studies, participants also wore headphones playing white noise.

Reactive feedback makes use of deflation patterns—the way the air moves out of the strap. Hence, such patterns are defined by how the valve opens and allows for release of air. Different patterns can be created by changing how long the valve is opened, e.g., releasing air in small consecutive bursts. Furthermore, pauses of varying length between individual pulses (air releases) can be used to differentiate patterns. For this study, we designed ten different deflation patterns (see Table 9.1).

Table 9.1: Overview of the 10 reactive feedback patterns used in our evaluation. Pulses (valve is open and air can escape) are shown as darker stretches of time.

Name	Description	Pattern
P0	Series of short pulses with breaks in between	
P1	Four short pulses followed by a break	
P2	Series of long pulses with breaks in between	
P3	Three long pulses followed by a break	
P4	Pulses of increasing length	
P5	Pulses of decreasing length	
P6	Heartbeat pattern	
P7	S.O.S. pattern	
P8	Series of very short pulses	
P9	Alternating short and long pulses	

Before a pattern can be played back, the strap is inflated to a pressure level of 8.2 kPa. While this is higher than the detection threshold levels found above, we chose to use a more fully inflated strap to be able to test more complex deflation patterns. If the internal pressure were only slightly above the detection threshold, only very short deflation patterns would be possible.

We recruited 13 additional participants for this study (3 female, age 20–53, $\bar{x} = 26.3$, $SD = 8.3$). All participants wore the strap on the wrist where they would wear a watch. Before the actual study, we then gave participants the opportunity to explore the available patterns. Each pattern was represented by an icon representing patterns as a sequence of vertical bars. Participants could click on a pattern to play it back, allowing them to build a connection between the visual representation and the haptic sensation. Once they started the actual study, it was not possible to go back to this exploration mode.

During the study, patterns were presented in randomized order. Each pattern was shown three times for a total of 30 trials. During a trial, participants could play back the pattern as often as they wanted. However, they only played back the pattern once in 74 % of the trials (on average, they played back 1.3 patterns per trial). Afterwards, they were asked to rate the pattern in four attribute dimensions: calm–hectic, soft–hard, rhythmic–arrhythmic, and pleasant–unpleasant. For each trial and dimension, participants used a continuous slider (0–1000) to indicate the level. Participants also indicated which pattern they felt was played back by picking from the visual representations they trained with before the study. They also rated their confidence in their pattern pick with another slider.

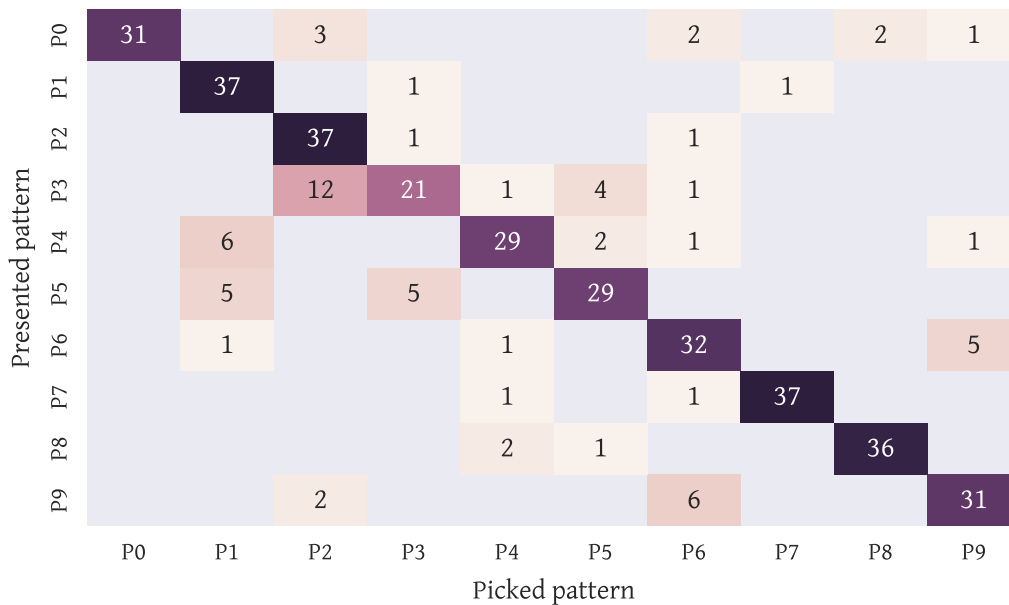


Figure 9.13: Presented with ten different reactive feedback patterns, participants correctly identified the pattern in 82 % of trials. The confusion matrix shows that while the results are overall good, patterns **P3** was slightly harder to identify than the other ones.

9.6.1.1 Results

Overall, participants were able to correctly identify the played back pattern in 82 % of trials. The confusion matrix for presented and picked patterns (see Figure 9.13) correspondingly shows peaks along the diagonal. This also shows that a subset of patterns were generally identified correctly. For example, patterns **P1**, **P2**, and **P7** all have recognition rates of 95 %. One pattern with slightly lower performance was **P3**, which was only identified correctly in 54 % of trials. Participants sometimes confused it with the steady series of pulses, indicating that the break between longer pulses was hard to detect.

So far we have only looked at quantitative measures of compression feedback. However, in the reactive feedback study, we used the opportunity to collect some qualitative ratings as well. As shown in Figure 9.14, participants rated compression feedback as slightly more soft, pleasant, and rhythmic. A paired samples t-test showed no significant difference for whether patterns tended towards *calm* or *hectic*; $t(120) = 0.24, p > 0.05$. However, paired samples t-tests showed that for the *soft*, *pleasant*, and *rhythmic* dimensions, participants rated the patterns significantly higher than the average rating of 500; $t(120) = [-3.90, -8.21, -5.92], p < 0.001$. While that difference in rating is not large, this does show that participants at least did not find the feedback unpleasant.

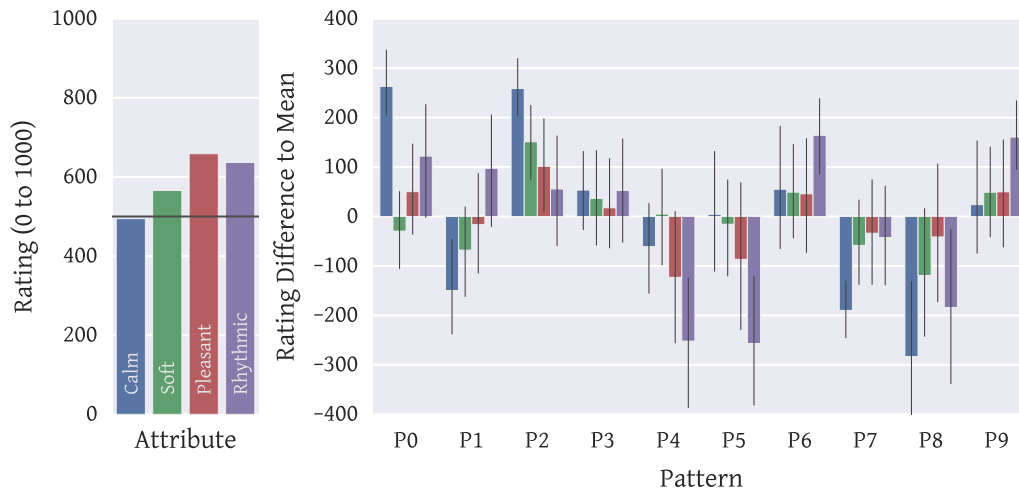


Figure 9.14: Asked to rate the patterns in four dimensions, participants found the patterns rather calm, pleasant, and rhythmic. This varies strongly by pattern though. Pattern **P8**, e.g., is particularly less calm than the others. Error bars show 95 % confidence intervals.

We also took a look at how ratings vary for different patterns. With a repeated-measures two-way ANOVA, we found significant main effects of pattern ($F(9, 108) = 9.47, p < 0.001$) and attribute ($F(3, 36) = 8.94, p < 0.001$). We also found a significant interaction of pattern and attribute ($F(27, 324) = 6.47, p < 0.001$). To investigate this, we ran pairwise permutational two-tailed t-tests with 1000 permutations for pattern combinations within an attribute (p-values adjusted for multiple comparisons with the Holm-Bonferroni method). These post hoc tests showed that there were no significant differences between patterns for the *pleasant* and *soft* attributes. However, the *calm* and *rhythmic* attributes both saw a large number of significant differences between patterns.

9.6.2 Discussion

The results show that reactive feedback is a viable format for notifications. Limiting a system to a smaller number of patterns (e.g., only for three app categories) would further boost the recognition rate. Participants also tended to rate the patterns favorably, particularly with respect to how pleasant they felt. We believe this kind of notifications could fill a gap in current systems. Not all notifications require immediate reaction, but can instead *pile up* over time and only reveal more details after the user queries them.

9.7 Vibration vs. Compression Feedback

In our studies so far, we have investigated compression feedback on its own. However, we were also wondering how it compares against vibration feedback. After all, if performance is much lower than with vibration feedback, if it is much more annoying, or if users find it not as comfortable, there is little chance compression feedback might actually get used for notifications. We thus ran a study to look at the differences between those two feedback methods.

For this study, we recruited 12 more participants (3 female, age 17–50, $\bar{x} = 25.0$, $SD = 8.3$). The study was a within-subjects design with feedback modality as the only factor. Participants sat in front of a computer, wearing headphones playing white noise. Participants wore our miniaturized mobile prototype on their wrist, housed in a 3d-printed bracelet-like enclosure. This device allowed us to attach both, compression feedback actuators and vibration feedback actuators to the controller (providing the first two feedback modalities). Furthermore, they kept a phone in their pocket. The vibration of the phone in the pocket also served as the third modality.

We designed four distinct feedback patterns for each, compression and vibrotactile feedback. This was intended to create some variability of feedback. For compression and vibrotactile feedback, respectively, these patterns were:

- | | |
|---|---|
| <ul style="list-style-type: none">• Inflate then deflate | <ul style="list-style-type: none">• Vibrate once for 1 s |
| <ul style="list-style-type: none">• Inflate, hold pressure briefly, deflate | <ul style="list-style-type: none">• Vibrate twice for 0.5 s with a 0.2 s pause in between |
| <ul style="list-style-type: none">• Inflate, let out some air, inflate again, deflate | <ul style="list-style-type: none">• Three short 0.2 s vibrations with 0.2 s breaks in between |
| <ul style="list-style-type: none">• Inflate in four equal-sized steps with brief plateaus in between then deflate | <ul style="list-style-type: none">• Two longer (1.5 and 2 s) vibrations with a 0.2 s pause in between |

The three feedback conditions were counterbalanced and patterns were randomized. As a primary task, participants repeatedly played a memory game on the computer for ten minutes in each condition. While playing the game, a notification would trigger every 40–60 s. If participants noticed this, they took out the phone and acknowledged the stimulus. Before the study, participants were given 10–15 minutes to familiarize themselves with the devices. Afterwards, participants filled out a questionnaire, rating the different feedback types.

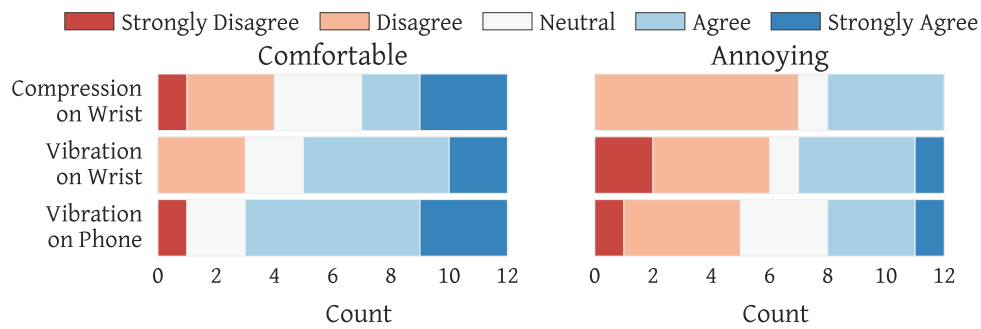


Figure 9.15: Comparing vibration and compression feedback, both elicited similar responses from participants asked to rate how comfortable and annoying they found them.

9.7.1 Results

We first checked whether performance in the primary task was impacted differently by the three feedback types with a repeated-measure one-way ANOVA. The used feedback method had no significant effect on how fast participants completed individual memory games ($F(2, 22) = 0.97, p > 0.05$), or how many mistakes they made in the game ($F(2, 22) = 0.87, p > 0.05$). How many notifications were missed was also not significantly different between conditions ($F(2, 22) = 0.58, p > 0.05$). However, the time it took participants to react to a notification was significantly different between the feedback types ($F(2, 22) = 22.96, p < 0.001$). To investigate this, we ran a pairwise permutational two-tailed t-test with 1000 permutations with p-values adjusted for multiple comparisons with the Holm-Bonferroni method. While there was no significant difference between the two vibration feedback conditions, compression feedback resulted in significantly ($p < 0.05$) slower reactions to the notifications. Participants in both vibration feedback conditions reacted in 9.1 s while it took them 12.0 s to react to pressure feedback stimuli.

If we compare how participants rated vibration and compression feedback after the study (see Figure 9.15) we see that there are no strong differences in how they were perceived. We checked for differences between the ratings with Friedman tests. For both the comfort ($\chi^2(2) = 0.84, p = 0.66$) and the annoyance ($\chi^2(2) = 0.55, p = 0.76$) rating we found no significant differences between the feedback modalities. Both modalities polarized users with respect to how annoying they found them. Asked whether they found the feedback comfortable, the vibration phone received slightly higher scores than the two wrist-mounted methods. This might be due to the unfamiliar prototype or the additional weight of it.

9.7.2 Discussion

Overall, this study shows that compression and vibrotactile feedback have similar impact on primary task performance. The only difference was that it took participants slightly longer to react to compression feedback stimuli. This is most likely the case because it takes a short moment for compression feedback to become noticeable. While vibrotactile feedback is instantaneous, compression feedback straps need to inflate beyond a certain level to become noticeable at all (as investigated in our psychophysical studies). However, this difference is rather small and would not prohibit use of compression feedback. Qualitative responses by participants also showed that this kind of feedback is not more annoying than vibration feedback. Comfort levels could be further improved by future prototypes that conform better to the wearer's wrist.

9.8 Beyond Notifications

So far, we have explored compression feedback as an alternative to vibrotactile feedback, with a focus on use for notifications. Thus, all the tests were also been run with the compression feedback apparatus on the wrist. While this provided us with in-depth data on this use case of compression feedback, it only represents a small part of the larger compression feedback design space. In this section, we explore other ways compression feedback can be used. This includes a way to use compression feedback for much more intense feedback, by inhibiting users. Such more intense feedback complements the rather subtle use for notifications we have explored so far. Where we ran several studies to investigate compression feedback use with notifications, the approach in this section is more exploratory. For several of the use cases and observations, we only conducted informal testing within the lab. While this is sufficient for showing the breadth of compression feedback, further studies are needed to more formally evaluate these kind of setups.

9.8.1 Inflation Profiles

We already saw in Sections 9.6 and 9.7 how compression feedback can do more than just basic inflation. Instead, inflation profiles, or *patterns*, can be used to change how the feedback feels. Inflation behavior can encode different messages, e.g., an urgent notification could be represented as strong inflations with periodic valve openings for sudden pressure releases. We found this gives a hectic and *yanking* sensation. On the other hand, a slow pressure build-up is useful for messages not requiring immediate attention. Here the act of inflating is not as noticeable and instead the feedback slowly rises in intensity as it shifts from the background to the foreground.

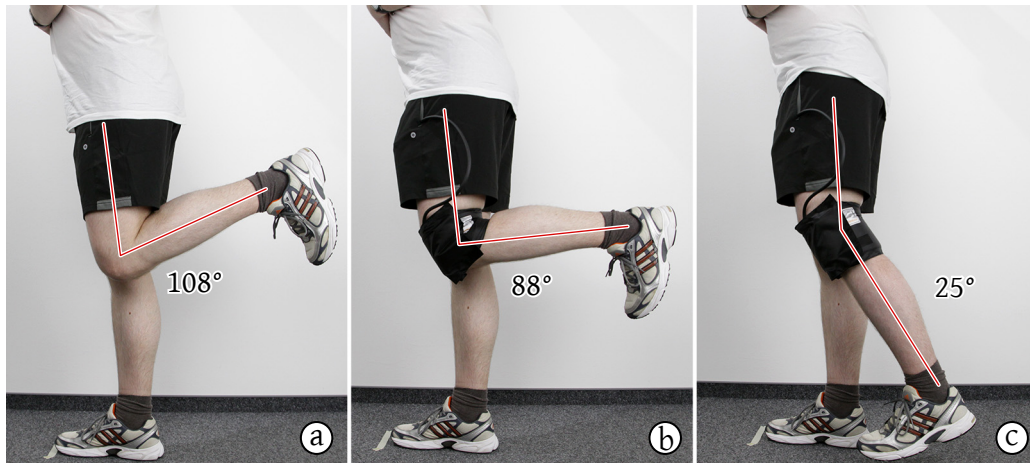


Figure 9.16: Freedom of leg movement changes when wearing (a) no strap, (b) a deflated strap over the knee, and (c) an inflated strap over the knee. While having a strap over the knee limits movement slightly, the effect is much stronger with an inflated strap.

When inflating and deflating so pressure behaves sinusoidal, the feedback gains lifelike properties and feels like a heartbeat or breathing motions. In our experiments we found that just changing the frequency of this pressure oscillation can provide strong feedback cues. While low frequencies are calming, with increased frequency the feedback quickly gains in urgency. This perception was also echoed in feedback by participants of our demonstration at CHI 2015 [236]. This can be used for general notifications, but has particularly large potential in intimate communication. Just the sound of a heartbeat can already help facilitate a feeling of intimacy in virtual communication [136]. While we have not explored this further, we believe compression feedback in such a configuration could be beneficial.

9.8.2 Jamming Bodies with Compression Feedback

In addition to exerting an inward force, compression feedback, with increased pressure levels and depending on strap placement, can also be used to restrict freedom of body movement. When placed over a joint and inflated, a strap can *lock* that joint in the resting position. For example, placing a strap over the knee allows control over how much the lower leg can be raised. As shown in Figure 9.16, full inflation (we inflated to ≈ 11.5 kPa) strongly decreases the range of motion. While just wearing the strap restricts movement a bit, the remaining range of movement is sufficient for, e.g., normal walking. Larger inflatable tubes, *air splints*, are already used in medicine to immobilize extremities and for physiotherapy. *Jamming User Interfaces* [80] explored similar changes between rigid and deformable states which occur when placing granular material inside the air bubble and vacuuming it.



Figure 9.17: Different strap positions enable different sensations on strong inflation. While lower and upper leg placement give a feeling of *weighing one down*, placement on the knee can effectively *lock* the leg. Thus one just *feels* restricting while the other one is inhibiting movement. Straps could be affixed at positions by sewing them into pants.

Placement of a strap directly over a joint is needed for direct mechanical locking of an extremity. However, other placements can also have an inhibiting effect on the user (as illustrated in Figure 9.17). For example, placing a strap around the lower leg and inflating it strongly provides a sensation as if the leg is *weighed down*, while not restricting freedom of movement. An inflated strap does put pressure on the underlying muscles, increasing the force needed for tensing.

9.8.3 Case Study: Compression Feedback for a Jogging Game

While we did explore jamming feedback in the lab, we have yet to see how this works in actual realistic use. We thus decided to implement a case study for a pervasive game that uses compression feedback. To keep things simple, we decided to create a single player game that could work with slow changing feedback (allowing us to use a small pump and develop a mobile system) and picked a jogging game.

Our game concept is based on a classic arcade game design pattern, *checkpoints*, introduced in *Namco's Pole Position*³ in 1982. Here a timer is running and players need to reach a checkpoint in a given amount of time. Should the timer run out before such a checkpoint is reached, the game ends. This added challenge can increase excitement [59]. The app *Zombies, Run!* [321] tries to create a similar situation by using audio feedback, to give players the impression that zombies are chasing them. Instead of having a virtual source of stress, we set out to create a physical stress force.

³See, e.g., <https://www.youtube.com/watch?v=FFs1Xc82Q0U>



Figure 9.18: We created a jogging game where runners are pushed to reach checkpoints in a given time via compression feedback. Runners wear a compression feedback strap around one knee. The strap is controlled from a controller board in a waist bag.

In our game, runners put a compression feedback strap around their knee (see Figure 9.18) and connect it to the mobile version of our compression feedback controller (lower right device in Figure 9.3). All components are carried by the participants in a waist bag (see Figure 9.7). This system is controlled by a phone app that tracks the jogger via GPS. During jogging, the strap around the runner's knee slowly inflates. The system is designed in a way that freedom-of-movement is not noticeably restricted in the initial state. As the strap inflates, joggers feel their corresponding leg being more and more inhibited. They need to reach virtual checkpoints to trigger the system to stop inflating and let out the air. This cycle of slowly increasing inhibition and release once a checkpoint is reached then repeats.

Checkpoints are spaced at 200 m intervals along the run. Runners are free to take any path they want, as the system only tracks overall distance and not the actual course. Due to the constriction on the leg, there is an added dimension of urgency added to their run. Some parts of the route (those closer to checkpoints) are more intense than others (just after release). By adjusting how much urgency is induced, the system acts as a kind of pace maker for the runner (like the quadcopters accompanying runners in [205]). As the pressure increases, users desire to achieve relief increases as well. Hence the system is designed in a way to push users towards *preemptively* changing their pacing, as they feel the onset of the feedback.

We tested our system in a park with 7 participants (1 female, age 19–26, $\bar{x} = 23.4$, $SD = 2.3$), three of whom were regular joggers. Participants ran for 6–10 minutes on a self-chosen course (i.e., we did not instruct them to run any specific route). We interviewed participants after they returned from their run.

Three participants rated the system as neither particularly comfortable nor uncomfortable, while two each rated comfort as a bit lower/higher than neutral. As this is a system that actively adds discomfort at regular intervals, this shows us we did not make the feedback too intense. All but one participant thought such a system could be an interesting companion for joggers, and three participants noted they themselves were open to using something like it. Participants gave varying reports on the effectiveness of the system: two reported not having been inhibited during their run, while two others thought the feedback was quite strong (one more participant noted too much pressure on the knee). This variance could be due to differences in individuals perception or to physiological differences, where the strap is more effective on some legs than others (e.g., due to placement on, or anatomy of the leg).

Our jogging game is but one example of a use case of compression feedback in pervasive games [13, 186]. There are general scenarios where compression feedback could augment pervasive gameplay. First of all, such feedback could be used to communicate game state. A tightening strap could indicate that a team is behind in points, a clock running out (as in our jogging game), or proximity to game objects (such as in a game of *hot and cold*). Pressure could also be directly related to player actions: The weight of a picked-up virtual object could be represented by compression feedback around the lower leg. A virtual path with rising and falling slope could be represented as increasing and decreasing pressure around the thighs. Increasing and decreasing pressure from bottom to top would be felt all over the body and could, e.g., simulate the effects of a potion.

More interestingly though, compression feedback could be used to physically inhibit some players or whole teams, with a direct impact on game-playing difficulty. This can be as simple as tightening a strap around a player's arm holding the egg in an egg-and-spoon race. High pressure on the arm reduces dexterity and agility, thus increasing the challenge in getting the egg safely to the finish line. Similarly, other limbs can be inhibited to, e.g., reduce throwing abilities or influence a runner—a scenario we will explore below. This could also be used to achieve real encumbrance when carrying virtual items. In general, the ability to temporarily inhibit players is useful for skill balancing and giving virtual objects and events a physical effect.

9.8.4 Beyond the Wrist

In our investigation of compression feedback we have mostly looked at placement on the wrist. However, in the jogging game evaluation, we already saw that this is not the only feasible location for compression feedback. In addition to the wrist, several other body locations are possible (as illustrated in Figure 9.19). The upper arm is already a common location for straps worn during workouts which could readily include compression feedback. Similarly, belts are an obvious place for compression. However, belts also require a large amount of air to achieve a compressive effect. Large amounts of tissue around the waist would also lead to a less pronounced effect than on, e.g., the wrist.

Figure 9.19: Wearables with compression feedback can be worn around many body parts. We imagine the wrist to be the most commonly used one. Other locations could be appropriate for more specific scenarios (e.g., wearing a strap on the upper arm is common during jogging).

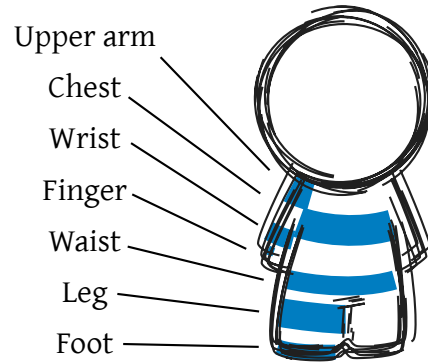


Figure 9.20: Neonatal blood pressure cuffs are small enough to be worn around fingers. Fixed-sized rings would be even more compact, but integration of circuitry and pneumatics is challenging at sizes this small. Instead, the feedback apparatus could also be integrated into a glove.



One interesting location are fingers, where rings could inflate to create compression feedback stimuli. We have experimented with neonatal cuffs worn around fingers (see Figure 9.20). In this setup, small inflations are felt easily, however, we found the range to be more limited than, e.g., on the wrist. Yet, the small strap size also means that only very little air is needed for inflation and no large pumps is needed. Miniaturizing a compression feedback device to fit into an actual ring is challenging though and is left to explore for future work.

The compression effect usually requires a strap to wrap around a body part. This requirement can be relaxed when the body part is already constrained (e.g., when wearing a bracelet as in Figure 9.22). We have also experimented with inserting inflatable bubbles into shoes (as illustrated in Figure 9.21). When inflated, this gives a sensation of the foot being pressed down. We found walking not to be restricted when inflated as the bubble above the foot does not limit freedom of movement of foot joints. If this is desired, a strap could be put around the ankle, or the shoe could include corresponding bubbles (only applicable to high-tops or boots). In the 80s, the *Reebok Pump* introduced inflation as a mean to ensure tight fit, a concept also used, e.g., in ski boots. Instead of only improving fit, we propose such inflation mechanisms could be changed to also provide haptic feedback to wearers.

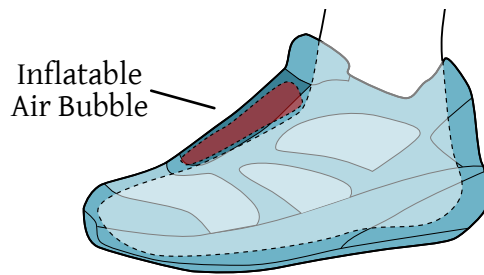


Figure 9.21: Instead of a strap around the foot, compression feedback is possible with only an added air bubble. As the foot is already strapped into the shoe, inflating that bubble results in an equivalent sensation.



Figure 9.22: Bracelets already wrap around the arm and thus, in combination with an air bubble attached anywhere along the strap, can be used to create compression feedback.

Instead of using straps, inflatable pockets can be directly integrated into clothing. Gloves, shirts, pants, shoes, belts, sweatbands, or socks already cover the relevant locations and could incorporate such sewn in pockets. Inflatable vests, e.g., are already available commercially⁴ and are used to provide a hug-like sensation. Such vests are, e.g., used to help reduce anxiety in autistic children [73]. Perovich et al. put pneumatic channels in a skirt to enable changing its form interactively [230]. Inflatable garments are also used in physical therapy, however, such garments do not resemble regular clothes. Costumes and fetish wear are areas where inflatable elements are used as well.

We see particular promise for integration of compression feedback into sports wear. Such clothing is often already form fitting and thus could easily integrate air bladders that only need slight inflation to be perceived. While this would make such garments adjustable (e.g., to provide dynamic support), they could also assist players, e.g., by providing context information during a game. One could imagine feeling a slight pressure on the left when another player is available for a pass on that side or when an opponent approaches. At lower inflation levels, compression feedback would not inhibit movement, but could provide subtle cues on what movements to perform. We already discussed such pervasive gaming scenarios above.

⁴e.g., from *Squease* (<http://www.squeasewear.com/>)

9.9 Summary

In this chapter, we have explored *compression feedback*. Compression feedback differs from other forms of pressure feedback or vibration feedback. It can be rather subtle but can scale to strong sensations as well. We have concentrated on the performance of compression feedback as a notification mechanism. In three psychophysical studies, we quantified minimal pressure levels at which a stimulus is felt (in the wild and in the lab) and how well two stimuli can be distinguished. We find that, on average, users can feel compression stimuli above 0.7 kPa (1.2 kPa when out in the wild). However, for some users this threshold can be as low as 0.22 kPa. Hence tuning feedback parameters to individual users would be worthwhile and can open up more of the pressure range. Moreover, users are able to distinguish two stimuli with 95% probability when the ratio between test level and base level is 2.7. We verified suitability for longer use with a 1 h long acceptability experiment.

In two additional studies we explored the properties of reactive compression feedback and compared compression feedback to vibration feedback. In both cases, different patterns are explored—an important feedback aspect to differentiate different types of notifications. We find that with a subset of three different patterns, participants were able to identify them in 95 % of cases—sufficient for real-world use. Results from those two studies thus further underline the general suitability of compression feedback for notifications.

We have also briefly taken a look at other uses of compression feedback. In particular, restricting users with stronger compression was tested in a jogging game. This shows that while compression feedback can be used for standard notification feedback, it also scales up to more invasive use cases. Having a feedback that is able to scale from low attention capture up to a level where it “forces” users to react fits well into the conceptual frame discussed in Chapter 7.

While compression feedback can already be mobile, the needed components are still relatively large. We expect to see further miniaturization enabling even better integration into wearables and other devices. Our smallest prototype only weights 60 g, including the battery, pump, and valve. However, the housing adds additional weight and the low integration of the components means the overall design is comparably bulky. Where we had to resort to parts from blood pressure monitors, these are designed for higher pressure ranges than required for compression feedback. Pumps and valves specifically designed for our purpose could help with further miniaturization. In this challenge, integrating the inflatable air bladders themselves is comparably straightforward. For example, as we have discussed, there have already been shoes on the market for decades that have an inflatable interior.

Many more design properties remain to be explored, though. For example, acceptability of sustained pressure feedback in everyday scenarios, attention capture quality when distractions are present, or differences of strap placement all require further investigation. With internal valves between adjacent chambers, a sensation of spreading pressure in a wave pattern could be achieved. Composite layers with cut patterns [325], could be used to further distinguish different pressure levels. Straps could stay rigid in some places, adding a displacement force to the feedback.

There are a large number of additional application areas to be explored, such as compression feedback when watching movies. Passengers waiting at an airport could be notified with increasing urgency of gate times or train passengers of approaching stops. Students could literally put pressure on professors giving a lecture when they struggle. Straps over elbows integrated in shirts could inflate when unhealthy food is detected. This would provide a noticeable resistance to eating said food which is not easily perceivable to bystanders. Straps could provide subtle information about what a user's pet is doing: if her cat is hunting the feedback is different than when it is napping. Similarly, sentiment of larger groups such as sports fans or celebrity followers could be communicated or a more personal grasp on your arm could be sent from distant friends and family. Instead of showing outside state, compression feedback can also be used to increase self-awareness. For example, mirroring a user's heartbeat in a strap could help users focus on and better understand their body's reactions. In general, compression feedback is an interesting addition to the haptic feedback repertoire of interactive systems, providing an additional design space to explore.

10 Conclusion

[...] obstinacy is perhaps the only human quality that matters at the end of the day, not only in the profession of the policeman but in many professions. At least in any that have something to do with the notion of truth.

— Michel Houellebecq, *The Map and the Territory*

In this dissertation we have described the concept of casual interaction and taken a look at several application areas and example systems that are built according to this concept. The underlying principle of casual interaction is to enable users to trade off some of their control over an interaction for gained convenience (or to allow for interaction in constrained settings). Hence, casual interaction fits best in scenarios where such lower engagement is possible and desirable. As technology is ever more closely weaved into our daily lives, such opportunities are more common than rare. For example, the smart home or the personal digital device both are ubiquitously available, yet we would not want to devote all our attention to them. On the other hand, casual interaction offers little to improve interaction with office suites on desktop workstations, or for traders on Bloomberg terminals¹.

However, many settings are more amenable to casual interactions. At home or during recreational activities users, e.g., might not want to expend too much effort. For example, as discussed in Chapter 4, a challenge of interaction in the home is how to offer simple means of interaction for increasingly complex systems. If each room contains multiple sensors and output channels (e.g., lighting systems), controlling everything individually can become overwhelming. By relinquishing some control to the home and using higher level commands (such as mood changing ones), users can stay in control while avoiding dealing with unwanted levels of detail. This increasing complexity of the world around us is not limited to the home. In fact, some are postulating a convergence of the physical world with the “*cyberspace*”, making computing truly pervasive [55]. Yet, if computation is embedded all around us there need to be ways to interact with this device ensemble that do not rely on close engagement. Finding higher level abstractions and allowing users to pick the one best suited for their current task (i.e., designing for casual interaction) is one approach this might be tackled.

¹A computer system for access to the *Bloomberg Professional* service with a complex control layout, designed for focused and highly specialized/trained work.

If we look at the wider definition of casual interaction that also includes casual communication, then the need for such systems is less driven by convenience and impacted more by the desire for informality. This brings with it a different kind of relinquishing control—one where ambiguity and playfulness are more acceptable. As we have seen in the discussion of existing mechanisms in the Apple Watch and in the work on emoji in Chapter 6, this can be applied to communication in several different ways. If we relate this back to the reasons for casual interaction we discussed in Chapter 2, then work on casual communication relates primarily to social constraints on interaction. Projecting a certain image or nourishing a connection is aided by such more informal means of messaging. Yet, casual communication is but one way to approach casual interaction, as is the effort avoidance one mentioned above. In fact, each of the projects explored in this dissertation brings a different aspect to the table.

We can take a broader view again and look back at the systems investigated so far and what they focus on. One common thread here has been effort avoidance. In Chapter 3, we started with the effort of having to reach for the phone and instead investigated moving interaction to the space around the device. Effort in this respect primarily refers to the physical effort, but can also incorporate the notion of not having to step back from something else. Apart from easing constraints on *where* to interact, within around-device interaction we also explored the concept of having coarser input and output. Distance to the device is a natural way to control such coarseness, but the general idea of changing granularity of input/output is one we revisit multiple times over the course of this dissertation. In addition to moving *off the device*, we also looked at using *proxies* for interaction, which fulfills the same purpose: allow for coarser interactions (also allowing for less focus due to guidance by objects' affordance).

After around-device interaction, we took a look at the smart home in Chapter 4. Where before we were more interested in specific interactions, the smart home was a good place to also look at wider patterns of casual interaction. For example, light control is something that should be implicit/incidental (but at least low effort) most of the time, and only require a close interaction sometimes. Compared to interaction with the phone, the home is a more likely place for repetitive and pervasive interaction patterns. Systems like our capacitive couch prototype can fill this role, by providing a steady stream of sensor data that most of the time is only used to reaffirm preexisting reaction patterns (e.g., keep the reading light on when someone sits on the sofa). Hence a crucial aspect of the home is how to make more control available when needed, but keep the control surface low in normal use. The couch and the one-button recognizer are ways to embed interaction in the environment, while our light control bracelet tried to combine the whole spectrum of control into one wearable.

In the following two Chapters (5 & 6), we turn to text as a medium for expression. While we focus on actually inputting text in the former, the latter takes a broader look at casual communication (as already described above). The continuous control of autocorrection in Chapter 5 does also support more expression (by giving users an easier way to enter words deviating from the standard vocabulary), but primarily shows how the control handoff need not be discrete. This is in contrast to most systems discussed in this dissertation that make a second (or more) channel available for interaction for users to choose from. Instead of having to choose, being able to continuously adjust the level of control saves users the (however small) transition cost between different input methods. But in Chapter 6, while still text entry related, we move slightly aback from concrete interfaces and investigate text entry from a more general casual perspective. Yet, this work is a necessary condition to then again provide concrete systems that allow users a way to casually express themselves. We saw one example of this with EmojiZoom, but the emoji modeling lays the foundation for more advanced systems in the future. Emoji are also but one example of casual expression and other means, such as animated GIFs or kaomoji, offer many opportunities for similar expressivity.

Last but not least, we investigated feedback for casual interaction. As this is quite a different angle than the more input-focused work so far, we first looked at the generally needed properties of such feedback in Chapter 7. Chapters 8 & 9 then detail specific aspects of concrete casual feedback systems. We explored two modalities that we believe add a more subtle alternative to feedback methods such as vibrotactile: indirect light feedback and compression feedback. In the latter case, we actually saw how this feedback can scale up in intensity and can also provide restrictive stimuli far from subtle. Both projects demonstrate that casual interaction does not just provide a way to frame input in interactive systems, but also takes into account how feedback can support less engaged interactions. We discussed this early on in Section 2.4, as well as in Chapter 3, when we adapted notification granularity depending on user engagement.

10.1 Limitations

As we have looked at several casual interaction systems we have already discussed some of their limitations as well. But on a more general level, there are some drawbacks inherent to all of casual interaction. Casual interaction systems might: (1) increase complexity of the overall system and thus require more effort from users to start using them, (2) fail to properly utilize the relegated control and thus frustrate the user, or (3) prevent skill progression by supporting unengaged user behavior. The severity of each of these problems depends on the specific casual interaction system.

One concern with casual interaction systems is the increased complexity of offering multiple ways to interact with them. For example, consider a wearable with several interaction modes, as seen in Chapter 4. Instead of training for one interface, users here need to (1) keep in mind several ways to interact, and (2) not confuse the interactions of different modes. However, casual interaction need not necessarily mean different modes. As we saw in Chapter 5, we can use variations inside a continuous input space to move between focused and casual interactions. Yet, once users consciously want to pick an engagement level, complexity issues might still present themselves. Another way to address this problem is to treat the different levels of engagement as experience levels. Users can start with a system in a more casual mode and pick up individual, more focused, interactions as needed. They then would not need to memorize the whole system at once, but delay learning to a later time where such an escalation is needed. However, such an approach would be less suitable to systems where the ability to take more control fast is critical and delayed learning would introduce unacceptable action delays.

As users yield control to the system in more casual interactions, a critical question is whether that yielded control is properly utilized. For example, the autocorrection system from Chapter 5 only offers a benefit to users if the corrections actually are sensible. If users engage less and only provide sloppy input, they need to be able to rely on the system fixing their writing. Note that this is less of a problem of the control-trade-off in casual interaction and more of an issue of the specific target area. However, if lower-engagement interaction does not need to acceptable results, trust in casual interactions in general could suffer. Thus, well-performing models to support casual interaction levels need to be in place to allow for users to interact with less engagement.

Finally, a more general problem is the question of how use of casual interaction systems could prevent skill progression of users. While complex computing systems can be a burden, mastering their complexity does allow users increased control and a larger say in the outcome of interactions. Instead, users might eschew that training and just settle for only casual interactions. This would require them to live with less control over the outcome of their interactions, but would also allow them to be less engaged (e.g., lazy) much more. However, from an overall perspective it would be problematic if this behavior becomes normal and all effort is avoided. But should we purposely make systems harder as not to let users slump to some lethargic state? While casual interaction might support less engagement the underlying fundamentals leading to effort-avoiding behavior are likely not fixed by artificially introducing hardship. As the law of less work and related concepts describe it, the desire for less effort is deeply rooted and we each individually have to find our own motivations to engage.

Overall, these limitations do not go against engaging less, but call for carefully designing casual interaction systems. We have to make sure to balance complexity, how much a system takes over, and what to surface to users. But we should also keep in mind that casual interaction should not mean just designing for low engagement, but to design for the full spectrum. In contrast to agent systems, users should be empowered to take control and not just to let the system take over completely.

10.2 Future Work

While we have explored several different application areas and techniques for casual interaction, there remain a large number of aspects to explore. In this final section, we will take a look at some of these and what could be next for casual interaction.

In Section 2.6 we learned about the reasons for which users might desire more casual interactions: social, physical, and mental. However, much of the work in this dissertation has mostly focused on providing interfaces that cater to users' desire to invest less work. For example, effort was a central question of the autocorrect work in Chapter 5, or the around device interaction work in Chapter 3. The one-button recognizer from Chapter 4 is one project that partly also caters to physical constraints. After all: even when, e.g., carrying bags, users might still be able to press a light switch, while entering a keycode or inserting a keycard might be out of the question. Yet, physical constraints have not received the same amount of focus as mental constraints.

The more obvious omission, though, are social constraints. While there is evidence that desire for coolness or a good impression with others can influence the way we use our devices, we have not built a system or ran a study to explore this. Designing such systems likely requires a different approach than the more engineering-driven one taken with the work presented so far. Yet, as technology permeates our daily lives, more explicitly designing for the way others perceive our interactions with it certainly is worthwhile. However, casual interaction only covers part of this area and a different approach might, e.g., ask how to design interfaces that make a user look powerful or dominant.

Designing casual interaction systems for social constraints would probably also require a different kind of user study setup than the ones we have applied. To adequately judge how perceptions of users of such systems change, a longer term and in situ deployment is desirable. While we have run in the wild studies, one even lasting a whole day (for indirect light feedback in Chapter 8), this is still not quite what would be desirable here. Areas where moving to such longitudinal, in situ, experiments would be most worthwhile are likely the smart home (as outlined in Chapter 4), but also personal mobile device (most likely the phone) use in general. One question to start with could be: "do users prefer systems that make them look cool to outside observers?" We already know that more beautiful devices are considered more usable [297], but does this transfer to ones signaling *casualness* to others?

Apart from different constraints, there is much open work in the models underlying casual interaction systems. With the current resurgence of artificial intelligence research, and in particular deep learning, models could be built for casual interaction. In fact, many of the projects described in this dissertation already make use of machine learning to guide the interaction once users delegate some control. The one-button recognizer is one example of this. If we can build more advanced models that better capture likely user intents, this could be used for assistive systems that relieve users of having to control everything by themselves. An example of this is the burgeoning personal assistant space, where systems like Microsoft's *Cortana* or Amazon's *Alexa* try to offer services, such as management of the user's calendar or reminders. The intention of these systems is to "make life easier" for their users, by offloading mundane tasks like scheduling or making a reservation. We can relate this back to the principles discussed in this dissertation and see that such assistants fulfill a role quite similar to casual interaction systems: they allow users to do *less*, but on the other hand *automate* some of the process. This shows, that the artificial intelligence methods underlying such systems are already applied for casual interaction. Hence, there is promise in further pushing for more detailed and intricate user models, in order to better be able to make use of control delegated by the user.

Conversational agents are an particularly interesting aspect of this wave of artificial intelligence. Recently, there has been a push² to bring more of such agents to applications like Skype, Slack, or just plain text messaging. The idea here is that instead of having dedicated applications for things like home control, restaurant reservations, or flight bookings, applications are represented as contacts in the user's address book and invoking them is replaced by starting a conversation with them. While this might be a fad, this interaction as conversation gives us an interesting twist on where we started: the definition of casual. Where a lot of previous work used casual to refer to casual communication, we went a different route (with the exception of the emoji research) and related more to the *relaxed and unconcerned* interpretation of the term. Yet, if systems become conversational, this brings casual communication back to the forefront. Current conversational agents are commonly transactional and users are taken through a series of steps to achieve their goal (such as booking a flight). However, we might wonder what a casual chat (reminiscent of the classic water cooler conversation) with such an agent would look like. Could a bot for our smart home be part of the family's *WhatsApp* group, and build its models of how best to assist us based on these conversations? These are exciting avenues to explore if we can indeed do a better job at learning models and inferring suitable behavior from them. Incorporating current artificial intelligence research findings into casual interaction systems thus certainly is something to explore more in-depth in the future.

²For example, Microsoft is facilitating such development with their *Bot Framework*: <https://dev.botframework.com/>

10.3 Closing Remarks

In describing casual interactions, I often find myself telling people what it is not: focused interactions. Yet, this is not ideal in that it places casual interactions solely as an antithesis to some common paradigm. Another way I found it useful to describe is as “interaction for lazy people”, but while this provides an immediate graspable idea, this is also an incomplete view of what casual interaction is. It can also be hard to communicate that no, automating everything would not really be casual interactions. The underlying concept at work here is that sometimes we would like to scale back and get away with doing less. But, importantly, we still make that choice and *give up* control—it is not taken from us. I feel this is a very positive aspect in an environment where we see massive growth of computing technology all around us, but very little answers to how to stay on top of it all. Clearly, we sometimes need to engage less, be it to engage with those around us or just because we are tired. Designing for casual interaction means designing to empower users to do so—whatever form this might take in a given situation or scenario.

References

- [1] Heba Abdelnasser, Moustafa Youssef, and Khaled A. Harras. “WiGest: A Ubiquitous WiFi-based Gesture Recognition System.” In: *2015 IEEE Conference on Computer Communications (INFOCOM)*. IEEE, 2015, pp. 1472–1480. DOI: 10 . 1109/INFOCOM . 2015.7218525 (cit. on p. 45).
- [2] Johnny Accot and Shumin Zhai. “Beyond Fitts’ Law: Models for Trajectory-Based HCI Tasks.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI ’97*. New York, New York, USA: ACM Press, 1997, pp. 295–302. DOI: 10 . 1145/258549 . 258760 (cit. on pp. 26, 45).
- [3] R. Rox Anderson and John A. Parrish. “The Optics of Human Skin.” In: *Journal of Investigative Dermatology* 77.1 (1981), pp. 13–19. DOI: 10 . 1111 / 1523 - 1747 . ep12479191 (cit. on pp. 212, 214, 215).
- [4] Ahmed Sabbir Arif and Wolfgang Stuerzlinger. “Pseudo-Pressure Detection and Its Use in Predictive Text Entry on Touchscreens.” In: *Proceedings of the 25th Australian Computer-Human Interaction Conference on Augmentation, Application, Innovation, Collaboration - OzCHI ’13*. New York, New York, USA: ACM Press, 2013, pp. 383–392. DOI: 10 . 1145/2541016 . 2541024 (cit. on pp. 116, 130).
- [5] Tanvir Aumi and Sven Kratz. “AirAuth: Evaluating In-Air Hand Gestures for Authentication.” In: *Proceedings of the 16th International Conference on Human-Computer Interaction with Mobile Devices and Services - MobileHCI ’14*. Toronto, ON, Canada: ACM, 2014 (cit. on p. 89).
- [6] Daniel Avrahami, Jacob O. Wobbrock, and Shahram Izadi. “Portico: Tangible Interaction on and around a Tablet.” In: *Proceedings of the 24th annual ACM symposium on User interface software and technology - UIST ’11*. New York, New York, USA: ACM Press, 2011, pp. 347–356. DOI: 10 . 1145/2047196 . 2047241 (cit. on pp. 46, 50).
- [7] Gilles Bailly, Antti Oulasvirta, Duncan P. Brumby, and Andrew Howes. “Model of Visual Search and Selection Time in Linear Menus.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI ’14*. New York, New York, USA: ACM Press, 2014, pp. 3865–3874. DOI: 10 . 1145/2556288 . 2557093 (cit. on p. 177).

- [8] Saeideh Bakhshi, David A. Shamma, Lyndon Kennedy, Yale Song, Paloma de Juan, and Joseph 'Jofish' Kaye. "Fast, Cheap, and Good: Why Animated GIFs Engage Us." In: *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems - CHI '16*. New York, New York, USA: ACM Press, 2016, pp. 575–586. DOI: 10.1145/2858036.2858532 (cit. on p. 199).
- [9] Saskia Bakker, Elise van den Hoven, and Berry Eggen. "Acting by hand: Informing interaction design for the periphery of people's attention." In: *Interacting with Computers* 24.3 (Apr. 2012), pp. 119–130. DOI: 10.1016/j.intcom.2012.04.001 (cit. on pp. 18, 22, 109).
- [10] Saskia Bakker, Elise van den Hoven, Berry Eggen, and Kees Overbeeke. "Exploring Peripheral Interaction Design for Primary School Teachers." In: *Proceedings of the Sixth International Conference on Tangible, Embedded and Embodied Interaction - TEI '12*. Vol. 1. 212. New York, New York, USA: ACM Press, 2012, pp. 245–252. DOI: 10.1145/2148131.2148184 (cit. on p. 111).
- [11] Saskia Bakker and Karin Niemantsverdriet. "The Interaction-Attention Continuum: Considering Various Levels of Human Attention in Interaction Design." In: *International Journal of Design* 10.2 (2016) (cit. on p. 18).
- [12] Thomas Baudel and Michel Beaudouin-Lafon. "Charade: Remote Control of Objects Using Free-Hand Gestures." In: *Communications of the ACM* 36.7 (1993), pp. 28–35. DOI: 10.1145/159544.159562 (cit. on p. 53).
- [13] Patrick Baudisch, Henning Pohl, Stefanie Reinicke, Emilia Wittmers, Patrick Lühne, Marius Knaust, Sven Köhler, Patrick Schmidt, and Christian Holz. "Imaginary Reality Gaming: Ball Games Without a Ball." In: *Proceedings of the 26th annual ACM symposium on User interface software and technology - UIST '13*. New York, New York, USA: ACM Press, 2013, pp. 405–410. DOI: 10.1145/2501988.2502012 (cit. on p. 258).
- [14] Matthew A. Baumann, Karon E. MacLean, Thomas W. Hazelton, and Ashley McKay. "Emulating Human Attention-Getting Practices with Wearable Haptics." In: *Proceedings of the 2010 IEEE Haptics Symposium*. IEEE, 2010, pp. 149–156. DOI: 10.1109/HAPTIC.2010.5444662 (cit. on pp. 209, 233, 235).
- [15] Roy F. Baumeister, Ellen Bratslavsky, Mark Muraven, and Dianne M. Tice. "Ego Depletion: Is the Active Self a Limited Resource?" In: *Journal of Personality and Social Psychology* 74.5 (1998), pp. 1252–1265. DOI: 10.1037/0022-3514.74.5.1252 (cit. on p. 22).
- [16] Joanna Bergstrom-Lehtovirta, Antti Oulasvirta, and Stephen Brewster. "The Effects of Walking Speed on Target Acquisition on a Touchscreen Interface." In: *Proceedings of the 13th International Conference on Human Computer Interaction with Mobile Devices and Services - MobileHCI '11*. New York, New York, USA: ACM Press, 2011, pp. 143–146. DOI: 10.1145/2037373.2037396 (cit. on p. 20).

- [17] Xiaojun Bi, Yang Li, and Shumin Zhai. “Fitts Law: Modeling Finger Touch with Fitts’ Law.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI ’13*. Paris, France: ACM, 2013, pp. 1363–1372. DOI: 10 . 1145/2470654 . 2466180 (cit. on p. 115).
- [18] Xiaojun Bi, Barton A. Smith, and Shumin Zhai. “Multilingual Touchscreen Keyboard Design and Optimization.” In: *Human-Computer Interaction 27.4* (2012), pp. 352–382. DOI: 10 . 1080/07370024 . 2012 . 678241 (cit. on p. 176).
- [19] Xiaojun Bi and Shumin Zhai. “Bayesian Touch - A Statistic Criterion of Target Selection with Finger Touch.” In: *UIST ’13: Proceedings of UIST 2013 - The ACM Symposium on User Interface Software and Technology*. New York, NY, USA, 2013, pp. 51–60 (cit. on pp. 115, 118).
- [20] Matthias Böhmer, Christian Lander, Sven Gehring, Duncan P. Brumby, and Antonio Krüger. “Interrupted by a Phone Call: Exploring Designs for Lowering the Impact of Call Notifications for Smartphone Users.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI ’14*. New York, New York, USA: ACM Press, 2014, pp. 3045–3054. DOI: 10 . 1145/2556288 . 2557066 (cit. on p. 150).
- [21] Daniel Boland and Roderick Murray-Smith. “Finding My Beat: Personalised Rhythmic Filtering for Mobile Music Interaction.” In: *Proceedings of the 15th International Conference on Human-Computer Interaction with Mobile Devices and Services - Mobile-HCI ’13*. New York, New York, USA: ACM Press, 2013, pp. 21–30. DOI: 10 . 1145/2493190 . 2493220 (cit. on p. 29).
- [22] S. J. Bolanowski, G. A. Gescheider, R. T. Verrillo, and C. M. Checkosky. “Four channels mediate the mechanical aspects of touch.” In: *The Journal of the Acoustical Society of America* 84 (1988), pp. 1680–1694. DOI: 10 . 1121 / 1 . 397184 (cit. on p. 235).
- [23] Alan Borning and Michael Travers. “Two Approaches to Casual Interaction Over Computer and Video Networks.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI ’91*. New York, New York, USA: ACM Press, 1991, pp. 13–19. DOI: 10 . 1145/108844 . 108847 (cit. on p. 8).
- [24] Peter Brandes. “Interaction with Compression Feedback on the Wrist.” Bachelor Thesis. Leibniz Universität Hannover, 2016 (cit. on pp. ix, 231).
- [25] Alan Bränzel, Christian Holz, Daniel Hoffmann, Dominik Schmidt, Marius Knaust, Patrick Lühne, René Meusel, Stephan Richter, and Patrick Baudisch. “GravitySpace: Tracking Users and Their Poses in a Smart Room Using a Pressure-Sensing Floor.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI ’13*. New York, New York, USA: ACM Press, 2013, pp. 725–734. DOI: 10 . 1145/2470654 . 2470757 (cit. on p. 89).

- [26] Andreas Braun, Reiner Wichert, Arjan Kuijper, and Dieter W. Fellner. “Capacitive proximity sensing in smart environments.” In: *Journal of Ambient Intelligence and Smart Environments* 7.4 (2015), pp. 483–510. DOI: 10.3233/AIS-150324 (cit. on p. 89).
- [27] Leo Breiman. “Technical Note: Some Properties of Splitting Criteria.” In: *Machine Learning* 24.1 (1996), pp. 41–47. DOI: 10.1007/BF00117831 (cit. on p. 94).
- [28] Stephen A. Brewster and Michael Hughes. “Pressure-Based Text Entry for Mobile Devices.” In: *Proceedings of the 11th International Conference on Human-Computer Interaction with Mobile Devices and Services - MobileHCI '09*. New York, New York, USA: ACM Press, 2009, 9:1–9:4. DOI: 10.1145/1613858.1613870 (cit. on p. 116).
- [29] MA Brown, Wolfgang Stuerzlinger, and E.J. Mendonça Filho. “The Performance of Un-Instrumented In-Air Pointing.” In: *Proceedings of the 2014 Graphics Interface Conference*. 2014, pp. 59–66 (cit. on p. 74).
- [30] Jesse Burstyn, Paul Strohmeier, and Roel Vertegaal. “DisplaySkin: Exploring Pose-Aware Displays on a Flexible Electrophoretic Wristband.” In: *Proceedings of the Ninth International Conference on Tangible, Embedded, and Embodied Interaction - TEI '15*. New York, New York, USA: ACM Press, 2015. DOI: 10.1145/2677199.2680596 (cit. on p. 111).
- [31] Karoline Busse. “Casual Interaction with a Bracelet.” Master Thesis. Leibniz Universität Hannover, 2014 (cit. on pp. ix, 86).
- [32] Alex Butler, Shahram Izadi, and Steve Hodges. “SideSight: Multi-”touch” interaction around small devices.” In: *Proceedings of the 21st annual ACM symposium on User interface software and technology - UIST '08*. New York, New York, USA: ACM Press, 2008, pp. 201–204. DOI: 10.1145/1449715.1449746 (cit. on p. 44).
- [33] Bill Buxton. “Integrating the Periphery and Context: A New Taxonomy of Telematics.” In: *Proceedings of Graphics Interface - GI '95*. 1995, pp. 239–246 (cit. on p. 17).
- [34] Christopher Campbell and Peter Tarasewich. “What Can You Say with Only Three Pixels?” In: *Proceedings of the 6th International Symposium on Mobile Human-Computer Interaction - MobileHCI '04*. 2004, pp. 1–12. DOI: 10.1007/978-3-540-28637-0_1 (cit. on p. 214).
- [35] Spencer Cappallo, Thomas Mensink, and Cees G.M. Snoek. “Image2Emoji: Zero-shot Emoji Prediction for Visual Media.” In: *Proceedings of the 23rd ACM international conference on Multimedia*. 2015, pp. 1311–1314. DOI: 10.1145/2733373.2806335 (cit. on pp. 161, 181).
- [36] Spencer Cappallo, Thomas Mensink, and Cees G.M. Snoek. “Query-by-Emoji Video Search.” In: *Proceedings of the 23rd ACM international conference on Multimedia*. 2015, pp. 735–736. DOI: 10.1145/2733373.2807961 (cit. on p. 161).

- [37] Andrew Carvey, Jim Gouldstone, Pallavi Vedurumudi, Adam Whiton, and Hiroshi Ishii. “Rubber Shark as User Interface.” In: *CHI '06 Extended Abstracts on Human Factors in Computing Systems - CHI EA '06*. New York, New York, USA: ACM Press, 2006, pp. 634–639. DOI: 10.1145/1125451.1125582 (cit. on pp. 46, 55).
- [38] Álvaro Cassinelli, Stéphane Perrin, and Masatoshi Ishikawa. “Smart Laser-Scanner for 3D Human-Machine Interface.” In: *CHI '05 Extended Abstracts on Human Factors in Computing Systems - CHI EA '05*. New York, New York, USA: ACM Press, 2005, pp. 1138–1139. DOI: 10.1145/1056808.1056851 (cit. on p. 45).
- [39] Jessica R. Cauchard, Janette L. Cheng, Thomas Pietrzak, and James A. Landay. “ActiVibe: Design and Evaluation of Vibrations for Progress Monitoring.” In: *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems - CHI '16*. New York, New York, USA: ACM Press, 2016, pp. 3261–3271. DOI: 10.1145/2858036.2858046 (cit. on p. 217).
- [40] Tao Chen and Min-Yen Kan. “Creating a live, public short message service corpus: the NUS SMS corpus.” In: *Language Resources and Evaluation* (Aug. 2012), pp. 1–37. DOI: 10.1007/s10579-012-9197-9 (cit. on p. 126).
- [41] Xiang ‘Anthony’ Chen, Tovi Grossman, and George Fitzmaurice. “Swipeboard: A Text Entry Technique for Ultra-Small Interfaces that Supports Novice to Expert Transitions.” In: *Proceedings of the 27th annual ACM symposium on User interface software and technology - UIST '14*. New York, New York, USA: ACM Press, 2014, pp. 615–620. DOI: 10.1145/2642918.2647354 (cit. on p. 164).
- [42] Xiang ‘Anthony’ Chen, Julia Schwarz, Chris Harrison, Jennifer Mankoff, and Scott E. Hudson. “Air+Touch: Interweaving Touch & In-Air Gestures.” In: *Proceedings of the 27th annual ACM symposium on User interface software and technology - UIST '14*. New York, New York, USA: ACM Press, 2014, pp. 519–525. DOI: 10.1145/2642918.2647392 (cit. on p. 46).
- [43] Kai-Yin Cheng, Rong-Hao Liang, Bing-Yu Chen, Rung-Huei Laing, and Sy-Yen Kuo. “iCon: Utilizing Everyday Objects as Additional , Auxiliary and Instant Tabletop Controllers.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '10*. New York, New York, USA: ACM Press, 2010, pp. 1155–1164. DOI: 10.1145/1753326.1753499 (cit. on pp. 46, 55).
- [44] Francesco Chinello, Mirko Aurilio, Claudio Pacchierotti, and Domenico Praticchizzo. “The HapBand: A Cutaneous Device for Remote Tactile Interaction.” In: *Proceedings of EuroHaptics - EuroHaptics '14*. 2014, pp. 284–291. DOI: 10.1007/978-3-662-44193-0_36 (cit. on pp. 233, 235).
- [45] Sangwon Choi, Jaehyun Han, Sunjun Kim, Seongkook Heo, and Geehyuk Lee. “ThickPad: A Hover-Tracking Touchpad for a Laptop.” In: *Proceedings of the 24th annual ACM symposium adjunct on User interface software and technology - UIST '11 Adjunct*. New York, New York, USA: ACM Press, 2011, pp. 15–16. DOI: 10.1145/2046396.2046405 (cit. on p. 44).

- [46] V. G. Chouvardas, a. N. Miliou, and M. K. Hatalis. “Tactile Displays: Overview and Recent Advances.” In: *Displays* 29 (2008), pp. 185–194. DOI: 10.1016/j.displa.2007.07.003 (cit. on p. 234).
- [47] Allan Christensen, Simon A. Pedersen, Per Bjerre, Andreas K. Pedersen, and Wolfgang Stuerzlinger. “Transition Times for Manipulation Tasks in Hybrid Interfaces.” In: *Proceedings of the 18th International Conference on Human-Computer Interaction*. 2014, pp. 138–150 (cit. on p. 42).
- [48] Karen Church, Denzil Ferreira, Nikola Banovic, and Kent Lyons. “Understanding the Challenges of Mobile Phone Usage Data.” In: *Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services - MobileHCI '15*. New York, New York, USA: ACM Press, 2015, pp. 504–514. DOI: 10.1145/2785830.2785891 (cit. on p. 136).
- [49] David A. Cieslak and Nitesh V. Chawla. “Learning Decision Trees for Unbalanced Data.” In: *Proc. PKKD*. 2008, pp. 241–256. DOI: 10.1007/978-3-540-87479-9_34 (cit. on p. 94).
- [50] Adrian Clark, Andreas Dünser, Mark Billingham, Thammathip Piumsomboon, and David Altimira. “Seamless Interaction in Space.” In: *Proceedings of the 23rd Australian Computer-Human Interaction Conference on - OzCHI '11*. New York, New York, USA: ACM Press, 2011, pp. 88–97. DOI: 10.1145/2071536.2071549 (cit. on p. 19).
- [51] Edward C. Clarkson, Shwetak N. Patel, Jeffrey S. Pierce, and Gregory D. Abowd. *Exploring Continuous Pressure Input for Mobile Phones*. Tech. rep. GIT-GVU-06-20. Georgia Tech, 2006, pp. 1–4 (cit. on p. 116).
- [52] A. Cockburn, P. Quinn, C. Gutwin, G. Ramos, and J. Looser. “Air Pointing: Design and Evaluation of Spatial Target Acquisition with and without Visual Feedback.” In: *International Journal of Human-Computer Studies* 69.6 (2011), pp. 401–414. DOI: 10.1016/j.ijhcs.2011.02.005 (cit. on p. 81).
- [53] Andrew Cockburn and Saul Greenberg. “Making Contact: Getting the Group Communicating with Groupware.” In: *Proceedings of the conference on Organizational computing systems - COCS '93*. New York, New York, USA: ACM Press, 1993, pp. 31–41. DOI: 10.1145/168555.168559 (cit. on p. 9).
- [54] Gabe Cohn, Daniel Morris, Shwetak Patel, and Desney Tan. “Humantenna: Using the Body as an Antenna for Real-Time Whole-Body Interaction.” In: *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems - CHI '12*. New York, New York, USA: ACM Press, 2012, pp. 1901–1910. DOI: 10.1145/2207676.2208330 (cit. on p. 44).

- [55] Marco Contia, Sajal K. Dasb, Chatschik Bisdikianc, Mohan Kumarb, Lionel M. Nid, Andrea Passarella, George Roussose, Gerhard Trösterf, Gene Tsudikg, and Franco Zambonellih. “Looking ahead in pervasive computing: Challenges and opportunities in the era of cyber-physical convergence.” In: *Pervasive and Mobile Computing* 8.1 (2012), pp. 2–21. DOI: 10 . 1016/j . pmc j . 2011 . 10 . 001 (cit. on p. 263).
- [56] Christian Corsten, Chat Wacharamanotham, and Jan Borchers. “Fillables: Everyday Vessels As Tangible Controllers with Adjustable Haptics.” In: *CHI '13 Extended Abstracts on Human Factors in Computing Systems - CHI EA '13*. New York, New York, USA: ACM Press, 2013, pp. 2129–2138. DOI: 10 . 1145/2468356 . 2468732 (cit. on pp. 46, 52).
- [57] Enrico Costanza, Samuel A. Inverso, and Rebecca Allen. “Toward Subtle Intimate Interfaces for Mobile Devices Using an EMG Controller.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '05*. New York, New York, USA: ACM Press, 2005, pp. 481–489. DOI: 10 . 1145 / 1054972 . 1055039 (cit. on p. 21).
- [58] Enrico Costanza, Samuel A. Inverso, Elan Pavlov, Rebecca Allen, and Pattie Maes. “eye-q: Eyeglass Peripheral Display for Subtle Intimate Notifications.” In: *Proceedings of the 8th Conference on Human-Computer Interaction with Mobile Devices and Services - MobileHCI '06*. New York, New York, USA: ACM Press, 2006, pp. 211–218. DOI: 10 . 1145/1152215 . 1152261 (cit. on pp. 204, 208, 217, 228).
- [59] Ben Cowley, Darryl Charles, Michaela Black, and Ray Hickey. “Toward an understanding of flow in video games.” In: *Computers in Entertainment* 6.2 (2008), p. 1. DOI: 10 . 1145/1371216 . 1371223 (cit. on p. 256).
- [60] Henriette Cramer, Paloma de Juan, and Joel Tetreault. “Sender-Intended Functions of Emojis in US Messaging.” In: *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services - MobileHCI '16*. New York, New York, USA: ACM Press, 2016, pp. 504–509. DOI: 10 . 1145/2935334 . 2935370 (cit. on p. 144).
- [61] Martin Culjat, Chih-Hung King, Miguel Franco, James Bisley, Warren Grundfest, and Erik Dutson. “Pneumatic Balloon Actuators for Tactile Feedback in Robotic Surgery.” In: *Industrial Robot: An International Journal* 35.5 (2008), pp. 449–455. DOI: 10 . 1108/01439910810893617 (cit. on p. 233).
- [62] Sajal K. Das, Diane J. Cook, Amiya Bhattacharya, Edwin O. Heierman, and Tze-Yun Lin. “The Role of Prediction Algorithms in the MavHome Smart Home Architecture.” In: *IEEE Wireless Communications* 9.6 (2002), pp. 77–84. DOI: 10 . 1109/MWC . 2002 . 1160085 (cit. on p. 85).
- [63] Dodge Daverman and Michael Olley. *Multi-touch force sensing touch-screen devices and methods*. EU Patent 2329341A2. 2011 (cit. on p. 130).

- [64] Bruno R. De Araujo, Géry Casiez, and Joaquim A. Jorge. “Mockup Builder: Direct 3D Modeling On and Above the Surface in a Continuous Interaction Space.” In: *Proceedings of the 2012 Graphics Interface Conference - GI'12*. 2012, pp. 173–180 (cit. on p. 45).
- [65] Alexander De Luca, Alina Hang, Frederik Brudy, Christian Lindner, and Heinrich Hussmann. “Touch me once and i know it’s you!: implicit authentication based on touch screen patterns.” In: *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems - CHI '12*. New York, New York, USA: ACM Press, 2012, pp. 987–996. DOI: 10.1145/2207676.2208544 (cit. on p. 88).
- [66] Marco Deneke. “Sammeln von Emoji-Bewertungen mit einem kompetitiven Spiel.” Bachelor Thesis. Leibniz Universität Hannover, 2016 (cit. on p. ix).
- [67] Anind K. Dey and Alan Newberger. “Support for Context-Aware Intelligibility and Control.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '09*. New York, New York, USA: ACM Press, 2009, pp. 859–868. DOI: 10.1145/1518701.1518832 (cit. on p. 88).
- [68] Carl DiSalvo, Francine Gemperle, Jodi Forlizzi, and Elliott Montgomery. “The Hug: An Exploration of Robotic Form for Intimate Communication.” In: *Proceedings of the 12th IEEE International Workshop on Robot and Human Interactive Communication - ROMAN '03*. 2003, pp. 403–408. DOI: 10.1109/ROMAN.2003.1251879 (cit. on p. 234).
- [69] Alan Dix. “Beyond intention - pushing boundaries with incidental interaction.” In: *Proceedings of Building Bridges: Interdisciplinary Context-Sensitive Computing*. Glasgow University, 2002, pp. 1–6 (cit. on pp. 17, 90).
- [70] Mark Dunlop and John Levine. “Multidimensional Pareto Optimization of Touchscreen Keyboards for Speed, Familiarity and Improved Spell Checking.” In: *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems - CHI '12*. New York, New York, USA: ACM Press, 2012, pp. 2669–2678. DOI: 10.1145/2207676.2208659 (cit. on p. 176).
- [71] Sezer Dursun. “Grob- und Feininteraktion mit einem ambienten Display.” Bachelor Thesis. Leibniz Universität Hannover, 2016 (cit. on p. ix).
- [72] Florian Echtler, Simon Nestler, Andreas Dippon, and Gudrun Klinker. “Supporting casual interactions between board games on public tabletop displays and mobile devices.” In: *Personal and Ubiquitous Computing 13.8* (2009), pp. 609–617. DOI: 10.1007/s00779-009-0246-3 (cit. on p. 9).
- [73] Stephen M. Edelson, Meredyth G. Edelson, David C. R. Kerr, and Temple Grandin. “Behavioral and Physiological Effects of Deep Pressure on Children With Autism: A Pilot Study Evaluating the Efficacy of Grandin’s Hug Machine.” In: *American Journal of Occupational Therapy 53.1979* (1999), pp. 145–152. DOI: 10.5014/ajot.53.2.145 (cit. on p. 260).

- [74] Darren Edge and Alan F. Blackwell. "Peripheral Tangible Interaction by Analytic Design." In: *Proceedings of the 3rd International Conference on Tangible and Embedded Interaction - TEI '09*. New York, New York, USA: ACM Press, 2009, pp. 69–76. DOI: 10.1145/1517664.1517687 (cit. on p. 18).
- [75] Mario Enriquez, Oleg Afonin, Brent Yager, and Karon Maclean. "A Pneumatic Tactile Alerting System for the Driving Environment." In: *Proceedings of the 2001 workshop on Perceptive user interfaces - PUI '01*. New York, New York, USA: ACM Press, 2001, pp. 1–7. DOI: 10.1145/971478.971506 (cit. on p. 233).
- [76] Thomas Erickson. "Some Problems with the Notion of Context-Aware Computing." In: *Communications of the ACM* 45.2 (2002), pp. 102–104. DOI: 10.1145/503124.503154 (cit. on p. 13).
- [77] Richard E. Fan, Martin O. Culjat, Chih-Hung King, Miguel L. Franco, Richard Boryk, James W. Bisley, Erik Dutson, and Warren S. Grundfest. "A Haptic Feedback System for Lower-Limb Prostheses." In: *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 16.3 (2008), pp. 270–277. DOI: 10.1109/TNSRE.2008.920075 (cit. on pp. 233, 235).
- [78] Thomas B. Fitzpatrick. "The Validity and Practicality of Sun-Reactive Skin Types I Through VI." In: *Archives of Dermatology* 124.6 (1988), pp. 869–871. DOI: 10.1001/archderm.1988.01670060015008 (cit. on p. 142).
- [79] Frank O. Flemish, Catherine A. Adams, Sheila R. Conway, Ken H. Goodrich, Michael T. Palmer, and Paul C. Schutte. *The H-Metaphor as a Guideline for Vehicle Automation and Interaction*. Tech. rep. TM-2003-212672. NASA, 2003 (cit. on p. 14).
- [80] Sean Follmer, Daniel Leithinger, Alex Olwal, Nadia Cheng, and Hiroshi Ishii. "Jamming User Interfaces: Programmable Particle Stiffness and Sensing for Malleable and Shape-Changing Devices." In: *Proceedings of the 25th annual ACM symposium on User interface software and technology - UIST '12*. New York, New York, USA: ACM Press, 2012, pp. 519–528. DOI: 10.1145/2380116.2380181 (cit. on pp. 233, 255).
- [81] Jodi Forlizzi, Carl DiSalvo, John Zimmerman, Bilge Mutlu, and Amy Hurst. "The SenseChair: The lounge chair as an intelligent assistive device for elders." In: *Proceedings of the 2005 conference on Designing for User eXperience - DUX '05*. New York, NY, USA: AIGA: American Institute of Graphic Arts, 2005, 31:1–31:13 (cit. on p. 89).
- [82] Euan Freeman, Stephen Brewster, and Vuokko Lantz. "Do That, There: An Interaction Technique for Addressing In-Air Gesture Systems." In: *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems - CHI '16*. New York, New York, USA: ACM Press, 2016, pp. 2319–2331. DOI: 10.1145/2858036.2858308 (cit. on p. 72).

- [83] Mayank Goel, Leah Findlater, and Jacob Wobbrock. “WalkType: Using Accelerometer Data to Accomodate Situational Impairments in Mobile Touch Screen Text Entry.” In: *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems - CHI '12*. New York, New York, USA: ACM Press, 2012, pp. 2687–2696. DOI: 10.1145/2207676.2208662 (cit. on pp. 20, 115).
- [84] Mayank Goel, Jacob O. Wobbrock, and Shwetak N. Patel. “GripSense : Using Built-In Sensors to Detect Hand Posture and Pressure on Commodity Mobile Phones.” In: 2012 (cit. on p. 130).
- [85] Erving Goffman. *The Presentation of Self in Everyday Life*. New York, NY: Anchor, 1959, p. 259 (cit. on p. 21).
- [86] Kristian Gohlke, Eva Hornecker, and Wolfgang Sattler. “Pneumatibles: Exploring Soft Robotic Actuators for the Design of User Interfaces with Pneumotactile Feedback.” In: *Proceedings of the TEI '16: Tenth International Conference on Tangible, Embedded, and Embodied Interaction - TEI '16*. New York, New York, USA: ACM Press, 2016, pp. 308–315. DOI: 10.1145/2839462.2839489 (cit. on p. 248).
- [87] Joshua Goodman, Gina Venolia, Keith Steury, and Chauncey Parker. “Language modeling for soft keyboards.” In: *Proc. AAAI 2002*. AAAI Press, 2002, pp. 419–424 (cit. on p. 115).
- [88] Bernhard Graimann, Gert Pfurtscheller, and Brendan Allison, eds. *Brain-Computer Interfaces: Revolutionizing Human-Computer Interaction*. The Frontiers Collection. Berlin, Heidelberg: Springer, 2010. ISBN: 978-3-642-02090-2. DOI: 10.1007/978-3-642-02091-9 (cit. on p. 16).
- [89] Saul Greenberg. “Peepholes: Low Cost Awareness of One’s Community.” In: *Conference Companion on Human Factors in Computing Systems - CHI '96*. New York, New York, USA: ACM Press, 1996, pp. 206–207. DOI: 10.1145/257089.257283 (cit. on p. 9).
- [90] Sven Karsten Greiner. “Direkte Interaktion im Nahbereich mobiler Geräte.” Bachelor Thesis. Leibniz Universität Hannover, 2014 (cit. on pp. ix, 41).
- [91] Tobias Grosse-Puppenthal, Andreas Braun, Felix Kamieth, and Arjan Kuijper. “Swiss-Cheese Extended: An Object Recognition Method for Ubiquitous Interfaces based on Capacitive Proximity Sensing.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '13*. New York, New York, USA: ACM Press, 2013, pp. 1401–1410. DOI: 10.1145/2470654.2466186 (cit. on p. 44).
- [92] Tobias Alexander Große-Puppenthal, Alexander Marinc, and Andreas Braun. “Classification of User Postures with Capacitive Proximity Sensors in AAL-Environments.” In: *Proceedings of the Second International Joint Conference on Ambient Intelligence - AmI'11*. Springer Berlin/Heidelberg, 2011, pp. 314–323. DOI: 10.1007/978-3-642-25167-2_43 (cit. on pp. 89, 102).

- [93] Tovi Grossman and Ravin Balakrishnan. "Pointing at Trivariate Targets in 3D Environments." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '04*. Vol. 6. 1. New York, New York, USA: ACM Press, 2004, pp. 447–454. DOI: 10.1145/985692.985749 (cit. on p. 83).
- [94] Asela Gunawardana, Tim Paek, and Christopher Meek. "Usability Guided Key-Target Resizing for Soft Keyboards." In: *Proceedings of the 15th international conference on Intelligent user interfaces - IUI '10*. New York, New York, USA: ACM Press, 2010, pp. 111–118. DOI: 10.1145/1719970.1719986 (cit. on pp. 115, 119).
- [95] Sidhant Gupta, Daniel Morris, Shwetak Patel, and Desney Tan. "SoundWave: Using the Doppler Effect to Sense Gestures." In: *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems - CHI '12*. New York, New York, USA: ACM Press, 2012, pp. 1911–1914. DOI: 10.1145/2207676.2208331 (cit. on p. 45).
- [96] Sean Gustafson, Patrick Baudisch, Carl Gutwin, and Pourang Irani. "Wedge: Clutter-Free Visualization of Off-Screen Locations." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '08*. New York, New York, USA: ACM Press, 2008, pp. 787–796. DOI: 10.1145/1357054.1357179 (cit. on p. 50).
- [97] Sean Gustafson, Daniel Bierwirth, and Patrick Baudisch. "Imaginary Interfaces: Spatial Interaction with Empty Hands and without Visual Feedback." In: *Proceedings of the 23rd annual ACM symposium on User interface software and technology - UIST '10*. New York, New York, USA: ACM Press, 2010, pp. 3–12. DOI: 10.1145/1866029.1866033 (cit. on p. 83).
- [98] Mark Hall, Hazeltine National, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. "The WEKA Data Mining Software: An Update." In: *SIGKDD Explorations* 11 (2009), pp. 10–18. DOI: 10.1145/1656274.1656278 (cit. on pp. 89, 102).
- [99] Michael Haller, Christoph Richter, Peter Brandl, Sabine Gross, Gerold Schossleitner, Andreas Schrempf, Hideaki Nii, Maki Sugimoto, and Masahiko Inami. "Finding the Right Way for Interrupting People Improving Their Sitting Posture." In: *Proceedings of the 13th International Conference on Human-Computer Interaction - INTERACT '11*. 2011, pp. 1–17. DOI: 10.1007/978-3-642-23771-3_1 (cit. on p. 232).
- [100] Jeffrey T. Hancock, Christopher Landrigan, and Courtney Silver. "Expressing Emotion in Text-Based Communication." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '07*. New York, New York, USA: ACM Press, 2007, pp. 929–932. DOI: 10.1145/1240624.1240764 (cit. on p. 150).
- [101] Rebecca Hansson and Peter Ljungstrand. "The Reminder Bracelet: Subtle Notification Cues for Mobile Devices." In: *CHI '00 extended abstracts on Human Factors in Computing Systems - CHI EA '00*. 2000, pp. 323–324 (cit. on p. 204).

- [102] Rebecca Hansson, Peter Ljungstrand, and Johan Redström. “Subtle and Public Notification Cues for Mobile Devices.” In: *Proceedings of Ubicomp 2001: Ubiquitous Computing*. Ed. by Gregory D Abowd, Barry Brumitt, and Steven Shafer. Berlin, Heidelberg: Springer, 2001, pp. 240–246. DOI: 10 . 1007 / 3 - 540 - 45427 - 6 _ 20 (cit. on p. 208).
- [103] Chris Harrison, John Horstman, Gary Hsieh, and Scott Hudson. “Unlocking the Expressivity of Point Lights.” In: *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems - CHI '12*. New York, New York, USA: ACM Press, 2012, pp. 1683–1692. DOI: 10 . 1145 / 2207676 . 2208296 (cit. on pp. 209, 214).
- [104] Chris Harrison and Scott E. Hudson. “Abracadabra: Wireless, High-Precision, and Unpowered Finger Input for Very Small Mobile Devices.” In: *Proceedings of the 22nd annual ACM symposium on User interface software and technology - UIST '09*. New York, New York, USA: ACM Press, 2009, pp. 121–124. DOI: 10 . 1145 / 1622176 . 1622199 (cit. on p. 43).
- [105] Chris Harrison and Scott E. Hudson. “Lightweight Material Detection for Placement-aware Mobile Computing.” In: *Proceedings of the 21st annual ACM symposium on User interface software and technology - UIST '08*. New York, New York, USA: ACM Press, 2008, pp. 279–282. DOI: 10 . 1145 / 1449715 . 1449761 (cit. on p. 55).
- [106] Chris Harrison and Scott E. Hudson. “Providing Dynamically Changeable Physical Buttons on a Visual Display.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '09*. New York, New York, USA: ACM Press, 2009, pp. 299–308. DOI: 10 . 1145 / 1518701 . 1518749 (cit. on p. 233).
- [107] Chris Harrison, Brian Y. Lim, Aubrey Shick, and Scott E. Hudson. “Where to Locate Wearable Displays? Reaction Time Performance of Visual Alerts from Tip to Toe.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '09*. New York, New York, USA: ACM Press, 2009, p. 941. DOI: 10 . 1145 / 1518701 . 1518845 (cit. on pp. 219, 228).
- [108] Chris Harrison, Munehiko Sato, and Ivan Poupyrev. “Capacitive Fingerprinting: Exploring User Differentiation by Sensing Electrical Properties of the Human Body.” In: *Proceedings of the 25th annual ACM symposium on User interface software and technology - UIST '12*. New York, New York, USA: ACM Press, 2012, pp. 537–544. DOI: 10 . 1145 / 2380116 . 2380183 (cit. on p. 89).
- [109] Khalad Hasan, David Ahlström, and Pourang Irani. “AD-Binning: Leveraging Around-Device Space for Storing , Browsing and Retrieving Mobile Device Content.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '13*. New York, New York, USA: ACM Press, 2013, pp. 899–908. DOI: 10 . 1145 / 2470654 . 2466115 (cit. on pp. 46, 54).

- [110] Liang He, Cheng Xu, Ding Xu, and Ryan Brill. “PneuHaptic: Delivering Haptic Cues with a Pneumatic Armband.” In: *Proceedings of the 2015 ACM International Symposium on Wearable Computers - ISWC '15*. New York, New York, USA: ACM Press, 2015, pp. 47–48. DOI: 10 . 1145/2802083 . 2802091 (cit. on p. 233).
- [111] Steven J. Henderson and Steven Feiner. “Opportunistic Controls: Leveraging Natural Affordances as Tangible User Interfaces for Augmented Reality.” In: *Proceedings of the 2008 ACM symposium on Virtual reality software and technology - VRST '08*. New York, New York, USA: ACM Press, 2008, pp. 211–218. DOI: 10 . 1145/1450579 . 1450625 (cit. on pp. 46, 52).
- [112] Niels Henze, Enrico Rukzio, and Susanne Boll. “Observational and experimental investigation of typing behaviour using virtual keyboards for mobile devices.” In: *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems - CHI '12*. ACM, 2012, pp. 2659–2668. DOI: 10 . 1145/2208636 . 2208658 (cit. on pp. 115, 118, 150).
- [113] Seongkook Heo and Geehyuk Lee. “Force Gestures: Augmenting Touch Screen Gestures with Normal and Tangential Forces.” In: *Proceedings of the 24th annual ACM symposium on User interface software and technology - UIST '11*. New York, New York, USA: ACM Press, 2011, pp. 621–626. DOI: 10 . 1145/2047196 . 2047278 (cit. on p. 115).
- [114] Seongkook Heo and Geehyuk Lee. “ForceTap: Extending the Input Vocabulary of Mobile Touch Screens by adding Tap Gestures.” In: *Proceedings of the 13th International Conference on Human Computer Interaction with Mobile Devices and Services - MobileHCI '11*. New York, New York, USA: ACM Press, 2011, pp. 113–122. DOI: 10 . 1145/2037373 . 2037393 (cit. on p. 130).
- [115] Anuruddha Hettiarachchi and Daniel Wigdor. “Annexing Reality: Enabling Opportunistic Use of Everyday Objects As Tangible Proxies in Augmented Reality.” In: *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems - CHI '16*. New York, New York, USA: ACM Press, 2016, pp. 1957–1967. DOI: 10 . 1145/2858036 . 2858134 (cit. on p. 46).
- [116] Otmar Hilliges, Shahram Izadi, Andrew D. Wilson, Steve Hodges, Armando Garcia-Mendoza, and Andreas Butz. “Interactions in the Air: Adding Further Depth to Interactive Tabletops.” In: *Proceedings of the 22nd annual ACM symposium on User interface software and technology - UIST '09*. New York, New York, USA: ACM Press, 2009, pp. 139–148. DOI: 10 . 1145/1622176 . 1622203 (cit. on p. 45).
- [117] Ken Hinckley, Randy Pausch, John C. Goble, and Neal F. Kassell. “Passive Real-world Interface Props for Neurosurgical Visualization.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '94*. Vol. 30. New York, New York, USA: ACM Press, 1994, pp. 452–458. DOI: 10 . 1145/191666 . 191821 (cit. on p. 46).

- [118] Ken Hinckley, Jeff Pierce, Eric Horvitz, and Mike Sinclair. “Foreground and Background Interaction with Sensor-Enhanced Mobile Devices.” In: *ACM Transactions on Computer-Human Interaction* 12.1 (Mar. 2005), pp. 31–52. DOI: 10 . 1145/1057237 . 1057240 (cit. on p. 17).
- [119] Ken Hinckley and Mike Sinclair. “Touch-Sensing Input Devices.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '99*. May. New York, New York, USA: ACM Press, 1999, pp. 223–230. DOI: 10 . 1145/302979 . 303045 (cit. on p. 41).
- [120] Alexis Hiniker, Sarita Y. Schoenebeck, and Julie A. Kientz. “Not at the Dinner Table: Parents’ and Children’s Perspectives on Family Technology Rules.” In: *Proceedings of the 19th ACM Conference on Computer-Supported Cooperative Work & Social Computing - CSCW '16*. New York, New York, USA: ACM Press, 2016, pp. 1374–1387. DOI: 10 . 1145/2818048 . 2819940 (cit. on p. 21).
- [121] Alexander Hoffmann, Daniel Spelmezan, and Jan Borchers. “TypeRight: A Keyboard with Tactile Error Prevention.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '09*. New York, New York, USA: ACM Press, 2009, pp. 2265–2268. DOI: 10 . 1145/1518701 . 1519048 (cit. on pp. 116, 127).
- [122] Franziska Hoheisel. “Kompressions-Feedback für Pervasive Games.” Bachelor Thesis. Leibniz Universität Hannover, 2015 (cit. on pp. ix, 231).
- [123] Christian Holz and Patrick Baudisch. “Fiberio: A Touchscreen That Senses Fingerprints.” In: *Proceedings of the 26th annual ACM symposium on User interface software and technology - UIST '13*. New York, New York, USA: ACM Press, 2013, pp. 41–50. DOI: 10 . 1145/2501988 . 2502021 (cit. on p. 88).
- [124] Christian Holz and Patrick Baudisch. “The Generalized Perceived Input Point Model and How to Double Touch Accuracy by Extracting Fingerprints.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '10*. New York, New York, USA: ACM Press, 2010, pp. 581–590. DOI: 10 . 1145/1753326 . 1753413 (cit. on pp. 74, 79, 115).
- [125] Jonggi Hong, Seongkook Heo, Poika Isokoski, and Geehyuk Lee. “SplitBoard: A Simple Split Soft Keyboard for Wristwatch-sized Touch Screens.” In: *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems - CHI '15*. New York, New York, USA: ACM Press, 2015, pp. 1233–1236. DOI: 10 . 1145/2702123 . 2702273 (cit. on p. 133).
- [126] Scott E. Hudson, Chris Harrison, Beverly L. Harrison, and Anthony LaMarca. “Whack Gestures: Inexact and Inattentive Interaction with Mobile Devices.” In: *Proceedings of the fourth international conference on Tangible, embedded, and embodied interaction - TEI '10*. New York, New York, USA: ACM Press, 2010, pp. 109–112. DOI: 10 . 1145/1709886 . 1709906 (cit. on p. 90).

- [127] Clark L. Hull. *Principles of Behavior*. New York, NY, USA: Appleton-Century-Crofts, 1943, p. 422 (cit. on p. 23).
- [128] Sungjae Hwang, Myungwook Ahn, and Kwang-yun Wohn. “MagGetz: Customizable Passive Tangible Controllers on and Around Conventional Mobile Devices.” In: *Proceedings of the 26th annual ACM symposium on User interface software and technology - UIST '13*. New York, New York, USA: ACM Press, 2013, pp. 411–416. DOI: 10.1145/2501988.2501991 (cit. on p. 43).
- [129] Sungjae Hwang, Andrea Bianchi, Myungwook Ahn, and Kwang-yun Wohn. “MagPen: Magnetically Driven Pen Interaction On and Around Conventional Smartphones.” In: *Proceedings of the 15th International Conference on Human-Computer Interaction with Mobile Devices and Services - MobileHCI '13*. New York, New York, USA: ACM Press, 2013, pp. 412–415. DOI: 10.1145/2493190.2493194 (cit. on p. 53).
- [130] Alexandra Ion, Edward Jay Wang, and Patrick Baudisch. “Skin Drag Displays: Dragging a Physical Tactor across the User’s Skin Produces a Stronger Tactile Stimulus than Vibrotactile.” In: *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems - CHI '15*. New York, New York, USA: ACM Press, 2015, pp. 2501–2504. DOI: 10.1145/2702123.2702459 (cit. on p. 210).
- [131] Hiroshi Ishii and Brygg Ullmer. “Tangible Bits: Towards Seamless Interfaces between People, Bits and Atoms.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '97*. March. New York, New York, USA: ACM Press, 1997, pp. 234–241. DOI: 10.1145/258549.258715 (cit. on p. 46).
- [132] Ken Iwasaki, Takashi Miyaki, and Jun Rekimoto. “Expressive Typing: A New Way to Sense Typing Pressure and Its Applications.” In: *CHI '09 Extended Abstracts on Human Factors in Computing Systems - CHI EA '09*. New York, New York, USA: ACM Press, 2009, pp. 4369–4374. DOI: 10.1145/1520340.1520668 (cit. on pp. 125, 130).
- [133] Bret Jackson, David Schroeder, and Daniel F. Keefe. “Nailing Down Multi-Touch: Anchored Above the Surface Interaction for 3D Modeling and Navigation.” In: *Proceedings of the 2012 Graphics Interface Conference - GI '12*. 2012, pp. 181–184 (cit. on p. 45).
- [134] R. J. Jagacinski and J. M. Flach. *Control Theory for Humans: Quantitative approaches to modeling performance*. New Jersey: Lawrence Erlbaum, 2003 (cit. on p. 13).
- [135] Mikkel R. Jakobsen, Yvonne Jansen, Sebastian Boring, and Kasper Hornbæk. “Should I Stay or Should I Go? Selecting Between Touch and Mid-Air Gestures for Large-Display Interaction.” In: *Proceedings of the 15th IFIP TC 13 International Conference - INTERACT '15*. Springer, 2015, pp. 455–473. DOI: 10.1007/978-3-319-22698-9_31 (cit. on p. 29).

- [136] Joris H. Janssen, Jeremy N. Bailenson, Wijnand A. Ijsselsteijn, and Joyce H.D.M. Westerink. "Intimate Heartbeats: Opportunities for Affective Communication Technology." In: *IEEE Transactions on Affective Computing* 1.2 (2010), pp. 72–80. DOI: 10.1109/T-AFFC.2010.13 (cit. on p. 255).
- [137] Joris H. Janssen, Wijnand A. Ijsselsteijn, and Joyce H.D.M. Westerink. "How Affective Technologies can Influence Intimate Interactions and Improve Social Connectedness." In: *International Journal of Human Computer Studies* 72.1 (2014), pp. 33–43. DOI: 10.1016/j.ijhcs.2013.09.007 (cit. on p. 137).
- [138] Brett Jones, Rajinder Sodhi, David Forsyth, Brian Bailey, and Giuliano Maciocci. "Around Device Interaction for Multiscale Navigation." In: *Proceedings of the 14th International Conference on Human-Computer Interaction with Mobile Devices and Services - MobileHCI '12*. New York, New York, USA: ACM Press, 2012, pp. 83–92. DOI: 10.1145/2371574.2371589 (cit. on p. 46).
- [139] Staas de Jong, Dünya Kirkali, Hanna Schraffenberger, Jeroen Jillissen, Alwin de Rooij, and Arnout Terpstra. "One-Press Control: A Tactile Input Method for Pressure-Sensitive Computer Keyboards." In: *CHI '10 Extended Abstracts on Human Factors in Computing Systems - CHI EA '10*. New York, New York, USA: ACM Press, 2010, pp. 4261–4266. DOI: 10.1145/1753846.1754136 (cit. on p. 116).
- [140] Wendy Ju, Brian A. Lee, and Scott R. Klemmer. "Range: Exploring Implicit Interaction through Electronic Whiteboard Design." In: *Proceedings of the ACM 2008 conference on Computer supported cooperative work - CSCW '08*. New York, New York, USA: ACM Press, 2008, pp. 17–26. DOI: 10.1145/1460563.1460569 (cit. on p. 18).
- [141] Shaun K. Kane, Daniel Avrahami, Jacob O. Wobbrock, Beverly Harrison, Adam D. Rea, Matthai Philipose, and Anthony LaMarca. "Bonfire: A Nomadic System for Hybrid Laptop-Tabletop Interaction." In: *Proceedings of the 22nd annual ACM symposium on User interface software and technology - UIST '09*. New York, New York, USA: ACM Press, 2009, pp. 129–138. DOI: 10.1145/1622176.1622202 (cit. on p. 46).
- [142] Shaun K. Kane, Jacob O. Wobbrock, and Ian E. Smith. "Getting off the treadmill." In: *Proceedings of the 10th International Conference on Human computer interaction with mobile devices and services - MobileHCI '08*. New York, New York, USA: ACM Press, 2008, pp. 109–118. DOI: 10.1145/1409240.1409253 (cit. on p. 20).
- [143] Andreas Karrenbauer and Antti Oulasvirta. "Improvements to Keyboard Optimization with Integer Programming." In: *Proceedings of the 27th annual ACM symposium on User interface software and technology - UIST '14*. New York, New York, USA: ACM Press, 2014, pp. 621–626. DOI: 10.1145/2642918.2647382 (cit. on p. 176).
- [144] Raghavendra S. Kattinakere, Tovi Grossman, and Sriram Subramanian. "Modeling Steering within Above-the-Surface Interaction Layers." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '07*. New York, New York,

- USA: ACM Press, 2007, pp. 317–326. DOI: 10 . 1145/1240624 . 1240678 (cit. on p. 45).
- [145] Michael Kearns and Dana Ron. “Algorithmic Stability and Sanity-Check Bounds for Leave-One-Out Cross-Validation.” In: *Neural Computation* 11.6 (1999), pp. 1427–1453. DOI: 10 . 1162/089976699300016304 (cit. on p. 95).
- [146] Simeon Keates, Faustina Hwang, Patrick Langdon, P. John Clarkson, and Peter Robinson. “Cursor Measures for Motion-Impaired Computer Users.” In: *Proceedings of the Fifth International ACM Conference on Assistive Technologies - Assets '02*. New York, New York, USA: ACM Press, 2002, pp. 135–142. DOI: 10 . 1145 / 638249 . 638274 (cit. on p. 20).
- [147] Hamed Ketabdar, Mehran Roshandel, and Kamer Ali Yüksel. “Towards Using Embedded Magnetic Field Sensor for Around Mobile Device 3D Interaction.” In: *Proceedings of the 12th International Conference on Human computer interaction with mobile devices and services - MobileHCI '10*. New York, New York, USA: ACM Press, 2010, pp. 153–156. DOI: 10 . 1145/1851600 . 1851626 (cit. on p. 43).
- [148] David E. Kieras and Anthony J. Hornof. “Towards Accurate and Practical Predictive Models of Active-Vision-Based Visual Search.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '14*. New York, New York, USA: ACM Press, 2014, pp. 3875–3884. DOI: 10 . 1145/2556288 . 2557324 (cit. on p. 177).
- [149] David Kim, Otmar Hilliges, Shahram Izadi, Alex D. Butler, Jiawen Chen, Iason Oikonomidis, and Patrick Olivier. “Digits: Freehand 3D Interactions Anywhere Using a Wrist-Worn Gloveless Sensor.” In: *Proceedings of the 25th annual ACM symposium on User interface software and technology - UIST '12*. New York, New York, USA: ACM Press, 2012, pp. 167–176. DOI: 10 . 1145/2380116 . 2380139 (cit. on p. 43).
- [150] Seoktae Kim, Hyunjung Kim, Boram Lee, Tek-Jin Nam, and Woohun Lee. “Inflatable Mouse: Volume-adjustable Mouse with Air-pressure-sensitive Input and Haptic Feedback.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '08*. New York, New York, USA: ACM Press, 2008, pp. 211–224. DOI: 10 . 1145/1357054 . 1357090 (cit. on pp. 233, 248).
- [151] Yeongmi Kim, Sehun Kim, Taejin Ha, Ian Oakley, and Woontack Woo. “Air-Jet Button Effects in AR.” In: *Proceedings of the 16th International Conference on Artificial Reality and Telexistence - ICAT'06*. Vol. 4282. Lecture Notes in Computer Science. Berlin, Heidelberg, 2006, pp. 384–391. DOI: 10 . 1007/11941354_39 (cit. on pp. 91, 233).
- [152] Jesper Kjeldskov, Mikael B. Skov, Benedikte S. Als, and Rune T. Høegh. “Is It Worth the Hassle? Exploring the Added Value of Evaluating the Usability of Context-Aware Mobile Systems in the Field.” In: *Proceedings of the 6th International Symposium on MobileHCI*. 2004, pp. 61–73. DOI: 10 . 1007/978-3-540-28637-0_6 (cit. on p. 217).

- [153] Konstantin Klamka and Raimund Dachsel. “Elasticcon: Elastic Controllers for Casual Interaction.” In: *Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services - MobileHCI '15*. New York, New York, USA: ACM Press, 2015, pp. 410–419. DOI: 10 . 1145 / 2785830 . 2785849 (cit. on p. 9).
- [154] R. L. Klatzky and S. J. Lederman. “Touch.” In: *Experimental Psychology*. Ed. by I. B. Weiner. Vol. 4. New York, NY, USA: Wiley, 2002, pp. 147–176 (cit. on p. 235).
- [155] Wouter Kool, Joseph T. McGuire, Zev B. Rosen, and Matthew M. Botvinick. “Decision Making and the Avoidance of Cognitive Demand.” In: *Journal of Experimental Psychology: General* 139.4 (2010), pp. 665–682. DOI: 10 . 1037 / a0020198 (cit. on p. 23).
- [156] Konrad P. Kording and Daniel M. Wolpert. “Bayesian decision theory in sensorimotor control.” In: *TRENDS in Cognitive Sciences* 10.7 (July 2006), pp. 319–326. DOI: 10.1016/j.tics.2006.05.003 (cit. on p. 14).
- [157] Tiiu Koskela and Kaisa Väänänen-Vainio-mattila. “Evolution towards smart home environments: Empirical evaluation of three user interfaces.” In: *Personal and Ubiquitous Computing* 8.3-4 (2004), pp. 234–240. DOI: 10 . 1007 / s00779 - 004 - 0283-x (cit. on p. 87).
- [158] Sven Kratz and Michael Rohs. “A \$3 Gesture Recognizer.” In: *Proceedings of the 15th international conference on Intelligent user interfaces - IUI '10*. New York, New York, USA: ACM Press, 2010, pp. 341–344. DOI: 10 . 1145 / 1719970 . 1720026 (cit. on p. 110).
- [159] Sven Kratz and Michael Rohs. “HoverFlow: Expanding the Design Space of Around-Device Interaction.” In: *Proceedings of the 11th International Conference on Human-Computer Interaction with Mobile Devices and Services - MobileHCI '09*. New York, New York, USA: ACM Press, 2009, 4:1–4:8. DOI: 10 . 1145 / 1613858 . 1613864 (cit. on pp. 25, 44).
- [160] Markus Krause. “A Behavioral Biometrics Based Authentication Method for MOOCs that is Robust Against Imitation Attempts.” In: *Proceedings of the first ACM Conference on Learning@Scale - L@S '14*. New York, New York, USA: ACM Press, 2014, pp. 201–202. DOI: 10 . 1145 / 2556325 . 2567881 (cit. on p. 89).
- [161] Bastian Krefeld. “Variable Ausgabeauflösung für den Nahbereich Mobiler Geräte.” Bachelor Thesis. Leibniz Universität Hannover, 2015 (cit. on pp. ix, 41).
- [162] Per Ola Kristensson. “Five Challenges for Intelligent Text Entry Methods.” In: *AI Magazin* 30.4 (2009), pp. 85–94. DOI: 10 . 1609 / aimag . v30i4 . 2269 (cit. on p. 115).

- [163] Per Ola Kristensson and Keith Vertanen. “The Inviscid Text Entry Rate and its Application as a Grand Goal for Mobile Text Entry.” In: *Proceedings of the 16th International Conference on Human-Computer Interaction with Mobile Devices and Services - MobileHCI '14*. New York, New York, USA: ACM Press, 2014, pp. 335–338. DOI: 10.1145/2628363.2628405 (cit. on p. 136).
- [164] Per-Ola Kristensson and Shumin Zhai. “Relaxing Stylus Typing Precision by Geometric Pattern Matching.” In: *Proceedings of the 10th international conference on Intelligent user interfaces - IUI '05*. New York, New York, USA: ACM Press, 2005, pp. 151–158. DOI: 10.1145/1040830.1040867 (cit. on p. 115).
- [165] Per-Ola Kristensson and Shumin Zhai. “SHARK²: A Large Vocabulary Shorthand Writing System for Pen-Based Computers.” In: *Proceedings of the 17th annual ACM symposium on User interface software and technology - UIST '04*. New York, New York, USA: ACM Press, 2004, pp. 43–52. DOI: 10.1145/1029632.1029640 (cit. on p. 115).
- [166] Henry Kučera and W. Nelson Francis. *Computational Analysis of Present-Day American English*. Providence, RI, USA: Brown University Press, 1967 (cit. on p. 147).
- [167] Jussi Kuittinen, Annakaisa Kultima, Johannes Niemelä, and Janne Paavilainen. “Casual Games Discussion.” In: *Proceedings of the 2007 conference on Future Play - Future Play '07*. New York, New York, USA: ACM Press, 2007, pp. 105–112. DOI: 10.1145/1328202.1328221 (cit. on p. 9).
- [168] Kostadin Kushlev, Jason Proulx, and Elizabeth W. Dunn. ““Silence Your Phones”: Smartphone Notifications Increase Inattention and Hyperactivity Symptoms.” In: *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems - CHI '16*. New York, New York, USA: ACM Press, 2016, pp. 1011–1020. DOI: 10.1145/2858036.2858359 (cit. on p. 208).
- [169] Gierad Laput, Robert Xiao, Xiang ‘Anthony’ Chen, Scott E. Hudson, and Chris Harrison. “Skin Buttons: Cheap, Small, Low-Powered and Clickable Fixed-Icon Laser Projectors.” In: *Proceedings of the 27th annual ACM symposium on User interface software and technology - UIST '14*. New York, New York, USA: ACM Press, 2014, pp. 389–394. DOI: 10.1145/2642918.2647356 (cit. on pp. 213, 230).
- [170] Mathieu Le Goc, Stuart Taylor, Shahram Izadi, and Cem Keskin. “A Low-cost Transparent Electric Field Sensor for 3D Interaction on Mobile Devices.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '14*. New York, New York, USA: ACM Press, 2014, pp. 3167–3170. DOI: 10.1145/2556288.2557331 (cit. on p. 44).
- [171] Lisa Lebduska. “Emoji, Emoji, What for Art Thou?” In: *Harlot 12* (2015) (cit. on p. 140).

- [172] Seungyon "Claire" Lee and Thad Starner. "BuzzWear: Alert Perception in Wearable Tactile Displays on the Wrist." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '10*. New York, New York, USA: ACM Press, 2010, pp. 433–442. DOI: 10.1145/1753326.1753392 (cit. on p. 209).
- [173] John Leggett, Glen Williams, Mark Usnick, and Mike Longnecker. "Dynamic identity verification via keystroke characteristics." In: *International Journal of Man-Machine Studies* 35.6 (1991), pp. 859–870. DOI: 10.1016/S0020-7373(05)80165-8 (cit. on p. 89).
- [174] Luis A. Leiva, Alireza Sahami, Alejandro Catala, Niels Henze, and Albrecht Schmidt. "Text Entry on Tiny QWERTY Soft Keyboards." In: *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems - CHI '15*. New York, New York, USA: ACM Press, 2015, pp. 669–678. DOI: 10.1145/2702123.2702388 (cit. on p. 164).
- [175] Christoph Manuel Lenz. "Eye Tracking für die beiläufige Interaktion." Bachelor Thesis. Leibniz Universität Hannover, 2015 (cit. on p. ix).
- [176] H. Levitt. "Transformed Up-Down Methods in Psychoacoustics." In: *The Journal of the Acoustical Society of America* 49.2B (1971), pp. 467–477. DOI: 10.1121/1.1912375 (cit. on p. 239).
- [177] Frank Chun Yat Li, David Dearman, and Khai N. Truong. "Virtual Shelves: Interactions with Orientation Aware Devices." In: *Proceedings of the 22nd annual ACM symposium on User interface software and technology - UIST '09*. New York, New York, USA: ACM Press, 2009, pp. 125–128. DOI: 10.1145/1622176.1622200 (cit. on pp. 54, 81).
- [178] Ying Liu and Qiqun Wang. "Chinese Pinyin Phrasal Input on Mobile Phone: Usability and Developing Trends." In: *Proceedings of the 4th international conference on mobile technology, applications, and systems and the 1st international symposium on Computer human interaction in mobile technology - Mobility '07*. Vol. 07. 11. New York, New York, USA: ACM Press, 2007, pp. 540–546. DOI: 10.1145/1378063.1378151 (cit. on p. 157).
- [179] Daniel P. Lopresti and Andrew Tomkins. "Approximate Matching of Hand-Drawn Pictograms." In: *Proceedings of the Third International Workshop on Frontiers in Handwriting Recognition*. 1993, pp. 102–111 (cit. on p. 160).
- [180] Andrés Lucero, Jussi Holopainen, and Tero Jokela. "Pass-Them-Around: Collaborative Use of Mobile Phones for Photo Sharing." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '11*. New York, New York, USA: ACM Press, 2011, pp. 1787–1796. DOI: 10.1145/1978942.1979201 (cit. on p. 59).

- [181] Andrés Lucero and Akos Vetek. “NotifEye: Using Interactive Glasses to Deal with Notifications While Walking in Public.” In: *Proceedings of the 11th Conference on Advances in Computer Entertainment Technology - ACE’14*. 2014, 17:1–17:10. DOI: 10.1145/2663806.2663824 (cit. on p. 204).
- [182] Marco Lui and Timothy Baldwin. “langid. py: An off-the-shelf language identification tool.” In: *Proceedings of the ACL 2012 System Demonstrations*. Association for Computational Linguistics. 2012, pp. 25–30 (cit. on p. 117).
- [183] Kent Lyons. “Visual Parameters Impacting Reaction Times on Smartwatches.” In: *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services - MobileHCI ’16*. New York, New York, USA: ACM Press, 2016, pp. 190–194. DOI: 10.1145/2935334.2935344 (cit. on pp. 228, 229).
- [184] Kent Lyons. “What can a Dumb Watch Teach a Smartwatch? Informing the Design of Smartwatched.” In: *Proceedings of the 2015 ACM International Symposium on Wearable Computers - ISWC ’15*. New York, New York, USA: ACM Press, 2015, pp. 3–10. DOI: 10.1145/2802083.2802084 (cit. on p. 208).
- [185] Laurens van der Maaten and Geoffrey Hinton. “Visualizing Data using t-SNE.” In: *Journal of Machine Learning Research* 9 (2008), pp. 2579–2605. DOI: 10.1007/s10479-011-0841-3 (cit. on p. 182).
- [186] Carsten Magerkurth, Adrian David Cheok, Regan L. Mandryk, and Trond Nilsen. “Pervasive Games: Bringing Computer Entertainment Back to the Real World.” In: *Computers in Entertainment* 3.3 (2005), pp. 1–19. DOI: 10.1145/1077246.1077257 (cit. on p. 258).
- [187] Neil J. Mansfield. *Human response to vibration*. Boca Raton, FL, USA: CRC Press, 2005. ISBN: 0-415-28239-X (cit. on p. 232).
- [188] Nicolai Marquardt, Ricardo Jota, Saul Greenberg, and Joaquim A. Jorge. “The Continuous Interaction Space: Interaction Techniques Unifying Touch and Gesture On and Above a Digital Surface.” In: *Proceedings of the 13th IFIP TC13 Conference on Human Computer Interaction - INTERACT ’11*. 2011, pp. 461–476 (cit. on p. 19).
- [189] Ramses V. Martinez, Carina R. Fish, Xin Chen, and George M. Whitesides. “Elastomeric Origami: Programmable Paper-Elastomer Composites as Pneumatic Actuators.” In: *Advanced Functional Materials* 22.7 (2012), pp. 1376–1384. DOI: 10.1002/adfm.201102978 (cit. on pp. 233, 235).
- [190] Akhil Mathur, Nicholas D. Lane, and Fahim Kawsar. “Engagement-Aware Computing: Modelling User Engagement from Mobile Contexts.” In: *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing - UbiComp ’16*. New York, New York, USA: ACM Press, 2016, pp. 622–633. DOI: 10.1145/2971648.2971760 (cit. on pp. 12, 13).

- [191] David C. McCallum, Edward Mak, Pourang Irani, and Sriram Subramanian. “PressureText: Pressure Input for Mobile Phone Text Entry.” In: *CHI '09 Extended Abstracts on Human Factors in Computing Systems - CHI EA '09*. New York, New York, USA: ACM Press, 2009, pp. 4519–4524. DOI: 10.1145/1520340.1520693 (cit. on p. 116).
- [192] Brandon T. McDaniel and Sarah M. Coyne. ““Technoference”: The interference of technology in couple relationships and implications for women’s personal and relational well-being.” In: *Psychology of Popular Media Culture* 5.1 (2016), pp. 85–98. DOI: 10.1037/ppm0000065 (cit. on p. 21).
- [193] Gregor McEwan and Saul Greenberg. “Supporting Social Worlds with the Community Bar.” In: *Proceedings of the 2005 international ACM SIGGROUP conference on Supporting group work - GROUP '05*. New York, New York, USA: ACM Press, 2005, pp. 21–30. DOI: 10.1145/1099203.1099207 (cit. on p. 9).
- [194] Marshall McLuhan. *Understanding Media: The Extensions of Man*. New York, NY, USA: McGraw-Hill, 1964. ISBN: 81-14-67535-7 (cit. on p. 132).
- [195] Justyna Medrek. “Indirekte Beleuchtung für Smartwatch-Benachrichtigungen.” Bachelor Thesis. Leibniz Universität Hannover, 2015 (cit. on pp. ix, 207).
- [196] Sarah Mennicken, James Scott, A. J. Bernheim Brush, and Asta Roseway. “Finding Roles for Interactive Furniture in Homes with EmotoCouch.” In: *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing Adjunct Publication - UbiComp '14 Adjunct*. New York, New York, USA: ACM Press, 2014, pp. 923–930. DOI: 10.1145/2638728.2641547 (cit. on p. 103).
- [197] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. “Distributed Representations of Words and Phrases and their Compositionality.” In: *Proceedings of NIPS*. 2013. arXiv: 1310.4546 (cit. on pp. 180, 182).
- [198] Hannah Miller, Jacob Thebault-Spieker, Shuo Chang, Isaac Johnson, Loren Terveen, and Brent Hecht. ““Blissfully happy” or “ready to fight”: Varying Interpretations of Emoji.” In: *International AAAI Conference on Web and Social Media*. 2016, pp. 259–268 (cit. on p. 140).
- [199] Pranav Mistry and Pattie Maes. “Mouseless: A Computer Mouse as Small as Invisible.” In: *CHI '11 Extended Abstracts on Human Factors in Computing Systems - CHI EA '11*. New York, New York, USA: ACM Press, 2011, pp. 1099–1104. DOI: 10.1145/1979742.1979715 (cit. on pp. 44, 53).
- [200] Takashi Mitsuda. “Pseudo Force Display that Applies Pressure to the Forearms.” In: *Presence: Teleoperators and Virtual Environments* 22.3 (2013), pp. 191–201. DOI: 10.1162/PRES_a_00150 (cit. on pp. 233, 236, 243).

- [201] Philipp Mock, Jörg Edelmann, Andreas Schilling, and Wolfgang Rosenstiel. “User Identification Using Raw Sensor Data From Typing on Interactive Displays.” In: *Proceedings of the 19th international conference on Intelligent User Interfaces - IUI '14*. New York, New York, USA: ACM Press, 2014, pp. 67–72. DOI: 10.1145/2557500.2557503 (cit. on p. 89).
- [202] Fabian Monroe and Aviel D. Rubin. “Keystroke dynamics as a biometric for authentication.” In: *Future Generation Computer Systems* 16.4 (2000), pp. 351–359. DOI: 10.1016/S0167-739X(99)00059-X (cit. on p. 89).
- [203] Miyuki Morioka and Michael J. Griffin. “Magnitude-dependence of equivalent comfort contours for fore-and-aft, lateral and vertical hand-transmitted vibration.” In: *Journal of Sound and Vibration* 295.3-5 (2006), pp. 633–648. DOI: 10.1016/j.jsv.2006.01.029 (cit. on p. 232).
- [204] Meredith Ringel Morris, Andreea Danielescu, Steven Drucker, Danyel Fisher, Bongshin Lee, m. c. Schraefel, and Jacob O. Wobbrock. “Reducing Legacy Bias in Gesture Elicitation Studies.” In: *interactions* 21.3 (2014), pp. 40–45. DOI: 10.1145/2591689 (cit. on pp. 61, 64).
- [205] Florian 'Floyd' Mueller and Matthew Muirhead. “Jogging with a Quadcopter.” In: *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems - CHI '15*. New York, New York, USA: ACM Press, 2015, pp. 2023–2032. DOI: 10.1145/2702123.2702472 (cit. on p. 257).
- [206] Florian Floyd Mueller, Frank Vetere, Martin R. Gibbs, Jesper Kjeldskov, Sonja Pedell, and Steve Howard. “Hug Over a Distance.” In: *CHI '05 Extended Abstracts on Human Factors in Computing Systems - CHI EA '05*. 2005, pp. 1673–1676. DOI: 10.1145/1056808.1056994 (cit. on p. 234).
- [207] Bilge Mutlu, Andreas Krause, Jodi Forlizzi, Carlos Guestrin, and Jessica Hodgins. “Robust, low-cost, non-intrusive sensing and recognition of seated postures.” In: *Proceedings of the 20th annual ACM symposium on User interface software and technology - UIST '07*. Vol. 4. 1. New York, New York, USA: ACM Press, 2007, pp. 149–1. DOI: 10.1145/1294211.1294237 (cit. on pp. 89, 102).
- [208] Suranga Nanayakkara, Roy Shilkrot, Kian Peen Yeo, and Pattie Maes. “EyeRing: A Finger-Worn Input Device for Seamless Interactions with our Surroundings.” In: *Proceedings of the 4th Augmented Human International Conference on - AH '13*. New York, New York, USA: ACM Press, 2013, pp. 13–20. DOI: 10.1145/2459236.2459240 (cit. on p. 43).
- [209] Alexander Ng, Stephen A. Brewster, and John Williamson. “The Impact of Encumbrance on Mobile Interactions.” In: *Proceedings of the 14th IFIP TC 13 International Conference on Human-Computer Interaction - INTERACT '13*. 2013, pp. 92–109. DOI: 10.1007/978-3-642-40477-1_6 (cit. on p. 20).

- [210] Alexander Ng, John Williamson, and Stephen Brewster. “The Effects of Encumbrance and Mobility on Touch-Based Gesture Interactions for Mobile Phones.” In: *Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services - MobileHCI '15*. New York, New York, USA: ACM Press, 2015, pp. 536–546. DOI: 10.1145/2785830.2785853 (cit. on p. 20).
- [211] Marketta Niemelä and Jukka Saarinen. “Visual Search for Grouped versus Ungrouped Icons in a Computer Interface.” In: *Human Factors: The Journal of the Human Factors and Ergonomics Society* 42.4 (2000), pp. 630–635. DOI: 10.1518/001872000779697999 (cit. on p. 177).
- [212] Ryuma Niiyama, Xu Sun, Lining Yao, Hiroshi Ishii, Daniela Rus, and Sangbae Kim. “Sticky Actuator: Free-Form Planar Actuators for Animated Objects.” In: *Proceedings of the Ninth International Conference on Tangible, Embedded, and Embodied Interaction - TEI '14*. New York, New York, USA: ACM Press, 2015, pp. 77–84. DOI: 10.1145/2677199.2680600 (cit. on p. 235).
- [213] Petra Kralj Novak, Jasmina Smailovic, Borut Sluban, and Igor Mozetic. “Sentiment of Emojis.” In: *PLoS ONE* 10.12 (2015), pp. 1–19. DOI: 10.1371/journal.pone.0144296. arXiv: 1509.07761 (cit. on p. 148).
- [214] Heather L. O'Brien and Elaine G. Toms. “The Development and Evaluation of a Survey to Measure User Engagement.” In: *Journal of the American Society for Information Science and Technology* 61.1 (2010), pp. 50–69. DOI: 10.1002/asi.21229 (cit. on pp. 10, 11).
- [215] Heather L. O'Brien and Elaine G. Toms. “What is User Engagement? A Conceptual Framework for Defining User Engagement with Technology.” In: *Journal of the American Society for Information Science and Technology* 59.6 (2008), pp. 938–955. DOI: 10.1002/asi.20801 (cit. on pp. 10, 11).
- [216] S. A. M. Offermans, H. A. van Essen, and J. H. Eggen. “User interaction with everyday lighting systems.” In: *Personal and Ubiquitous Computing* 18.8 (Dec. 2014), pp. 2035–2055. DOI: 10.1007/s00779-014-0759-2 (cit. on p. 104).
- [217] Akio Ogihara, Hiroyuki Matsumura, and Akira Shiozaki. “Biometric Verification Using Keystroke Motion and Key Press Timing for ATM User Authentication.” In: *2006 International Symposium on Intelligent Signal Processing and Communications*. IEEE, 2006, pp. 223–226. DOI: 10.1109/ISPACS.2006.364872 (cit. on p. 89).
- [218] Stephen Oney, Chris Harrison, Amy Ogan, and Jason Wiese. “ZoomBoard: A Diminutive QWERTY Soft Keyboard Using Iterative Zooming for Ultra-Small Devices.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '13*. New York, New York, USA: ACM Press, 2013, p. 2799. DOI: 10.1145/2470654.2481387 (cit. on pp. 133, 164, 165).

- [219] Antti Oulasvirta and Joanna Bergstrom-Lehtovirta. “Ease of Juggling: Studying the Effects of Manual Multitasking.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '11*. New York, New York, USA: ACM Press, 2011, pp. 3103–3112. DOI: 10.1145/1978942.1979402 (cit. on p. 20).
- [220] Antti Oulasvirta, Anna Reichel, Wenbin Li, Yan Zhang, Myroslav Bachynskzi, Keith Vertanen, and Per Ola Kristensson. “Improving Two-Thumb Text Entry on Touchscreen Devices.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '13*. 2013, pp. 2765–2774 (cit. on p. 117).
- [221] Minna Pakanen, Ashley Colley, Jonna Häkkinä, Johan Kildal, and Vuokko Lantz. “Squeezy Bracelet: Designing a Wearable Communication Device for Tactile Interaction.” In: *Proceedings of the 8th Nordic Conference on Human-Computer Interaction Fun, Fast, Foundational - NordiCHI '14*. New York, New York, USA: ACM Press, 2014, pp. 305–314. DOI: 10.1145/2639189.2639238 (cit. on p. 248).
- [222] Dimitris Papanikolaou, A.J. Bernheim Brush, and Asta Roseway. “BodyPods: Designing Posture Sensing Chairs for Capturing and Sharing Implicit Interactions.” In: *Proceedings of the Ninth International Conference on Tangible, Embedded, and Embodied Interaction - TEI '15*. New York, New York, USA: ACM Press, 2015, pp. 375–382. DOI: 10.1145/2677199.2680591 (cit. on p. 89).
- [223] Joseph A. Paradiso and Mark Feldmeier. “A Compact, Wireless, Self-Powered Push-button Controller.” In: *Proceedings of the 3rd International Conference on Ubiquitous Computing - Ubicomp '01*. 2001, pp. 399–304 (cit. on p. 99).
- [224] Jerome Pasquero, Scott J. Stobbe, and Noel Stonehouse. “A Haptic Wristwatch for Eyes-Free Interactions.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '11*. New York, New York, USA: ACM Press, 2011, pp. 3257–3266. DOI: 10.1145/1978942.1979425 (cit. on pp. 65, 209).
- [225] Patrick E. Patterson and Judd A. Katz. “Design and Evaluation of a Sensory Feedback System that Provides Grasping Pressure in a Myoelectric Hand.” In: *Journal of Rehabilitation Research and Development* 29.1 (1992), pp. 1–8. DOI: 10.1682/JRRD.1992.01.0001 (cit. on pp. 233, 236).
- [226] A. Peacock and M. Wilkerson. “Typing patterns: a key to user identification.” In: *IEEE Security & Privacy Magazine* 2.5 (2004), pp. 40–47. DOI: 10.1109/MSP.2004.89 (cit. on p. 89).
- [227] Jennifer Pearson, Simon Robinson, and Matt Jones. “It’s About Time: Smartwatches as Public Displays.” In: *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems - CHI '15*. New York, New York, USA: ACM Press, 2015, pp. 1257–1266. DOI: 10.1145/2702123.2702247 (cit. on pp. 65, 230).

- [228] Esben Warming Pedersen and Kasper Hornbæk. “mixiTUI: A Tangible Sequencer for Electronic Live Performances.” In: *Proceedings of the 3rd International Conference on Tangible and Embedded Interaction - TEI '09*. New York, New York, USA: ACM Press, 2009, pp. 223–230. DOI: 10.1145/1517664.1517713 (cit. on p. 47).
- [229] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesna. “Scikit-learn: Machine Learning in Python.” In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830 (cit. on p. 182).
- [230] Laura Perovich, Philippa Mothersill, and Jennifer Broutin Farah. “Awakened Apparel: Embedded Soft Actuators for Expressive Fashion and Functional Garments.” In: *Proceedings of the 8th International Conference on Tangible, Embedded and Embodied Interaction - TEI '14*. New York, New York, USA: ACM Press, 2013, pp. 77–80. DOI: 10.1145/2540930.2540958 (cit. on p. 260).
- [231] Simon T. Perrault, Eric Lecolinet, James Eagan, and Yves Guiard. “WatchIt: Simple Gestures and Eyes-free Interaction for Wristwatches and Bracelets.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '13*. New York, New York, USA: ACM Press, 2013. DOI: 10.1145/2470654.2466192 (cit. on p. 105).
- [232] Martin Pielot and Rodrigo de Oliveira. “Peripheral Vibro-Tactile Displays.” In: *Proceedings of the 15th International Conference on Human-Computer Interaction with Mobile Devices and Services - MobileHCI '13*. New York, New York, USA: ACM Press, 2013, pp. 1–10. DOI: 10.1145/2493190.2493197 (cit. on p. 204).
- [233] Stefania Pizza, Barry Brown, Donald McMillan, and Airi Lampinen. “Smartwatch in vivo.” In: *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems - CHI '16*. New York, New York, USA: ACM Press, 2016, pp. 5456–5469. DOI: 10.1145/2858036.2858522 (cit. on p. 208).
- [234] Henning Pohl. “Casual Interaction: Moving Between Peripheral and High Engagement Interactions.” In: *Peripheral Interaction: Challenges and Opportunities for HCI in the Periphery of Attention*. Ed. by Saskia Bakker, Doris Hausen, and Ted Selker. Berlin, Heidelberg: Springer, 2016, pp. 117–135. DOI: 10.1007/978-3-319-29523-7_6 (cit. on pp. viii, 5).
- [235] Henning Pohl. “Casual Interaction: Scaling Interaction for Multiple Levels of Engagement.” In: *CHI '15 Extended Abstracts on Human Factors in Computing Systems - CHI EA '15*. New York, New York, USA: ACM Press, 2015, pp. 223–226. DOI: 10.1145/2702613.2702625 (cit. on p. vii).

- [236] Henning Pohl, Dennis Becke, Eugen Wagner, Maximilian Schrapel, and Michael Rohs. “Wrist Compression Feedback by Pneumatic Actuation.” In: *CHI '15 Extended Abstracts on Human Factors in Computing Systems on - CHI EA '15*. 2015. DOI: 10.1145/2702613.2725427 (cit. on pp. vii, 231, 255).
- [237] Henning Pohl, Peter Brandes, Hung Ngo Quang, and Michael Rohs. “Squeezeback: Pneumatic Compression for Notifications.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '17*. New York, New York, USA: ACM Press, 2017. DOI: 10.1145/3025453.3025526 (cit. on pp. viii, 231).
- [238] Henning Pohl, Christian Domin, and Michael Rohs. “Beyond Just Text: Semantic Emoji Similarity Modeling to Support Expressive Communication 🧑🧑📱😊.” In: *ACM Transactions on Computer-Human Interaction* 24.1 (2017). DOI: 10.1145/3039685 (cit. on pp. viii, 131).
- [239] Henning Pohl, Markus Hettig, Oliver Karras, Hatice Öztürk, and Michael Rohs. “CapCouch: Home Control with a Posture-Sensing Couch.” In: *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2015 ACM International Symposium on Wearable Computers - UbiComp '15*. New York, New York, USA: ACM Press, 2015, pp. 229–232. DOI: 10.1145/2800835.2800932 (cit. on p. 86).
- [240] Henning Pohl, Franziska Hoheisel, and Michael Rohs. “Inhibiting Freedom of Movement with Compression Feedback.” In: *CHI '17 Extended Abstracts on Human Factors in Computing Systems - CHI EA '17*. New York, New York, USA: ACM Press, 2017. DOI: 10.1145/3027063.3053081 (cit. on pp. viii, 231).
- [241] Henning Pohl, Markus Krause, and Michael Rohs. “One-Button Recognizer: Exploiting Button Pressing Behavior for User Differentiation.” In: *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing - UbiComp '15*. New York, New York, USA: ACM Press, 2015, pp. 403–407. DOI: 10.1145/2750858.2804270 (cit. on pp. viii, 86).
- [242] Henning Pohl, Bastian Krefeld, and Michael Rohs. “Multi-Level Interaction with an LED-Matrix Edge Display.” In: *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services adjunct - MobileHCI '16 Adjunct*. 2016. DOI: 10.1145/2957265.2961855 (cit. on pp. viii, 41).
- [243] Henning Pohl, Justyna Medrek, and Michael Rohs. “ScatterWatch: Subtle Notifications via Indirect Illumination Scattered in the Skin.” In: *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services companion - MobileHCI '16*. 2016. DOI: 10.1145/2935334.2935351 (cit. on pp. viii, 207).
- [244] Henning Pohl and Roderick Murray-Smith. “Focused and Casual Interactions: Allowing Users to Vary Their Level of Engagement.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '13*. New York, New York,

- USA: ACM Press, 2013, pp. 2223–2232. DOI: 10 . 1145/2470654 . 2481307 (cit. on pp. vii, 5, 53).
- [245] Henning Pohl and Michael Rohs. “Around-Device Devices: My Coffee Mug is a Volume Dial.” In: *Proceedings of the 16th International Conference on Human-Computer Interaction with Mobile Devices and Services - MobileHCI '14*. 2014. DOI: 10 . 1145/2628363 . 2628401 (cit. on pp. vii, 41).
- [246] Henning Pohl, Michael Rohs, and Roderick Murray-Smith. “Casual Interaction: Scaling Fidelity for Low-Engagement Interactions.” In: *Workshop on Peripheral Interaction: Shaping the Research and Design Space at CHI 2014*. 2014 (cit. on p. vii).
- [247] Henning Pohl, Dennis Stanke, and Michael Rohs. “EmojiZoom: Emoji Entry via Large Overview Maps 🗺️🔍.” In: *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services companion - MobileHCI '16*. 2016. DOI: 10 . 1145/2935334 . 2935382 (cit. on pp. viii, 131).
- [248] Qian Qin, Michael Rohs, and Sven Kratz. “Dynamic Ambient Lighting for Mobile Devices.” In: *Proceedings of the 24th annual ACM symposium adjunct on User interface software and technology - UIST '11 Adjunct*. New York, New York, USA: ACM Press, 2011, pp. 51–52. DOI: 10 . 1145/2046396 . 2046418 (cit. on pp. 66, 229).
- [249] Hung Ngo Quang. “Compression Feedback for Notifications.” Master Thesis. Leibniz Universität Hannover, 2016 (cit. on pp. ix, 231).
- [250] Jeffrey M. Quinn and Tuan Q. Tran. “Attractive Phones Don’t Have to Work Better.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '10*. New York, New York, USA: ACM Press, 2010, pp. 353–362. DOI: 10 . 1145/1753326 . 1753380 (cit. on p. 12).
- [251] Raf Ramakers, Davy Vanacken, Kris Luyten, Karin Coninx, and Johannes Schöning. “Carpus: A Non-Intrusive User Identification Technique for Interactive Surfaces.” In: *Proceedings of the 25th annual ACM symposium on User interface software and technology - UIST '12*. New York, New York, USA: ACM Press, 2012, pp. 35–44. DOI: 10 . 1145/2380116 . 2380123 (cit. on p. 89).
- [252] Gonzalo Ramos, Matthew Boulos, and Ravin Balakrishnan. “Pressure Widgets.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '04*. New York, New York, USA: ACM Press, 2004, pp. 487–494. DOI: 10 . 1145/985692 . 985754 (cit. on p. 115).
- [253] Parisa Rashidi and Diane J. Cook. “Keeping the resident in the loop: Adapting the smart home to the user.” In: *IEEE Transactions on Systems, Man, and Cybernetics Part A: Systems and Humans* 39.5 (2009), pp. 949–959. DOI: 10 . 1109/TSMCA . 2009 . 2025137 (cit. on p. 88).
- [254] Radim Řehůřek and Petr Sojka. “Software Framework for Topic Modelling with Large Corpora.” In: *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. Valetta, Malta: ELRA, 2010, pp. 45–50 (cit. on p. 180).

- [255] Stephan Richter, Christian Holz, and Patrick Baudisch. “Bootstrapper: Recognizing Tabletop Users by Their Shoes.” In: *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems - CHI '12*. New York, New York, USA: ACM Press, 2012, pp. 1249–1252. DOI: 10.1145/2207676.2208577 (cit. on p. 89).
- [256] Julie Rico and Stephen Brewster. “Usable Gestures for Mobile Interfaces: Evaluating Social Acceptability.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '10*. New York, New York, USA: ACM Press, 2010, pp. 887–896. DOI: 10.1145/1753326.1753458 (cit. on p. 20).
- [257] James A. Roberts and Meredith E. David. “My life has become a major distraction from my cell phone: Partner phubbing and relationship satisfaction among romantic partners.” In: *Computers in Human Behavior* 54 (2016), pp. 134–141. DOI: 10.1016/j.chb.2015.07.058 (cit. on p. 21).
- [258] Simon Robinson, Parisa Eslambolchilar, and Matt Jones. “Exploring Casual Point-and-Tilt Interactions for Mobile Geo-Blogging.” In: *Personal and Ubiquitous Computing* 14.4 (2010), pp. 363–379. DOI: 10.1007/s00779-009-0236-5 (cit. on p. 9).
- [259] Simon Rogers, John Williamson, Craig Stewart, and Roderick Murray-Smith. “FingerCloud: Uncertainty and Autonomy Handover in Capacitive Sensing.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '10*. New York, New York, USA: ACM Press, 2010, pp. 577–580. DOI: 10.1145/1753326.1753412 (cit. on pp. 25, 122).
- [260] Sven Röttering. “Casual Interaction with a Smartwatch.” Master Thesis. Leibniz Universität Hannover, 2016 (cit. on pp. ix, 5).
- [261] Anne Roudaut, Henning Pohl, and Patrick Baudisch. “Touch Input on Curved Surfaces.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '11*. New York, New York, USA: ACM Press, 2011, pp. 1011–1020. DOI: 10.1145/1978942.1979094 (cit. on p. 74).
- [262] Thijs Roumen, Simon T. Perrault, and Shengdong Zhao. “NotiRing: A Comparative Study of Notification Channels for Wearable Interactive Rings.” In: *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems - CHI '15*. New York, New York, USA: ACM Press, 2015, pp. 2497–2500. DOI: 10.1145/2702123.2702350 (cit. on p. 228).
- [263] Wenjie Ruan, Quan Z. Sheng, Lei Yang, Tao Gu, Peipei Xu, and Longfei Shangguan. “AudioGest: Enabling Fine-Grained Hand Gesture Detection by Decoding Echo Signal.” In: *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing - UbiComp '16*. New York, New York, USA: ACM Press, 2016, pp. 474–485. DOI: 10.1145/2971648.2971736 (cit. on p. 45).

- [264] Dmitry Rudchenko, Tim Paek, and Eric Badger. “Text Text Revolution: A Game That Improves Text Entry on Mobile Touchscreen Keyboards.” In: *Proceedings of the 9th International Conference on Pervasive Computing - Pervasive '11*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 206–213. DOI: 10.1007/978-3-642-21726-5_13 (cit. on p. 115).
- [265] T. Scott Saponas, Chris Harrison, and Hrvoje Benko. “PocketTouch: Through-Fabric Capacitive Touch Input.” In: *Proceedings of the 24th annual ACM symposium on User interface software and technology - UIST '11*. Figure 1. New York, New York, USA: ACM Press, 2011, pp. 303–308. DOI: 10.1145/2047196.2047235 (cit. on p. 100).
- [266] T. Scott Saponas, Desney S. Tan, Dan Morris, Ravin Balakrishnan, Jim Turner, and James a. Landay. “Enabling Always-available Input with Muscle-computer Interfaces.” In: *Proceedings of the 22nd annual ACM symposium on User interface software and technology - UIST '09*. New York, New York, USA: ACM Press, 2009, pp. 167–176. DOI: 10.1145/1622176.1622208 (cit. on p. 43).
- [267] Albrecht Schmidt. “Implicit Human Computer Interaction Through Context.” In: *Personal Technologies 4.2* (2000), pp. 191–199. DOI: 10.1007/BF01324126 (cit. on p. 18).
- [268] Dominik Schmidt, Ming Ki Chong, and Hans Gellersen. “HandsDown: Hand-Contour-Based User Identification for Interactive Surfaces.” In: *Proceedings of the 6th Nordic Conference on Human-Computer Interaction Extending Boundaries - NordiCHI '10*. New York, New York, USA: ACM Press, 2010, pp. 432–441. DOI: 10.1145/1868914.1868964 (cit. on p. 88).
- [269] Eike Clas Schulze. “Punktwolkenvisualisierung mittels Bewegungsparallaxe.” Master Thesis. Leibniz Universität Hannover, 2014 (cit. on p. ix).
- [270] Julia Schwarz, Scott Hudson, Jennifer Mankoff, and Andrew D. Wilson. “A framework for robust and flexible handling of inputs with uncertainty.” In: *Proceedings of the 23rd annual ACM symposium on User interface software and technology (UIST '10)*. ACM, 2010, pp. 47–56. DOI: 10.1145/1866029.1866039 (cit. on p. 116).
- [271] Philipp Seelig. “Attribute-Driven Soft Keyboard for Emoji Entry.” Bachelor Thesis. Leibniz Universität Hannover, 2015 (cit. on p. ix).
- [272] Claude E. Shannon. “A Mathematical Theory of Communication.” In: *Bell System Technical Journal 27.3* (1948), pp. 379–423. DOI: 10.1002/j.1538-7305.1948.tb01338.x (cit. on p. 15).
- [273] Kang Shi, Pourang Irani, Sean Gustafson, and Sriram Subramanian. “PressureFish : A Method to Improve Control of Discrete Pressure-based Input.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '08*. New York, New York, USA: ACM Press, 2008, pp. 1295–1298. DOI: 10.1145/1357054.1357256 (cit. on p. 116).

- [274] Weidong Shi and Jun Yang. “SenGuard: Passive user identification on smartphones using multiple sensors.” In: *2011 IEEE 7th International Conference on Wireless and Mobile Computing, Networking and Communications - WiMob’11*. IEEE, 2011, pp. 141–148. DOI: 10.1109/WiMOB.2011.6085412 (cit. on p. 89).
- [275] Katie A. Siek, Yvonne Rogers, and Kay H. Connelly. “Fat Finger Worries: How Older and Younger Users Physically Interact with PDAs.” In: *Proceedings of Human-Computer Interaction - INTERACT’05*. 2005, pp. 267–280. DOI: 10.1007/11555261_24 (cit. on pp. 114, 115).
- [276] Ronit Slyper and Jessica Hodgins. “Prototyping Robot Appearance, Movement, and Interactions Using Flexible 3D Printing and Air Pressure Sensors.” In: *Proceedings of the 21st IEEE International Symposium on Robot and Human Interactive Communication - RO-MAN’12*. Vol. 1. IEEE, 2012, pp. 6–11. DOI: 10.1109/ROMAN.2012.6343723 (cit. on pp. 234, 248).
- [277] Sunghyun Song, Geeyoung Noh, Junwoo Yoo, Ian Oakley, Jundong Cho, and Andrea Bianchi. “Hot & Tight: Exploring Thermo and Squeeze Cues Recognition on Wrist Wearables.” In: *Proceedings of the 2015 ACM International Symposium on Wearable Computers - ISWC ’15*. New York, New York, USA: ACM Press, 2015, pp. 39–42. DOI: 10.1145/2802083.2802092 (cit. on p. 209).
- [278] Daniel Spelmezan, Caroline Appert, Olivier Chapuis, and Emmanuel Pietriga. “Controlling Widgets With One Power-Up Button.” In: *Proceedings of the 26th annual ACM symposium on User interface software and technology - UIST ’13*. New York, New York, USA: ACM Press, 2013, pp. 71–74. DOI: 10.1145/2501988.2502025 (cit. on p. 91).
- [279] Martin Spindler, Marcel Martsch, and Raimund Dachsel. “Going Beyond the Surface: Studying Multi-Layer Interaction Above the Tabletop.” In: *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems - CHI ’12*. New York, New York, USA: ACM Press, 2012, pp. 1277–1286. DOI: 10.1145/2207676.2208583 (cit. on p. 45).
- [280] Mandayam A. Srinivasan and Jyh-shing Chen. “Human Performance in Controlling Normal Forces of Contact with Rigid Objects.” In: *Proc. Advances in Robotics, Mechatronics, and Haptic Interfaces - ASME’93*. Vol. 49. 1993, pp. 119–125 (cit. on p. 122).
- [281] Dennis Stanke. “Ein Zoombares Emoji Keyboard.” Bachelor Thesis. Leibniz Universität Hannover, 2016 (cit. on pp. ix, 131).
- [282] Christian Steins, Sean Gustafson, Christian Holz, and Patrick Baudisch. “Imaginary Devices: Gesture-Based Interaction Mimicking Traditional Input Devices.” In: *Proceedings of the 15th International Conference on Human-Computer Interaction with Mobile Devices and Services - MobileHCI ’13*. New York, New York, USA: ACM Press, 2013, pp. 123–126. DOI: 10.1145/2493190.2493208 (cit. on pp. 47, 53).

- [283] Angus Stevenson and Lindberg Christine A., eds. *New Oxford American Dictionary*. 3rd editio. Oxford: Oxford University Press, 2010. ISBN: 978-0-19-539288-3 (cit. on p. 8).
- [284] Craig Stewart, Michael Rohs, Sven Kratz, and Georg Essl. “Characteristics of Pressure-Based Input for Mobile Devices.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '10*. New York, New York, USA: ACM Press, 2010, pp. 801–810. DOI: 10.1145/1753326.1753444 (cit. on p. 115).
- [285] S. Strachan and R. Murray-Smith. “Bearing-based selection in mobile spatial interaction.” In: *Personal and Ubiquitous Computing* 13.4 (2009), pp. 265–280 (cit. on p. 14).
- [286] Sriram Subramanian, Dzimitry Aliakseyeu, and Andrés Lucero. “Multi-Layer Interaction for Digital Tables.” In: *Proceedings of the 19th annual ACM symposium on User interface software and technology - UIST '06*. New York, New York, USA: ACM Press, 2006, pp. 269–272. DOI: 10.1145/1166253.1166295 (cit. on p. 45).
- [287] Sriram Subramanian, Dzimitry Aliakseyeu, and Andrés Lucero. “Multi-layer Interaction for Digital Tables.” In: *UIST '06: Proceedings of the 19th Annual ACM Symposium on User Interface Software and Technology*. Montreux, Switzerland: ACM, 2006, pp. 269–272. DOI: 10.1145/1166253.1166295 (cit. on p. 122).
- [288] Katja Suhonen, Kaisa Väänänen-Vainio-Mattila, and Kalle Mäkelä. “User Experiences and Expectations of Vibrotactile, Thermal and Squeeze Feedback in Interpersonal Communication.” In: *Proceedings of the 26th Annual BCS Interaction Specialist Group Conference on People and Computers - BCS-HCI '12*. 2012, pp. 205–214 (cit. on p. 233).
- [289] Jared Suttles and Nancy Ide. “Distant Supervision for Emotion Classification with Discrete Binary Values.” In: *Proceedings of the 14th International Conference on Computational Linguistics and Intelligent Text Processing - CICLing '13*. Vol. 7817 LNCS. PART 2. 2013, pp. 121–136. DOI: 10.1007/978-3-642-37256-8_11 (cit. on p. 148).
- [290] Jaakko Suutala and Juha Röning. “Methods for person identification on a pressure-sensitive floor: Experiments with multiple classifiers and reject option.” In: *Information Fusion* 9.1 (2008), pp. 21–40. DOI: 10.1016/j.inffus.2006.11.003 (cit. on p. 89).
- [291] Yoshiki Takeoka, Takashi Miyaki, and Jun Rekimoto. “Z-touch: An Infrastructure for 3D Gesture Interaction in the Proximity of Tabletop Surfaces.” In: *ACM International Conference on Interactive Tabletops and Surfaces - ITS '10*. New York, New York, USA: ACM Press, 2010, pp. 91–94. DOI: 10.1145/1936652.1936668 (cit. on p. 44).

- [292] Yuzuru Tanahashi and Kwan-liu Ma. “Stock Lamp: An Engagement-Versatile Visualization Design.” In: *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems - CHI '15*. 2015, pp. 595–604. DOI: 10 . 1145/2702123 . 2702323 (cit. on p. 204).
- [293] C.C. Tappert, C.Y. Suen, and T. Wakahara. “The State of the Art in Online Handwriting Recognition.” In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 12.8 (1990), pp. 787–808. DOI: 10 . 1109/34 . 57669 (cit. on p. 160).
- [294] Camilo Tejeiro, Cara E. Stepp, Mark Malhotra, Eric Rombokas, and Yoky Matsuoka. “Comparison of Remote Pressure and Vibrotactile Feedback for Prosthetic Hand Control.” In: *Proceedings of the IEEE RAS and EMBS International Conference on Biomedical Robotics and Biomechatronics*. 2012, pp. 521–525. DOI: 10 . 1109/BioRob . 2012 . 6290268 (cit. on pp. 233, 236).
- [295] Kazuhiro Terajima, Takashi Komuro, and Masatoshi Ishikawa. “Fast Finger Tracking System for In-Air Typing Interface.” In: *CHI '09 Extended Abstracts on Human Factors in Computing Systems - CHI EA '09*. New York, New York, USA: ACM Press, 2009, pp. 3739–3744. DOI: 10 . 1145/1520340 . 1520564 (cit. on p. 53).
- [296] Chad C. Tossell, Philip Kortum, Clayton Shepard, Laura H. Barg-Walkow, Ahmad Rahmati, and Lin Zhong. “A Longitudinal Study of Emoticon Use in Text Messaging from Smartphones.” In: *Computers in Human Behavior* 28.2 (2012), pp. 659–663. DOI: 10 . 1016/j . chb . 2011 . 11 . 012 (cit. on p. 150).
- [297] Noam Tractinsky. “What is beautiful is usable.” In: *Interacting with Computers* 13.2 (2000), pp. 127–145. DOI: 10 . 1016/S0953-5438(00)00031-X (cit. on pp. 12, 267).
- [298] Dzmitry Tsetserukou. “HaptiHug: A Novel Haptic Display for Communication of Hug Over a Distance.” In: *Haptics: Generating and Perceiving Tangible Sensations - EuroHaptics '10*. 2010, pp. 340–347. DOI: 10 . 1007/978-3-642-14064-8_49 (cit. on pp. 233–235).
- [299] Yuki Urabe, Rafal Rzepka, and Kenji Araki. “Emoticon Recommendation System for Effective Communication.” In: *Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining - ASONAM '13*. New York, New York, USA: ACM Press, 2013, pp. 1460–1461. DOI: 10 . 1145/2492517 . 2492594 (cit. on p. 161).
- [300] Cati Vaucelle, Leonardo Bonanni, and Hiroshi Ishii. “Design of Haptic Interfaces for Therapy.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '09*. CHI. ACM Press, 2009, pp. 467–470. DOI: 10 . 1145/1518701 . 1518776 (cit. on pp. 234, 236).

- [301] Marynel Vázquez, Eric Brockmeyer, Ruta Desai, Chris Harrison, and Scott E. Hudson. “3D Printing Pneumatic Device Controls with Variable Activation Force Capabilities.” In: *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems - CHI '15*. New York, New York, USA: ACM Press, 2015, pp. 1295–1304. DOI: 10.1145/2702123.2702569 (cit. on pp. 234, 248).
- [302] Keith Vertanen, Haythem Memmi, Justin Emge, Shyam Reyal, and Per Ola Kristensson. “VelociTap: Investigating Fast Mobile Text Entry using Sentence-Based Decoding of Touchscreen Keyboard Input.” In: *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems - CHI '15*. New York, New York, USA: ACM Press, 2015, pp. 659–668. DOI: 10.1145/2702123.2702135 (cit. on p. 133).
- [303] Leticia Vidal, Gastón Ares, and Sara R. Jaeger. “Use of Emoticon and Emoji in Tweets for Food-Related Emotional Expression.” In: *Food Quality and Preference* 49 (2015), pp. 119–128. DOI: 10.1016/j.foodqual.2015.12.002 (cit. on p. 148).
- [304] Daniel Vogel and Ravin Balakrishnan. “Interactive Public Ambient Displays: Transitioning from Implicit to Explicit, Public to Personal, Interaction with Multiple Users.” In: *Proceedings of the 17th annual ACM symposium on User interface software and technology - UIST '04*. New York, New York, USA: ACM Press, 2004, pp. 137–146. DOI: 10.1145/1029632.1029656 (cit. on pp. 18, 42, 204).
- [305] Robert Y. Wang and Jovan Popović. “Real-time Hand-tracking with a Color Glove.” In: *ACM Transactions on Graphics* 28.3 (2009), 63:1–63:8. DOI: 10.1145/1531326.1531369 (cit. on p. 43).
- [306] M. Warrington, M. Younger, and J. Williams. “Student Attitudes, Image and the Gender Gap.” In: *British Educational Research Journal* 26.3 (June 2000), pp. 393–407. DOI: 10.1080/01411920050030914 (cit. on p. 21).
- [307] Jane Webster and Hayes Ho. “Audience Engagement in Multimedia Presentations.” In: *ACM SIGMIS Database* 28.2 (1997), pp. 63–77. DOI: 10.1145/264701.264706 (cit. on p. 11).
- [308] Daryl Weir, Henning Pohl, Simon Rogers, Keith Vertanen, and Per Ola Kristensson. “Uncertain Text Entry on Mobile Devices.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '14*. New York, New York, USA: ACM Press, 2014, pp. 2307–2316. DOI: 10.1145/2556288.2557412 (cit. on pp. vii, 113, 114, 129, 176).
- [309] Daryl Weir, Simon Rogers, Roderick Murray-Smith, and Markus Löchtefeld. “A User-Specific Machine Learning Approach for Improving Touch Accuracy on Mobile Devices.” In: *Proceedings of the 25th annual ACM symposium on User interface software and technology - UIST '12*. New York, New York, USA: ACM Press, 2012, pp. 465–476. DOI: 10.1145/2380116.2380175 (cit. on pp. 79, 115, 117).

- [310] Mark Weiser. “The Computer for the 21st Century.” In: *Mobile Computing and Communication Review* 3.3 (1991), pp. 3–11. DOI: 10.1145/329124.329126 (cit. on p. 85).
- [311] Mark Weiser and John Seely Brown. “The Coming Age of Calm Technology.” In: *Beyond Calculation: The Next Fifty Years of Computing*. Vol. 01. July. New York, NY, USA: Copernicus, 1997, pp. 75–85 (cit. on p. 99).
- [312] Elliott Wen, Winston Seah, Bryan Ng, Xuefeng Liu, and Jiannong Cao. “UbiTouch: Ubiquitous Smartphone TouchPads using Built-in Proximity and Ambient Light Sensors.” In: *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing - UbiComp '16*. New York, New York, USA: ACM Press, 2016, pp. 286–297. DOI: 10.1145/2971648.2971678 (cit. on p. 44).
- [313] Christopher D. Wickens. “Multiple Resources and Performance Prediction.” In: *Theoretical Issues in Ergonomics Science* 3.2 (2002), pp. 159–177. DOI: 10.1080/14639220210123806 (cit. on p. 12).
- [314] John Williamson. “Continuous Uncertain Interaction.” PhD Thesis. University of Glasgow, 2006, p. 247 (cit. on p. 116).
- [315] John Williamson, Roderick Murray-Smith, and Stephen Hughes. “Shoogle: Excitatory Multimodal Interaction on Mobile Devices.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '07*. New York, New York, USA: ACM Press, 2007, pp. 121–124. DOI: 10.1145/1240624.1240642 (cit. on p. 247).
- [316] Karl D. D. Willis and Ivan Poupyrev. “12Pixels: Exploring Social Drawing on Mobile Phones.” In: *Proceedings of the 8th International Conference on Pervasive Computing - Pervasive '10*. Berlin, Heidelberg: Springer, 2010, pp. 391–408. DOI: 10.1007/978-3-642-12654-3_23 (cit. on p. 199).
- [317] Andrew D. Wilson. “PlayAnywhere: A Compact Interactive Tabletop Projection-Vision System.” In: *Proceedings of the 18th annual ACM symposium on User interface software and technology - UIST '05*. New York, New York, USA: ACM Press, 2005, pp. 83–92. DOI: 10.1145/1095034.1095047 (cit. on p. 48).
- [318] Andrew D. Wilson, Shahram Izadi, Otmar Hilliges, Armando Garcia-Mendoza, and David Kirk. “Bringing Physics to the Surface.” In: *Proceedings of the 21st annual ACM symposium on User interface software and technology - UIST '08*. New York, New York, USA: ACM Press, 2008, pp. 67–76. DOI: 10.1145/1449715.1449728 (cit. on p. 9).
- [319] Graham Wilson, Martin Halvey, Stephen A. Brewster, and Stephen A. Hughes. “Some Like it Hot: Thermal Feedback for Mobile Devices.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '11*. CHI. ACM Press, 2011, pp. 2555–2564. DOI: 10.1145/1978942.1979316 (cit. on pp. 209, 239).

- [320] Raphael Wimmer, Matthias Kranz, Sebastian Boring, and Albrecht Schmidt. “CapT-able and CapShelf - Unobtrusive Activity Recognition Using Networked Capacitive Sensors.” In: *2007 Fourth International Conference on Networked Sensing Systems*. IEEE, 2007, pp. 85–88. DOI: 10 . 1109/INSS . 2007 . 4297395 (cit. on p. 89).
- [321] Emma Witkowski. “Running from Zombies.” In: *Proceedings of The 9th Australasian Conference on Interactive Entertainment Matters of Life and Death - IE '13*. New York, New York, USA: ACM Press, 2013, pp. 1–8. DOI: 10 . 1145 / 2513002 . 2513573 (cit. on p. 256).
- [322] Jacob O. Wobbrock, Meredith Ringel Morris, and Andrew D. Wilson. “User-defined Gestures for Surface Computing.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '09*. New York, New York, USA: ACM Press, 2009, pp. 1083–1092. DOI: 10 . 1145/1518701 . 1518866 (cit. on p. 60).
- [323] Cheng Xu and Kent Lyons. “Shimmering Smartwatches: Exploring the Smartwatch Design Space.” In: *Proceedings of the Ninth International Conference on Tangible, Embedded, and Embodied Interaction - TEI '14*. New York, New York, USA: ACM Press, 2015, pp. 69–76. DOI: 10 . 1145/2677199 . 2680599 (cit. on p. 209).
- [324] Xing-Dong Yang, Khalad Hasan, Neil Bruce, and Pourang Irani. “Surround-See: Enabling Peripheral Vision on Smartphones during Active Use.” In: *Proceedings of the 26th annual ACM symposium on User interface software and technology - UIST '13*. New York, New York, USA: ACM Press, 2013, pp. 291–300. DOI: 10 . 1145/2501988 . 2502049 (cit. on pp. 42, 44).
- [325] Lining Yao, Ryuma Niiyama, Jifei Ou, Sean Follmer, Clark Della Silva, and Hiroshi Ishii. “PneUI: Pneumatically Actuated Soft Composite Materials for Shape Changing Interfaces.” In: *Proceedings of the 26th annual ACM symposium on User interface software and technology - UIST '13*. New York, New York, USA: ACM Press, 2013, pp. 13–22. DOI: 10 . 1145/2501988 . 2502037 (cit. on pp. 233, 235, 262).
- [326] Chun Yu, Xu Tan, Yue Shi, and Yuanchun Shi. “Air Finger: Enabling Multi-scale Navigation by Finger Height above the Surface.” In: *Proceedings of the 13th international conference on Ubiquitous computing - UbiComp '11*. New York, New York, USA: ACM Press, 2011, pp. 495–496. DOI: 10 . 1145/2030112 . 2030188 (cit. on p. 19).
- [327] Shumin Zhai, Michael Hunter, and Barton A. Smith. “The Metropolis Keyboard - An Exploration of Quantitative Techniques for Virtual Keyboard Design.” In: *Proceedings of the 13th annual ACM symposium on User interface software and technology - UIST '00*. New York, New York, USA: ACM Press, 2000, pp. 119–128. DOI: 10 . 1145/354401 . 354424 (cit. on p. 176).
- [328] Cheng Zhang, Anhong Guo, Dingtian Zhang, Yang Li, Caleb Southern, Rosa I. Arriaga, and Gregory D. Abowd. “Beyond the Touchscreen: An Exploration of Extending Interactions on Commodity Smartphones.” In: *ACM Transactions on Interactive Intelligent Systems* 6.2 (2016), 16:1–16:23. DOI: 10 . 1145/2954003 (cit. on p. 42).

- [329] Wei Zhang, Jian Xu, and Dan Li. *Electronic equipment, pressure detecting method and pressure detecting device*. US Patent 20120313872. 2012 (cit. on p. 130).
- [330] Yang Zhang, Junhan Zhou, Gierad Laput, and Chris Harrison. “SkinTrack: Using the Body As an Electrical Waveguide for Continuous Finger Tracking on the Skin.” In: *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems - CHI '16*. New York, New York, USA: ACM Press, 2016, pp. 1491–1503. DOI: 10.1145/2858036.2858082 (cit. on p. 230).
- [331] Zhengyou Zhang, Ying Wu, Ying Shan, and Steven Shafer. “Visual Panel: Virtual Mouse, Keyboard and 3D Controller with an Ordinary Piece of Paper.” In: *Proceedings of the 2001 workshop on Percetive user interfaces - PUI '01*. New York, New York, USA: ACM Press, 2001, pp. 1–8. DOI: 10.1145/971478.971522 (cit. on p. 53).
- [332] Chen Zhao, Ke-yu Chen, Md Tanvir Islam Aumi, Shwetak Patel, and Matthew S. Reynolds. “SideSwipe: Detecting In-Air Gestures Around Mobile Devices Using Actual GSM Signal.” In: *Proceedings of the 27th annual ACM symposium on User interface software and technology - UIST '14*. 2014. DOI: 10.1145/2642918.2647380 (cit. on p. 45).
- [333] Yi Zheng, Qi Liu, Enhong Chen, Yong Ge, and J. Leon Zhao. “Time Series Classification Using Multi-Channels Deep Convolutional Neural Networks.” In: *International Conference on Web-Age Information Management - WAIM '14*. 2014, pp. 298–310. DOI: 10.1007/978-3-319-08010-9_33 (cit. on p. 94).
- [334] Ying Zheng and John B. Morrell. “Haptic Actuator Design Parameters that Influence Affect and Attention.” In: *2012 IEEE Haptics Symposium - HAPTICS'12*. Ieee, 2012, pp. 463–470. DOI: 10.1109/HAPTIC.2012.6183832 (cit. on pp. 233–235).
- [335] Ying Zheng, Ellen Su, and John B. Morrell. “Design and evaluation of pactors for managing attention capture.” In: *2013 World Haptics Conference (WHC)*. IEEE, 2013, pp. 497–502. DOI: 10.1109/WHC.2013.6548458 (cit. on pp. 232, 233).
- [336] Thomas G. Zimmerman, Joshua R. Smith, Joseph A. Paradiso, David Allport, and Neil Gershenfeld. “Applying Electric Field Sensing to Human-Computer Interfaces.” In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '95*. New York, New York, USA: ACM Press, 1995, pp. 280–287. DOI: 10.1145/223904.223940 (cit. on p. 89).
- [337] George Kingsley Zipf. *Human Behavior and the Principle of Least Effort: An Introduction to Human Ecology*. Cambridge, MA: Addison-Wesley Press, 1949. ISBN: 978-1614273127 (cit. on p. 24).

Curriculum Vitae

Full Name Henning Sebastian Clemens Pohl
Date of Birth December 23rd, 1982
Place of Birth Achim, Germany

Education

2007 – 2010 Computer Science (Master of Science)
Technische Universität Darmstadt, Germany
2008 – 2009 Digital Arts and Sciences (Visiting on Fulbright Scholarship)
University of Florida, USA
2005 New Media (Visiting as exchange student)
Indiana University Purdue University Indianapolis, USA
2003 – 2007 Digital Media (Bachelor of Science)
Universität Bremen, Germany
2002 Abitur
Schulzentrum Walle, Bremen, Germany

Professional Experience

Jan 2017 – *present* Postdoctoral researcher
University of Copenhagen, Denmark
Nov 2012 – Oct 2016 Research associate
Leibniz Universität Hannover, Germany
Jun 2012 – Oct 2012 Vacation scholar
University of Glasgow, UK
Jun 2010 – May 2012 Scholarship holder
Hasso Plattner Institute, Germany

Colophon

This dissertation was typeset with \LaTeX 2 ϵ and compiled with Lua \TeX version 1.0.0. The custom design is built on top of the *scrprpt* class from the KOMA-Script package.

The body text is set in 11 pt Gentium, designed by Victor Gaultney. Headings are set in Roboto, designed by Christian Robertson.

Plots prepared with matplotlib and seaborn.