3rd Conference on Production Systems and Logistics

Controlling A Vacuum Suction Cup Cluster Using Simulation-Trained Reinforcement Learning Agents

Georg Winkler[1]

[1] *Fraunhofer Institute for Machine Tools and Forming Technology IWU, Reichenhainer Straße 88, 09126 Chemnitz, Germany*

Abstract

Using compressed air in industrial processes is often accompanied by a poor cost-benefit ratio and a negative impact on the environmental footprint due to usual distribution inefficiencies. Compressed air-based systems are expensive regarding installation and lead to high running costs due to pricey maintenance requirements and low energy efficiency due to leakage. However, compressed air-based systems are indispensable for various industrial processes, like handling parts with Class A surface requirements such as outer skin sheets in automobile production. Most of those outer skin parts are solely handled by vacuum-based grippers to minimize any visible effect on the finished car. Fulfilling customer expectations and simultaneously reducing the running costs of decisive systems requires finding innovative strategies focused on using the precious resource of compressed air as efficiently as possible.

This work presents a sim2real reinforcement learning approach to efficiently hold a workpiece attached to a vacuum suction cup cluster. In addition to pure energy-saving, reinforcement learning enables those agents to be trained without collecting extensive data beforehand. Furthermore, the sim2real approach makes it easy and parallelizable to examine numerous agents by training them in a simulation of the testing rig rather than at the testing rig itself. The possibility to train various agents fast additionally facilitates focusing on the robustness and simplicity of the found agents instead of only searching for strategies that work, making training an intelligent system scalable and effective. The resulting agents reduce the amount of energy necessary to hold the workpiece attached by more than 15% compared to a reference strategy without machine learning and by more than 99% compared to a conventional strategy.

Keywords

Reinforcement Learning; Sim2Real; Energy Efficiency; Automotive; Gripper; Suction Cups; Compressed Air; Vacuum-Based Handling; Car Body Shop; Body-In-White.

1. Introduction

Today's automotive sector is highly automated and competitive. In addition to the possibility of differentiating oneself from the competition through high-quality processing, the ecological impact of a product plays an increasingly important role in the product's evaluation by both customers and stakeholders. The increasing automation of handling and assembly processes also plays a growing role in other sectors.

Class A surface requirements apply to specific components in automotive production to meet the high expectations on processing quality. The deformation of these components must be minimized in the handling and assembly process. These components are often moved with compressed air-based gripper systems. Based on the Venturi principle, compressed air can be used to create a vacuum which, in contrast to clamping, has less influence on the shape of the component. While these gripper systems may be adequate for handling processes, compressed air is often connected to leakage and thus high energy demand. It is necessary to reduce the amount of energy consumed by compressed air-based systems to fulfill the rising requirements regarding the ecological footprint and simultaneously remain competitive regarding quality and price. Three potential approaches can be identified to do so.

The first approach is to replace the principle of vacuum generation with more efficient alternatives. Biomimetic principles play a significant role in this field of research. One possibility is to imitate muscle-like systems with deformable membranes. Various principles are presented in [1–4] to create such systems based on different deformation mechanisms. Alternatively, the size of suction cups can be adjusted to fit the actual loads to be handled and thus reduce waste of energy due to over dimensioning as presented in [5]. The work conducted in [6] uses an origami-like structure to fit diverse surfaces to extend efficient gripping to non-flat surfaces. The zero pressure difference method presented in [7] additionally achieves efficient aspiration on porous surfaces, reducing the leakage by minimizing the pressure difference between the environmental pressure and the pressure in the outermost border of the suction zone.

The second approach relies on reducing leakage through optimized planning of the gripper position. The experiments presented in [11,8,10,9] use artificial neural networks (ANNs) to find suitable gripping positions based on point clouds extracted from CAD or depth images. In [12], those approaches are extended by optimizing the gripping positions and the amount and dimensioning of the grippers.

The work presented here can be placed in the third category focusing on the optimal control of the gripping system. Research in this area is sparse, especially if the focus is on the optimized gripper control via reinforcement learning (RL) agents. The closest experiments to compare this work with are presented in [13], where Deep-Q-Networks (DQNs) are used to control the compressed air supplied in different phases of package movement. This work can be delineated from the following three points of view. On the one hand, the fine-grained regulation of the compressed air supply is replaced by only deciding to switch it on or off entirely, resulting in a more straightforward control mechanism. On the other, switching on or off is based on measured pressure values in the vacuum chambers instead of measuring accelerations in the handling process. Finally, in [13], a handling process with actual movement is considered instead of a stationary process. This process and the actual setup are presented in section 2. The work presented in the following is based mainly on the master thesis [14].

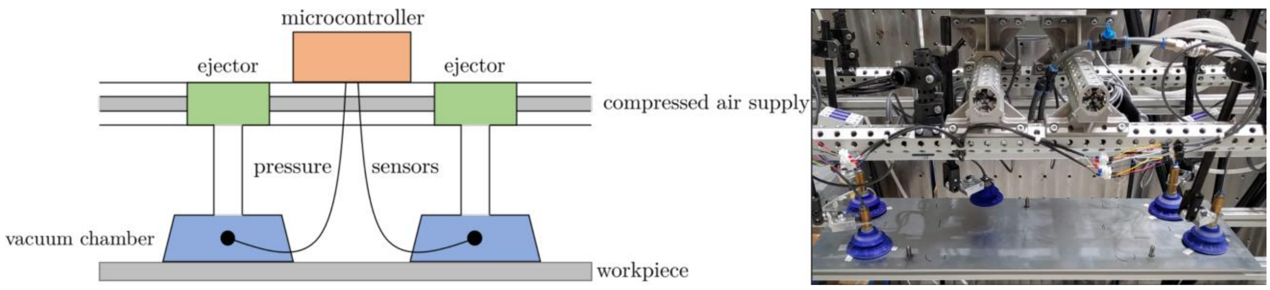

Figure 1: The vacuum suction cup cluster. Left: Schematic illustration. Right: Photography.

2. Setup

The left side of Figure 1 shows a schematic illustration of the vacuum suction cup cluster to be controlled as efficiently as possible. It consists of four vacuum chambers that are colored blue and evacuated by four ejectors based on the Venturi principle. Those are colored green, and they are connected to a Raspberry Pi 4B microcontroller. Approximately every second, the microcontroller queries pressure sensors integrated into the vacuum chambers to get the current pressure state. Based on this pressure vector, the microcontroller controls the ejectors. For every timestep, the microcontroller can decide between doing nothing or activating one of the ejectors, leading to an immediate pressure drop through the evacuation of the chamber. The sucks per hour (sph) can be used as a surrogate measure for compressed air demand for a given control scheme. This control task aims to hold a metal sheet, the grey-colored workpiece. Photography of the real experimental rig is depicted on the right side. [14]

3. Methods

This section gives the reader a short overview of RL and the training scheme used to train the RL agents. Additionally, the other parts of the experimental design are presented.

3.1 A short dive into reinforcement learning

The presented setup is to be controlled via RL agents. RL is one of three machine learning (ML) domains, with supervised and unsupervised learning being the others. In contrast to the other domains, RL does not require collecting extensive amounts of data beforehand. Instead, the agents explore a given environment and generate the data to learn from during this exploration. The reference work by Sutton and Barto [15] gives an in-depth description of the theory behind RL.

This environment's mathematical formalization can be considered a Markov Decision Process (MDP) consisting of four main properties. Those are a set of states $s_i \in S$ the system can take, a set of actions $a_i \in A$ the agent can choose to execute, the transition function $p(s'|s,a): S \times A \rightarrow S$ determining the state following to a specific action in a particular state and the reward function $r(s,a,s'): S \times A \times S \rightarrow \mathcal{R}$ determining the reward the agent collects with this transition.

The goal of the exploration process is to visit many different states, try many different actions and learn what reward can be expected in the future by choosing specific actions. To make good decisions for every timestep t in an episode with length T, it is not only necessary to greedily look on the next immediate reward, but to also take possible future rewards into consideration. Therefore, the discounted sum of rewards denoted as the return R, with $\gamma \in [0,1]$ being the discount factor, is taken into consideration instead. The calculation of the return is given in Equation (1).

$$R_t = r_{t+1} + \gamma\, r_{t+2} + \cdots + \gamma^{T-t-1}\, r_T = \sum_{k=0}^{T} \gamma^k\, r_{t+k+1} \qquad (1)$$

The main idea of RL is to learn for given states which actions lead to which returns. The agent's decision making is given by the policy π and is often implemented to greed for the highest expected return. However, for continuous state or actions spaces, it is not possible to visit every state and try every action. Instead, it is necessary to find an approximator for expected rewards in future time steps. Therefore, different mathematical models are trained to learn to predict the Q-values of given state-action-pairs with

$$Q^{\pi}(s,a) = \mathbb{E}_{\pi}[R_t | s_t = s, a_t = a] \qquad (2)$$

The theory behind this Q-learning approach is heavily researched. Interested readers are referred to [15], which illuminates all the essential basics of the methods used in this work and [14] for an overview.

3.2 Approximation of Q with regression models

In theory, the approximation of Q-values can be realized with every regression model, from the simple linear regression approach to ANNs with unlimited depth and complexity. This work compares combinations of the RL scheme and a variety of different regression models. Diving into the theory of all these models would be beyond the scope of this paper. Background information can be found in [14].

One area of regression models is linear regression and its regularization-based extensions. Those are the LASSO, Ridge, and Elastic Net regression. The second domain is represented by support vector regression (SVR), whereby the ϵ- and the ν-SVR are used. Additionally, a k-nearest neighbors regression (KNNR) is used. The fourth domain includes decision tree-based methods. The decision tree regression relying on a single decision tree represents the most straightforward approach. The ensemble methods random forest regression, XGBoost regression and gradient boosting regression extend this approach by combining multiple decision trees. For all models except the XGBoost regression, the implementation in scikit-learn is used. The XGBoost regression is implemented in an individual package.

Finally, ANNs based on the Keras framework are used to predict the Q-values in the given use case. The plain use of ANNs comes with two significant drawbacks: They are susceptible to highly-correlated inputs and changing or non-stationary target values, both typical for RL environments. DQNs add an experience replay memory (ERM) and an additional target network to circumvent those two problems. Those DQNs can further be extended with prioritization in the sampling process [16], the consideration of double learning [17] and using a dueling architecture [18] to further improve and stabilize the learning procedure.

The ERM is combined with all the models from above. The models explore the environment during the training process, and the collected steps consisting of the initial state, chosen action, reward, and following state are saved in this finite memory. New data points replace the most outdated data points if the ERM is filled. This memory limitation is necessary to decrease the influence of outdated data points, such as state-action pairs that are unlikely to be chosen by an agent trained further because they might lead to bad states.

3.3 Experimental Design

The conducted experiments can be split into four major phases. In the first phase, baseline methods to control the vacuum suction cup cluster without artificial intelligence are evaluated. The second phase is used to examine the behavior of the vacuum suction cup cluster and create a simulation based on it. In the third phase, the simulation is exploited to train various RL agents and transfer and test them at the real-world testing rig in phase four. It follows a more detailed description of the four phases. [14]

3.3.1 Baseline methods

Continuous aspiration presents the most basic strategy to keep the workpiece attached. Assuming that this strategy is equal to evacuating every suction cup at every timestep, this leads to a baseline of 14400 sph. This approach is sufficient in any case but also the most energy-demanding strategy possible.

A simple alternative is to use a threshold-based strategy. Therefore, a threshold is chosen, and whenever one of the measured pressure values exceeds this fraction of the ambient pressure, the corresponding ejector is activated. The sufficient threshold of 80 % found by stepwise reduction starting from a high threshold leads to 21 sph needed. However, this sufficient threshold does not imply that the workpiece falls off whenever it is exceeded, since this depends on the interplay of all four suction cups.

3.3.2 Creation of the simulation and the training environment

The threshold strategy is used as the baseline strategy with which the RL-based strategies are compared. The rest of the experiments aim to reduce the ejector time further and thus beat this baseline. Therefore, the second phase aims to find a simulation to train those agents. The question may arise why a simulation is used to train the agents instead of training them on the real-world testing rig. The reason is simple: time.

The advantage of simulation-based training is obvious. It is possible to train an unrestricted number of agents in parallel and simulate thousands of timesteps in the blink of an eye. Although there are approaches to train agents in parallel in real-world environments, the exploitation of those approaches is, next to other challenges with those approaches, limited for the given experiments by the fact that there is only one testing rig.

An approach like the threshold agent, but with a threshold of 96 %, is used to collect data for creating the simulation. If the workpiece falls off, it is automatically reattached. Table 1 shows the resulting mean values and standard deviations of the pressure difference per timestep. It must be noted that the used pressure sensors are not calibrated. Instead, the raw pressure values from the pressure sensors are used, and thus also the differences cannot be connected to a typical pressure unit like Pascal.

Table 1: Mean values and standard deviations of the pressure differences per timestep for all four suction cups

	Cup 1	Cup 2	Cup 3	Cup 4
Mean	0.99	1.86	1.10	1.16
Standard deviation	1.75	2.01	2.04	2.41

This simulation is integrated into the RL environment, which complements the state and action spaces with the reward function. A high penalty of -100000 points should keep the agent from letting the workpiece fall, while a penalty of -10 is connected to every sucking decision to avoid the workpiece being resucked too often. Doing nothing results in a reward of $+1$. Those values are set based on a trial-and-error scheme.

3.3.3 Training scheme

There are three methods to sample from the ERM. The first option is to sample randomly. The second option is to prioritize sampling based on the difference between expected and realized reward as in [16]. The third option is to use every data point contained in the ERM for every training step. This approach is referred to as "batch equal buffer" (beb) approach. This third approach is not used in combination with the DQNs since it would extend the training time of the ANNs to an infeasible dimension.

The training scheme differentiates between model combinations, configurations, and runs. In this case, the term model combination denotes combining a mathematical model, say the linear regression, with possible modifications. For all models but the DQNs, possible options are random sampling, prioritized sampling or the beb approach. For the DQNs, the options are different. ANNs with one to three hidden layers are used as mathematical background models. For them, it is freely combined whether prioritization is used, if double learning is considered, and if a dueling architecture is implemented instead of solely learning the Q-values.

Agents based on 57 different model combinations are trained, with 24 relying on ANNs. For every of those model combinations, 300 different configurations are trained and evaluated during the training process. Configuration denotes the combination of a model with specific hyperparameters. Those hyperparameters can further be dissected in model-inherent and environmental hyperparameters. All model combinations share the limits for environmental hyperparameters. The limits for the model-inherent hyperparameters are set based on experience and around default values. The package Optuna, based on a tree-structured parzen estimator, is used for hyperparameter optimization.

In supervised learning, model validation methods like cross-validation are used to ensure a generalized assertion about the capability of agents and that success or failure not only depends on good or bad luck but also prevents challenges like overfitting. The RL scheme makes the use of such strategies hard since the distinction between test and training data is not easily possible. Three training runs are performed to ensure some validation, and the mean score over all three runs is used as estimate of a configuration's performance.

Every run consists of 500 episodes of training, with a maximum episode length of 300 steps. The chosen maximum episode length presents a trade-off between reducing the training time to a minimum while ensuring enough exploration to learn the whole environment. A test episode with 1500 steps is performed every ten training episodes, leading to 50 validation episodes per run. Those episodes assess how well the agent performs with the current training state. Following the given reward scheme of the environment, a theoretical score between $+1500$ and -115000 points can be reached in every validation episode.

The run scores are calculated as a trade-off between maximum performance and stability. Taking more validation episodes into consideration places more weight on stability than on peak performance. On the one hand, only focusing on performance in RL can lead to preferring agents that might tend to catastrophic forgetting, which can negatively impact the usefulness in real world use cases. On the other hand, taking all validation episodes into account may discriminate agents that learn slow but stable with high performance in the end. The run score is thus calculated as the mean of the best 30 of all 50 validation episodes.

3.3.4 Transfer to real-world tests

Having 50 validation episodes for every run and thus 150 possible test candidates for real-world tests per configuration and a total of 17100 configurations with only limited testing capacities makes it non-trivial to decide which exact models should be tested. This decision is thus based on a heuristic. First, all unsuccessful configurations are sorted out. A configuration is stated as being successful if the configuration score lies above -20000 points. Configurations with equal or better scores perform well regarding stability and peak performance. Further reduction is achieved by limiting the maximum number of configurations tested per model combination to five. Following these restrictions, the amount of test candidates is bounded to an upper maximum of 285. Also, it is necessary to decide which training state of the agent to use for the real-world test, with 50 model states corresponding to the 50 validation episodes being possible options and three possible runs. It is not trivial to compare the three runs due to the strong influence of randomness in the exploration and the evolution of the pressure values in the simulation. Thus, the decision is made to test the training state connected to the highest validation score of every run.

The real-world tests are performed in two phases. Short ten-minute pre-tests are performed first. The agents can fail if the work piece already falls off in those ten minutes or if the agent decides to resuck more than 18 times, which can be extrapolated to a demand of more than 108 sph and thus five times more than the simple threshold strategy. If more than one configuration passes the pre-test for any model combination, only the one with the best pre-test performance in terms of sph needed is tested in a long-time test. This limitation is again imposed by limited testing capabilities and bounds the amounts of long-time tests to 171, with a long-time test per run of the model combination. The respective agents are then tested over a period of two hours to examine if they can hold the workpiece attached and how many sph they need to do so.

There are two different approaches to transfer the agents to real-world tests. The first one is to use the raw measured pressure values. This option is easier to implement, but the gap between simulation and the real-world system can potentially decrease the capability to hold the workpiece attached. The second approach measures the ambient pressure and scales the measured pressure values to fit the simulated pressure values. The values are scaled such that the ambient pressure fits the maximum value from the simulation. In every test case, the pressure values are scaled up, leading to a higher probability of the agents holding the workpiece attached, but on the price of making the agents less efficient due to more frequent resucking.

This scheme is used to answer three guiding questions:

1. Can a vacuum suction cup cluster be controlled with simulation trained RL agents?
2. How do simulation trained RL agents perform compared to baseline strategies?
3. Are there individual methods that stand out throughout the tests?

4. Results

This section presents the results of the experiments described above. First, it gives an overview of the findings of the simulated training. The second part describes the results from transferring the agents to work on the real-world testing rig. [14]

4.1 Performance in simulated training

Of all 17100 configurations tested, 1901 can be classified as successful based on the restrictions mentioned above. Of all 57 model combinations, 24 lead to successful agents. From those 24 successful model combinations, seven rely on using ANNs, with a total of 178 successful configurations.

Combining the beb approach with the ν-SVR reaches the highest individual score with -2305. In contrast, the combination with the elastic net regression leads to the highest number of successful configurations with 183 and the highest mean performance over all 300 configurations with a mean of -24301. A more in-depth analysis and comparison of the simulation results can be found in [14].

A total of 110 configurations fulfill the requirements described above and thus are transferred to real-world tests. For the linear regression, only the combination with a prioritized ERM leads to success and thus to five transferred configurations. Five configurations for each the prioritized and the beb approach are transferred for the ridge and LASSO regression. The LASSO regression adds four more successful configurations using the basic ERM. Five configurations for each sampling method are transferred for the elastic net regression, the KNNR, and the ν-SVR, while the ϵ-SVR is not successful with the beb approach. A single successful configuration is found for neither the basic decision tree regression nor the ensemble methods based on it. Thus, those methods are not represented in real-world trials.

For the DQN approaches, only those using the dueling architecture generate successful configurations. Four configurations with one hidden layer and one configuration using three hidden layers are transferred only using this modification. With prioritization added to the combinations, another five configurations are transferred for one and three hidden layers. Considering double learning instead leads to five transferred DQNs with one and one with two layers. Combining all three modifications is only successful for one layer DQNs, where five more are tested in the real-world tests.

Table 2: Performances for all model combinations tested in two-hour real-world tests with test results given in sucks per hour and differences compared to the threshold agent in percent

Regression model	Sampling Method	Scaled Test [sph] (Diff [%])	Unscaled Test [sph] (Diff [%])
Ridge	Prioritized ERM	24.7 (+17.6)	23.3 (+11.0)
	Basic ERM	22.9 (+9.0)	21.6 (+2.9)
LASSO	Batch equal buffer	54.8 (+161.0)	69.7 (+231.9)
Elastic Net	Basic ERM	17.7 (-15.7)	18.3 (-12.9)
	Prioritized ERM	37.8 (+80.0)	36.6 (+74.3)
	Batch equal buffer	19.1 (-9.0)	18.9 (-10.0)
KNNR	Basic ERM	17.5 (-16.7)	Failed
ϵ-SVR	Basic ERM	66.7 (+217.6)	55.2 (+162.9)
ν-SVR	Basic ERM	29.5 (+40.5)	25.5 (+21.4)
	Prioritized ERM	22.0 (+4.8)	19.5 (-7.1)

4.2 Performance in real-world tests

Agents from twelve of the 24 model combinations transferred pass the pre-test. The only successful model combinations are the combination of ridge regression with a prioritized ERM memory or using the beb approach, the LASSO regression with using the beb approach, the elastic net and the KNNR with all three sampling schemes, the ϵ-SVR combined with the basic ERM and the ν-SVR with a basic or prioritized ERM. It is noticeable that none of the agents using DQNs pass the pre-test.

Following those results, 36 long-time tests are conducted. The results of tests where the agent can hold the workpiece attached are illustrated in Table 2. The agents using the KNNR do all fail to keep the workpiece attached if they are tested without the pressure scaling. If pressure scaling is applied, the combination with the basic ERM can keep the workpiece attached and, at the same time, is the most efficient in the field with only 17.5 sph to do so. This demand is equivalent to a reduction of 16.7 % compared to the threshold agent. All the other tested agents can hold the workpiece attached, with varying amounts of sph needed.

5. Discussion

Based on the experimental results, this section answers the research questions posed at the outset and identifies weaknesses in the experimental design. [14]

5.1 Answering the research questions

5.1.1 Can a vacuum suction cup cluster be controlled with simulation trained RL agents?

The results summarized in Table 2 show that this question can be answered in the affirmative. Nine different agents can hold the workpiece attached for two hours without regard if the pressure values are scaled. With the combination of the basic ERM and the KNNR, another combination can be added for scaled pressure values.

5.1.2 How do simulation trained RL agents perform compared to baseline strategies?

The threshold-based strategy needs 21 sph to keep the workpiece attached. As Table 2 implies, the comparison shows that no statement is possible whether the RL-based agents perform better or worse in general. However, three model combinations perform more efficiently than the threshold agent, with savings of up to $16.7\,\%$ combining the KNNR with the basic ERM and pressure scaling. The other two use the elastic net regression with the basic ERM or the beb approach.

Additionally, combining the ridge regression with the basic ERM and the ν-SVR with the prioritized version, two more model combinations perform comparably well with differences in the single-digit percentage range. The difference between the scaled and the unscaled test for combining the elastic net regression with the basic ERM shows that although it is assumed that the agents should work more efficiently without scaling, this does not necessarily hold for the actual experimental results. This unexpected difference can be attributed to the fact that the number of sucks required fluctuates in correlation with the fluctuation of the ambient conditions. Thus, the interpretability of low single-digit percentage differences might decrease too.

5.1.3 Are there individual methods that stand out throughout the tests?

This question can be answered from two points of view. On the one hand, it is interesting to look at the results of the simulated experiments. Since only a fraction of configurations is tested in the real-world tests, the simulation results give a broader view about stability in the learning process and susceptibility against the hyperparameter changes. From the simulation point of view, two models stand out positively. The elastic net regression and the ν-SVR produce by far the most successful model combinations, being the only two models that produce more than 100 successful configurations using either of the three sampling methods. Additionally, the ν-SVR results in the best individual score for all three sampling methods. The elastic net regression achieves the best mean results for all 300 trained configurations for all three sampling methods.

On the other hand, good performance in the simulated experiments might indicate that a model might be stable in training and promising for use in real-world scenarios, but what matters is the actual performance in the real-world tests. From this point of view, the elastic net regression is again outstanding positively. Combined with the basic ERM or the beb approach, the elastic net regression saves $15.7\,\%$ and $9.0\,\%$ in the scaled and $12.9\,\%$ and $10.0\,\%$ in the unscaled tests. The only model performing more efficiently is the KNNR combined with a basic ERM and scaled pressure values. However, this model combination fails using unscaled pressure values. The susceptibility to a broader gap between the simulation and the real-world behavior implies that this method thus might not be optimally suitable for such transfer scenarios.

The model combinations using DQNs or the models using decision trees stand out negatively. For the DQNs, only 178 of all 7200 configurations lead to success in the simulation, but none of the transferred models can hold the workpiece attached in the real-world tests. For the decision tree-related models not a single of all 3600 configurations is successful. The interpretation of those two negative examples must be put into the context of the used training scheme. It cannot be concluded that those models are generally unsuitable for the given task. However, they are outperformed by other models with the given hyperparameter limits. No statement can be made about whether the same or even similar results would be obtained if other hyperparameter limits were chosen.

5.2 Experimental limitations and starting points for further research

The experiments in this work are subject to various limitations. This section presents an excerpt of these limitations to allow the reader to put the results into context and potentially identify interesting starting points for further research. These limitations can be roughly divided into three categories.

First, there are challenges resulting from the use of a simulation. As with every simulation, the one used in this work is not a perfect digital representation of the actual physical system, especially since only a simple linear relation between the pressure differences over time is assumed. This imperfection might lead to the trained agents performing well in most situations but failing in special situations unseen before. The additional influence of wear and tear might further change the system's dynamics, widen the gap between simulation and real-world and cause a decrease in the ability to control the vacuum suction cup cluster.

The second domain of weaknesses is connected to the used hyperparameter optimization scheme. The performance of most models depends heavily on choosing hyperparameters that fit the problem. Even heuristics and recommendations for finding optimal hyperparameters cannot guarantee that they lie within the chosen limits. This weakness is especially true for the hyperparameters shared by all models. However, it is also not practicable to widen the limits. The widening of the limits might increase the probability that the optimal hyperparameter values lie inside the limits. However, due to the curse of dimensionality, it does not necessarily mean that they can be found with the given number of configurations to be tested.

The third category of weaknesses results mainly from the approach used to transfer the agents from the simulation to real-world use. One central point is saving only the best-performing model regarding collected rewards for every run. Saving only that model might lead to savings in memory demand and relives the experimenter of deciding which model to take manually. However, it does not ensure that this model performs best in the real-world test case compared to all other potential candidates. As already mentioned, the experimental setup is also constantly influenced by changing ambient conditions and effects like wear and tear. Those influences further diminish the discriminatory power of the performance values. Repeated testing and testing a broader spectrum of candidates could potentially minimize those effects. However, those two extensions are left open for future research due to limited testing capacities.

6. Summary and outlook

This paper presents a way to train simulation-based RL agents to control a vacuum suction cup cluster and compares their performance in the real-world use case with alternative baseline methods without ML. The baseline methods are continuous sucking and a threshold-based agent that sucks whenever the measured pressure values inside the suction cups surpass the ambient pressure measured before.

While continuous sucking leads to a demand of 14400 sucks per hour, the threshold agent gets by with only 21 sucks per hour. A variety of RL agents is found that outperform the threshold agent. The most efficient is based on a KNNR and only needs 17.5 sph, saving 16.7 % in comparison. Other promising agents are those based on the elastic net regression and the ν-SVR. They are even more successful in simulated experiments. Their real-world application leads to respective savings of 15.7 % and 7.1 %**.** The conducted experiments show that simulation-trained RL agents have the potential to control a real-world vacuum suction cup cluster and at the same time outperform baseline strategies without ML. Thus, the presented work can be viewed as a proof of concept for the proposed simulation-based RL training scheme.

A variety of starting points could be used to extend both the simulation-based and the real-world experiments. Limitations of the experiments are listed, and a comparison to changes resulting from fixing these issues could improve the significance and informative value of the conducted experiments. Furthermore, more attention could be paid to aspects like computing effort or stability against intrinsic and extrinsic influences rather than pure performance.

Acknowledgement

This work resulted from a project funded by the European Union Grant number 100359269.

Europäische Union

Europa fördert Sachsen.
EFRE
Europäischer Fonds für regionale Entwicklung

References

[1] Follador, M., Tramacere, F., Mazzolai, B., 2014. Dielectric elastomer actuators for octopus inspired suction cups. Bioinspiration & Biomimetics 9 (4), 46002.

[2] Haines, C.S., Li, N., Spinks, G.M., Aliev, A.E., Di, J., Baughman, R.H., 2016. New twist on artificial muscles. Proceedings of the National Academy of Sciences 113 (42), 11709–11716.

[3] Welsch, F., Kirsch, S.-M., Motzki, P., Schmidt, M., Seelecke, S., 2018. Vacuum Gripper System Based on Bistable SMA Actuation, in: ASME 2018 Conference on Smart Materials, Adaptive Structures and Intelligent Systems. American Society of Mechanical Engineers Digital Collection.

[4] Zhang, P., Kamezaki, M., Otsuki, K., He, Z., Sakamoto, H., Sugano, S., 2020. Development of a Vacuum Suction Cup by Applying Magnetorheological Elastomers for Objects with Flat Surfaces, in: 2020 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM). IEEE.

[5] Gabriel, F., Fahning, M., Meiners, J., Dietrich, F., Dröder, K., 2020. Modeling of vacuum grippers for the design of energy efficient vacuum-based handling processes. Production Engineering 14 (5-6), 545–554.

[6] Zhakypov, Z., Heremans, F., Billard, A., Paik, J., 2018. An Origami-Inspired Reconfigurable Suction Gripper for Picking Objects With Variable Shape and Size. IEEE Robotics and Automation Letters 3 (4), 2894–2901.

[7] Shi, K., Li, X., 2020. Vacuum suction unit based on the zero pressure difference method. Physics of Fluids 32 (1), 17104.

[8] Mahler, J., Matl, M., Liu, X., Li, A., Gealy, D., Goldberg, K., 2018. Dex-Net 3.0: Computing Robust Vacuum Suction Grasp Targets in Point Clouds Using a New Analytic Model and Deep Learning, in: 2018 IEEE International Conference on Robotics and Automation (ICRA). IEEE.

[9] You, F., Mende, M., Stogl, D., Hein, B., Kroger, T., 2018. Model-Free Grasp Planning for Configurable Vacuum Grippers, in: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE.

[10] Wan, W., Harada, K., Kanehiro, F., 2019. Planning Grasps for Assembly Tasks.

[11] Jiang, P., Ishihara, Y., Sugiyama, N., Oaki, J., Tokura, S., Sugahara, A., Ogawa, A., 2020. Depth Image-Based Deep Learning of Grasp Planning for Textureless Planar-Faced Objects in Vision-Guided Robotic Bin-Picking. Sensors (Basel, Switzerland) 20 (3), 706.

[12] Gabriel, F., Baars, S., Römer, M., Dröder, K., 2021. Grasp Point Optimization and Leakage-Compliant Dimensioning of Energy-Efficient Vacuum-Based Gripping Systems. Machines 9 (8), 149.

[13] Gabriel, F., Bergers, J., Aschersleben, F., Dröder, K., 2021. Increasing the Energy-Efficiency in Vacuum-Based Package Handling Using Deep Q-Learning. Energies 14 (11), 3185.

[14] Georg Winkler, 2021. Smart suction cup cluster handling using sim2real reinforcement learning. Master Thesis, Chemnitz University of Technology.

[15] Sutton, R.S., Barto, A.G., 2018. Reinforcement Learning, second edition: An Introduction. MIT Press.

[16] Schaul, T., Quan, J., Antonoglou, I., Silver, D., 2015. Prioritized Experience Replay. http://arxiv.org/pdf/1511.05952v4.

[17] Hado van Hasselt, Arthur Guez, David Silver, 2016. Deep Reinforcement Learning with Double Q-Learning. Proceedings of the AAAI Conference on Artificial Intelligence 30 (1).

[18] Ziyu Wang, Tom Schaul, Matteo Hessel, Hado Hasselt, Marc Lanctot, Nando Freitas, 2016. Dueling Network Architectures for Deep Reinforcement Learning. International Conference on Machine Learning, 1995–2003.

Biography

Georg Winkler (*1996) received his B.Sc. in Physics and his M.Sc. in Data Science at the University of Technology Chemnitz. Since 2019, he has been part of the research department for body shop, assembly and disassembly at the Institute for Machine Tools and Forming Technology (IWU) with focus on smart systems and the integration of artificial intelligence in industrial processes.