Optical autonomous sensor module communicating with a smartphone using its camera

Dudko, Uliana, Pflieger, Keno, Overmeyer, Ludger

Uliana Dudko, Keno Pflieger, Ludger Overmeyer, "Optical autonomous sensor module communicating with a smartphone using its camera," Proc. SPIE 10922, Smart Photonic and Optoelectronic Integrated Circuits XXI, 109220I (4 March 2019); doi: 10.1117/12.2506777

SPIE.

Event: SPIE OPTO, 2019, San Francisco, California, United States

Optical autonomous Sensor Module communicating with a Smartphone using its Camera

Uliana Dudko*, Keno Pflieger, Ludger Overmeyer

Institute for Transport and Automation Technology, Leibniz University Hannover,
An der Universität 2, 30823 Garbsen, Germany

ABSTRACT

Wireless optical communication is a viable alternative to conventional RF technology. Our novel design combines optical communication and energy harvesting in one device with a size of 30 x 10 x 5 mm using the latest innovations in low-power electronics and solar cell technology. In our study, we implement visible light communication between a sensor module and a smartphone. The proposed system design and a communication protocol are specifically developed for environments with illumination levels of 100 – 500 lux, like industrial halls. The sensor integrated into the module can vary according to application requirements. As an example, in our work, we use a temperature and pressure sensor and an accelerometer. A bright flash from a smartphones build-in LED activates the module. The module takes measurements and sends the result in form of an optical data signal, which is then received by the smartphone camera. This technique is able to provide reliable communication despite low-power restrictions of energy harvesting. By using a smartphone this approach offers more convenience to a user and enables flexible deployment of the modules in industrial machinery.

Keywords: Visible light communication, autonomous sensor module, smartphone, energy harvesting, low-power

1. INTRODUCTION

Smartphones and tablets over recent years have become prevalent in manufacturing environments, offering faster access to critical information [1]. They are used for data acquisition, monitoring and representation of factory statistics in intelligible format. Internet of things (IoT) devices and wireless sensor nodes embedded in industrial machinery are common sources for such data. Wireless sensor modules, which can work autonomously by harvesting energy from indoor ambient light have found numerous applications in condition monitoring and process automation [2]. They spend most of the time in standby mode and are activated periodically by an external or internal signal to send the measurement data. Generally, such devices use radio-frequency (RF) technology for communication. However, there are manufacturing environments, where optical communication is preferable, since it is more secure and robust to strong electromagnetic fields (e.g. nuclear or refinery plants). Most smartphones available on the market can be employed for bi-directional visible light communication (VLC) using its flashlight to transmit data and a built-in camera to receive the optical signal.

In [3] we introduced a concept of an optical autonomous sensor module, which takes measurement and emits data using a white LED when woken up by an external optical signal. In order to provide the power autonomy, the module is supplied with energy from an embedded solar cell. Several such modules can comprise a chain or a network with the smartphone as an end terminal, controlled by a user. The device dimensions tend to be as small as possible; however, they are restricted by the size of the solar cell required for their power needs. Therefore, the miniaturization tendency limits the amount of the available power of the module and consequently, the LED's intensity. According to the current conceptual design the sensor module has a size of 30 x 10 x 5 mm (Figure 1).

In this study, we present the visible light communication between the designed low-power optical sensor module and a smartphone. In particular, we focus on the optical data reception and demodulation using a CMOS sensor of a smartphone camera. The complexity and originality of this work lies in the enhanced real time image processing algorithms for demodulation of low intensity light signals. We provide an analysis on how the camera parameters, such as exposure time, frame rate, and image resolution affect the resulting quality of the image data. Furthermore, we introduce accurate time management methods for real time data processing. We then evaluate the communication methods by looking at the scaling between the data rate and the distance between transmitter and receiver.

*uliana.dudko@ita.uni-hannover.de; phone +49 511 762 18157; fax +49 511 762 4007; ita.uni-hannover.de

Smart Photonic and Optoelectronic Integrated Circuits XXI, edited by Sailing He, El-Hang Lee, Proc. of SPIE Vol. 10922, 109220I · © 2019 SPIE
CCC code: 0277-786X/19/$18 · doi: 10.1117/12.2506777

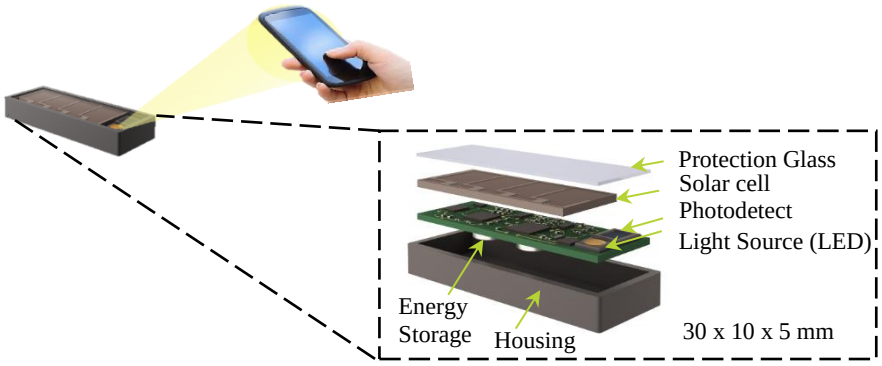


Figure 1. Design of an autonomous sensor module communicating with a smartphone by means of visible light.

Section 2 begins by laying out the theoretical principals of light signal reception by a CMOS camera sensor and explores the related work to the LED-to-smartphone communication. Section 3 provides an overview on the system description and employed communication protocol. Section 4 is concerned with the proposed methods and algorithms for light signal demodulation. In section 5, we evaluate the performance of the designed communication system.

2. STATE OT THE ART

The idea of using a smartphone camera for visible light communication was first introduced in 2012 by Danakis et al. [4]. The authors proposed an algorithm for data reception, where the captured information is represented in the form of light and dark bands provided by the rolling shutter effect of the CMOS camera sensor. This idea made it possible to achieve data rates much higher than the camera frame rate (typically 30, 60 or 240 frames per second) since a change in the state of the LED is still detectable.

2.1 Rolling Shutter Effect

CMOS sensors have a method of image acquisition, which is called the rolling shutter effect. When using this method an entire image or a video frame is not captured at once but acquired by sequential scanning of the exposed columns of pixels (scan lines). The columns are merged together in order to form a single image. Although in photography the rolling shutter causes distortions of fast-moving objects, in VLC it is employed as the key mechanism of light signals recognition.

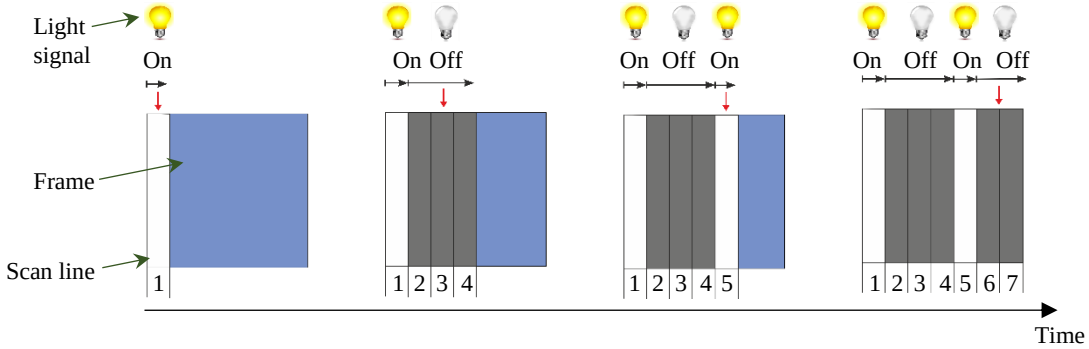


Figure 2. Rolling shutter effect. The width of the stripes corresponds to the time of the on and off states of an LED.

The width of the bright and dark bars is directly related to the time of the on and off states of an LED according to a communication protocol (Figure 2). The pattern of bright and dark stripes can be translated into binary data. Greater amount of scan lines or higher image resolution leads to higher data rate. Therefore, the recognizable transmitter signal frequency is defined not only by frame rate but also by image resolution. Theoretically, the upper limit of the data rate can be achieved when the width of a scan line is one pixel and it corresponds to one state of an LED. In practice, the neighboring scan lines affect each other's brightness resulting in hardly distinguishable low and high brightness levels. For this reason, higher data rates require complicated thresholding algorithms.

By employing the rolling shutter effect for the communication, we must consider the fact that the whole data sequence will not necessarily fit into one frame. In this case, the data is captured in several frames, but some information between the consecutive frames can still be lost. For this reason, the data must be split into several blocks. The transmission time for one block must be shorter than the total capture time for a frame. Thus, at least one block is visible within one frame and can be completely decoded.

2.2 Software Implementation and Related Works

The main procedure for the image processing by a smartphone app introduced by Danakis et al. [4] has become a reference for all the other following works. The general stages depicted in Figure 3 are present in all studies. However, the methods and software implementation of these stages differ in order to improve communication performance. The studies [4]–[6] are focused on applications where a transmitter is represented as an LED light bulb, which illuminates the whole room and has a light distribution close to uniform. Duque et al. in several of his studies [7], [8] conduct the research on LED-to-camera communication, where an LED is a part of an IoT device or a sensor node and its brightness is concentrated only in a small spot of the image.

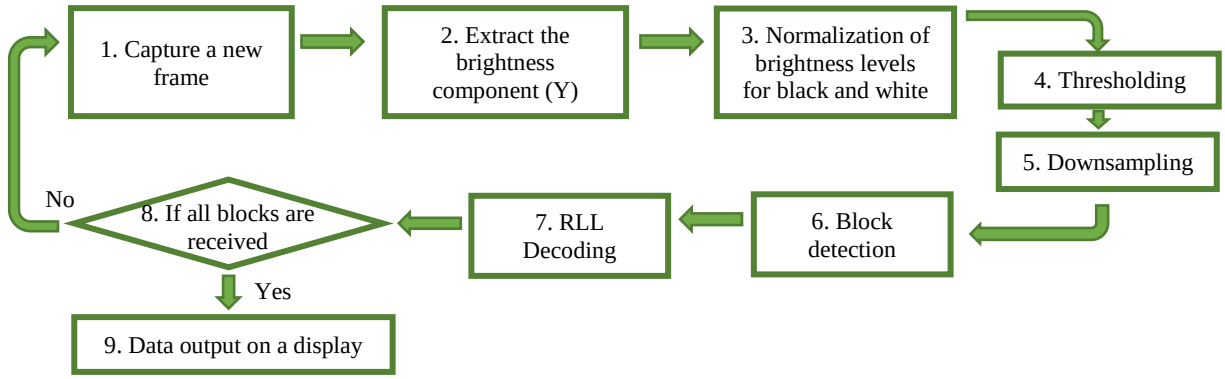


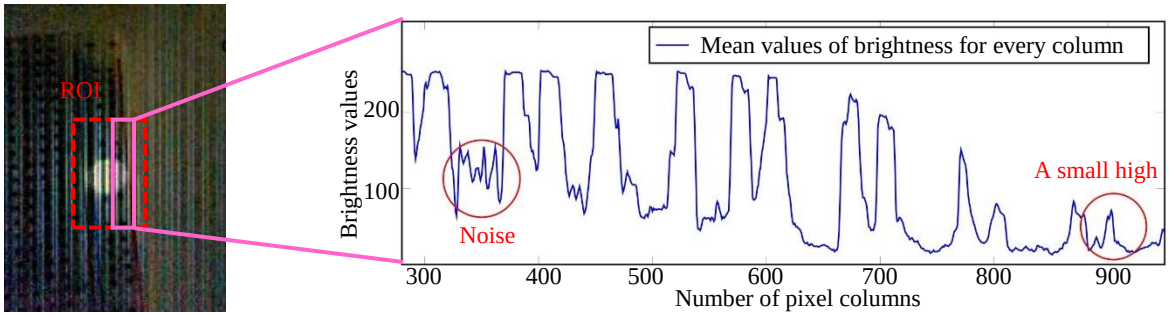
Figure 3. The general steps for decoding workflow.

When the preview mode of the Android camera application is enabled, the resulting images are stored in the buffer. The YUV image format (or Y-Cb-Cr for iOS [6]) is the most appropriate one since all information aside from the Y component can be disregarded, as the Y or "luma" component carries all luminance information and is used to distinguish between the white and black bands. In [7] Duque et al. consider the usage of other image formats and confirm that uncompressed YUV_420_888 is the most relevant format for the current task. In the first step, the Y component is converted to a two-dimensional data array. The preview image then presents a matrix of *numbers of scan lines* x *brightness value in the range from 0 to 255*.

In step 2, matrix is transformed into a one-dimensional array, where the luminance values of a scan line (pixels column) are normalized. The challenge of this step is that the image is not homogeneous in terms of brightness, some parts of the image are darker than others due to receiver instability. This happens even in the case when the transmitter is an LED light bulb. In [4] for normalization, the authors use a 3rd degree polynomial regression, which significantly increases the computational complexity for real time processing. In order to minimize it, the authors of [5], [6] instead of applying polynomial regression simply compute the average of every pixels column and do not consider the brightness inhomogeneity. In [7]. Duque et al. propose a real-time algorithm for region of interest (ROI) detection for small LEDs (Figure 4.a). For every frame the algorithm searches a bright spot of the transmitter and operates reliably even when image capturing is unstable due to user's hands movements. Since in our study a small LED is used as transmitter, we base our ROI detection algorithm on the idea of Duque et al. We modified this algorithm to make it suitable for our system and present the clear explanation in Section 4.2.

In the third step, a threshold is applied to the data sequence to determine low and high brightness levels. For this purpose, we have to define the threshold level. In [8] and [6] this procedure is implemented by computing the average values of the one-dimensional array obtained in step 2. In [5] this value is set empirically and in [4] it is equal to 0 as a result of polynomial regression. When using a small LED, this step is very important, since even in the area of detected ROI the luminous levels are highly inhomogeneous. Therefore, the same threshold value cannot be applied to all parts of the image

(Figure 4.b). In our study, we compute the threshold value, which dynamically changes depending on brightness levels over the data array (Section 4.1). This thresholding method was specifically developed for weak LED transmitters, whose brightness is restricted by the energy harvesting. Our novel thresholding approach is robust to background noise and provides a low signal-to-noise ratio (SNR) requirement.



(a) (b)

Figure 4. a. The region of interest (ROI) is the LED spot marked by the red frame. b. The challenge of thresholding: similar amplitude of noise and a small high level.

After applying the threshold, low and high brightness levels represented as a sequence of zeros and ones must be converted into the bit symbols according to the communication protocol. This step is called downsampling. Next, we recover the block order and then translate the run length limited (RLL) code to the initial data (see Section 3.2). Finally, if all blocks are obtained the reception is stopped and the data is shown on the display. Otherwise, we capture a new frame and repeat the image processing algorithm.

In order to operate in real-time without information loss, the frame processing time has to be lower than *1/frame rate*. For instance, for capturing with the frame rate of 30 fps the processing time is approx. 33 ms and for the frame rate of 240 fps, it is only 4 ms. If the processing time is longer than 1/frame time, the message needs to be sent more often to be able to receive all blocks successfully. Every time a frame is captured, the image data is acquired in the camera background thread. The efficient time management can only be achieved by allocating an image buffer. This buffer adds the next image to the queue (usually limited to 5) if the processing of the current frame takes too long [7]. In the earliest implementation [4], the authors measured the processing time of around 182 ms, which lead to the frequent frame skip. In the further work [5] for 30 fps the time of approx. 33 ms was achieved by minimizing the algorithm complexity. Image processing for [8] took only 18.4 ms since only the pixels in ROI were considered. The authors of [6] succeeded in implementing the real time processing with a frame rate of 240 fps (ca. 4 ms per frame) without frame skip, however, they do not present any time efficiency analysis. We assume they were using lower image resolutions, and therefore processing less data per frame.

The algorithm we present in this paper differs from the previous works since we consider the low-power restrictions of the transmitter, which is the autonomous sensor module that we design. When the transmitter brightness is significantly limited, the SNR is low even at the distances of several centimeters. Despite the requiring complicated data analysis, whole image processing is still able to operate in real-time for 30fps with processing completing on average in less than 33ms.

3. COMMUNICATION SYSTEM

In this study, we focus on line-of-sight (LOS) communication between the sensor module and a smartphone. We consider communication in one direction only. The sensor module transmits the measured data in form of light pulses according to the modulation described in 3.2. Commonly the data of one measurement from the most typical sensors (temperature and pressure sensors, magnetic switches, accelerometers, etc.) takes max. 3 bytes. Each byte is designated with a block number to reconstruct the data order on the receiver. The smartphone camera captures an image and demodulates the light signals repeatedly until all the blocks are received and the message is complete.

3.1 System Description

Transmitter. The sensor module is powered from a solar cell and therefore, the supplying energy is limited. Due to these limitations, the module consists of ultra-low-power electronic components which limit the performance of the

communication and therefore the bandwidth to approx. 10 kHz. The transmitter in the sensor module is implemented as a warm white light LED (LM301Z) with dimensions of 3 x 3 mm**.** The LED is controlled by a microcontroller through a MOSFET. The module operation voltage and current are 2.8 V and 9 mA respectively. The embedded LED consumes the major amount of current, which is 8 mA, while the electric circuit of the module requires only 1 mA in active mode. Therefore, the power usage of the module is 25.2 mW. A major part of it is consumed by the LED and comprises 22.4 mW. The measured optical power of the transmitting LED is 7.4 mW, which is relatively low in comparison with typical LED lamp used for illumination purposes (several Watts).

Receiver. We use a Samsung Galaxy S7 (SM-G930F) smartphone with Android 7.0 operating system as a receiver. The rear camera of the smartphone has a maximum resolution of 12 megapixels with an aspect ratio of 4032 x 3024 pixels (px). The Galaxy S7 works with the Android application programming interface (API) level 24. This API supports the camera2 class, which allows fine tuning of camera hardware, such as exposure time, ISO value or frame rate.

3.2 Communication protocol

The On-Off keying (OOK) modulation combined with Manchester RLL coding is the most common and straightforward implementation of a communication protocol for VLC. We implemented this approach for the communication between two autonomous sensor modules, however, the results of the reliability tests were not satisfying (30% error rate at 10 cm distance). Therefore, we designed a protocol based on a modified version of Pulse Position Modulation (PPM) and 4B6B RLL code.

In pulse position modulation, the position of a pulse in the time interval is shifted to the one of possible required states. Different intervals between pulses are used for self-synchronization and error reduction during demodulation. We set the duration of a symbol to T_s = 0.5 ms and duration of a chip to T_{chip} = 0.1 ms. The pulse for one "1" logical state has always the second position within the T_s and for zero "0" it is always the forth. The start bit has pulses in both second and forth position of T_s (Figure 5). The significant difference of time intervals between pulses 0.2 ms and 0.6 ms explicitly identifies the sequence of "01" and "10" respectively. Time interval of 0.4 ms denotes the duplication of the last logical state. The interval of 0.1 ms stands for a start bit. The data encoded this way can be easily demodulated without using additional synchronization symbols, unlike [7], [5].

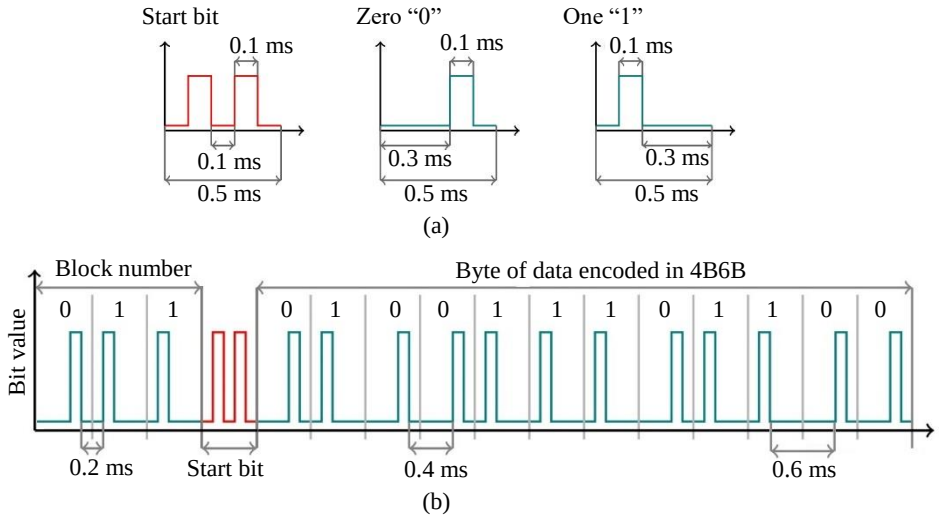


Figure 5. a. Duration of a chip and a symbol in PPM. b. Communication protocol

The use of RLL line codes helps to detect problems in data recovery and to avoid long runs of 1s and 0s that can possibly cause flicker. We have chosen 4B6B for data bytes encoding as recommended in IEEE 802.15.7 standard [9]. According to 4B6B, each 4-bit sequence is changed to DC balanced 6-bit corresponding to a special table [10], where counts of 1 and 0 are equal.

Prior to the start bit, we add the 3-bit sequence for a block number, which is necessary to reconstruct data order from a set of camera frames. Since a typical message from a sensor takes only 3 bytes, 3 bits for a block number is sufficient.

3.3 Camera Parameters

The camera parameters, which affect the quality of the image and, therefore, the decoding capability are exposure time, CMOS sensitivity and image resolution. The choice of parameters values has to be optimized in order to achieve the maximum distinguishability of the bright and dark bands. In this work, we select the parameters manually based on the experimental analysis and set them to fixed values prior to frame capturing.

Typically, the budget smartphones (e.g. Huawei Y II) have "legacy" hardware support, where most of the camera parameters are controlled automatically by the operating system and cannot be manually programmed. The programming of exposure time and sensitivity is only available for smartphones with "full" hardware support, which has API level starting from 21 and can work with camera2 class, like the Samsung Galaxy S7s that we use in our study.

Exposure time describes the period a scanline is exposed to light. For CMOS sensor, the scanlines readout is not performed simultaneously but with a constant delay for each consecutive pixel column. Therefore, there is an overlap between the scanlines. As a result, at long exposure time, the bright bands imbricate the black ones making the decoding impossible (Figure 6). When reducing the exposure time, the delay of scanline readout remains the same, however, the overlap between the consecutive columns decreases. For this reason, the exposure time should be as low as possible to sharpen the bright and dark stripes and thus enhance the distinguishability. For Samsung Galaxy S7 the lowest possible value is 1/8000 s.

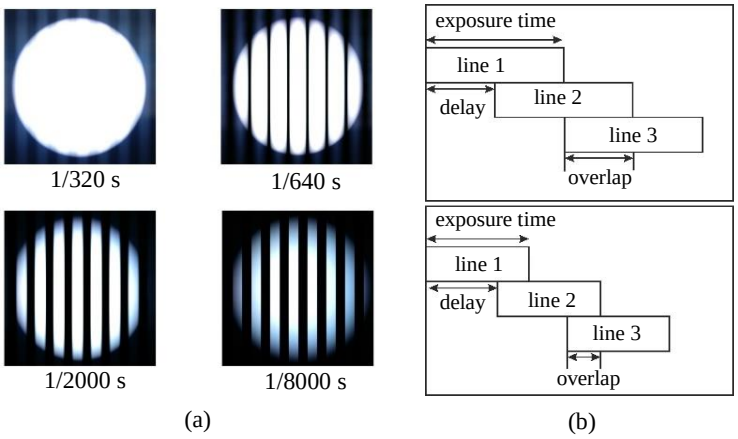


Figure 6. a. Photos of an LED flickering at 240 Hz for different exposure time. b. Scanlines overlapping in rolling shutter

Sensitivity. At short exposure time settings, the overall image brightness decreases making the bands recognition at farther distances more challenging. In order to increase the visibility of the bright stripes, a good light amplification is required. The camera sensitivity parameter determines the quantity of light needed for a good exposure. A higher sensor sensitivity enhances the image brightness and allows the recognition of bright bands from farther distances. However, this leads to higher noise, which is also not desirable due to random changes in the light intensity of pixels. As a result, there is a goal conflict between decreasing the exposure time and increasing the sensitivity. In our system, we empirically set the sensitivity value to 10000.

Image resolution determines the level of detail when analyzing the matrix of brightness values. The higher the resolution is, the more pixels will define the brightness of a band. Therefore, the higher accuracy and noise tolerance can be achieved. Since high resolution increases the processing time per frame, it should satisfy the time limits of the frame rate. For the frame rate of 30 fps in this study, we could manage to work with the highest possible resolution of 4032 x 3024 pixels with the processing time close to 33 ms. With the increase of the frame rate to 240 fps the resolution that can be processed in the required time decreases to 1280 x 720 px for the majority of smartphones.

The maximal possible data rate that can be achieved for 240 fps for the proposed communication protocol can be theoretically estimated as follows:

$$Bit\ rate = \frac{2^3}{\left(T_{chip\ min} \cdot N_{chips}\right) \cdot \left(N_{symbols} + Start\ bit + N_{block}\right)}, \quad (1)$$

where N_{chips} is the number of chips per symbol equal to 5 according to the protocol. $N_{symbols}$ stands for the number of symbols per one encoded byte in 4B6B coding, which is equal to 12 for 8 bits. N_{block} represents the number of symbols to encode a block number of a byte. Assuming that the minimal chip duration $T_{chip\ min}$ corresponds to 1 pixel, it can be calculated as follows:

$$T_{chip\ min} = \frac{1/frame\ rate}{horizontal\ resolution\ in\ pixels}. \quad (2)$$

As a result, for $T_{chip\ min}$ = 5.787 µs the maximal theoretically possible bit rate is 17 280 bits/s. However, in this case, the signal recognition works only for extremely short distances (< 2 cm). This condition is far from the realistic application scenario. Therefore, in our study, we focus on the communication implementation at a frame rate of 30 fps and increased symbol duration T_s in order to achieve the maximal feasible communication distance.

4. IMAGE PROCESSING

We base the image processing algorithm on the general structure explained in 2.2. After the frame is captured we start the processing with the ROI detection. In this step, we extract the brightest part of the image, which contains the transmitted signal. Once we shortened the data, the matrix is converted to a one-dimensional array by calculating the average luminance level of every column. Then, the thresholding algorithm is used to recognize a luminance value as a one (high) or a zero (low). Since the average brightness is not consistent over the image, we set the threshold dynamically according to the changing brightness levels instead of using a fixed value. The resulting binary array is then decoded in two steps. At first, we analyze the sequences of consecutive ones and zeros based on the communication protocol to obtain the decoded data in form of bits. In the second step, the chips are decoded according to the 4Bit6Bit coding algorithm. Finally, if all blocks are received the data is reconstructed using the block sequence number (BSN) and saved in an array to display the resulting message on the screen.

4.1 Region of Interest (ROI)

Algorithm. The ROI algorithm runs a loop over the whole image matrix row by row. We try to find the row with the maximum number of bands (the ROI origin) by counting the number of bright and dark bands. For this purpose, we analyze the luminance values of the pixels, which are in the range from 0 to 255. Here, 0 corresponds to the luminance of the darkest band and 255 denotes the luminance of the brightest band. To reduce the computation time every 8th pixel (Δx) in every 25th row (Δy) are considered (Figure 7).

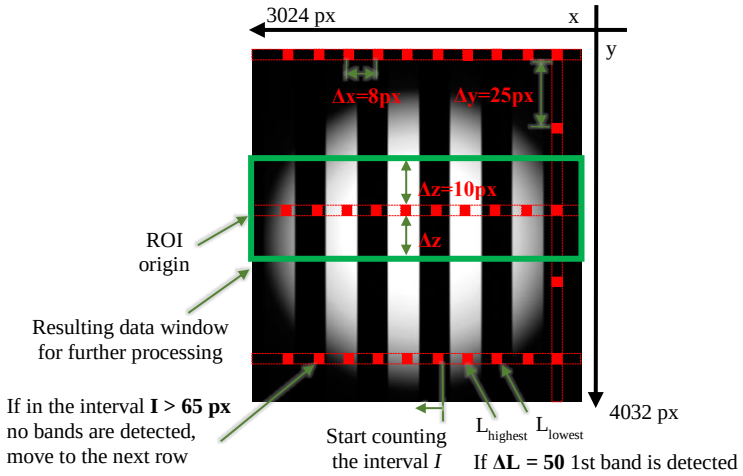


Figure 7. An image frame with an LED spot. The green window is a result of the ROI detection algorithm, which is taken for further processing.

When analyzing a row, we detect a band if the difference between the lowest and the highest luminance values is more than $\Delta L = 50$. Once a band is detected, we start to count the interval $I = 65$ px looking for another significant change in luminance. This process repeats until in the interval I there are no bands found. In this case, we stop consideration of the rest of the row and go to the next one.

When all the rows of the image are analyzed, we know the position of the row with the maximum number of bands. This is the ROI origin, which contains the most amount of data. Additionally, 10 pixels (Δz) above and 10 below the origin row are taken in the further processing algorithm.

Parameters choice. The step of 8 px is calculated with regard to the narrowest interval in the communication protocol of 0.1 ms for the start bit. For the horizontal resolution of 3024 px at a frame rate of 30 fps, the time interval of 0.1 ms corresponds to a band of 10 px. Therefore, the chosen step of 8 px satisfies the requirement of $\Delta x < 10\ px$ in order to be able to recognize every single band. Since for the column analysis such level of detail is not required, we increased the interval between rows to 25 px.

The broadest band corresponding to the longest possible period 0.6 ms of the protocol takes 55 px in the image. Therefore, we set the interval $I = 65$ px. If the interval since last recognized band exceeds 65 px, that would certainly mean, that there are no more bands can occur in the current row.

The distinguishing luminous value between bright and dark bands $\Delta L = 50$ is set empirically. If ΔL is lower than 50, the random changes in the light intensity caused by noise affect negatively the detection of the bands.

4.2 Thresholding

Once the ROI algorithm has narrowed the signal area to the two-dimensional array of 20 x 3024 px, the mean values of every column are calculated. As a result, we have a one-dimensional array as an input data for thresholding, where we decide if the luminance value should be considered as a one or a zero. In a simple implementation, we could set the threshold to 128 as the mean value between 0 and 255, so the values below and above 128 would be decoded as 0 and 1 respectively. However, due to inhomogeneous distribution of the LED light intensity, this would be incorrect for the side parts of the ROI, where the luminous values for high state are even smaller than 100. With the increase of the distance between a transmitter and receiver, this method would not recognize any bright bands at all. For this reason, in the proposed algorithm we set the threshold value for every high state (bright band) individually.

The thresholding algorithm can be divided into two stages. At the first stage, we go over the data array searching for every high state. Once a high state is detected, we save the position in pixels of the start and the end of the high interval. Then we calculate the threshold as a mean value of maximum and minimum for every high. This process is repeated until the end of the data. Thus, we form an additional list with the threshold values and the intervals where it must be applied. At the second stage, the data array is analyzed again by applying the threshold to the luminance values in the corresponding intervals. As a result of the thresholding, we obtain a binary array with the sequences of ones and zeros, representing the low and high states of the signal (Figure 8). This is, however, not the final data, since the binary array still must be decoded with regard to the communication protocol.

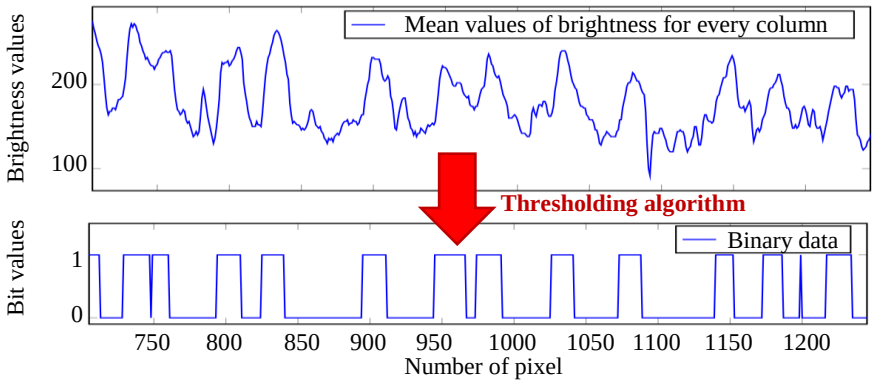


Figure 8. The result of thresholding algorithm.

The first stage of the algorithm contains the detection of high states, which must be additionally clarified (Figure 9). The workflow is based on two procedures: searching for an increase (rising edge) and for a decrease (falling edge) in the light intensity. During the search for an increase, the current value in the data array is saved if it is the lowest one (minimum). If the current value is higher than the sum of the lowest value and the distinguishing value (condition 1), the increase is found. Then we calculate the threshold, save the position of the current value and update the distinguishing value *D*. After that we search for a decrease. Here we save the current value if it is the highest one in order to define a peak of the light intensity. If the current value is less than the sum of the highest value and the distinguishing value *D*, we detect a decrease (condition 2). In this case, the distinguishing value must be updated and the search for the new increase starts again. The example of the first stage of thresholding is shown in Figure 10.

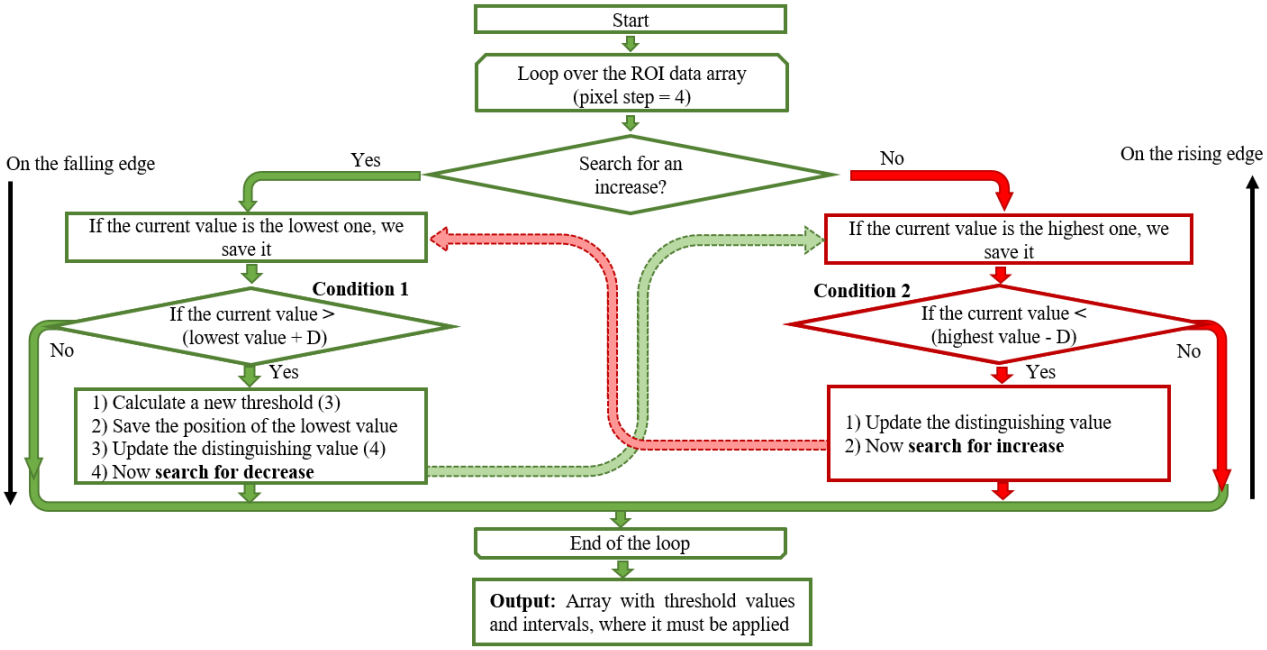


Figure 9. The simplified block diagram of the high state detection and calculation of its threshold value.

It is noteworthy to mention that the new threshold value is calculated every time the first procedure determines an increase. The borders of the interval are defined by the positions of two low values: the previous lowest value (L_{old}) and the new lowest value (L_{new}) as a start and the end of the interval respectively. The threshold is calculated as follows:

$$T_i = \frac{(max\ (L_{old}, L_{new}) + highestValue)}{2}, \tag{3}$$

where *i* is a number of the high state interval. The distinguishing value is a window, which represents the noise tolerance. This value is set dynamically as 30% of the difference between the highest and the lowest value:

$$D = (highestValue - L_{new}) \cdot 30\%\,. \tag{4}$$

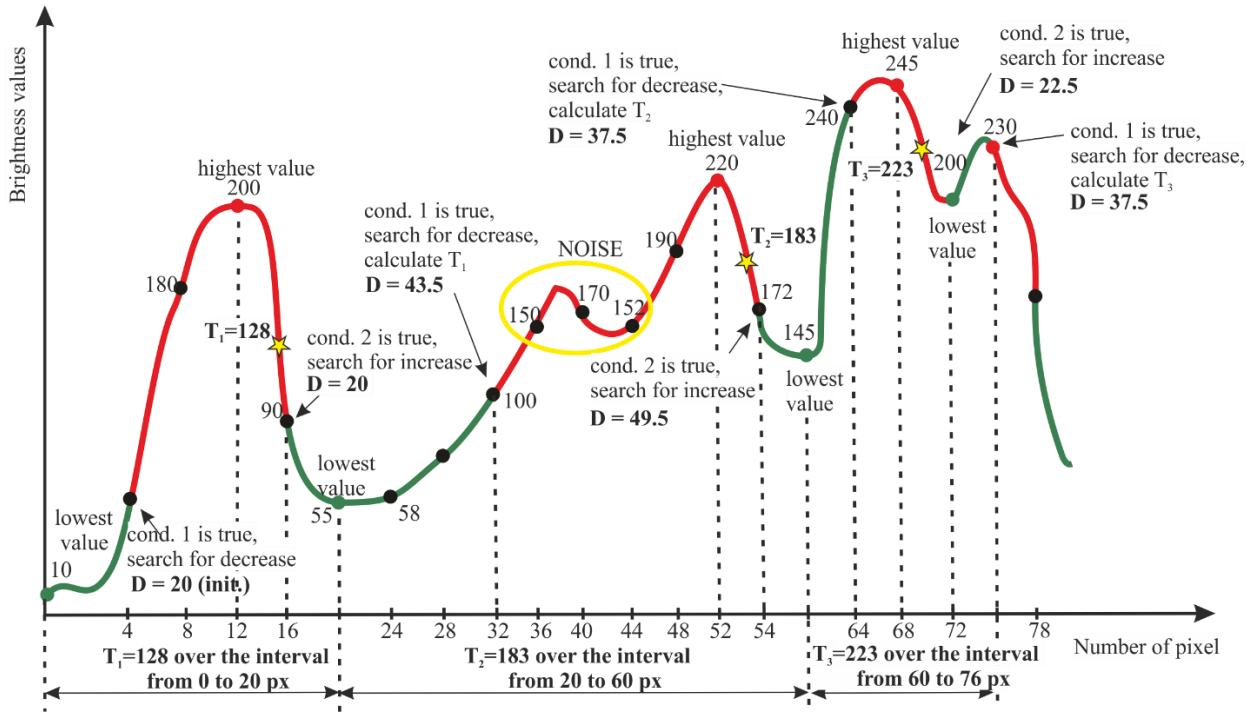


Figure 10. The example of the first stage of the thresholding for a data array sample.

The procedures of searching for the rising and falling edges are repeated until the end of the data array. Then at the second stage of the thresholding using the list of the threshold values and the intervals, we receive the binary array for the further decoding.

5. EVALUATION

Several evaluation tests were performed to estimate the communication reliability, the data rate and the impact of the surrounding illumination as a noise factor. The measurements were carried out for two different scenarios: in an opaque enclosure with a fixed illumination and in industry hall conditions. During the tests, we found out, that the background of the transmitter influences the data rate. Therefore, we provided measurements for black and white backgrounds to assess this impact.

The measurement setup consists of a slide rail with a centimeter scale to perform the measurements at a fixed distance. The smartphone holder is mounted on the sliding carrier. The transmitting sensor module is positioned in line-of-sight to the smartphone camera on a white or black background.

Data rate. For the measurements in the opaque enclosure, we use an LED lamp to control and maintain the desired illumination level without the presence of ambient light. For the first experiment, we set the illumination level to 500 lx, which is recommended for the areas where continuous work is carried out [11]. We use a lux meter to measure the illumination at the position of the transmitting LED. In order to estimate the data rate in the realistic environment conditions, additionally, we carry out the same experiment in industry hall with a similar average illumination of 500 lx to perform the comparison.

The transmitter continuously sends 3 bytes, which simulate the sensor measurements. In the smartphone app, we measure the *throughput* as a time required to receive 3 bytes including doubled blocks. Typically, several frames are necessary to reconstruct the initial byte order, which takes more time to receive an original message. This time is also calculated and is defined as a *goodput*. We estimate the data rate as a ratio of the number of information bits sent to the reception time for throughput and goodput respectively. The results of the measurements in the opaque enclosure and the industry hall for white and black backgrounds are depicted in figure 11. Each measuring point corresponds to the average of three error-free runs for each distance.

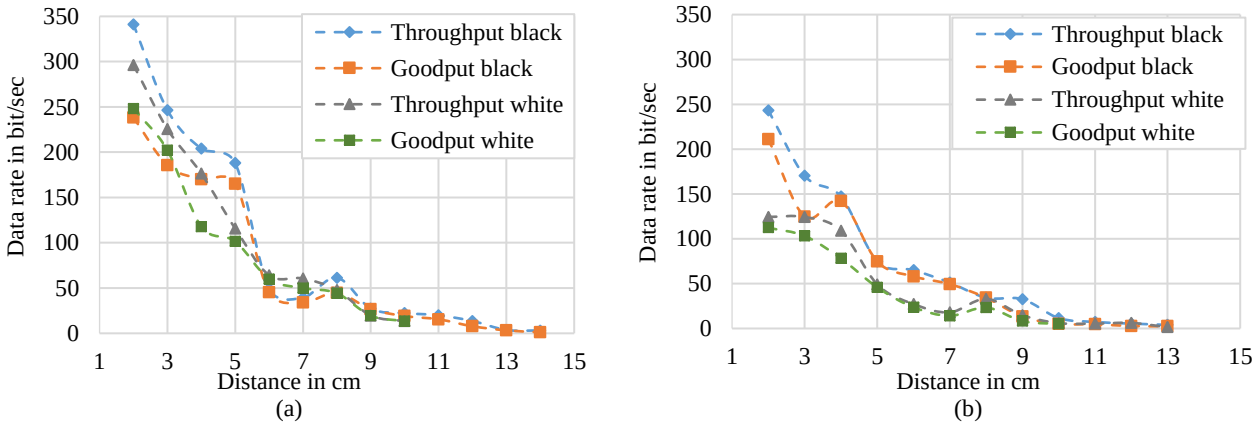


Figure 11. a. Data rate measurements in the opaque enclosure at 500 lx. b. Data rate measurements in the industry hall at an average illumination of 500 lx.

The maximum throughput of 341 bit/s and maximal communication distance 14 cm was achieved for the measurements in the opaque enclosure with the black background. The lowest measured throughput time is 66 ms, which exactly corresponds to 2 frames, required for 3 bytes reception. Compared the results for the black and white backgrounds, it can be seen that data rate for a black one is from 50 to 100 bits/s higher for the distances up to 5 cm. The white background affects the data rate negatively due to high surface reflectance, which decreases the distinguishability of the bright bands of the image.

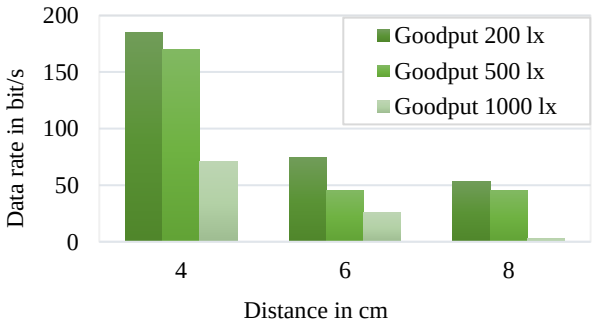


Table 1. BER at different distances.

Distance	Errors	BER
4 cm	0	0 %
6 cm	1	0.04 %
8 cm	1	0.04 %

Figure 12. Data rate measurements in different illumination conditions.

Environmental impact. We performed the measurements of goodput for illumination of 200 lx, 1000 lx and compared them to the measurements at 500 lx (Figure 12). It can be observed that the brighter the environment is illuminated the more negatively it affects the data rate. Therefore, the data rate in the industry hall is generally lower due to the noise coming from unpredictable changes in sunlight illumination and additional reflectance of the nearby objects.

Bit error rate. In order to analyze the communication reliability, the transmitter sends 2500 bits continuously and the bit rate is counted on the receiver. The tests were carried out at the illumination of 500 lx in the opaque enclosure at the manageable communication distances of 4 cm, 6 cm, and 8 cm. At 4 cm all bits were received correctly. At two farther distances, one error bit was detected for each distance respectively (Table 1). For the total transmitted sequence, the bit error rate resulted to 0.04%. Overall, these results indicate that the communication between the sensor module and the smartphone is reliable even at the increased distances.

Processing time. In these series of tests, we analyzed the average time, which is required to perform the proposed image processing algorithm for one frame. We send 100 packages 15 times at a distance of 5 cm and with an illumination level of 500 lx. The average processing time per frame is saved chronologically for every run. The histogram (Figure 13.a)

shows the distribution of the samples over the average processing time per frame. The shortest average processing time per frame is 31 ms and the highest is 40 ms. The arithmetic average of the first 10 runs is 32.4 ms. Such behavior shows that in the first 10 runs the algorithm is efficient and capable to keep up with the camera frame rate (processing time ≤ 33 ms). When the run number exceeds 10, the processing time rises, so that the average for all 15 runs becomes 34.2 ms and communication efficiency decreases.

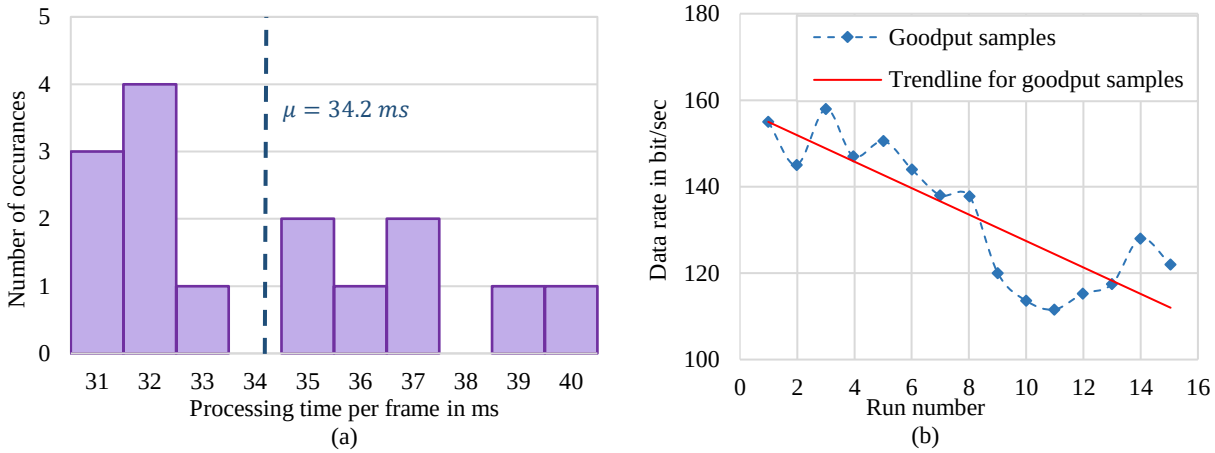
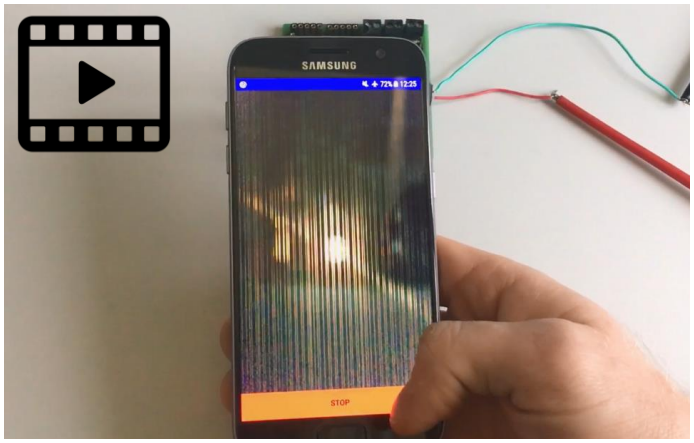


Figure 13. Frequency of the occurrence of different processing times per frame; b. Data rate dependence on the run number

With multiple runs, the temperature of the smartphone increases. This happens due to continuous computation during the image processing. Figure 13.b shows the impact of the run number on the average goodput. It can be observed that the data rate decreases with the increased run number after the 8th run.

In the real application after one successful data transmission, the designed autonomous sensor module must be recharged. Depending on the size of a solar cell this process takes from 2 to 4 minutes. This time would potentially be enough for the smartphone to cool down to prevent the decrease of communication efficiency.



Video 14. Demo Video of visible light communication between a sensor evaluation board and the smartphone using its camera: http://dx.doi.org/10.1117/12.2077147.1

Video 1 demonstrates the working communication between a sensor evaluation board and the smartphone. White frame in the middle of the evaluation board outlines the size of the designed module. In the video instead of real sensor data we use tree letters “ITA” as 3 data bytes for the test and demonstration purposes.

CONCLUSION

We demonstrated an enhanced image processing algorithm specially developed for the communication between an autonomous sensor module and a smartphone camera. In this study, we prioritized the communication reliability rather than the data rate, due to the challenge of data transfer for the transmitter powered by energy harvesting. We developed a thresholding algorithm and improved the region of interest (ROI) detection to increase the readability from the low-power transmitting LED of the module. Although the theoretical maximum of the data rate was not achieved, the design evaluation proved the reliability of the communication with the bit error rate of 0.04 %. We reached the maximal communication distance of 14 cm and the maximal goodput of 248 bit/s. The obtained results have shown that the developed system can be successfully employed in industrial applications with environmental illumination of approx. 500 lx with the maximal goodput of 112 bit/s. The achieved data rate is sufficient enough for the initial application purposes since the common length of the transmitted message from the sensor module does not exceed three bytes. The possibility to interpret the weak optical signals in industrial environments will offer new applications for wireless optical communications in the future.

ACKNOWLEDGMENTS

We gratefully acknowledge the generous support of the Lower Saxony Ministry for Science and Culture (Germany) within the framework of the “Tailored light” project. Additionally, the exceptional work of Frederik Berg (B.Sc.) was also invaluable to the progress of this research. His contribution to this project and all of his efforts have been highly appreciated.

REFERENCES

[1] Daymon Thompson, "Smartphones and tablets in manufacturing" Control Engineering, 8 October 2015, <https://www.controleng.com/articles/smartphones-and-tablets-in-manufacturing/> (09 January 2019).

[2] Erdelj, M., Mitton, N. and Natalizio, E., "Applications of industrial wireless sensor networks," Industrial Wireless Sensor Networks: Applications, Protocols, and Standards, 1-22 (2013).

[3] Dudko, U., Overmeyer, L., "Intelligent Photoelectric Sensor Module Utilizing Light for Communication and Energy Harvesting," DGaO Proceedings (2017).

[4] Danakis, C., Afgani, M., Povey, G., Underwood, I. and Haas, H., "Using a CMOS camera sensor for visible light communication," Globecom Workshops (GC Wkshps), IEEE, 1244-1248 (2012).

[5] Ferrandiz-Lahuerta, J., Camps-Mur, D. and Paradells-Aspas, J., "A Reliable Asynchronous Protocol for VLC Communications Based on the Rolling Shutter Effect," GLOBECOM, IEEE, 1-6 (2015).

[6] Schmid, S., Arquint, L. and Gross, T. R., "Using smartphones as continuous receivers in a visible light communication system," Proceedings of the 3rd Workshop on Visible Light Communication Systems, ACM, 61–66 (2016).

[7] Duque, A., Stanica, R., Rivano, H. and Desportes, A., "Unleashing the power of LED-to-camera communications for IoT devices," Proceedings of the 3rd Workshop on Visible Light Communication Systems, ACM, 55–60, (2016).

[8] Duquel, A., Stanica, R., Rivano, H. and Desportes, A., "Decoding methods in LED-to-smartphone bidirectional communication for the IoT," LIFI Congress (GLC), 1-6 (2018).

[9] Rajagopal, S., Roberts, R. and Lim, S.-K., "IEEE 802.15.7 visible light communication. Modulation schemes and dimming support," IEEE Communications Magazine 50, no. 3, 72–82 (2012).

[10] Jones, E.V. and Zhu, S., "Data sequence coding for low jitter timing recovery," Electronics Letters, 337-338, (1987).

[11] Standardization IO. "Lighting for Work Places Part 1: Indoor (ISO: 8995-1: 2002 E)," International Commission on illumination, Vienna (2002).