Research on Debugging Interaction of IoT Devices Based on Visible Light Communication

Jiefan Qiu[1(✉)], Chenglin Li[1], Yuanchu Yin[1], and Mingsheng Cao[2]

[1] Zhejiang University of Technology, Hangzhou 310023, China
qiujiefan@zjut.edu.cn
[2] University of Electronic Science and Technology of China, Chengdu 610054, China

Abstract. Wireless sensors normally deployed in inaccessible areas. Once the wireless communication of sensor node fails, the lost node cannot be repaired by debugging interaction which depends on this communication. Visible light communication (VLC) is a supplement of the traditional radio-wave wireless communication by and needs a dedicated device or module. Thus, VLC is hard to be applied in low-cost sensor nodes. We implement a hybrid duplexing debugging interaction system (HDDIS) based on VLC in general smartphone and sensor node. The smartphone is taken as a VLC gateway to send debugging codes to the sensor node. In order to improve the VLC transmission rate of debugging codes, we propose a novel debugging code compression method for data source and channel coding. With regard to the data source, we analyze the binary instructions and reuse opcodes and leverage a bit-mask technique to compress operands. The average compression rate of binary instructions reaches 84.11%. For channel coding, we optimize the dual-header pulse interval modulation (DH-PIM) and propose the overlap DH-PIM (ODH-PIM) by introducing a LED half-on state. The LED half-on state can improve the representation ability of each symbol. The experiment results illustrate that our modulation reduces transmission time by 10.71% compared with DH-PIM.

Keywords: Sensor nodes · Debugging · Visible light communication · Instruction · Modulation

1 Introduction

The wireless sensor network is composed of decentralized and self-organized sensor nodes. Once the network deployment, it is difficult to maintain the nodes, especially the nodes are deployed in the inaccessible area, such as in a museum show in cases [1]. The reparation and debug of sensor nodes depend on the built wireless network infrastructure.

However, some failures of the sensor network are related to unreliable wireless communication. If the sensor nodes on the critical network path are temporarily or permanently lost, and they cannot be repaired and work, the performance of the network system is possibly degenerated, and even invalid the entire system of the network. Meanwhile, it is difficult to employ an alternatives interaction method to complete a debugging node

H. Gao et al. (Eds.): CollaborateCom 2020, LNICST 349, pp. 132–147, 2021.
https://doi.org/10.1007/978-3-030-67537-0_9

due to the limited resources. In practice, developers have to abandon these lost nodes and replace them by backup nodes to which the tasks running on lost nodes are migrated [2–4]. The problem is that developers rely on experience and intuition to determine whether the tasks should be migrated, and that is usually inaccurate. In addition, developers cannot figure out the reason for the node lost and prevent node loss. Even if task migration is successful, the current status of a task is not stored. That finally affects the performance of entire networks. To this end, we hope to find an alternative debugging interaction method without modifying the local hardware of sensor nodes.

In recent years, with the development of visible light technology, the general sensor node, and smartphone equip various visible light components such as LED, camera, and ambient light sensor. These components provide a potential capacity for visible light communication (VLC). More and more traditional sensor devices have begun to use VLC as a complement of radio wave communication (RWC) for traditional wireless communication. VLC adopts unidirectional propagation different from the traditional RWC with broadcast way. This kind of propagation is not easy to be intercepted or eavesdropped and improves transmission security. In addition, VLC does not occupy the bandwidth resources of current sensor networks and affects the wireless communication between sensor nodes [5]. Therefore, we try to apply VLC in debugging interaction.

However, current VLC requires dedicated visible light components or modules. For example, Fan L et al. applied VLC to the access control system. Therefore, it is necessary to modify the existing system by adding a special VLC circuit. The circuit composes of the photodiode, amplifier, comparator, and MCU [6]. In order to improve the transmission rate and solve the problem of multi-channel transmission, Wang Y et al. designed a duplexing indoor communication system based on visible light RGB-LED. The duplexing communication system employs RGB light emitting diodes, low pass filters, electronic amplifiers, and photodiodes [7]. VLC components are sensitive to environmental optical noise. Adiono T et al. proposed a solution to reduce the influence of optical noise. This solution requires an analog filter which is taken as the front-end receiver [8]. Such extra hardware modification increase cost and not suitable for deployed sensor nodes, which is a bottleneck of the VLC application in IoT.

In this paper, we implemented a hybrid duplexing debugging interaction system (HDDIS) based on VLC applied in general sensor nodes. We tap the potential capacity of VLC components in smartphone and sensor nodes and regard the smartphone as a VLC-based gateway for debugging node in field. The debugging interaction uplink mainly transmits debugging information from the malfunction sensor node. It leverages the smartphone's optical camera as the signal receiver, and the node's LED as the signal transmitter. The debugging interaction downlink mainly transmits updating codes to fix nodes. It leverages the smartphone's flashlight as the signal transmitter and the node's ambient light sensor as a signal receiver. In this system, since the data transmitted by downlink are relative to binary code for an update, and its size is large compared with debugging information data. In addition, the general ambient light sensor equipped in a node has a serious light sensitivity latency, which prolongs data transmission and lower transmission rate.

To this end, we have proposed a set of compression schemes in source and channel coding. In source coding, we respectively compress opcode and operand of one instruction. In practice, a large number of redundancies exist in opcodes. The same part of the opcodes of the instruction can be reused. In addition, a part of the operand, such as a base address, is also represented by bit-mask to further shorten the length of each instruction. In channel coding, we designed the overlap dual-header pulse interval modulation (ODH-PIM). DH-PIM is an isochronous pulse time modulation in which data are encoded as discrete time slots between adjacent pulses. The width of the low power is used to represent a symbol. We add a LED half-on state as an overlapping mark in DH-PIM. This mark realizes an overlap representation to compress two data into one symbol and further reduces the number of light pulses.

Besides, since the optical camera completes shooting in units of image frames, the interval at which the node LED sends data needs to be consistent with the optical camera's shooting time to transmit debugging information successfully. For this reason, we further propose a feedback frame synchronization scheme for rolling shutter. In this synchronization solution, the smartphone communicates with the node to coordinate signal transmission and reception synchronization. Real-time detection of the collected images to ensure synchronous communication.

Finally, we conducted a serial of experiments to verify our method from compression rate, bit error rate (BER), transmission time, and energy consumption. The experiment results illustrate our compression method reduce 80% of redundant codes. At the same time, the ODH-PIM owns the lower bit error rate than PWM, and the less transmission time than DH-PIM at the cost of an extra 17.36% energy overhead.

The rest of this paper is structured as follows: Sect. 2 shows relative work; Sect. 3 gives the overview of the HDDIS; Sect. 4 describes the design of instruction compression based on bitmask and ODH-PIM. Section 5 describes the uplink synchronization scheme. Section 6 describes experimental and evaluates our approach and Sect. 7 closes with a conclusion.

2 Related Work

Chen P Y et al. [9] introduced a mask-based compression method and used it in real-time embedded systems. Kumar R N et al. [10] proposed a method of combining a lookup table (LUT) and a mask, which increases the matching range of the lookup table. Early compression work [11, 12] used masks for code compression in embedded systems, which significantly improves the code compression rate. However, the above articles need to transfer some dictionaries or lookup tables, which increases the amount of data transferred. We expect to judge the type of data only when compressing and perform data compression without transferring the dictionary.

At present, intensity modulation/direct detection (IM/DD) is mainly used in VLC to communicate. DPIM modulation transmits data in units of groups, and Ghassemlooy et al. [13] applied DPIM modulation for the first time in the field of optical communications. Subsequently, Aldibbiat et al. [14] proposed double-head pulse interval modulation (DH-PIM) based on DPIM. Kaili Yao et al. [15] proposed a novel scheme using pulse position to improve the BER performance instead of amplitude of Carrier-less Amplitude

Phase. Agha Yasir et al. [16] specifically combined the PPM and pulse shape modulation, this scheme provided has improved in system performance along with computational complexity by increasing the bandwidth and the number of pulses. However, there are few modulations designed to compress transmission data.

CMOS sensors also have the function of receiving and converting optical signals into electrical signals, so many researchers have begun to study the role of optical cameras in VLC. T. S. Fuzile and R. Heymann et al. [17] implemented a three-color LED VLC system on the mobile terminal, but the data transmission rate was low due to the global shutter and the simple modulation and demodulation method.

In response to the above research work, we have designed a HDDIS based on visible light using the existing components on the existing nodes and the general sensor module of the Android device.

3 System Overview

To repairing the wireless communication lost node, we design HDDIS based on VLC. As shown in Fig. 1, in HDDIS, the smartphone is taken as VLC gateway, and send binary codes to a sensor node. The node downloads codes and preforms the repair process. Finally, debugging results is reported bake to smartphone. The two links use different sensor components and communication methods to achieve node debugging. The downlink uses traditional VLC technology to update and repair the sensor node. It uses a smartphone flashlight as the signal transmitter and the ambient light sensor on the sensor node as the signal receiver. The sensor node then feeds back the debugging

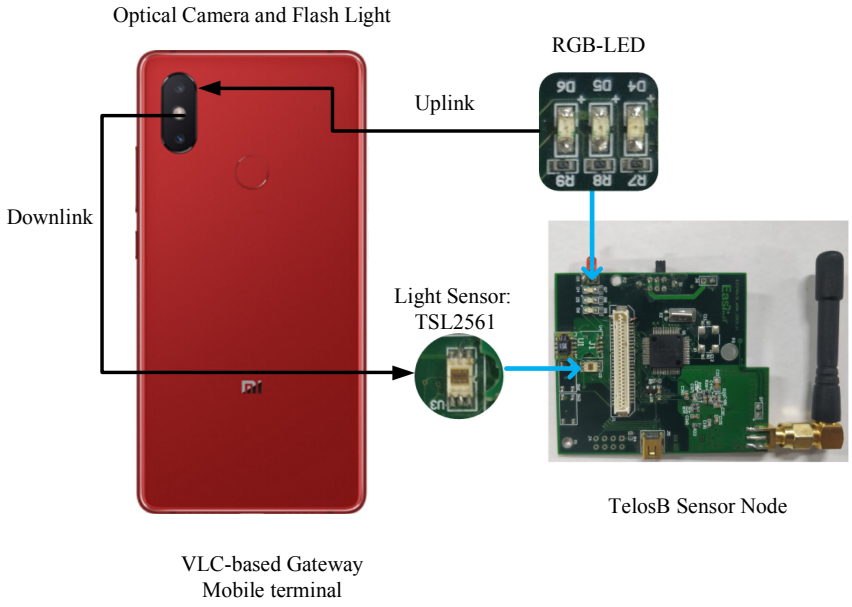


Fig. 1. Hardware interaction diagram

information to the smartphone from the uplink. In the uplink, the LED on the node acts as a signal transmitter, and the optical camera of the smartphone acts as a signal receiver using OCC to collect LED light information.

4 Downlink: Instruction Compression Based on Bitmask and Pulse Compression Method

The downlink transmits a large amount of binary data to repair the sensor node. However, the ambient light sensor on the node has low accuracy, resulting in a long transmission time of the data unit and a low transmission rate of the overall codes. To this end, we respectively optimize the instruction representation for source coding and the symbol representation for channel coding.

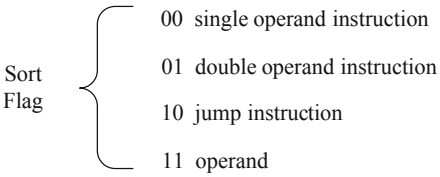
4.1 Compress the Binary Instruction

When transmitting the new code, we only transmit the modified function, which can effectively reduce the size of the transmitted data. These functions will be converted by the compiler into instructions for the next analysis, and finally into compressed binary data. In this paper, instructions are divided into three types: single operand instructions, double operand instructions, and jump instructions. Each type of instruction has the same part in the opcode. And single operand instructions, double operand instructions may occupy different numbers of bytes due to different addressing modes. Therefore, we have analyzed the instruction structure and remove some fixed opcode binary data, and further compressed the operand of multiple byte instructions. When instructions occupy multiple bytes, their operand will occupy following bytes of the first byte. The specific plan taking the MSP430 instruction set as an example is as follows:

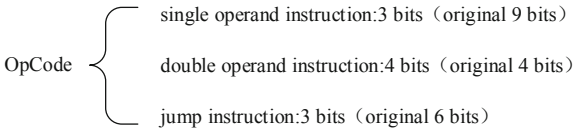
First, binary instructions will be divide into a lot of sequences. The opcode data in sequences will be compressed. The opcode encoding format is shown in Fig. 2(a). Sort flag divides sequences into four categories as shown in Fig. 2(b). In the Opcode segment, fix sequences are omitted and only changed sequences is transmitted. Figure 2(c) shows the number of transmitting bits in each kind of opcode. Then, the other parts are transmitted directly according to the original sequences.

Second, regarding the operand data of multiple bytes instruction, we considered a method of replacing the all-zero sequence to compress the data. The operand data encoding format as it shows in Fig. 3.

(a) First byte encoding format



(b) Value of "Sort Flag"



(c) The compression effect of opcode.

Fig. 2. Opcode compression coding diagram

Sort Flag	Pattern	Group Number	Non-zero Data
(11) 2 bits	2 bits	2 bits	4 bits/8 bits

Fig. 3. Operand compression coding diagram

The sort flag corresponds to that in Fig. 2(a), indicates that this is an operand sequence. The pattern means that different schemes are used for compression. The group number indicates the group number where non-zero data appear. And the non-zero data shows specific non-zero data content. Next, we introduce the schemes adopted by each pattern shown in Table 1. 16 bits of data are divided into four groups, each group has 4 bits.

Table 1. Four different pattern

Pattern	Value	Data size and number of groups
Pattern 1	00	-, -
Pattern 2	01	4 bit, 1 group
Pattern 3	10	8 bits, 2 groups
Pattern 4	11	8 bits, 2 groups

(1) Pattern 1:
Direct transmission of the original sequence;

(2) Pattern 2:
Dividing 16 bits of data into four groups, each group has 4 bits. The group number represents different groups, which means that specific 4 bits of non-zero data appear in the group (Fig. 4);

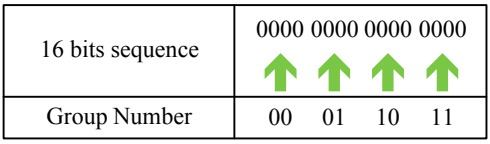


Fig. 4. Pattern 2 diagram

(3) Pattern 3:
Dividing 16 bits of data into four groups, each group has 4 bits. The group number represents a combination of different groups, which means that specific 8 bits of non-zero data appears in the combination;

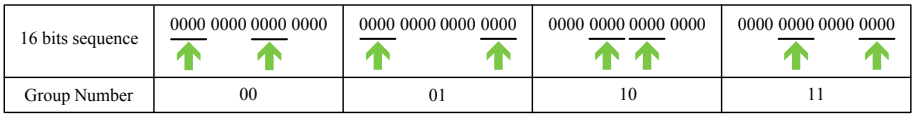


Fig. 5. Pattern 3 diagram

(4) Pattern 4:
Dividing 16 bits of data into two groups, each group has 8 bits. The group number" represents different groups, which means that specific 4 bits of non-zero data appears in the group (Fig. 6).

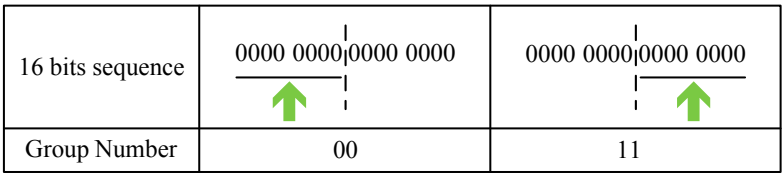


Fig. 6. Pattern 4 diagram

In some cases, the sequence transmitted includes 16 bits of data whose bits are all-zero. At this time, since two group number are consumed in pattern 4, group number can be set to 10 to indicate that this is a group of all-zero sequences. By analyzing the original binary sequence of the code, it can be found that the frequency of the sequence is not low.

Take the instructions in the MSP430 instruction set as an example as shown in Fig. 7. The "Push" instruction is a single-byte single operand instruction, so there is no need to consider additional operand compression. The "Call" instruction is also a single-operand instruction, but it occupies two bytes due to different addressing modes, so the operand of the second byte can be compressed.

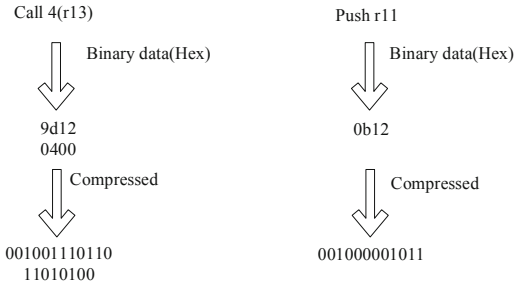


Fig. 7. Example of instruction compression

4.2 Introduce LED Half-on State for Compression Modulation

After the first step of compressing the binary instruction, a better modulation method can be designed to further compress the code. However, the commercial off-the-shelf device usually has low accuracy, making it difficult to achieve high-speed data transmission. Therefore, there is necessary to improve existing modulation methods to solve this problem.

Using the ambient light sensor, the Measuring process of light intensity needs a duration of light-on state. If we intend to shorten the duration, the sensor captures a new LED state different from LED On/Off state. We mark the new state as LED half-on state. Moreover, we designed the ODH-PIM based on the existing DH-PIM modulation method by adding LED half-on state. DH-PIM is an isochronous pulse time modulation in which data are encoded as discrete time slots between adjacent pulses. A symbol that encodes M bits of data is represented by k slots of low power and followed by one pulse of constant power, where $0 \leq k \leq L/2-1$ and $L = 2^M$. The pulse of constant power will last t_r or $2 * t_r$ for two numbers which are a radix-minus-one complement. For example, the number "0100" has 4 slots of low power and 1 slot pulse of constant power. The number "1011" is a radix-minus-one complement of "0100" and has 4 slots of low power and 2 slots pulse of constant power. A symbol of the DH-PIM usually contains 4 bits of data. However, the ODH-PIM can compress 8 bits of data into one symbol because two sets of data can be overlapped.

Because different sets of data have different time slot sizes when two sets of data pulses are overlapped together for modulation, and special pulses need to be used to separate them. At this time, the LED half-on state plays an important role. As shown in Fig. 8, given M = 4, after representing the contents of two sets of data, a time slot pulse is introduced as an order symbol, indicating the order of the two sets of data. And a set of inverted marks is used to determine which set of data needs to be complemented. In ODH-PIM modulation, the order flag indicates that the order of the two sets of data needs to be changed, and if there is no flag, the data set with a shorter period is first. Besides, there are four cases of the reverse flag.

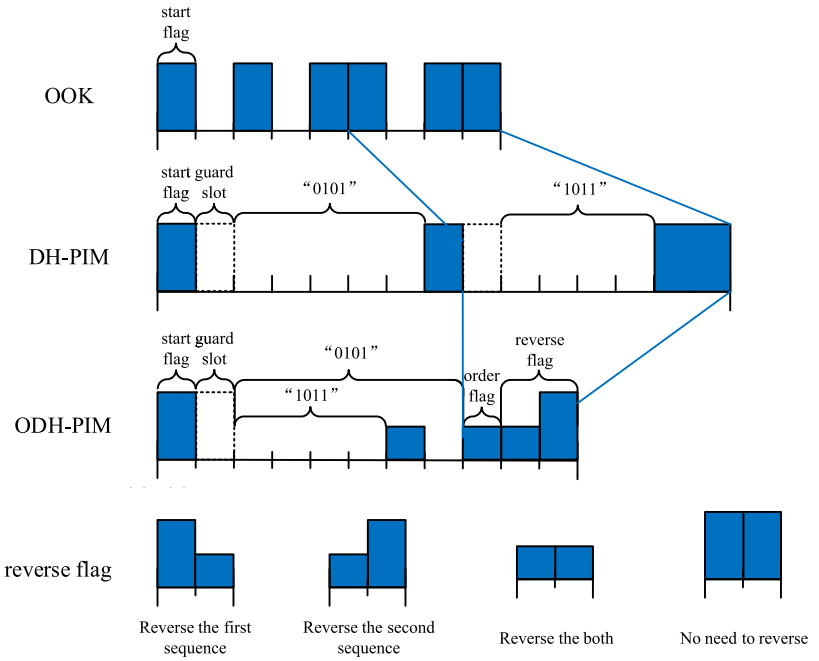


Fig. 8. Three modulation comparison. Bit series "01011011"

5 Uplink: Synchronization Scheme for Rolling Shutter Camera

In optical camera communication, synchronization has always been one of the most important issues. First, the sampling of the optical camera is performed randomly. It may occur during any symbol of the signal, so any symbol may be lost at any time. Secondly, the frame sampling interval is variable, which depends on the characteristics of the image sensor and optical channel. When the camera collects node LED stroboscopic information, the data symbols will be lost due to the lack of uniform turn-on time at the transmitter and receiver and the optical camera's unstable frame rate.

To this end, we propose a frame synchronization scheme for the rolling shutter camera. In this synchronization solution, the smartphone communicates with the node to coordinate signal transmission and reception synchronization. In the communication process, due to the smartphone's unstable frame rate, the data stripes' position in the collected image frame will shift. Due to the hardware device's limitation, the time accuracy that can be controlled is low. It is impossible to guarantee that a complete data frame is collected in the effective area. To solve this problem, we take the green light as the indicator for detecting the synchronization status, and the yellow and red lights continuously send the same data frame within the duration of the image frame to ensure that one image frame can successfully receive one data frame. Due to the instability of the optical cameras' frame rate, two adjacent data frames are simultaneously collected in one image frame. Therefore, after sending a data frame, let the green light send a low pulse to indicate a data frame's end. If the dark stripes in the green light are detected in the image frame, the image frame contains two data frames. Furthermore, synchronization has expired. At this time, the feedback information needs to be sent by the smartphone's flashlight, and the node end receives the feedback information and resynchronizes. The program consists of two parts:

(1) Initial synchronization at the beginning of transmission;
(2) Resynchronization after transmission interruption (Fig. 9).

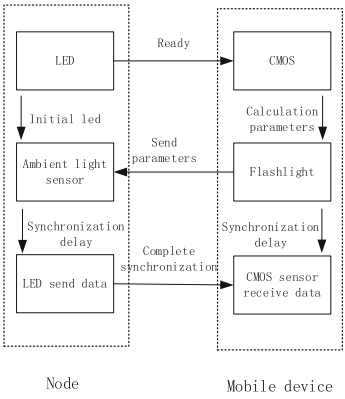


Fig. 9. Initial synchronization diagram

In the initial stage, we have a lot of preparations to do. When the node is in the ready state, the three LEDs are always on, the smartphone takes pictures and collects and determines the value of the synchronization parameter according to the result of the image processing. The synchronization parameter is shown in Fig. 10. The smartphone calculates the effective area width according to the position of the LED light in the image, which is marked as X. Then the smartphone will send the parameter code to the node end, and the node end can calculate the LED emission frequency suitable for the effective area width according to the parameter.

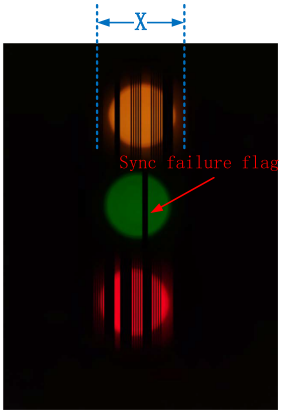


Fig. 10. Transmission parameter diagram

After the initial synchronization, the node LED repeatedly sends the corresponding data frame within each image frame's duration. However, due to the unstable frame rate, the data stripes will shift in the image frame. When the offset exceeds one image frame's duration, the same image frame may contain two adjacent data frames. The green light is used as the detection of the synchronization state. Whenever a data frame is sent, the green light will send a low pulse to separate two adjacent data frames. When the green light in the image frame detects dark stripes, the data collected in an effective area containing two adjacent data frames. The synchronization has failed and needs to be resynchronized. The smartphone side feeds back the image frame's sequence number with synchronization failure to the node side through the flashlight. After the node side receives and decodes, it resynchronizes the transmission from the data frame indicated by the sequence number.

6 Evaluation

6.1 Experiment Environment

Based on the Android operating system and general sensor nodes, we realize a HDDIS using a commercial off-the-shelf Android smartphone and popular sensor nodes TelosB. In this system, smartphone need to be equipped with a flashlight, optical camera, and its installed Android support for Camera2 API. The TelosB owns RGB-LED and ambient light sensor.

6.2 Experiment Content

We conduct a series of experiments about updating code blocks from five different functions shown in Table 2. The Timer and LS Ctrl function relative to controlling sensor node. Bubble sort, Dijkstra, and Horspool is sort of algorithms.

As shown in Fig. 11, we compared the compressed code size with the original code, and the compression rate of the five code blocks is given in Table 2. It is obviously

that the size of the code block is reduced as least 8%, especially in LS Ctrl Fun with 20% reduction. Due to more reused bits from the opcode of single operand instructions than multi-byte instructions, it can achieve better effect that applying our compression scheme in the code block including more single operand instructions, such as Timer Fun and Bubble Sort.

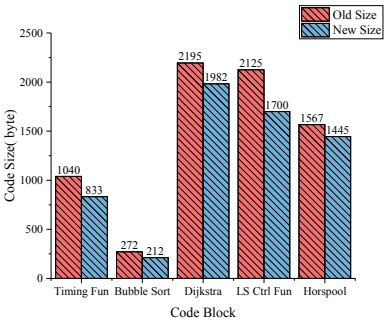


Fig. 11. Compressed Code Size

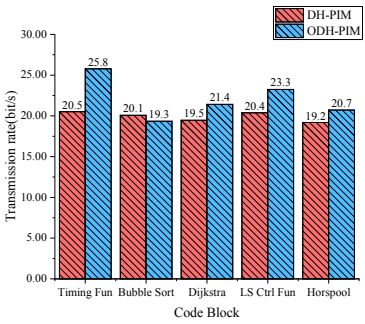


Fig. 12. Transmission rate

The transmission rate of ODH-PIM is given in Fig. 12. We compared ODH-PIM with DH-PIM. In order to neatly compares, these two methods are used to transmit origin code without compression. In the most cases of transmitting code blocks, our method is better than DH-PIM. Especially in Timing Fun case, ODH-PIM is 25.8% faster than DH-PIM because ODH-PIM can effectively reduce the length of a symbol. In Bubble Sort case, we found a large number of consecutive zero existing in binary codes which ODH-PIM hard to reduce, and the transmission rate of ODH-PIM remains about 3.9% below DH-PIM.

Table 2. Compression Rate

Code Block	Brief	Size(bit)	Compression Rate	Complete time(s)
Timer Fun	Control timer	9360	80.10%	364.2
Bubble Sort	Data sorting	2128	77.94%	118.7
Dijkstra	Find the shortest path	17560	90.30%	821.5
LS Ctrl Fun	Control light sensor	17000	80%	732.3
Horspool	Character match	12536	92.21%	606.4

We also apply the binary code compression and ODH-PIM at the same time, and record the transmission completion time shown in Table 2. Table 2 list the transmission completion time of each code block. The instruction compression based on bitmask reduces the size of the code block and ODH-PIM shorten the length of each symbol. Both methods guarantee that the commercial off-the-shelf devices complete the debugging interaction by VLC.

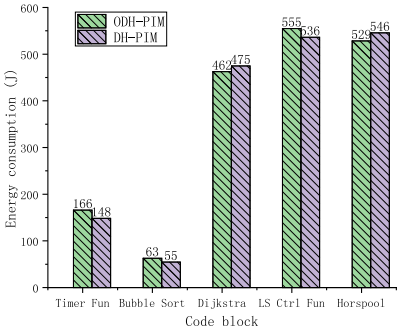


Fig. 13. Energy consumption

Next, we compared the energy consumption of the ODH-PIM and DH-PIM. The result is shown in Fig. 13. We use ODH-PIM and DH-PIM to transmit the above five code blocks respectively, and finally figure out the average energy consumption. The DH-PIM consumes 352 J of energy. However, ODH-PIM consumes 355 J energy. Since realizing the LED half-on state in ODH-PIM requires flashlight remaining a duration of high-speed flash, it will pay extra 7.36% energy consumption than DH-PIM. However, the energy consumption of flashlight is from smartphone and the extra part is slight comparing with the transmission rate improvement by LED half-on state.

The bit error rate (BER) is an important fact to evaluate performance of ODH-PIM. Figures 14 and 15 demonstrate the BER of ODP-PIM with different angles and distances. In the downlink, different distances between the flashlight to the ambient light sensor result in different light intensities captured by the sensor. From Fig. 14, it is obviously that BER is very high if the transmission distance is over 20 cm. The reason is that the sensor is difficult to detect the LED half-on state once exceeding 20 cm and BER boost due to lack of the state.

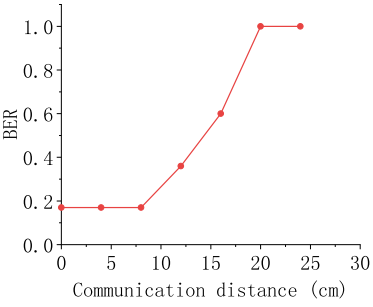


Fig. 14. BER under different distances

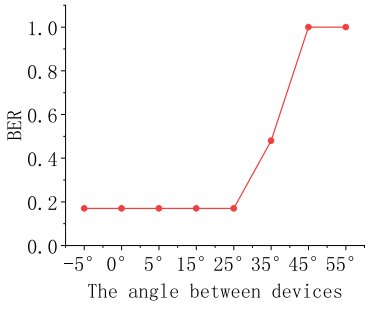


Fig. 15. BER under different angles

Because VLC owns the directionality character, the transmission angle is also factor to impact performance of transmission. As shown in Fig. 15, when the angle between both devices exceeds 45°, VLC is unavailable using smartphone and TelosB sensor. The angle also affects the light intensity collected by the sensor. When the angle is less than 25°, the sensor is able to capture enough light intensity from the flashlight, so the VLC

can work with a low BER. When the angle is between 25° and 45°, the flashlight provide enough light intensity for communication, and that leads to boost BER.

The debugging information stroboscopic sequence captured by the optical camera is saved as an image for decoding in the uplink. However, its own hardware parameters, such as exposure time, will also affect imaging. The experimental results are shown in Fig. 16. When the exposure time is the minimum exposure time of 10 microseconds because each line's exposure time is short, the transition zone around the effective area is narrow. The high-frequency frame header is easy to identify decode. When the exposure time is increased to 100 microseconds, it can be clearly observed that the light information is also collected outside the LED's actual light-emitting area, and the transition band is wider, which affects the identification and detection of the high-frequency frame head. On the other hand, larger exposure time can increase the width of the effective area, and at the same time increase the data capacity of a single frame of data. So the appropriate exposure time can be set according to actual hardware conditions to achieve the best communication effect.

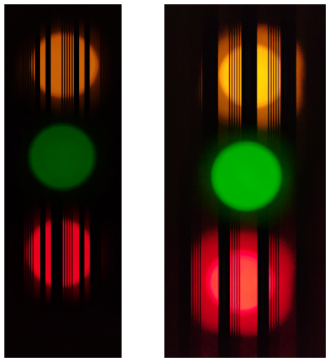


Fig. 16. Comparison of imaging under different exposure time conditions

7 Conclusion

In this paper, we apply VLC-based debugging interaction applying in the sensor nodes which malfunction caused by radio communication failure, and further design a hybrid duplexing debugging interaction system based on commercial off-the-shelf device. This system adopts different visible light sensing means in the physical layer of uplink and downlink. In view of the unstable frame rate of general CMOS optical cameras, we also designed a frame synchronization scheme to ensure fast and reliable transmission of debugging data. Due to transmitting debugging binary code with downlink, two methods from source and channel coding are proposed to improve downlink transmitting performance. For source coding, the character of transmitted binary code makes bitmask-based instruction compression available to reduce the size of transmitted data.

For channel coding, we optimize DH-PIM by introducing LED half-on state, and put forward Overlap DH-PIM to increase the transmission rate. Experimental results demonstrate the validity of the above two methods. In future work, by adding error correction in VLC, the transmission rate can be improved. In addition, the drone equipping flashlight and camera owns the potential capacity of VLC. By sending the drone to place nearby the lost sensor node, it extends VLC-based debugging interaction to most of field-deployed application.

Acknowledgment. This research work is supported by Zhejiang Provincial Natural Science Foundation of China under Grant (LY20F020026).

References

1. Li, D., Liu, W., Cui, L.: EasiDesign: an improved ant colony algorithm for sensor deployment in real sensor network system. In: Proceedings of IEEE GLOBECOM, pp. 1–5 (2010)
2. Du, W.Z., Chen, H.M., Li, D., Cui, L.: Research on a gateway switching mechanism based on gateway's energy-harvesting for wireless sensor networks. Chin. High Technol. Lett. **26**(6), 631–642 (2016)
3. Shah, P.A., Awan, K.M., Rehman, Z., et al.: A route optimized distributed IP-based mobility management protocol for seamless handoff across wireless mesh networks. Mob. Networks Appl. **23**(4), 752–774 (2018)
4. Liu, J., Chung, S.H.: An efficient load balancing scheme for multi-gateways in wireless mesh networks. J. Inf. Process. Syst. **9**(3), 365–378 (2013)
5. Shao, S., Khreishah, A., Rahaim, M.B., et al.: An indoor hybrid WiFi-VLC internet access system. In: Mobile Ad Hoc and Sensor Systems, pp. 569–574 (2014)
6. Fan, L., Liu, Q., Jiang, C., et al.: Visible light communication using the flash light LED of the smart phone as a light source and its application in the access control system. In: 2016 IEEE MTT-S International Wireless Symposium, pp. 1–4 (2016)
7. Das, S., Chakraborty, A., Chakraborty, D., Moshat, S.: PC to PC data transmission using visible light communication. In: Computer Communication and Informatics, pp. 1–5 (2017)
8. Adiono, T., Pradana, A., Putra, R.V.W., et al.: Analog filters design in VLC analog front-end receiver for reducing indoor ambient light noise. In: IEEE Asia Pacific Conference on Circuits and Systems, pp. 581–584 (2017)
9. Chen, P.Y., Wu, C.C., Jiang, Y.J.: Bitmask-based code compression methods for balancing power consumption and code size for hard real-time embedded systems. Microprocess. Microsyst. **36**(3), 267–279 (2012)
10. Kumar, R.N., Chandran, V., Valarmathi, R.S., et al.: Bitstream compression for high speed embedded systems using separated split look up tables (LUTs). J. Comput. Theor. Nanosci. **15**(5), 1719–1727 (2018)
11. Seong, S.W., Mishra, P.: Bitmask-based code compression for embedded systems. IEEE Trans. Comput. Aided Design Integrated Circuits Syst. **27**(4), 673–685 (2008)
12. Haider, S., Nazhandai, L.: A hybrid code compression technique using bitmaskand prefix encoding with enhanced dictionary selection. In: Proceedings of Conference on Compilers, Architecture, and Synthesis for Embedded Systems, pp. 58–62. ACM, New York (2007)
13. Ghassemlooy, Z., Hayes, A.R., Seed, N.L., et al.: Digital pulse interval modulation for optical communications. IEEE Commun. Mag. **36**(12), 95–99 (1998)

14. Aldibbiat, N.M., Ghassemlooy, Z., McLaughlin, R.: Error performance of dual header pulse interval modulation (DH-PIM) in optical wireless communications. IEEE Proc Optoelectron **148**(2), 91–96 (2001)
15. Yao, K., Wu, N., Wang, X., et al.: A novel power efficient modulation scheme for VLC systems. In: 2016 IEEE/CIC International Conference on Communications in China (ICCC). IEEE (2016)
16. Ali, A.Y., Zhang, Z., Zong, B.: Pulse position and shape modulation for visible light communication system. In: International Conference on Electromagnetics in Advanced Applications. IEEE (2014)
17. Fuzile, T.S., Heymann, R.: Investigating a visible light communication channel with an LED transmitter and a smartphone receiver. In: IEEE Africon 2017 Proceeding, pp. 354–358 (2017)