

Visible Light Communication (VLC) between Electronic Devices

Thesis

Bachelor of Engineering in the Undergraduate School of The Ohio State University

By

Meenwook Ha, B.S.

Honors Undergraduate Research Distinction Thesis Project

The Ohio State University

2019

Thesis Committee:

C. Emre Koksal, Adviser

Lee C Potter

Copyright by

Meenwook Ha

2019

ABSTRACT

Visible Light Communication (VLC) is a form of wireless communication that uses lightwaves to transmit and receive data. The use of VLC technology has gained significant popularity in recent years. Li-Fi is a popular example of VLC, in which separate equipment needs to be installed to the mobile devices to decode data transmitted from Li-Fi enabled bulbs. The objective of this project is to build a VLC system, utilizing the existing components of existing mobile devices. In particular, we will utilize the screen as the transmitting unit of the information and the camera as the unit, capturing the transmitted signal, on which we encode English text. To that end, we have implemented the transceiver as a Matlab application. Our system has a position recovery, synchronization, and error correction layers built in. Our main objective was to achieve robust communication, albeit at low rates, due to the limitations of the components and the channel.

DEDICATION

Dedicated to the students at The Ohio State University

ACKNOWLEDGEMENTS

This Undergraduate Honors Research Distinction Thesis Project is sponsored by College of Engineering at The Ohio State University, and I would like to express my sincere gratitude to Professor C. Emre Koksal, who gave me a precious opportunity to conduct an individual thesis project in the field of Visible Light Communication and advised the overall project. I also want to deliver my gratitude to professors who taught and gave knowledge to me at The Ohio State University.

VITA

March 2011.....Ogum High School

2012 to 2016.....Korean National University, Transport

2014.....Academic Award Scholarship

2016 to present.....Undergraduate, The Ohio State University

2016 to present.....Gee Scholarship

2016 to present.....College of Engineering Dean’s List

2018.....Honors Research Scholarship

FIELD OF STUDY

Major Field: Electrical Engineering

TABLE OF CONTENTS

INTRODUCTION.....	1
Figure 1. Visible Light Spectrum.....	1
METHODOLOGY	2
Figure 2. Big Picture of Visible Light Communication	2
# 1. Encoding	3
Figure 3. Information Matrix of ‘Mathwork’	3
Figure 4. Adding Pilot Vectors and Encoding Process	3
# 2. Transmitting Lightwaves.....	4
Figure 5. Matlab App Screen When Transmitting ‘M’	4
# 3. Receiving the Transmitted Signal.	5
# 3-1. Sampling Rate	5
# 3.2 Noise Effect from External Lights.....	5
Figure 6. Noise from External Lights.....	5
# 3.3. Color Detecting Function	6
Figure 7. Color Threshold App Before Setting Ranges	6
Figure 8. Color Threshold App After Setting Ranges.	6
# 3.4 Receiving.....	7
Figure 9. Receiving Process	7
# 4. Decoding	8
#4-1. Error Correction Using Syndrome Vectors.	8
Figure 10. Error Detection Using Syndrome Vector.....	8
#4-2. Connect Two Decoded Codebooks and Truncate.	8
Figure 11. Truncation Based on Pilot Vectors	9
#4-3. Error Detection from Synchronization.	9
Figure 12. Synchronization Error Detection Process	9
Figure 13. Output Message in Matlab Command	10
CONCLUSION	11
REFERENCES.....	12

INTRODUCTION

Visible Light Communication (VLC) is a part of Optical Wireless Communication (OWC). OWC uses lightwaves to transmit and receive data, and the advantages of OWC will be fast communication speed and the fact that users do not have to pay to use the light spectrum.

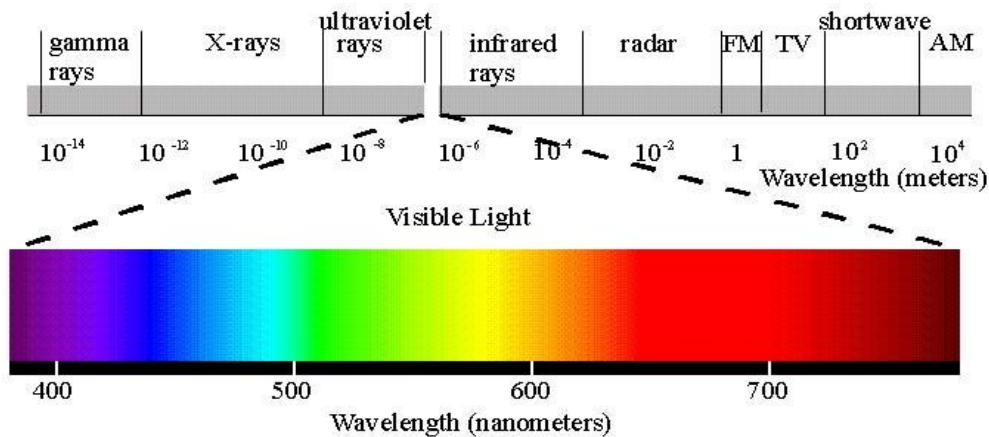


Figure 1. Visible Light Spectrum

OWC covers the entire light spectrum, and VLC belongs to OWC using the Visible Light spectrum. The Smart Bulb from Philips is a successful example of VLC. Using the smart bulbs, people can easily manage all bulbs in a room. If a user changes the settings of a bulb using a mobile app, lightwaves are created from the change in a main bulb and transmitted to nearby bulbs so that every bulb in a room has the same setting. However, most VLC systems require installing new equipment. In the case of the Smart Bulb, if we want to construct the VLC system in a room, it is required to replace all bulbs to smart bulbs designed for VLC. In this project, the existing components of existing mobile devices are utilized to build a VLC system. Most mobile devices such as tablet PCs, smartphones, and laptops have screens and cameras. To build a VLC system, the project utilizes the screen as the transmitting unit of the information and the camera as the unit that receives the transmitted signal. Therefore, the ultimate goal of this project is to achieve robust VLC utilizing existing components of existing mobile devices.

METHODOLOGY

The VLC system of this project is comprised of four steps: encoding, transmitting, receiving, and decoding, and the big picture of the system is shown below.

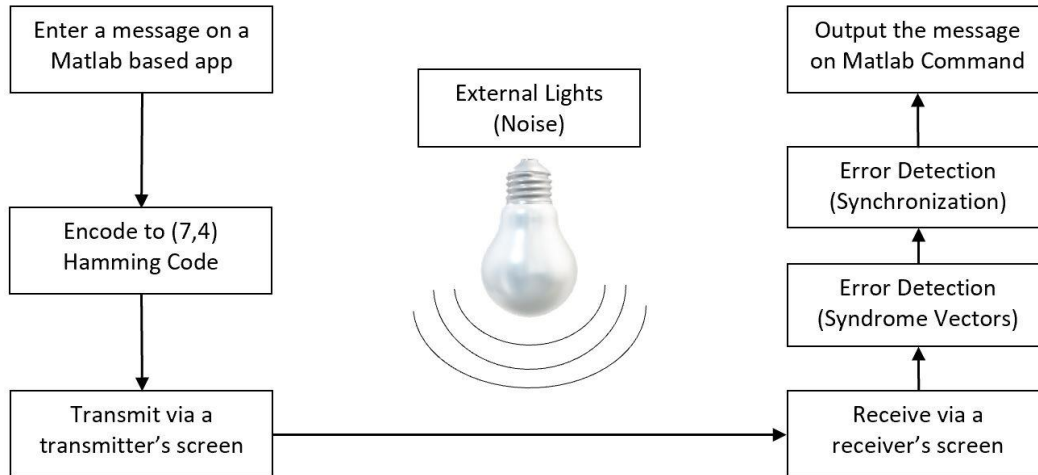


Figure 2. Big Picture of Visible Light Communication

A Matlab based app was built to have the purpose of inputting and encoding messages. Users can run the app on their mobile devices by accessing the Matlab-Online browser. Each number, character, and letter in the message is converted to the corresponding ASCII-Code, 8-digit binary number. The binary numbers form an information matrix, and the matrix is encoded to two (7,4) Hamming Codes. The encoded two matrices are transmitted as forms of lightwaves, and receivers receive not only the lightwaves but also noises. In this binary channel, there can be noise effects from the external lights which need to be corrected in decoders. Errors in the received signal are corrected by passing two error correction steps, and the message is outputted on the screen of receivers. The following are detailed explanations for the four steps.

1. Encoding

Suppose a user inputs ‘Mathwork’ in the app. The letters in the message are accepted as corresponding ASCII-Codes, 8-digit binary numbers, and the numbers form an information matrix as Figure 3 shows below.

M	a	t	h	w	o	r	k
0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1
0	1	1	1	1	1	1	1
0	0	1	0	1	0	1	0
1	0	0	1	0	1	0	1
1	0	1	0	1	1	0	0
0	0	0	0	1	1	1	1
1	1	0	0	1	1	0	1

Figure 3. Information Matrix of ‘Mathwork’

Each binary number is vertically stored and therefore, the first row has the leftmost bits in binary numbers, and the last row has the rightmost bits in binary numbers. For the next step, pilot vectors are added at the front and back of the information matrix as the left picture of Figure 4 shows below.

$$\begin{array}{c}
 \begin{array}{|c|cccccccc|c|}
 \hline
 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
 \hline
 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
 \hline
 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
 \hline
 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\
 \hline
 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 \\
 \hline
 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 \\
 \hline
 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\
 \hline
 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \\
 \hline
 \end{array}
 \begin{array}{l}
 x_1 \longrightarrow c_1 = G \cdot x_1 = \\
 x_2 \longrightarrow c_2 = G \cdot x_2 =
 \end{array}
 \begin{array}{c}
 \begin{array}{|c|cccccccc|}
 \hline
 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
 \hline
 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
 \hline
 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
 \hline
 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\
 \hline
 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
 \hline
 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\
 \hline
 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\
 \hline
 \end{array} \\
 \\
 \begin{array}{|c|cccccccc|}
 \hline
 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\
 \hline
 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \\
 \hline
 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 1 \\
 \hline
 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \\
 \hline
 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 \\
 \hline
 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 \\
 \hline
 \end{array}
 \end{array}$$

Figure 4. Adding Pilot Vectors and Encoding Process

The largest 8-digit binary number is used as the pilot number. In order to encode this matrix to (7,4) Hamming code, the matrix is separated to two matrices, x_1 and x_2 . By multiplying the generator matrix (G) for (7,4) Hamming code with x_1 and x_2 , two codebooks, c_1 and c_2 , are generated.

2. Transmitting Lightwaves.

Figure 5 below shows the screen of the transmitter app when it is transmitting letter 'M.' Users can input a message on the input section, and the message is encoded to 2 codebooks as explained in the encoding section. To transmit the two codebooks, the codewords in the codebooks are expressed as consecutive light frames. The top of the screen expresses the codewords in c_2 and the bottom of screen expresses the codewords in c_1 . Red is used to express bits in codewords, and blue is used to express boundaries between bits. Red is used to express bits in codewords, and blue is used to express boundaries between bits.

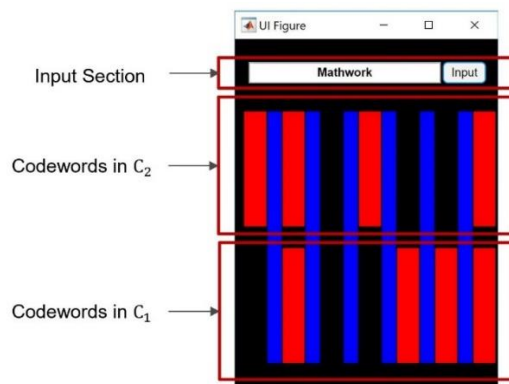


Figure 5. Matlab App Screen When Transmitting 'M'

If the bit is 1 at a certain bit's location, red is displayed at that location. For example, the codeword for letter 'M' in the second codebook is $1101001_{(2)}$. Therefore, there are red columns at the leftmost two, center, and rightmost bits' locations. Decoders can find the locations of bits by comparing the center coordinates of red and blue columns. Therefore, if there are no blue columns, it is difficult to find the exact locations of bits although it can find the relative positions between bits.

Each image expresses each letter number, or character and lasts for 0.3. The reason of the image time use is that the time produces the fastest communication speed by correcting all possible synchronization errors. It will be explained in decoding section.

[The transmitting process will be shown at the link: <https://youtu.be/r1IjKyWx2hk>]

3. Receiving the Transmitted Signal.

Before explaining the receiving process, there are three important concepts: sampling rate, noise effect, and the Color Threshold App of Matlab, need to be explained.

3-1. Sampling Rate

By multiplying, the image time and frame rate of Matlab, the sampling rate is calculated. The fastest frame rate of Matlab is 10 FPS (Frame/Second) so that this frame rate is used, and the rate means that if a user takes 100 frames, it will take 10 seconds. However, in actual, it takes approximately from 10 to 12 seconds because it takes more time to process given commands in a loop. This minimal time delay causes synchronization errors which need to be corrected during decoding process. By multiplying the framerate and image time, the sampling, 3, is calculated as it is shown below.

$$\text{Sampling rate} = 10\text{Frame/Second} \times 0.3 \text{ Second/Image} = 3 \text{ Frame/Image.}$$

The sampling rate means that the receiver will have three samples for each letter, number, or character from the transmitted signal if there are no synchronization errors.

3.2 Noise Effect from External Lights.

There can be noise effects from external lights during receiving the transmitted signal. External lights can block any bits on the screen of transmitters. If external light hits the center of the screen of a transmitter and reflects out to a receiver, it is impossible for the receiver to accept the center bit because the light blocks the center bit.

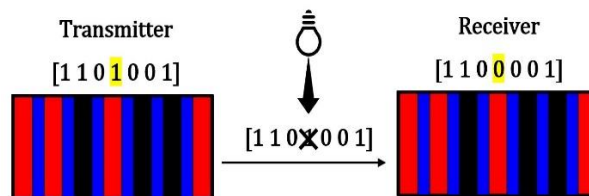


Figure 6. Noise from External Lights

Figure 6 shows the noise effect in visual. When a codeword, $1101001_{(2)}$, is transmitted, the external light can block the middle information bit, and the codeword can be received as $1100001_{(2)}$.

3.3. Color Detecting Function

To detect colors on a received frame, color detecting functions are created by using the Color Threshold App of Matlab. Figure 7 shows the Color Threshold app. In the app, if a user sets certain ranges for red, blue, and green colors by moving the minimum and maximum bars and saves the setting, this app automatically creates a function to detect colors satisfying all the three conditions.

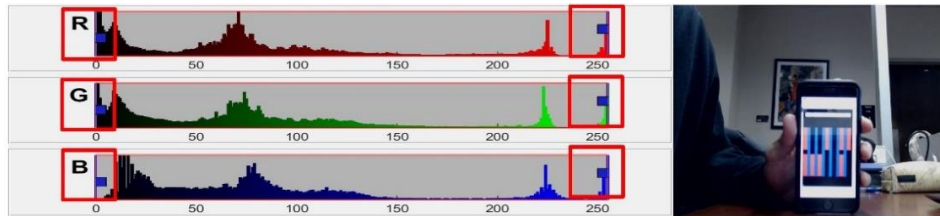


Figure 7. Color Threshold App Before Setting Ranges

If a user wants to create a function for detecting red colors, the user can lower the maximum bar of green and blue colors and raise the minimum bar of red, and then the function is created. Figure 8 below shows the result from the changes. In this way, color threshold functions for red and blue colors are created.

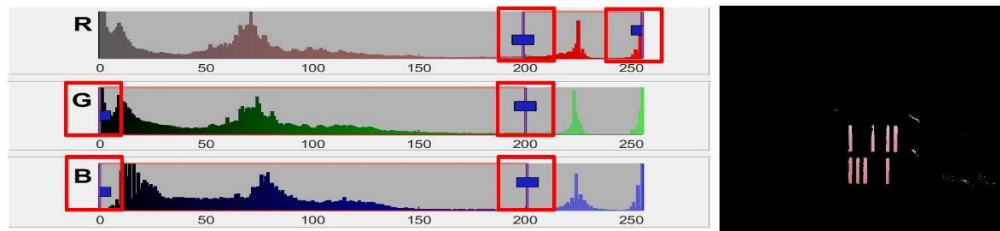


Figure 8. Color Threshold App After Setting Ranges.

3.4 Receiving

Using the color threshold functions, the bits from the transmitted signal are stored, and received frames are shown on the screen of receiver. Figure 9 shows the receiving process in visual. In Step 1, to detect red columns, the decoder binarizes the captured frame using the color threshold functions. The right picture shows the binarized frame.

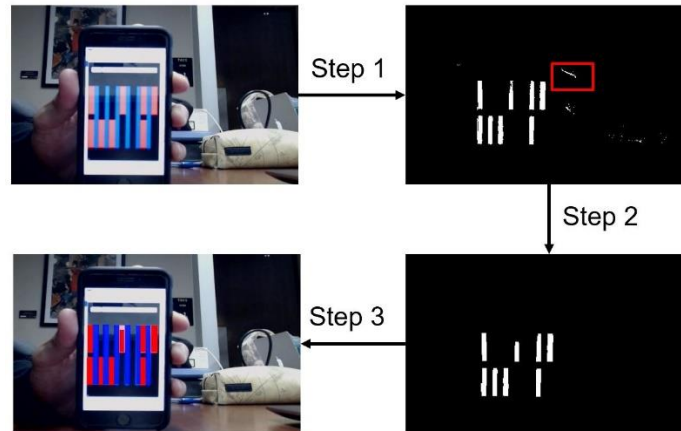


Figure 9. Receiving Process

Step 2 is noise filtering process. Before detecting the locations and number of bits, the noises are filtered out to prevent possible errors. In the decoder, there is a certain range for red columns. Only if sizes of red columns are in the range of 150 to 100,000 pixels, they are accepted as bits. To filter out noises in the range, the noise filtering process is required. In this process, if columns are smaller than 5×5 pixels or the shapes of noises are not rectangular, they are removed on the frame. As the result of this process, minimal noises are filtered out, and the noise marked with the red box in Figure 9 is also removed because of its shape although the size is in the range. Using the cleaned frame, the receiver displays the accepted frame on the screen. The red columns and blue columns shown on the screen of the receiver are solid colors by comparing with the screen of the transmitter. Based on the solid columns, users can know the current communication status.

[The receiving process will be shown at the link: <https://youtu.be/utl7pWNf3WE>]

4. Decoding

There are two error correction steps. One is using syndrome vectors, and the other is synchronization.

#4-1. Error Correction Using Syndrome Vectors.

The first error correction is using syndrome vectors. The decoder receives two codebooks having error bits, and (7,4) Hamming code allows to correct one error bit per codeword. Figure 10 below expresses the error correction process in visual.

$$\hat{c}_1 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} \quad s = H\hat{c}_1 = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & \color{red}{\boxed{0}} \\ 0 & 1 & 1 & 1 & 0 & 1 & \color{red}{\boxed{0}} \\ 1 & 1 & 0 & 1 & 0 & 0 & \color{red}{\boxed{0}} \end{bmatrix} \quad \hat{c}_1 = \begin{bmatrix} \color{red}{\boxed{0}} \\ \color{red}{\boxed{0}} \\ \color{red}{\boxed{0}} \\ \color{red}{\boxed{1}} \\ \color{red}{\boxed{0}} \\ \color{red}{\boxed{1}} \\ \color{red}{\boxed{1}} \end{bmatrix} \quad c_1 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \end{bmatrix}$$

Reverse the digit \downarrow

Figure 10. Error Detection Using Syndrome Vector

When a received codeword is multiplied by the parity-check matrix (H) for (7,4) Hamming code, the syndrome vector for the codeword is created. If all three elements in the syndrome vector are zeros, it means that there is no error in the received codeword, and if it is not, the syndrome vector points to the location of the digit having the error bit and reverses the digit. In Figure 10, the syndrome vector points to the 7th column of the parity check matrix, and therefore the 7th element in the codeword is reversed. If there is more than one error bit in a codeword, the codeword is undetectable. The syndrome vector might point to the wrong bit's location so that the codeword will be incorrectly decoded. In this step, the possibility is ignored because it can be corrected in the synchronization step.

#4-2. Connect Two Decoded Codebooks and Truncate.

Before correcting synchronization errors, the two corrected codebooks are connected to one matrix after removing parity bits, and the matrix is truncated based on pilot vectors as Figure 11 shows below.

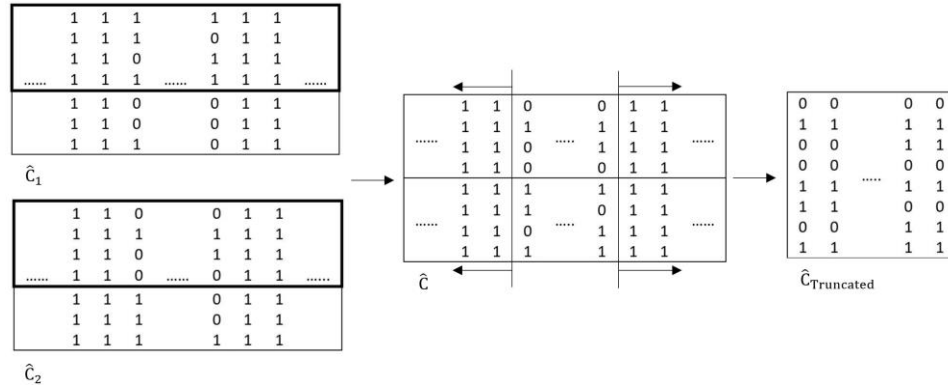


Figure 11. Truncation Based on Pilot Vectors

#4-3. Error Detection from Synchronization.

Before down-sampling the matrix, synchronization errors were corrected. In Figure 12, the left matrix is before synchronization error corrections, and the right matrix is after the corrections.

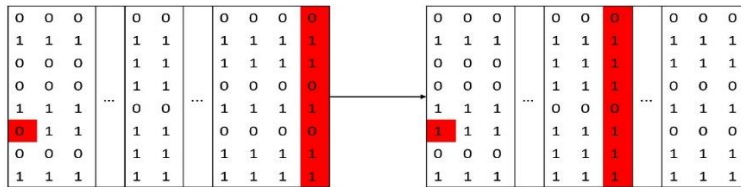


Figure 12. Synchronization Error Detection Process

First, vectors can still have incorrect information bits, and to correct the errors, the principle of repetition code is applied. For instance, in the left matrix, the 6th row of the leftmost vector was supposed to be 1 like the consecutive two vectors. To correct these errors, rows of three consecutive vectors are compared and voted for the majority carriers. Therefore, it is possible to correct the error if there is one error bit in a row. Secondly, under-sampled vectors are corrected. Based on the sampling rate, 3, if the number of the same consecutive vectors is 2, one vector is copied and added. Thirdly, over-sampled vectors are also corrected. If four identical vectors are continued, one vector is removed.

Here is the reason why this system needs to use image time of 0.3 Second/Image. First, in order to vote for majority carriers, it is better to have an odd number of sampling rate. It is also possible to use even numbers, but in the case of even numbers, the result of the vote could be half and half. Therefore,

the use of an odd number of sampling rate is recommended. In addition, if the sampling rate is 1 and if a vector is under-sampled, it means that the vector will not be accepted in a decoder which means that it is impossible to recover the under-sampled vector. Therefore, a sampling rate of 3 is used to correct all possible synchronization errors and have a fast communication speed. After this error detection process, the corrected matrix is down-sampled, and the received message is outputted in Matlab command as Figure 13 shows below.

```
The received message is: Mathwork
```

Figure 13. Output Message in Matlab Command

[The whole process will be shown at the link: <https://youtu.be/NCjbWIkDJkY>]

CONCLUSION

In conclusion, it was possible to build VLC systems between mobile devices that were capable of robust communications. However, there are still two points which need to be improved for better communication and commercialization. Therefore, I want to suggest two future research directions.

First, it will be better if the communication system is converted to real-time communication. In the current system, decoders need to receive and decode the transmitted signals separately. Therefore, it is not a real-time communication. As the length of a message is longer, it will take more time to decode the received signal and therefore, users will have to wait to receive decoded message after communication. This inconvenience can be resolved if the current communication system is improved to a real-time communication system.

Secondly, since the processing speeds of some mobile devices such as tablet PCs and smartphones are slow to process 10 FPS, they could not be used as receivers. Laptops can process the frame rate, but it is difficult for other mobile devices. 5 FPS and 1 FPS were also tried, but there was still much lag when capturing frames. In order to resolve this problem, the Matlab code for decoders was converted to a Matlab web-app so that users simply could use the app instead of directly running the code. However, the processing speeds of mobile devices are still problematic, and in the case of Matlab web-app, it is possible for users to access the host's URL only if they are in the same intranet. Therefore, although they are in the same Wi-Fi zone, users cannot access to the host's URL if the users are not in the same intranet as the host.

One solution related to the two suggested directions can be to convert Matlab scripts for receivers and transmitters to Java. Java is specialized for designing apps. While many commands and tools are limited to build apps in Matlab, diverse tools are provided to build apps in Java. If the app for receivers can achieve a fast frame rate and lead the decoder to receive and decode simultaneously, the two suggested directions will be resolved.

REFERENCES

- [1] Git-hub Transmitter and Receiver Code Repository: <https://github.com/Meenwook/VSL>
- [2] “Friends of Hue.” Philips. [Online]. Available: <http://www2.meethue.com/en-us/friends-of-hue>. [Accessed: 02-Oct-2018].
- [3] “Light Fidelity (Li-Fi): towards all optical networking.” SPIE.DIGITAL LIBRARY. [Online]. Available: <https://www.spiedigitallibrary.org/conference-proceedings-of-spie/9007/900702/Light-fidelity-Li-Fi-towards-all-optical-networking/10.1117/12.2044649.full> [Accessed: 15-Mar-2019]
- [4] “Li-Fi is a paradigm-shifting 5G technology.” ScienceDirect. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S2405428317300151> [Accessed: 15-Mar-2019]
- [5] “Receiving Process Video.” Available: <https://youtu.be/utl7pWNf3WE>
- [6] “Whole Communication Process Video.” Available: <https://youtu.be/NCjbWIkDJkY>
- [7] “Transmitting Process Video.” Available: <https://youtu.be/r1IjKyWx2hk>
- [8] “What is Visible Light Communication?” Firefly Li-Fi. [Online]. Available: <https://www.fireflylifi.com/visible-light-communications.html> [Accessed: 16-Mar-2019]
- [9] “Web Apps.” MathWorks. [Online]. Available: <http://www.mathworks.com/help/compiler/web-apps.html>. [Accessed: 15-Mar-2019].