Wireless Communication through High Frequency Modulation of Light-Emitting-Diode Lightbulbs

Daniel Connolly, William Fairman, Qingmu Deng
December 13th, 2019
Introduction to Analog and Digital Communications, Fall 2019
Olin College of Engineering
Needham, MA

***Abstract*—The discovery of high frequency switchings of commercial LED lightbulbs offers a new approach to wireless communication through visible light. In this report, we present our experiments in transmitting images through LED lightbulbs over distance comparable to that of a desktop lamp and the methods we adopted for signal filtering, noise suppression, and error correction in our communication system.**

I. Introduction and System Overview

Light fidelity, Li-Fi for short, is a wireless communication scheme that transmits data within the visible light spectrum. The advent of LED technologies has made high frequency switching accessible to even commercial indoor lighting equipment. Mathematically, by exploiting the orthogonality of

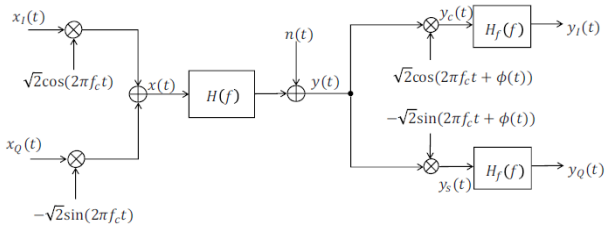

Fig. 1. The block diagram for the quadrature amplitude modulation (QAM) scheme.

cosines and sines, we can multiply two independent streams of bits by cosine and sine respectively at the transmitter end. At the receiver end, we would discern the two data streams by again multiplying by cosine and sine. While n-QAM schemes exist that account for the amplitude of the transmitted and received signals, we focused on 4-QAM, or quadrature phase shift keying (QPSK), in order to achieve a symbol rate of 2 bits per symbol.

A. Physical System

Our physical setup exists in two separate parts, one for the transmitter and one for the receiver. On the transmitter end, depicted in Figure 2, a laptop connects to an Universal Software Radio Peripheral (USRP), which transmits our data through an amplifier before it modulates the brightness of the LED lightbulb. A bias voltage is needed on the lightbulb so that the relationship between the energy of the emitted light and the applied voltage remains linear. On the receiver

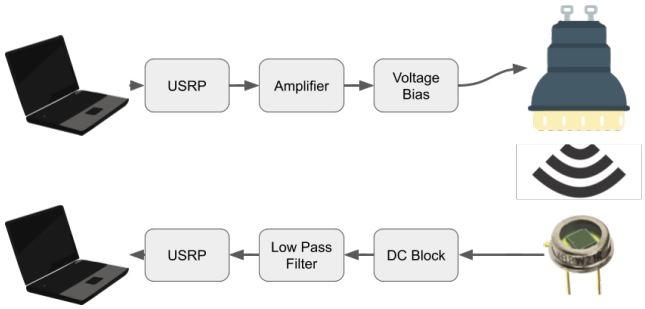

Fig. 2. The block diagram for the LIFI system.

end, we used a ThorLabs avalanche photodetector to capture the variation in the output of the lightbulb. Specifically, it measures the power of the light it receives. The photodetector then connects to a DC block and a low pass filter before another USRP communicates the received data back to a second laptop.

B. Software Implementation

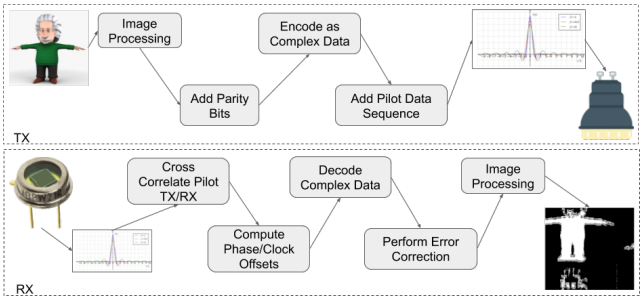

Fig. 3. The flow diagram of our data processing flow diagram.

1) TX Data Preparation: When a user inputs an image, the image is read into Matlab in black and white with an appropriate contrast. The image matrix is then flattened out into a vector. If we choose to incorporate the (7,4) Hamming Code we implemented in our signal in order to perform error correction, as shown in Appendix A, we would add in three parity bits for every 4 bits of actual data. This scales up the amount of data we have to send by $7/4$. Theoretically, only one error in every seven bits of the received data can be

corrected with this level of parity. Higher error rates or burst errors (multiple errors in a small portion of the data) have the potential to compound the number of errors in our data.

To make the data processing easier on both the TX and RX ends, we standardized our data packet structure as shown in Figure 4. In order to determine the length of our pilot signal, we calculated the likelihood of finding the sequence in our full data packet as $\frac{\text{packet length}-\text{pilot length}}{2^{\text{pilot length}}}$. We ultimately chose a 64-bit sequence, represented in 4-QAM as 32 complex numbers, making the probability of our pilot sequence occurring in our data packet 1.08×10^{-14}. As a result, we include in our packet a 64-bit pilot signal known by both the TX and the RX, metadata, which includes relevant data such as the dimensions of an image, the data we intend to transmit, and zero padding that ensured the length of data packet is always 200,000 bits.

Fig. 4. The structure of data packets in our system.

To mitigate the effects of the channel and possible inter-symbol interference, we chose a root raised cosine matched filter with a rolloff of 0.5 and 50 data points in between zero crossings (pulse width). Thanks to the forgiving nature of root raised cosine and the relatively clean channel in the visible light spectrum, we would convolve our standardized packet directly with the root raised cosines after upsampling our data by 50, the effective width of our pulses. The consequent transmission data is $50 \times 200,000 = 10,000,000$ bits long.

2) RX Data Processing: Once we received the data, we needed to perform a significant amount of processing. To ensure we were mitigating the effects of the channel, we again convolved our signal with the same root raised cosine we utilized on the transmitter end, effectively choosing this as our filter F(f) and making our Q(f) approximately a raised cosine.

$$Q(f) = P(f) \times H(f) \times F(f) \tag{1}$$

Because the receiver knew the seed of the random generator the transmitter used to create the pilot data sequence at the beginning of each data packet, it could recreate the sequence, pass it through a root raised cosine filter, and cross correlate the result with the received signal. The peaks of the cross correlation plot tell us when a new packet is being sent. We can then trim the received data to begin at this point and be of the same size as our standard data packets.

If the cosine signals at the transmitter and receiver are not perfectly synchronized, we need to correct for their phase offsets in order to accurately decode the received data. In order to do this, we took two approaches: a DFT-based approach and a Costas loop approach. In the DFT-based approach [1], we take advantage of the fact that we know our transmitted signal $x[k]$ is $\pm 1 \pm 1j$, or $\exp j(\frac{\pi}{4} + m\frac{\pi}{2})$, where $m = 0, 1, 2,$ or 3. We know our received signal $y[k] = x[k] \exp j(f_\Delta + \theta) + n[k]$, where f_Δ is the frequency offset between the transmitter and receiver clocks and θ is the constant phase offset of the signal. Assuming noise is negligible, we can raise our normalized received signal to the fourth power to find $(y[k])^4 = \exp j(4f_\Delta + 4\theta + \pi)$. Plotting the discrete Fourier transform of this signal, we can use the location of the peak to find f_Δ and the height of the peak to find θ. In the Costas loop method [2], we implement a control loop containing proportional and integral terms. It takes a blind approach, as we are able to generate an estimate of the error without actually knowing what data was transmitted.

Once we had corrected for the phase and frequency offsets between our transmitter and receiver modules, we decoded the complex data into a single bit stream downsampled it by sampling at the mid-point of our raised cosine pulses. Because our system had a particularly clean channel and the raised cosine decays quickly in comparison to a sinc, for example, we were somewhat robust to errors that occur as a result of choosing the wrong sampling point for each symbol. Finally, we corrected for errors that could be found with the parity bits introduced by our hamming code software and performed the necessary image processing to recover the original input image.

II. Primary Results

One of the major challenges we faced in creating our lifi system was correcting for the phase and frequency offsets between the transmitter and receiver, specifically when sending large data packets such as our standard 10 million data point packets. As discussed earlier, we tried two distinct approaches to solving this problem - the DFT method and the Costas loop method - although our final system ultimately used the DFT approach exclusively, as it showed more promising approaches. As shown in Figure 5, the Costas loop approach produced consistently reasonable constellations, but we found that any errors that occurred in early iterations of its control loop quickly compounded until the results were no longer useful, with error rates consistently around 20%. In contrast, we were able to achieve far better results with the DFT-based approach, the results of which are shown in Figure 6. After adjusting for phase offsets using this method, we were able to consistently reduce our error rates to below 10%.

The best result we have is the error corrected transmission of an Einstein image over 25 inches with an error rate just below 2% at a sampling rate of two million samples per second, giving a data rate of 80kbits/s. A side-by-side comparison between the TX data and RX data is given in Figure 7.

The successful transmission, albeit error prone, prove the possible applications of the LiFi technology in a common desktop lamp setting. Interestingly, without the error correction code, our error rate would increase to over 3% when we transmitted in the identical environment. The effectiveness of our error control code can be more dramatically seen when the error rate was higher by default, such as in Figure 8 and 9.

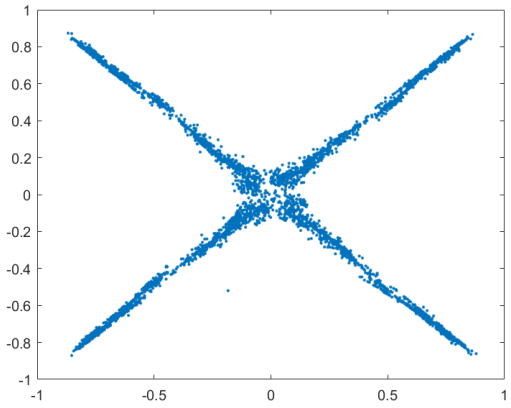

Fig. 5. The received 4-QAM constellation after using the Costas loop approach to correcting for phase offsets.

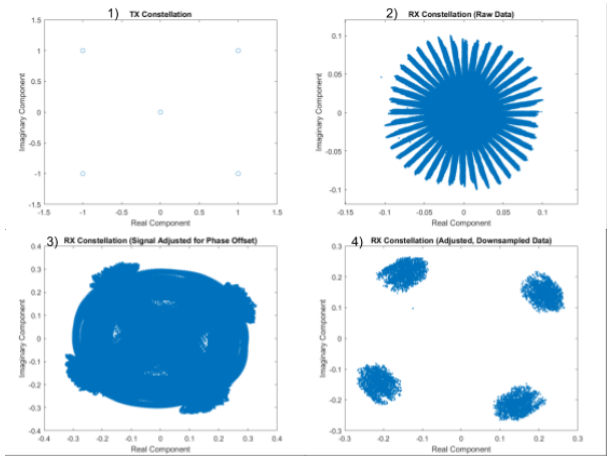

Fig. 6. In the top row, we have the raw 4-QAM transmission constellation on the left and the raw receiver constellation on the right. In the bottom row, we have the rx constellation after we have used the DFT approach to correcting for phase offsets on the left and the final downsampled rx constellation on the right.

Fig. 7. On the left is the decoded image we transmitted it through our LiFi system. On the right is the original image data.

III. Summary

We have demonstrated the possibility of visible light communication by transmitting signals in a standard LED lightbulb for 80kbit/s with the best error rate under 2 percent. To attest to the usefulness of LiFi, we produced "Lifi - An ADC Final" available now on Youtube (www.youtube.com/watch?v=TNK1LMGcHHc). However, we were not able to transmit without errors, which compound as the transmission data gets larger. Our code repository for this project can be found on github at https://github.com/QingmuDeng/ADComm_Lifi.

Fig. 8. A high error rate (≈5%) transmission over 25 inches with error control code.

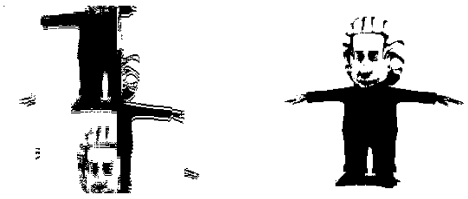

Fig. 9. A high error rate (≈10%) transmission over 25 inches with no error control code. Einstein's head has looped around the image.

1) Decay in signal strength: When an energetic photon hits an electron within the junction of the semiconductor photodetector, an electron-hole pair is created. As the hole and electron are swept to the opposite ends of the junction, a detectable current/voltage change is created. The particle-wave duality of the photon also allows it to be viewed as an electromagnetic wave. The strength of the electric and magnetic fields decays as one over the sqaure of the distance, and the energy of an electromagnetic wave is proportional to the square of the field strength. Therefore, the overall weakening of our LiFi signal is one over distance to the fourth power, increasing our error rate significantly as we try to move further away.

2) Size of data packets: No physical crystals can oscillate at exactly the same frequency just as there are no white noise because infinite power does not exist. Given the same sampling frequency, as time drags on longer for larger data packets, the difference in frequency between the TX and RX clocks would become large enough that the RX data would drift further away from where they are meant to be on the phase plane. For relatively short data packets (less than 10,000 bits, for instance), the drift is almost negligible for the transmission in under a second. If the length got over 50,000 bits, significant drift become easily visible as a four-point constellation smears

into a curve or even a circle. If the constellation happened to line up near the real and imagery axes, a spike in error rate could also be seen as our signal estimation only looks at the sign of our data. Therefore, one challenge of achieving faster data rates remains in reliable and consistent phase offset adjustment with large data packets.

3) Possible Future Work: Building upon our current work, we can imagine several novel extensions to improve the system. First, one could continue to improve the robustness of our software in correcting for the phase and frequency offsets of our transmitter and received. This might include further development of the Costas loop or an approach that mixes the advantages of the discrete fourier transform method with that of the Costas loop. To that end, we might also consider shortening the length of our data packets and developing the required software to break up a signal into multiple packets. This would limit the amount of drift in the phase and frequency offsets that could occur within the transmission of a single packet, potentially improving the robustness of our offset correction algorithms. We found that with small data packets and our relatively clean channel, the phase and frequency offsets caused almost no errors when transmitting up to 10k data points in a packet. Another way to reduce error in our system would be to continue building on top of our current error correction algorithm to reduce the effect of small numbers of bit flips. Additionally, one could imagine encoding the type of data being transmitted into the signal in order to allow us to transmit words, sounds, and videos in addition to its current image transmission capabilities. One could also consider pushing the data rate by performing some initial compression of the data, increasing the sampling rate of the USRP devices, or reducing the pulse width of our root raised cosine filters. Finally, to make our system more viable in future systems, one might try to improve the performance of the system at distances greater than the maximum of twenty-five inches that we used during development and testing.

APPENDIX

A. Hamming Code (7,4)

Introducing redundancy in the information to be transmitted is the basic principle of error control codes. By add three extra parity bits to four bits of original bit data, we can at maximum correct randomly distributed bit flipping as long as the probability of error is below $\frac{1}{7}$ or roughly 14%. As shown in Figure 10, imagine three parity bits map to three mutually intersecting circles with the four intersections mapping to the four bits of data. If all the data bits within a parity bit circle add up to an even number, the parity bit is zero; if the sum is odd, the parity is one. Once all three parity bits are calculated, the original data sequence

$$[D1\ D2\ D3\ D4]$$

is modified so it looks like

$$[P1\ D1\ P2\ D2\ P3\ D3\ D4]$$

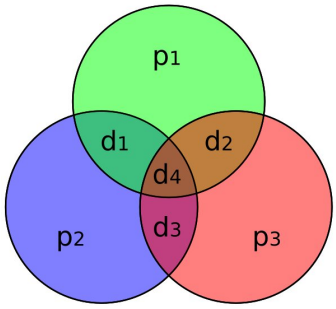

Fig. 10. Graphical representation of Hamming Code(7,4).

If any one of the data bits is flipped, there are at least two parity bits mismatch after a rough of recalculation. The intersection of the mismatched parity bits are the data bit with error.

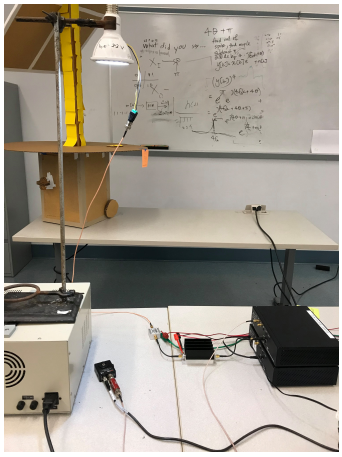

Fig. 11. The physical setup used for communicating through the high frequency modulation of LED lightbulbs in this project.

REFERENCES

[1] Govindasamy, Siddhartan. Carrier Synchronization: DFT-based Approach.
[2] Govindasamy, Siddhartan. Carrier Synchronization: Digital Costas' Loop Approach.