Adaptive Software-Defined Visible Light Communication Networks

Stefan Schmid
Dept. of Computer Science
ETH Zurich, Switzerland

Benjamin von Deschwanden
Dept. of Computer Science
ETH Zurich, Switzerland

Stefan Mangold
Lovefield Wireless
Bern, Switzerland

Thomas R. Gross
Dept. of Computer Science
ETH Zurich, Switzerland

ABSTRACT

A platform for software-based Visible Light Communication (VLC) can be built with simple hardware components and VLC can be integrated into a room's lighting system. VLC is therefore a good match for indoor setups that need to support a large number of nodes with moderate bandwidth requirements. But designing a VLC system for worst-case operating conditions is more restrictive than necessary. This paper presents a software-based VLC system that adapts link sensitivity and capacity to match environment constraints. Compared to a fixed sensing scheme, an adaptive scheme increases channel capacity by almost an order of magnitude under favorable conditions or significantly extends the communication range. This adaptive link sensitivity allows communication even without a direct line of sight between two devices, opening new opportunities to use VLC systems in novel scenarios involving different types of embedded devices – ranging from network nodes that only include a single Light Emitting Diode (LED), such as consumer electronics, toys, and wearables, to LED light bulbs that run Linux and provide an indoor VLC communication fabric.

CCS CONCEPTS

•Networks →Wireless personal area networks; Network protocol design;

KEYWORDS

Visible Light Communication; Free-Space Optics; Communication Protocols.

ACM Reference format:
Stefan Schmid, Benjamin von Deschwanden, Stefan Mangold, and Thomas R. Gross. 2017. Adaptive Software-Defined Visible Light Communication Networks. In *Proceedings of The 2nd ACM/IEEE International Conference on Internet-of-Things Design and Implementation, Pittsburgh, PA USA, April 2017 (IoTDI 2017),* 12 pages.
DOI: 10.1145/3054977.3054983

IoTDI 2017, Pittsburgh, PA USA

978-1-4503-4966-6/17/04…$15.00
DOI: 10.1145/3054977.3054983

1 INTRODUCTION

Visible Light Communication (VLC) combines illumination and communication and is therefore an attractive technology for a ubiquitous indoor communication system. Already deployed lighting infrastructure can be reused for communication purposes without effort. Light sources built from Light Emitting Diodes (LEDs) are an attractive choice for such a VLC system as LEDs are inexpensive, readily available, economical, and can be used to emit light (for transmission of data) as well as to sense light (for receiving data). Furthermore, VLC does not interfere with the use of the scarce radio spectrum and cannot be easily overheard from another room – to observe a message exchange that is communicated via a VLC channel, the eavesdropping party needs either direct or indirect (through reflection) line-of-sight access.

To allow the use of VLC for emerging scenarios like the Internet of Things (IoT), networked toys [4, 21], or sensor networks that have modest bandwidth demands, the endpoints of a VLC system must be as simple as possible. A minimal VLC node should be able to operate just with a single LED. Battery-powered devices that only include a single LED (or a small number of LEDs), such as consumer electronics, toys, electronic tags, or wearables are examples of possible endpoints of a VLC network.

LED light bulbs[1] combine multiple LEDs and are significantly brighter than single LEDs radiating light in different directions; they are low-cost [18, 23] and have been proposed for many novel indoor applications. To enable the sensing of incoming signals from other light-emitting devices, LED light bulbs can be enhanced with simple light receiving electronics based on photodiodes. Such LED light bulbs are powerful enough to establish a communication link over several meters, but are still based on a software-based Physical (PHY) and Medium Access Control (MAC) layer [19], making the system compatible with low-cost LED-only systems and thereby allowing the communication system to connect a wide range of devices.

The key insight that allows different LED-based systems to communicate with each other (LED-only devices or LED-based light bulbs) is to let software handle the communication protocols. The software-based PHY layer faces two core problems: (i) all devices within range must be synchronized, i.e., agree when to engage in communication, and (ii) the analog signal detected by an LED or a photodiode must be sensed and digitized. This latter issue imposes

[1] Modern indoor LED light sources come in various shapes and designs, often still in the shape of a conventional light bulb. Our system can be used with such light bulbs, or added to any other LED light source of arbitrary shape. We use "light bulb" as synonym for all kinds of LED light sources.

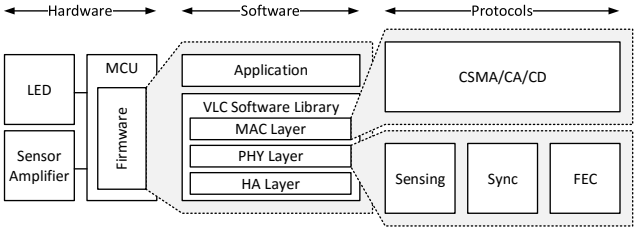

Figure 1: The VLC platform uses only simple hardware components. All communication protocols are implemented in software to provide a flexible research and prototyping platform.

constraints on the transmitter but it is really the receiver design that determines the properties of the communication system. Both of these problems, synchronization and sensing, can be accomplished in software, resulting in a flexible and adaptive communication system that can serve as a platform for the IoT. The system's ability to adapt link capacity based on channel conditions (i.e., based on the environment, the capabilities of and the distance between the communication partners) is critical for a practical VLC system. The paper presents

- a discussion of the software architecture that allows inter-operation of heterogeneous devices (Section 2);
- the description of an adaptive software-based VLC system to allow communication between simple single-LED devices and VLC-enabled light bulbs (Section 3); and
- an evaluation of the communication link and protocol for different network setups (Section 4).

2 SOFTWARE-BASED VLC

This section describes the hardware and software components necessary to build a flexible VLC system. The basic building blocks are a receiving front-end (LED or photodiode), a transmitter (LED), and a programmable microcontroller (MCU) running the communication protocols for the PHY and MAC layers.

2.1 Hardware Independence

A VLC system that connects devices, e.g., in the context of sensor networks and the IoT must have minimal hardware requirements (so that low-cost/low-part count implementations are possible) yet allow inter-operation with other devices. A software-centric approach is attractive as it isolates hardware dependencies and allows the reuse of the software base. As shown in Figure 1, most of the layers are device independent (i.e., PHY, MAC), only a thin Hardware Adaptation (HA) layer is device-specific. This setup supports two popular hardware platforms: either single LEDs are used as receiving front-end, or photodiodes, possibly connected to an amplifier, sense incoming light. The transmitter is in either case a single LED or a group of LEDs, e.g., as found in an LED light bulb.

This software-centric approach offers a communication system running on off-the-shelf and low-cost microcontrollers but is still robust and stable and fulfills the data rate requirements for scenarios requiring moderate data rates. Many devices already use a microcontroller and LEDs and could benefit from VLC by only relying on software changes. The software-based design yields many benefits (in addition to support for different kinds of stations): the system is flexible (software development proceeds faster than construction of hardware) and extensible (easily realized by software customization).

2.2 Hardware Building Blocks

To allow the reader to follow the discussion of adapting the link sensitivity, we describe here the hardware building blocks of the VLC platform. It relies only on a few low-cost and off-the-shelf hardware parts summarized in the following sections.

2.2.1 Transceiver: LED. The physical data transmission can be implemented using a single LED as a transmitter and receiver. Emitting a light pattern is straightforward: simply turn the LED on and off based on predefined timings. Using an LED to sense light can be accomplished by charging it in reverse-bias and measuring the remaining charge after some time has passed. The amount of charge left over is directly influenced by the photoelectric effect, and thus approximately proportional to the amount of light shone onto the LED during the period of measurement. This behavior can be used to decode bits encoded in on-off light patterns [5, 20]. A low-complexity LED-only system can be built with only an LED as a transceiver, and a microcontroller for processing.

2.2.2 Receiver: Photodiode and Amplifier. A dedicated light-to-voltage converter is an alternative to transmitting and receiving with a single LED. Such a device essentially consists of a photodiode coupled with a transimpedance amplifier (built from a single operational amplifier and only a few passive components). A dedicated sensor has the main advantage that much fainter signals can be detected and thus allows for significantly higher transmission distances. In contrast to a setup based on a single LED, charging the sensor is no longer necessary, since the measurement with a light-to-voltage converter directly yields a voltage value. This setup is more involved, since more components are needed, yet it still only requires two microcontroller pins and only slight changes in the software part responsible for sensing the input. A dedicated sensor is mandatory when the LEDs employed by the target device do not have good receiving characteristics, e.g., LED light bulbs cannot receive through the high power white LEDs, therefore dedicated sensors are employed [18, 23].

2.2.3 Microcontroller. All the necessary PHY and MAC layer protocols run on a single microcontroller. Any processor with the necessary peripherals: (timers and an Analog-to-Digital Converter (ADC)) can be used. The computational requirements are moderate, e.g., 8-bit AVR processors [2] or 32-bit ARM M0 processors [24] can handle the processing load. The communication protocols are built on top of a hardware abstraction to support multiple platforms and make adopting of new hardware architectures effortless. The microcontroller operates the LED, reading out LED or sensor, encodes and decodes data, runs forward error correction, and handles MAC layer collision avoidance and retransmissions. Using a single processor for the complete communication system and providing a flexible software platform makes the solution applicable to many already existing devices.

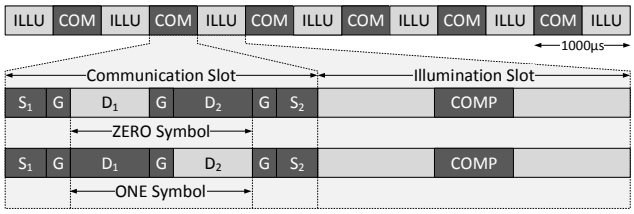

Figure 2: Slot-based communication protocol. *ILLU* and *COM* slots alternate, providing communication and networking capabilities while still being able to provide illumination. Dark gray areas symbolize no light output and light gray areas depict light emission.

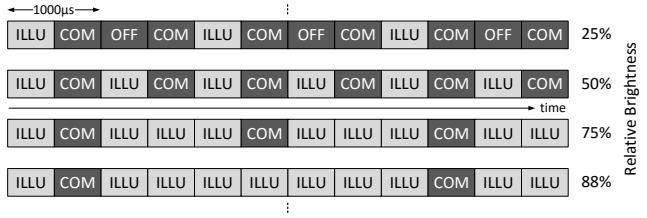

Figure 3: Adaptive light output for slot-based communication protocols. Light brightness can be controlled by adding or removing illumination slots. Dark gray areas symbolizes no light output and light gray areas depict light emission.

2.3 Software Building Blocks

The software part of the presented VLC platform implements a complete PHY and MAC layer. All functionality is encapsulated into a library – *libvlc* – and presented through a well-defined API to user programs. The following paragraphs summarize the core parts of *libvlc*.

2.3.1 Slot-based Communication and Illumination. As shown in Figure 2, the implementation of the PHY layer utilizes fixed-length periods of a duration of 1 ms, containing communication (*COM*) and illumination (*ILLU*) slots. The dark gray areas symbolizes no light, where the light gray regions depict constant light output. The *COM* slot is used to handle synchronization (explained in the next paragraph) and data modulation, and the *ILLU* slot handles illumination (and possible light compensation). Thus, at a frequency of 1 kHz, the brightness of the transmitting (or receiving) LED appears constant to a human observer and can therefore be used as a lighting device without flickering. To transmit one bit of data, ON-OFF Keying (OOK) with Pulse Width Modulation (PWM) is used. Light output is either enabled during D_1 or D_2, encoding a `ZERO` or `ONE` symbol. For decoding, no threshold separating a `ZERO` from `ONE` symbol is needed since light values resulting from D_1 and D_2 can be directly compared. How light is actually measured using different setups is omitted here since it is explained in detail in Section 3.2. The data slots are surrounded by guard (G) intervals to prevent light leaking into neighboring intervals. While transmitting data, the additional light output of D_1 or D_2 must be compensated during the next illumination slot (*COMP* interval) to maintain constant illumination and to prevent the light source from flickering. This slot-based concept clearly separates communication and illumination, providing a stable PHY layer that enables distributed networking protocols like Carrier Sense Multiple Access with Collision Avoidance CSMA/CA or even Collision Detection (CD) [19].

2.3.2 Synchronization. Figure 2 shows that the measurement at the receiver must occur in a perfectly aligned and synchronous way, which is achieved using a software-based synchronization method [19]. There are two reasons why synchronization is needed: (1), the initial start-up offset of two devices needs to be compensated and (2), the accumulated imprecision of inexpensive oscillators needs to be counteracted. The synchronization slots S_1 and S_2 measure light from neighboring devices. When the measuring device is perfectly in sync with its neighbors, the amount of light sensed in both synchronization slots is equal, since no light is emitted during S_1 and S_2. In the case where all neighbors are idle, if the measuring device is ahead of its (synchronized) neighbors the amount of light detected in S_1 will be less than in S_2. Similarly, if the measuring device is lagging behind its neighbors, the amount of light detectable in S_1 is higher than in S_2. This mechanism not only allows a device to detect whether it is out of sync with its neighbors, but it also enables the device to figure out whether the misalignment is positive or negative. This information is then used to adapt the length of the last *ILLU* slot slightly to shift towards alignment. This step is repeated several times until the device is (again) aligned with the pattern of its neighbors.

2.3.3 Adaptive Brightness. With the scheme shown in Figure 2, a lighting device outputs light only half of the time, being only half as bright as it could be when not following the presented communication protocol. If the light source only needs to provide static brightness, the lighting system can be designed such that 50 percent duty cycle is equal to the target brightness. An adaptive light source can follow the concept depicted in Figure 3. Light output can be reduced replacing *ILLU* slots with additional slots where no light is emitted or even replace *ILLU* slots with *COM* slots to increase communication capacity. To increase brightness, additional *ILLU* slots can be introduced at the cost of *COM* slots and communication performance.

2.3.4 Forward Error Correction. The system uses highly optimized Reed-Solomon [17] codes for Forward Error Correction (FEC). The codes are directly implemented on the microcontroller as part of the PHY layer and operate on the byte level and can correct byte errors resulting from 1 to 8 bit errors within a single byte. Reed-Solomon adds a number of parity bytes (in our case 16, but since it is a software implementation, it can be changed to any reasonable power of 2). This redundancy allows to find and correct 8 byte errors, or correct 16 byte errors if the error location is known. Since the PHY layer decoder is also part of the same software, it is possible to mark bytes where bit decoding decisions were close for later correction. The algorithm is implemented to start step-by-step the calculations while receiving data. FEC is controlled by a threshold value. If the PHY layer payload is below this threshold, only a 16 bit Frame Check Sequence (FCS) is used and if the payload is equal or larger than the threshold, FEC is enabled. The receiver is informed via a flag in the PHY header.

3 ADAPTIVE SENSING

The modular and layered structure of the VLC platform allows easy porting of the VLC system to different microcontroller families and architectures. As of the moment, an AVR (ATmega328p, [2]) and ARM (STM32F0, [24]) are supported by the HA layer. Neither of them currently reaches its limit in terms of computational power and memory.

Therefore good channel conditions allow a more densely modulated medium (shortening data interval duration, at the cost of additional computation), which will increase the link capacity of the system. This section describes how a fully adaptive system that dynamically adapts to current channel conditions and capabilities of participating communication partners can be built using the building blocks of Section 2.

3.1 Physical Layer Modes

To make use of available processing power and good channel conditions, the data interval duration within a communication slot can be shortened to build communication slots with different numbers of data interval pairs to increase channel capacity. These PHY modes are shown in Figure 4; different PHY modes can be built by changing the number of data intervals and their duration inside a communication slot. The top of the figure shows the `SINGLE` or `BASIC` PHY mode; it is identical to the one shown in Figure 2 and uses identical timings. By equally partitioning D_1 and D_2 into two intervals each, PHY mode `DOUBLE` contains four data sub-intervals $D_{1,1}$, $D_{1,2}$, $D_{2,1}$, and $D_{2,2}$. By repeating this procedure, the construction of PHY mode `QUAD` and `OCTA` is straightforward (see Figure 4). There are in total 16 sub-intervals for the `OCTA` PHY mode. Since the overall communication slot duration stays the same, the sub-intervals are getting shorter for higher PHY modes, e.g., 21 µs for an `OCTA` sub-interval. Shorter slots mean that less light is received (for LED-only systems), and the synchronization must be (more) accurate. Hence, the better the channel condition the higher the PHY mode that can be used.

3.1.1 Data Encoding. For all PHY modes, two sub-intervals $D_{1,i}$ and $D_{2,i}$ encode a data symbol `ZERO` or `ONE`. Figure 5 shows an example of an `OCTA` communication slot. The dark gray background symbolizes lights turned off, where the light gray areas mean lights turned on. Below the slot visualization, the light on/off signal is shown. The corresponding sub-intervals are labeled by the same numbers. Always $D_{1,k}$ and $D_{2,k}$, for $k \in \{1, \ldots, N\}$, encode a single symbol. Corresponding sub-intervals are always encoded with inverted signals so that for the decoder a simple light value comparison (as explained in Section 2) is enough. Spatially splitting up the corresponding sub-intervals makes the decoder more robust against light leaking into neighboring slots during imprecise synchronization.

3.1.2 Theoretical Speedup. The theoretical improvement of the data throughput for a given slot layout with N sub-intervals and a payload of size P is obtained by dividing the throughput achieved with N sub-intervals by the throughput of the base case with only a single sub-interval (Equation 1).

$$S_{N,P} = \frac{G_{N,P}}{G_{1,P}} \tag{1}$$

The throughput itself is calculated by dividing the payload size by the time it takes to successfully transmit that payload ($T_{N,P}$), as shown in Equation 2.

$$G_{N,P} = \frac{P}{T_{N,P}} \tag{2}$$

The time $T_{N,P}$ of a successful transmission of a payload of size P (using N sub-intervals) depends on whether FEC is used for resilience or not and described by Equation 3 (with FEC) or Equation 4 (w/o FEC). H_P is the 4 B PHY header of a frame includinga the Start Frame Delimiter (SFD), which always needs to be transmitted in the basic PHY mode. The 4 B MAC header is denoted by H_M. Depending on whether FEC is used or not, the MAC header includes either 16 B FEC or 2 B CRC, respectively. We use a superscript (*FEC* or *CRC*) to distinguish the two cases. Agaian, P signifies the payload size, which amounts to 200 B at its maximum, and N is the number of sub-intervals used.

$$T_{N,P}^{FEC} = \frac{\left(H_P + \frac{H_M^{FEC}+P}{N}\right) + \left(H_P + \frac{H_M^{CRC}}{max(1,N-1)}\right)}{Capacity} \tag{3}$$

$$T_{N,P}^{CRC} = \frac{\left(H_P + \frac{H_M^{CRC}+P}{N}\right) + \left(H_P + \frac{H_M^{CRC}}{max(1,N-1)}\right)}{Capacity} \tag{4}$$

The first addend of the numerator in Equation 3 and Equation 4 describes the frame containing the message of payload P. Its MAC header H_M, as well as its payload, can be sent using the given number of sub-intervals N, while the PHY header H_P is transmitted in the basic PHY mode. The second addend in the numerator represents the MAC layer acknowledgment (ACK), which needs to be received by the sender, before the transmission is considered successful and a new message can be sent. The ACK does not have a MAC payload and its MAC header, which always includes CRC, is transmitted using one PHY mode lower then its corresponding data frame. Dividing the numerator by the channel capacity, which is equal to 1 kbit/s (for the PHY mode `SINGLE`), yields the time $T_{N,P}$ taken for a payload of size P, with N sub-slots, to be transmitted successfully.

Table 1 shows the theoretical transmission times, as well as the resulting throughput and maximum theoretical speedup, for values between 1 and 8 for the number of sub-slots N, based on Equations 3 and 4. The capacity, as opposed to the throughput, grows linearly with the number of sub-intervals N, reaching 8 kbit/s at $N = 8$. Yet, the measure of throughput is far more significant on the MAC level and yields a more meaningful assessment of the implementation or comparison with other systems.

3.2 Receiving Strategies

Different receiving and decoding methods can be used depending on the receiving channel (LED or dedicated sensor) and the microcontroller's capabilities. Figure 6 summarizes the three approaches (for PHY mode `OCTA`, analogue for lower modes). The first (top) example (1) shows the layout of the communication slot for a receiving LED channel. The first row shows the on/off light signal of a transmitter, and the second row visualizes the reception logic. Since the LED is not paired with an electrical amplifier, the channel cannot directly be sampled. The LED is first charged in reverse

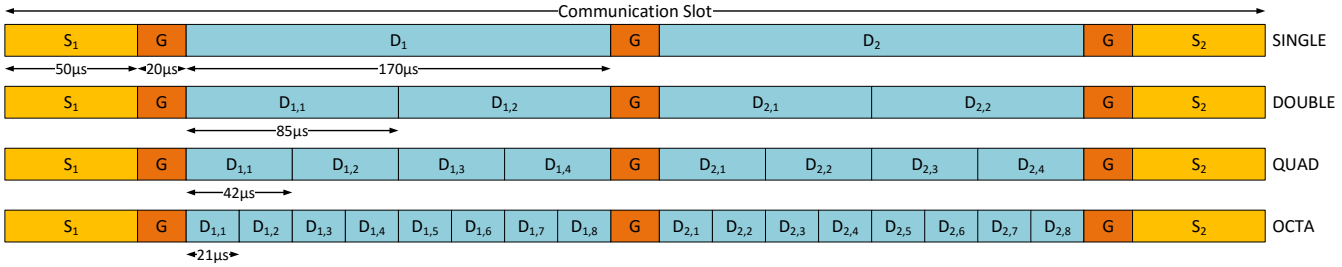

Figure 4: Visualization of the four available PHY modes. Construction is done by simply splitting one data interval into two, repetitively. More data intervals per communication slot increase channel capacity.

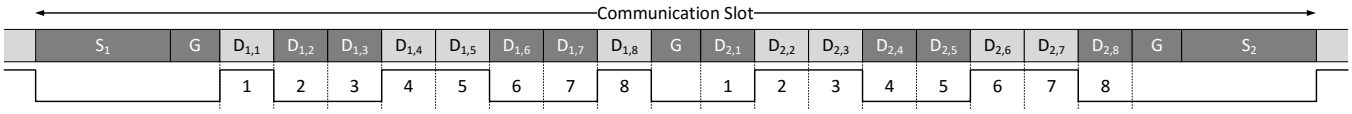

Figure 5: Example communication slot, showing 1 byte of data. The first row shows the slot layout and the second row the on/off light signal. Intervals $D_{1,k}$ and $D_{2,k}$ together encode a `ZERO` or `ONE` symbol.

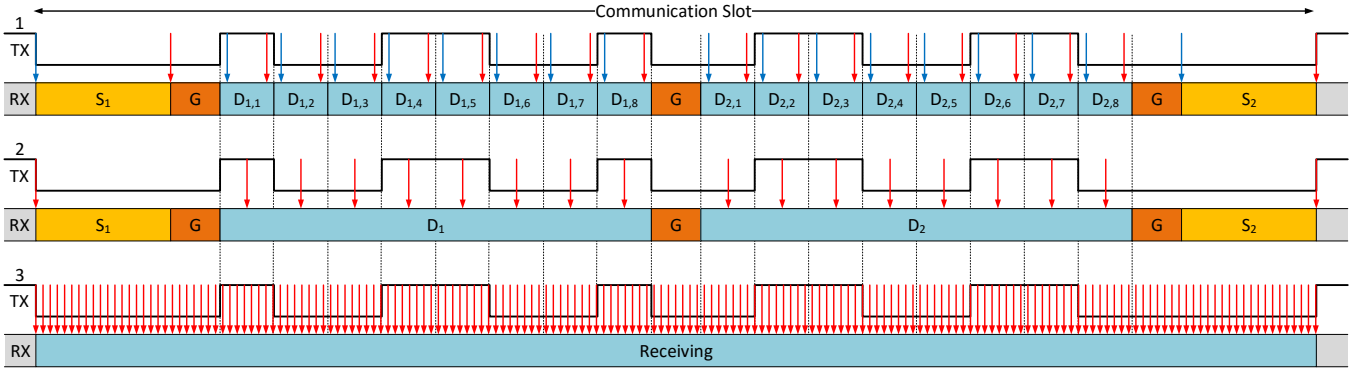

Figure 6: Different receiving methods based on available hardware and peripherals. (1) LED-based receiving, (2) dedicated sensor-based receiving, and (3) timer-based sampling with DMA [from top to bottom].

Table 1: Theoretical transmission time, throughput and speedup, using a payload of maximum size (200 B), depending on the number of sub-intervals; with only FCS or with FEC.

	FCS			FEC		
N	T_N [s]	G_N [b/s]	S_N	T_N [s]	G_N [b/s]	S_N
1	1.76	909.12	1.00	1.87	854.72	1.00
2	0.94	1709.44	1.88	0.99	1612.88	1.89
3	0.64	2510.48	2.76	0.67	2371.52	2.77
4	0.49	3252.00	3.58	0.52	3076.96	3.60
5	0.41	3944.80	4.34	0.43	3738.32	4.37
6	0.35	4594.16	5.05	0.37	4360.48	5.10
7	0.31	5204.48	5.72	0.32	4947.04	5.79
8	0.28	5779.12	6.36	0.29	5500.96	6.44

bias and the remaining voltage is measured after some time[5, 20]. The remaining voltage correlates to the received light since the charge. The blue arrow indicates a charge and the red arrow denotes the starting point of a voltage measurement. Therefore, every sub-interval needs its own processing; as a consequence higher PHY modes (`QUAD` and `OCTA`) are only available on more recent MCUs (e.g., the ARM STM32F0). The second communication slot (2) shows the implementation of a dedicated sensor channel. Here, the voltage can directly be read from the output of the transimpedance amplifier. The processor triggers a sample for each sub-interval timed at the center of the interval. Samples can easily be collected and later be processed at the end of the communication slot. For a microcontroller supporting Direct Memory Access (DMA), the last shown method (3) can be used. The complete communication slot can be simplified into one state. A timer can be instructed to trigger samples (reading voltage values) during the complete slot and have them transferred to memory using DMA. The samples are processed later (during an *ILLU* slot) where they are aged and assigned to synchronization and data sub-intervals according to the used PHY mode. This method removes all processing from the communication slot, and the microcontroller can be used to process higher layer protocols during this time. This method also opens up space for further improvements discussed in the next section.

3.3 Synchronization Correction

PHY mode `OCTA` requires precise synchronization since the data sub-slots are quite short. The synchronization is done in software

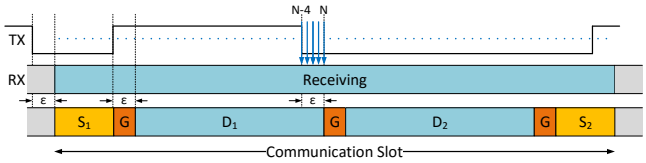

Figure 7: Transmitter and receiver are slightly misaligned by ϵ. Using the DMA sampling method, the edge of a data interval can be detected and the synchronization offset for the payload sent in a higher PHY mode can be calculated.

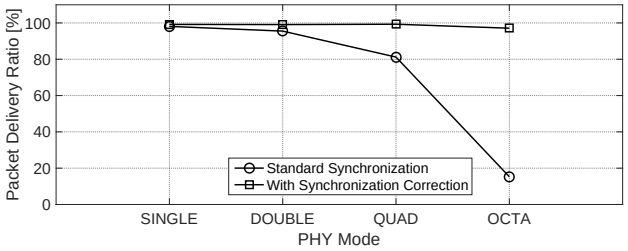

Figure 8: Packet delivery ratio for a light bulb setup for the different PHY modes. Using the synchronization corrections drastically improves the successful reception for mode `QUAD` and `OCTA`.

and only relies on the simple method described in Section 2, and therefore might not be as accurate as needed. Figure 7 shows such a scenario where the synchronization offset is ϵ. When using the DMA sampling method, this offset can be recognized and corrected. During the transmission of the PHY header, PHY mode `SINGLE` is always used. For all PHY header bits, there is either a falling edge at the end of D_1 or a rising edge at the beginning of D_2. Since it is known where this edge should be (if perfectly synchronized) and the edge can be detected using the collected samples, the resulting offset (in samples) can be calculated and be considered when assigning samples to sub-intervals. The example in Figure 7 shows that the edge is expected at sample N but detected at sample N-4. This offset is calculated for all PHY header bits and averaged to improve the PHY payload decoding. Figure 8 shows an experiment where the packet delivery ratio for a transmitting and receiving light bulb is measured. When using only the standard synchronization method, for PHY mode `QUAD`, 20 percent of the packets are lost, and for PHY mode `OCTA` even more than 80 percent. Applying the offset correction leads to drastically improved results: nearly every packet is received successfully. This method helps when a sensible sensor setup is used where already small changes in light intensity and temperature can offset synchronization, e.g., in a light bulb setup. This method shows another way to improve capacity by adaptively selecting the correct samples.

3.4 Frame Format

Figure 9 depicts the frame format used to transmit data. The PHY header, mainly containing operational meta-data in the first byte, as well as the rest of the frame's length, is made resilient using 1 B CRC. The PHY header is preceded by a 1 B SFD (not shown), which serves to detect the beginning of a frame. Following the PHY header, the MAC header contains information about the sender,

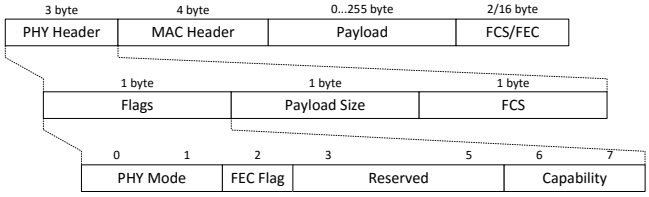

Figure 9: PHY and MAC frame format used by *libvlc*. The PHY header is used to encode PHY mode, activated FEC, and capabilities.

receiver, and the MAC layer frame itself. The aforementioned 8 B header can be followed by up to 200 B of payload, which is either concluded with 2 B of Frame Check Sequence (FCS) or 16 B for FEC, depending on whether FEC is being used or not. The presence (or absence) of FEC can be defined in the PHY header.

The first byte of the PHY header encodes the employed PHY mode for each transmission, using the first two bits. The third bit defines if FEC or just FCS is used, so that the receiver can process the incoming data accordingly. Bit 3 to 5 are not in use yet, and the last two bits define the capabilities (supported PHY modes) of the transmitting device. These two bits are always transmitted with every frame, informing all neighbors about the PHY modes supported by this specific device. The PHY header is always transmitted using the `SINGLE` PHY mode to allow the receiver to decode its contents and then switch to the requested PHY mode to receive the PHY payload.

3.5 Dynamic PHY Mode Adaptation

To be able to react to changes in the environment, we leverage the information available from collected statistics packet statistics to switch between PHY modes in an automated manner. One of the first published algorithm to make use of transmission and failure counters to adapt the transmission mode was Adaptive Rate Fallback (ARF) [9]. This algorithm works by periodically checking whether a faster transmission mode would work, using so-called probing packets. Whenever a certain number of probing packets are transmitted successfully, the sender switches to the faster mode. Reversely, when the number of lost frames exceeds a given threshold, the sender scales back to a slower mode.

In a comparison of different adaptation schemes [13] and the introductory paper for ARF [12], the disadvantages of ARF are elaborated. The major drawback of ARF is its inability to leverage long-term stability. ARF probes the channel capabilities frequently, regardless of the result of former inquiries. This behavior leads to an unnecessarily high number of failed probing packets, which diminishes throughput. Adaptive ARF (AARF) tries to eliminate this problem by adapting the interval between probing. Whenever a given number of probing packets fail, the threshold determining when to send the next probing packet is doubled (up to a certain limit). This yields a substantial reduction in wasted probing packets in a stable environment and still allows AARF to use a faster PHY mode, should the conditions improve.

Since the scheme of AARF is straightforward to implement, given the information available from the statistics, an adapted version of AARF is used in *libvlc*.

4 EVALUATION

This section discusses measurements for different VLC-enabled devices and setups. LED-only devices with different capabilities, light bulbs communicating with LED-only devices, and light bulb networks are evaluated to show the effectiveness of the presented adaptive PHY layer together with the CSMA/CA-based MAC layer. All results show MAC layer throughput, and the error bars in all figures indicate the standard deviation.

4.1 LED-to-LED Communication

LED-only devices, which use the same LED to send and receive, are evaluated for point-to-point communication for multiple distances and in a network with up to 12 devices.

4.1.1 Point-to-Point. The point-to-point performance is measured in terms of throughput for an LED-only setup. Here, both sender and receiver are based on ARM M0 prototype boards, using a single LED each to transmit and receive.

Figures 10 to 13 show the results of an LED-only setup for fixed PHY modes `SINGLE` through `OCTA`. For each of those plots, messages of various sizes between 1 B and 200 B are transmitted from a dedicated sending device and acknowledged by a dedicated receiver (the MAC protocol always uses a more resilient PHY mode than the one which was used to receive). Data frames are transmitted at saturation, i.e., the next frame is immediately sent whenever the previous frame is done (either ACK'ed or timed out). Only frames that have been successfully acknowledged count towards throughput. For each fixed PHY mode, as well as the adaptive mode, the sender tries to transmit messages of size {1, 10, 20, 50, 100, 150, 200} B using an FEC threshold of 32.

The plots show that PHY mode `SINGLE` (Figure 10) works reliably between 0 cm and 160 cm. Using 200 B messages, we achieve a maximum throughput of about 0.85 kbit/s for the aforementioned distances. It matches the theoretical maximum throughput as calculated in Section 3.1.2. The furthest distance where successful transmissions are still possible is at about 170 cm, albeit with a throughput of about 0.1 kbit/s.

PHY mode `DOUBLE` (Figure 11) works reliably between 0 cm and 140 cm, for a maximum throughput of 1.55 kbit/s. Again, this result lies within the magnitude of the theoretical maximum of 1.61 kbit/s, deviating from the calculated value by 4 %. Communication is possible up to 160 cm, but with heavy losses in throughput.

PHY mode `QUAD` (Figure 12) works reliably between 0 cm and 90 cm, for which the maximum throughput is 2.85 kbit/s. Comparing this value to the theoretical maximum of 3.07 kbit/s, it can be seen that 92 % of the theoretically possible performance is achieved.

Maximum throughput is achieved with PHY mode `OCTA` (Figure 13). The throughput has increased to 4.72 kbit/s, which is equal to 86 % of its theoretical maximum of 5.50 kbit/s, and communication is possible up to 60 cm.

To summarize, the results reflect the assumption that longer messages yield better throughput values for distances below a certain distance. The point at which shorter messages perform better than longer ones is reached when the probability for bit errors is so high that sending shorter messages with a higher framing overhead yields a higher expected throughput, due to a smaller number of lost messages.

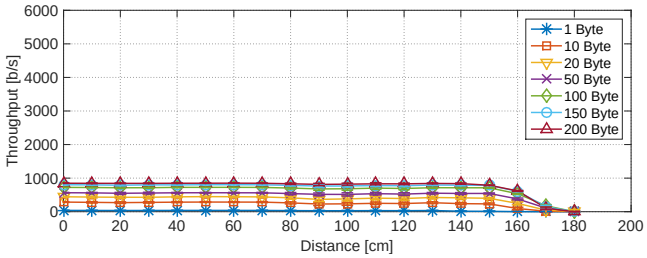

Figure 10: Throughput for LED point-to-point communication for different packet sizes and variables distances using PHY mod `SINGLE`

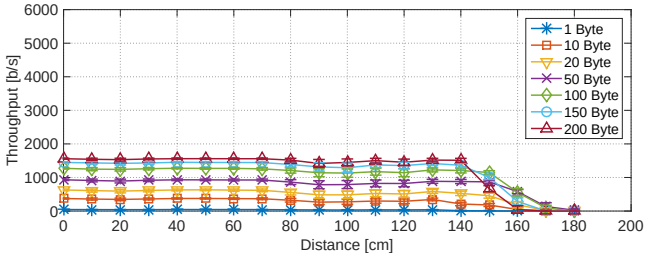

Figure 11: Throughput for LED point-to-point communication for different packet sizes and variables distances using PHY mod `DOUBLE`.

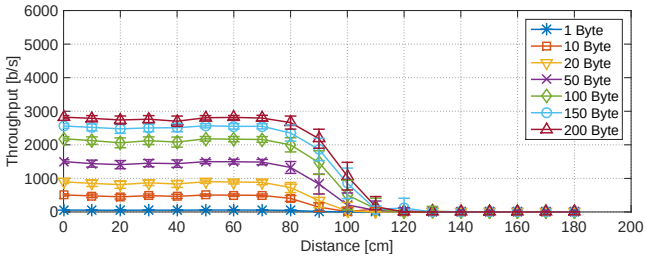

Figure 12: Throughput for LED point-to-point communication for different packet sizes and variables distances using PHY mod `QUAD`.

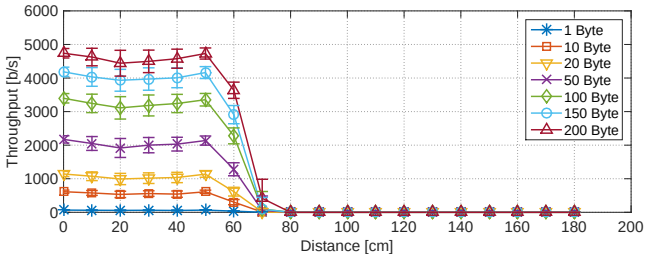

Figure 13: Throughput for LED point-to-point communication for different packet sizes and variables distances using PHY mode `OCTA`.

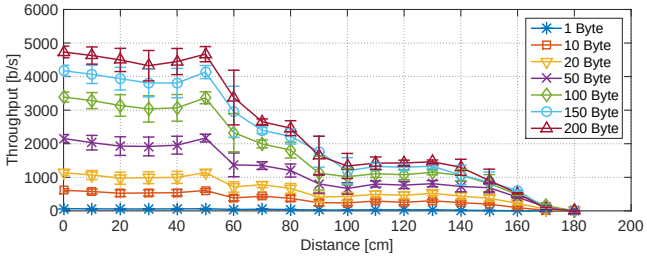

Figure 14: Throughput for LED point-to-point communication for different packet sizes and variables distances using adaptive PHY mode selection.

For an evaluation of the effectiveness of adaptivity, we compare the adaptive PHY mode selection to the results achieved with a fixed PHY mode. The throughput is measured again with the same setup as before, but without specifying the mode. Thus, *libvlc* uses the modified AARF algorithm to select the optimal PHY mode dynamically. The results of this experiment are shown in Figure 14.

The adaptive PHY mode selection procedure always starts at the slowest mode and probes for the applicability of the next higher mode after a couple of frames have been transmitted successfully. If the probing packet succeeds, the PHY mode is changed accordingly and an attempt to transmit faster occurs again some frames later. Should the probing packet fail, the current PHY mode is maintained and the waiting period for the next probing packet is doubled. This behavior has the advantage that it is always able to detect the fastest possible PHY mode, as becomes evident when comparing the graphs. One drawback of this tentative approach is the increased uncertainty in the transmission duration and thus, throughput, which is visible in the considerably higher error bars in Figure 14. Nonetheless it shows that the adaptive method is succeeding to chose the best available PHY mode at all distances resulting in optimal throughput.

4.1.2 Network. In this measurement series, we evaluate the adaptive PHY mode selection in a network of 12 devices arranged in a star shape, using LED channels [21]. The network consists of 6 AVR and 6 ARM boards, which are added one by one to the network in an alternating fashion. For each number of devices, throughput is measured for the complete system for various message sizes.

The AVR boards are capable of running PHY mode `DOUBLE` and the ARM boards support up to PHY mode `QUAD` in this configuration. These capabilities are set by the measurement application and used by *libvlc* to transmit at optimal rates. For this measurement, all devices transmit at saturation, selecting a random receiver (not themselves) for each transmission.

The throughput resulting from these measurements for each number of devices and messages sizes can be seen in Figure 15. When only two devices (one AVR and one ARM board) are present, *libvlc* can transmit using PHY mode `DOUBLE`, which is the maximum supported by both devices. The two devices can transmit using only small contention windows, since collisions in this setup are unlikely, which leads to the maximum throughput possible defined by PHY mode `DOUBLE`.

Adding more devices to the network diminishes the throughput due to the higher probability of collisions. The relatively stable behavior of the system over a large range of number of devices (from three to nine) shows that the collision avoidance provided by the MAC layer works reliably. Nevertheless, it can be seen that using more than nine transmitters considerably diminishes the total system throughput, since collisions and congestion become more likely. Additionally, it becomes evident that the adaptive PHY mode selection's behavior is too conservative. Figure 16 shows throughput resulting from the use of the optimal PHY mode for each device, instead of using the adaptive PHY mode selection. The average throughput reached is around 1.2 kbit/s for this mixed setup.

Figure 17 shows the result of measurements using the same star setup for 2 to 12 devices, but now using only devices with ARM

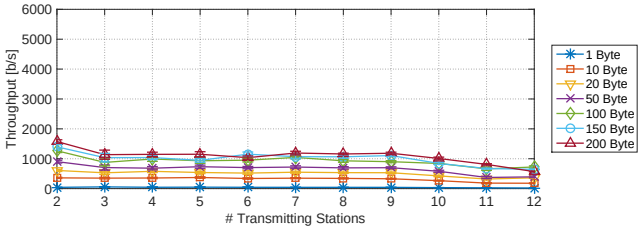

Figure 15: Throughput for a network of mixed LED-based devices (AVR and ARM) for different numbers of participating devices and packet sizes, using adaptive PHY mode selection.

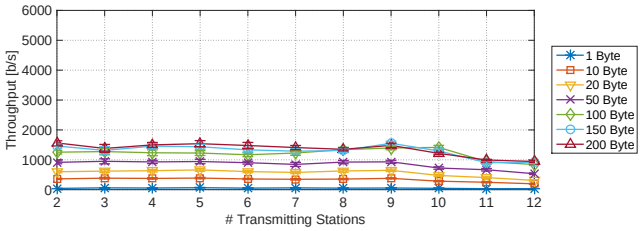

Figure 16: Throughput for a network of mixed LED-based devices (AVR and ARM) for different numbers of participating devices and packet sizes, using fixed maximum available PHY modes.

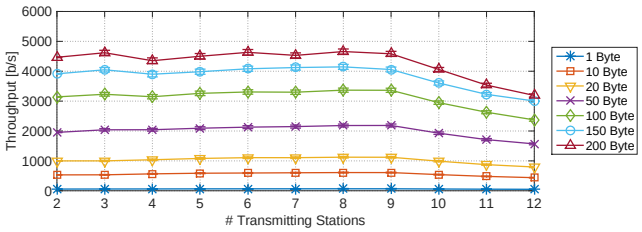

Figure 17: Throughput for a network of LED-based ARM devices for different numbers of participating devices and packet sizes, using adaptive PHY mode selection.

MCUs. Devices employ adaptive PHY mode selection and randomly select a communication partner for each transmission. Since now only ARM-based devices are present, which can use PHY mode `OCTA`, throughput is significantly higher than with a mixed setup. The network shows the typical behavior of a CSMA/CA protocol. Throughput is slightly increasing (when adding more devices) until saturation is reached; at this point, collisions start to be more probable and throughput decreases again for each additional device, but communication stays stable. For 200 B packets, throughput can reach up to almost 5 kbit/s for 9 devices participating in the network.

4.2 Light Bulb-to-LED Communication

Since *libvlc* does not only support LED-LED channels, but also allows the use of LED-only devices in combination with LED light bulbs with dedicated sensors, we want to investigate whether *libvlc* is still compatible with a heterogeneous hardware setup. We measure the performance of a system where the light bulb sends messages to an LED receiver. Unlike before, we send broadcast messages for these measurements to also prsent the performance for unacknowledged data transmission.

Figures 18 to 21 show the results of the light bulb-to-LED measurements. Since the adaptive PHY mode selection is based on transmission statistics, it cannot be used for broadcast messages due to the lack of ACKs. It becomes evident from the plots that the maximum throughput is higher than with acknowledged messages, which is understandable, since the sender does not need to wait for an ACK before sending the next message. The difference in throughput is most noticeable for PHY mode OCTA, where the LED setup with ACK achieves a maximum throughput of about 4.72 kbit/s and the light bulb-to-LED setup reaches 5.46 kbit/s for 200 B messages, an increase of about 15 %. Thanks to the higher light intensity emitted by the light bulb, this setup can span further distances than the LED-only case. Yet, it is limited by the small-angle sensitivity of the LED, which leads to a relatively modest improvement in the maximum distance of about 20 cm. The results show that the protocols also work for heterogeneous networks.

4.3 Light Bulbs

This section presents measurements gathered with LED light-bulbs equipped with sensors for reception. In addtion, all light bulbs are equipped with a Wi-Fi-enabled Linux board for control and measurement collection [18].

4.3.1 Network. Whereas Section 4.2 reported results for a mixed setup of LED light bulbs and LED-only devices, we now evaluate the performance of a network of multiple light bulbs. For this experiment, six light bulbs are arranged in a star-like shape (each around 2 meters apart from the center), using the devices on the edges as dedicated senders and the light bulb in the center of the star as a dedicated receiver. Figures 22 to 26 show the results for each fixed PHY mode as well as the effect of adaptive mode selection. All plots show the total network throughput, which is equal to the sum of the throughput achieved at each sender.

The results for all four PHY modes depict a stable throughput throughout the varying number of senders. PHY mode SINGLE can transmit at 0.85 kbit/s maximum (its theoretical maximum) using five senders. The performance of PHY mode DOUBLE (1.60 kbit/s, using three senders) is also within a 1 % margin of its theoretical maximum. PHY mode QUAD performs at 91 % of its theoretical maximum, which is 2.79 kbit/s. As before, the maximum achieved throughput using PHY mode OCTA (4.71 kbit/s, using three senders) is again around 15 % less than the theoretical maximum of 5.50 kbit/s.

The adaptive PHY mode selection performs slightly worse than mode OCTA with more than three senders and messages sizes that trigger the use of FEC. Its maximum throughput reaches 4.65 kbit/s with only one sender. This suggests that the parameters for the adaptive selection, which determines how the algorithm reacts to dropped or retransmitted frames, are set too conservatively in a scenario with higher contention probability. This leads to hasty scale-backs, where a slower PHY mode is selected too quickly, and the waiting time for the next probing packet is too long to quickly adapt to the recovery characteristics of a congested channel.

4.3.2 No-Line-of-Sight Communication. The highly sensitive light bulb sensors and the improved synchronization methods allow for line-of-sight communication over more than 5 meters (measurements not shown in this paper for space reasons). One of the most

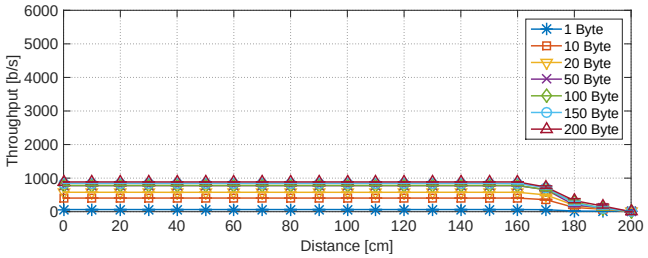

Figure 18: Throughput for light bulb-to-LED point-to-point communication at variable distance for different packet sizes using PHY mode SINGLE.

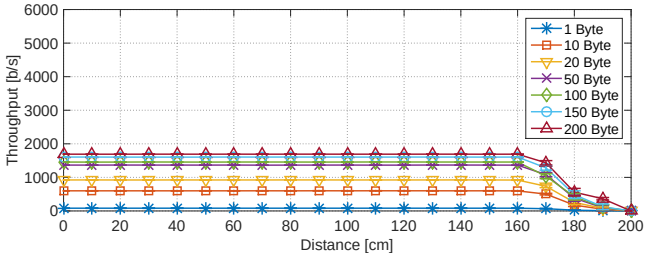

Figure 19: Throughput for light bulb-to-LED point-to-point communication at variable distance for different packet sizes using PHY mode DOUBLE.

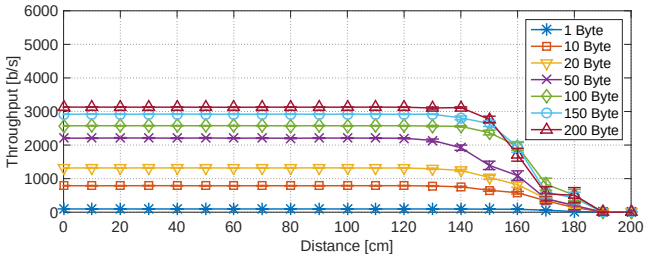

Figure 20: Throughput for light bulb-to-LED point-to-point communication at variable distance for different packet sizes using PHY mode QUAD.

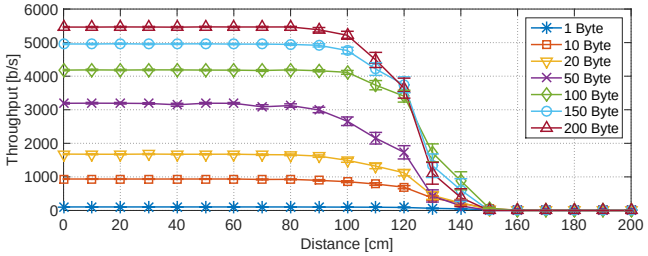

Figure 21: Throughput for light bulb-to-LED point-to-point communication at variable distance for different packet sizes using PHY mode OCTA.

often heard critique points for VLC system is that only line-of-sight communication is possible [26]. Figure 27 shows that also no-line-of-sight communication is possible if accurate enough sensing is available. A light bulb (LB1) is moved from line-of-sight to a no-line-of-sight location while communicating with a static second light bulb (LB2). The plot shows RSSI measurements. The RSSI value is defined as the average difference between a low and high light level

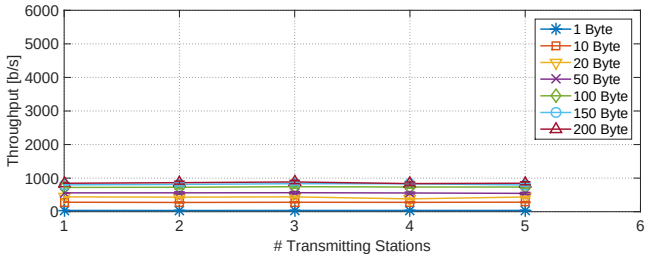

Figure 22: Network throughput for 1 to 5 transmitting light bulbs, one receiving bulb, different packet sizes, and PHY mode `SINGLE`.

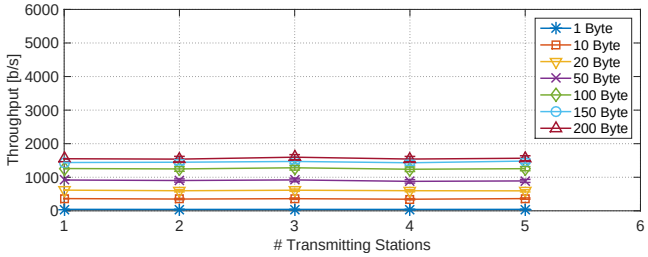

Figure 23: Network throughput for 1 to 5 transmitting light bulbs, one receiving bulb, different packet sizes, and PHY mode `DOUBLE`.

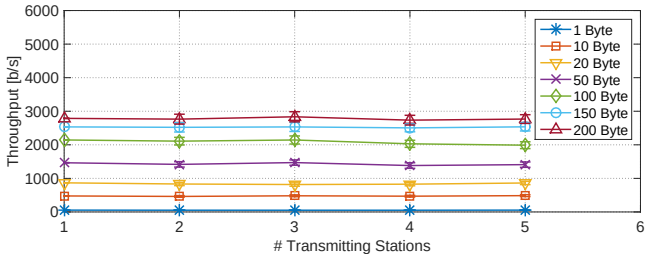

Figure 24: Network throughput for 1 to 5 transmitting light bulbs, one receiving bulb, different packet sizes, and PHY mode `QUAD`.

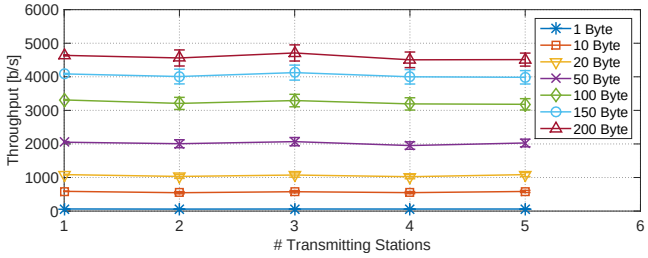

Figure 25: Network throughput for 1 to 5 transmitting light bulbs, one receiving bulb, different packet sizes, and PHY mode `OCTA`.

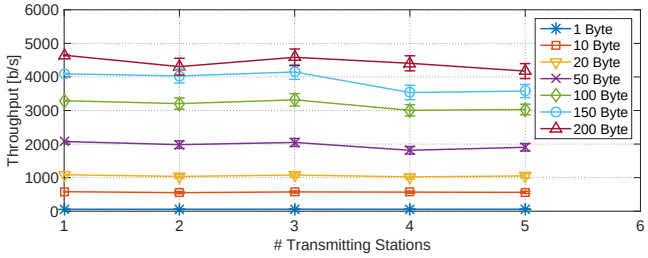

Figure 26: Network throughput for 1 to 5 transmitting light bulbs, one receiving bulb, different packet sizes, and adaptive PHY mode selection.

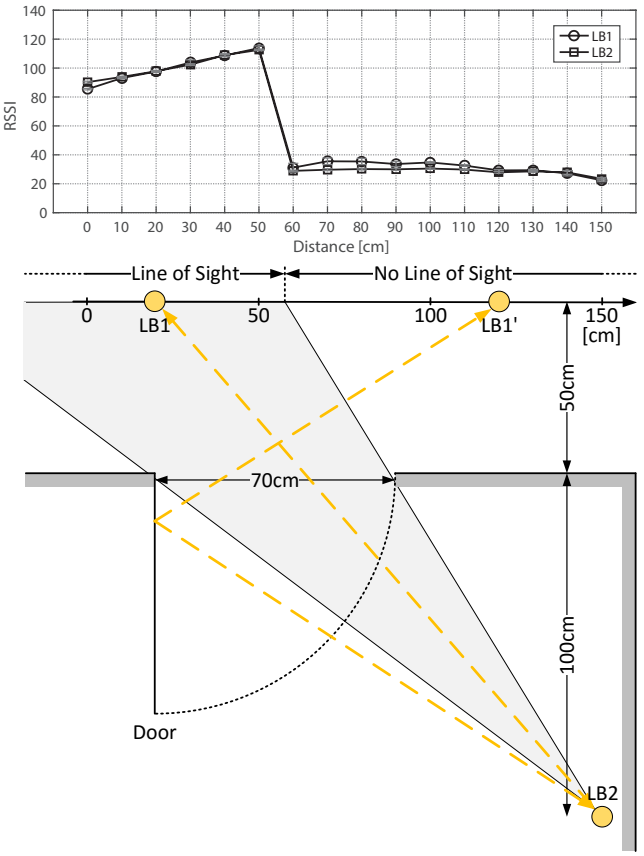

Figure 27: Visualization and RSSI measurements for line-of-sight and no-line-of-sight communication between two light bulbs.

measured in ADC units (10 bit resolution). While in line-of-sight, LB1 moves closer to LB2, and therefore the RSSI value is slightly increasing. At 60 cm, RSSI drops to 30. This is the point where the light bulbs start to communicate via the reflection on the door. The communication also stays stable when moving LB1 further away from the door. At 150 cm LB1 is at the same horizontal coordinate but separated by a wall, and still communicating with LB2. *libvlc* is configured to distinguish symbols if they are at least 20 RSSI units apart. This value can still be lowered in software to be able to reach even further, but such a setting also increases the probability of false positive SFD detection and therefore causes wasted processing time. Reflection not only works on doors but also an walls and ceilings and evens allow communication between different floor levels.

5 RELATED WORK

Since the vision of using VLC light bulbs for data communication has been formulated [6], many research challenges have been addressed in the field of low-cost practical VLC systems and recent results have been collected in an overview [15]. Schmid et al. describe a flexible software-based VLC system that supports indoor room-area networking. Their protocol is inspired by the MAC layer of 802.11 and uses CSMA/CA. The target applications

for their approach are (1) low-cost communication networks for consumer electronics [21] and (2) indoor localization with networks of LED light bulbs [22]. Kuo et al. [11] explore building blocks for a software-defined lighting (SDL) architecture, aiming to leverage research on software-defined networks and software-defined radios.

This paper introduces multiple physical layer modulation modes to enable dynamic adaptive link sensitivity into VLC. The evaluation shows that substantial benefits (regarding the capacity of the channel as well as the distance between sending and receiving station) can be realized.

The benefit of exploiting dynamic link adaptation (closed- and open-loop, centralized/distributed) has been demonstrated for Wi-Fi IEEE 802.11 and cellular networks [1, 16]). This work employs open-loop distributed approaches.

The protocol and platform design presented here do not completely implement the IEEE standard for VLC, 802.15.7 [3, 7, 14] and tradeoff performance for the benefit of flexible prototyping and protocol exploration. The VLC standard IEEE 802.15.7 specifies multiple PHY and MAC layers, of which some are similar to this software-based implementation.

Wang et al. [25] present an alternative low-cost hardware and software platform centered around a printed circuit board (OpenVLC1.0 cape) that implements an optical front-end; its primary application is to allow other groups to explore VLC by prototyping MAC and PHY protocols and has been used for an experimental characterization of the performance of VLC channels [8].

Klaver et al. [10] investigate multi-hop VLC and present a novel platform that can be connected to many prototyping and sensor boards.

6 CONCLUSION

To realize the vision of the IoT, we need a communication platform that allows the inter-operation of a great variety of devices. A software-based VLC system can work with a wide range of sensing hardware. In scenarios that must work with minimal hardware components (e.g., because cost or energy consumption are critical parameters), a single LED can be used as a sender and receiver. If the environment allows use of an LED light bulb (with a dedicated sensor for receiving), then larger distances can be covered. As LED light bulbs and single-LED systems use the same PHY and MAC layer (all in software), they can inter-operate and form a convenient platform for indoor room-area networks.

Realistic VLC systems must work across diverse environmental conditions - close to a window that brings sunlight into a room, for mobile devices (maybe attached as tags to physical objects), or without direct line-of-sight (either because of the placement of nodes or because a moving object/person blocks temporarily the view). The software-centric approach described here allows to easily adapt the link sensitivity based on the strength of the input signal. Adaptivity can deal with varying distances between sender and receiver, the hardware capabilities (sensor properties and processor features), or environmental conditions. Compared to a static system, an adaptive VLC system can translate the increased capacity into up to 8 times higher bit rates, can communicate over significantly larger distances, or allow communication without a direct line-of-sight, allowing VLC around a corner or between different floors of a building.

An adaptive VLC system is an attractive platform for room-area networks that include different types of embedded devices – such as network nodes which only use simple single LEDs or LED light bulbs that provide the backbone of an indoor VLC communication fabric for the IoT.

REFERENCES

[1] M. S. Alouini and A. J. Goldsmith. 1999. Capacity of Rayleigh Fading Channels Under Different Adaptive Transmission and Diversity-Combining Techniques. *IEEE Transactions on Vehicular Technology* 48, 4 (Jul 1999), 1165–1181. DOI: http://dx.doi.org/10.1109/25.775366

[2] Atmel. 2015. Atmel 8-Bit Microcontroller Datasheet. http://www.atmel.com/images/doc8161.pdf. (2015). [Online; accessed 25-November-2015].

[3] A. Boucouvalas, P. Chatzimisios, Z. Ghassemlooy, M. Uysal, and K. Yiannopoulos. 2015. Standards for Indoor Optical Wireless Communications. *Communications Magazine, IEEE* 53, 3 (March 2015), 24–31. DOI: http://dx.doi.org/10.1109/MCOM.2015.7060515

[4] G. Corbellini, K. Aksit, S. Schmid, S. Mangold, and T. R. Gross. 2014. Connecting Networks of Toys and Smartphones with Visible Light Communication. *IEEE Communications Magazine* 52, 7 (July 2014), 72–78. DOI: http://dx.doi.org/10.1109/MCOM.2014.6852086

[5] P. Dietz, W. Yerazunis, and D. Leigh. 2003. Very Low-Cost Sensing and Communication Using Bidirectional LEDs. In *UbiComp 2003: Ubiquitous Computing*. Springer, 175–191.

[6] H. Elgala, R. Mesleh, and H. Haas. 2009. Indoor Broadcasting via White LEDs and OFDM. *Consumer Electronics, IEEE Transactions on* 55, 3 (2009), 1127–1134. DOI: http://dx.doi.org/10.1109/TCE.2009.5277966

[7] C. Gavrincea, J. Baranda, and P. Henarejos. 2014. Rapid Prototyping of Standard-Compliant Visible Light Communications System. *Communications Magazine, IEEE* 52, 7 (July 2014), 80–87. DOI: http://dx.doi.org/10.1109/MCOM.2014.6852087

[8] Milad Heydariaan, Shengrong Yin, Omprakash Gnawali, Daniele Puccinelli, and Domenico Giustiniano. 2016. Embedded Visible Light Communication: Link Measurements and Interpretation. In *Proceedings of the 2016 International Conference on Embedded Wireless Systems and Networks (EWSN '16)*. Junction Publishing, USA, 341–346. http://dl.acm.org/citation.cfm?id=2893711.2893797

[9] Ad Kamerman and Leo Monteban. 1997. WaveLAN-II: A High-Performance Wireless LAN for the Unlicensed Band. *Bell Labs technical journal* 2, 3 (1997), 118–133.

[10] L. Klaver and M. Zuniga. 2015. Shine: A Step Towards Distributed Multi-Hop Visible Light Communication. In *Proc. 2015 IEEE 12th International Conference on Mobile Ad Hoc and Sensor Systems (MASS)*,. IEEE, IEEE, 235 – 243.

[11] Ye-Sheng Kuo, Pat Pannuto, and Prabal Dutta. 2014. System Architecture Directions for a Software-defined Lighting Infrastructure. In *Proceedings of the 1st ACM MobiCom Workshop on Visible Light Communication Systems (VLCS '14)*. ACM, New York, NY, USA, 3–8. DOI: http://dx.doi.org/10.1145/2643164.2643166

[12] Mathieu Lacage, Mohammad Hossein Manshaei, and Thierry Turletti. 2004. IEEE 802.11 Rate Adaptation: A Practical Approach. In *Proceedings of the 7th ACM International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM '04)*. ACM, New York, NY, USA, 126–134. DOI: http://dx.doi.org/10.1145/1023663.1023687

[13] Alexander W Min and Kang G Shin. 2008. An Optimal Transmission Strategy for IEEE 802.11 Wireless LANs: Stochastic Control Approach. In *Sensor, Mesh and Ad Hoc Communications and Networks, 2008. SECON'08. 5th Annual IEEE Communications Society Conference on*. IEEE, 251–259.

[14] S.K. Nobar, K.A. Mehr, and J.M. Niya. 2015. Comprehensive Performance Analysis of IEEE 802.15.7 CSMA/CA Mechanism for Saturated Traffic. *Optical Communications and Networking, IEEE/OSA Journal of* 7, 2 (February 2015), 62–73. DOI: http://dx.doi.org/10.1364/JOCN.7.000062

[15] P. H. Pathak, X. Feng, P. Hu, and P. Mohapatra. 2015. Visible Light Communication, Networking, and Sensing: A Survey, Potential and Challenges. *IEEE Communications Surveys & Tutorials* 17, 4 (November 2015), 2047 – 2077.

[16] Daji Qiao, Sunghyun Choi, and K. G. Shin. 2002. Goodput analysis and link adaptation for IEEE 802.11a wireless LANs. *IEEE Transactions on Mobile Computing* 1, 4 (Oct 2002), 278–292. DOI: http://dx.doi.org/10.1109/TMC.2002.1175541

[17] I. S. Reed and G. Solomon. 1960. Polynomial Codes Over Certain Finite Fields. *J. Soc. Indust. Appl. Math.* 8, 2 (1960), 300–304. DOI: http://dx.doi.org/10.1137/0108018

[18] Stefan Schmid, Theodoros Bourchas, Stefan Mangold, and Thomas R. Gross. 2015. Linux Light Bulbs: Enabling Internet Protocol Connectivity for Light Bulb Networks. In *Proceedings of the 2nd International Workshop on Visible Light Communications Systems (VLCS '15)*. ACM, New York, NY, USA, 3–8. DOI: http:

//dx.doi.org/10.1145/2801073.2801074

[19] S. Schmid, G. Corbellini, S. Mangold, and T.R. Gross. 2014. Continuous Synchronization for LED-to-LED Visible Light Communication Networks. In *Optical Wireless Communications (IWOW), 3rd International Workshop*. 45–49. DOI: http://dx.doi.org/10.1109/IWOW.2014.6950774

[20] Stefan Schmid, Giorgio Corbellini, Stefan Mangold, and Thomas R Gross. 2012. An LED-to-LED Visible Light Communication System with Software-based Synchronization. In *Globecom Workshops (GC Wkshps), 2012 IEEE*. IEEE, 1264–1268.

[21] Stefan Schmid, Giorgio Corbellini, Stefan Mangold, and Thomas R. Gross. 2013. LED-to-LED Visible Light Communication Networks. In *Proceedings of the Fourteenth ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc '13)*. ACM, New York, NY, USA, 1–10. DOI: http://dx.doi.org/10.1145/2491288.2491293

[22] Stefan Schmid, Thomas Richner, Stefan Mangold, and Thomas R. Gross. 2016. EnLighting: An Indoor Visible Light Communication System based on Networked Light Bulbs. In *Sensing, Communication, and Networking (SECON), 13th Annual IEEE International Conference on*.

[23] Stefan Schmid, Josef Ziegler, Giorgio Corbellini, Thomas R. Gross, and Stefan Mangold. 2014. Using Consumer LED Light Bulbs for Low-cost Visible Light Communication Systems. In *Proceedings of the 1st ACM MobiCom Workshop on Visible Light Communication Systems (VLCS '14)*. ACM, New York, NY, USA, 9–14. DOI: http://dx.doi.org/10.1145/2643164.2643170

[24] STMicroelectronics. 2016. ARM-based 32-bit MCU Datasheet. http://www.st.com/web/en/resource/technical/document/datasheet/DM00039193.pdf. (2016). [Online; accessed 25-November-2015].

[25] Qing Wang, Domenico Giustiniano, and Daniele Puccinelli. 2015. An Open Source Research Platform for Embedded Visible Light Networking. *IEEE Wireless Communications* 22, 2 (April 2015), 94 – 100.

[26] Bo Xie, Guang Tan, and Tian He. 2015. SpinLight: A High Accuracy and Robust Light Positioning System for Indoor Applications. In *Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems (SenSys '15)*. ACM, New York, NY, USA, 211–223. DOI: http://dx.doi.org/10.1145/2809695.2809713