Bidirectional Visible Light Communications for the Internet of Things

Alexis Duque

N° d'ordre NNT : 2018LYSEI072

THÈSE DE DOCTORAT DE L'UNIVERSITÉ DE LYON
opérée au sein de
l'INSA de Lyon

École Doctorale ED512
InfoMaths

Spécialité de doctorat : Informatique

Soutenue publiquement le 09/10/2018, par :
Alexis Duque

Bidirectional Visible Light Communications for the Internet of Things

---

Devant le jury composé de :

Emmanuel CHAPUT	Professeur des Universités	INP Toulouse	Rapporteur
Anne JULIEN-VERGONJANNE	Professeur des Universités	Univ. Limoges	Rapporteur
Josep PARADELLS ASPAS	Professeur	UPC	Rapporteur
Luc CHASSAGNE	Professeur des Universités	UVSQ	Examinateur
Valeria LOSCRI	Chargé de Recherche	INRIA Lille	Examinateur
Hervé RIVANO	Professeur des Universités	INSA Lyon	Directeur de thèse
Razvan STANICA	Maître de Conférences	INSA Lyon	co Directeur de thèse

Département FEDORA – INSA Lyon - Ecoles Doctorales – Quinquennal 2016-2020

SIGLE	ÉCOLE DOCTORALE	NOM ET COORODONNÉES DU RESPONSABLE
CHIMIE	**CHIMIE DE LYON** http://www.edchimie-lyon.fr Sec : Renée EL MELHEM Bat Blaise Pascal 3e étage secretariat@edchimie-lyon.fr Insa : R. GOURDON	M. Stéphane DANIELE Institut de Recherches sur la Catalyse et l'Environnement de Lyon IRCELYON-UMR 5256 Équipe CDFA 2 avenue Albert Einstein 69626 Villeurbanne cedex directeur@edchimie-lyon.fr
E.E.A.	**ELECTRONIQUE, ELECTROTECHNIQUE, AUTOMATIQUE** http://edeea.ec-lyon.fr Sec : M.C. HAVGOUDOUKIAN Ecole-Doctorale.eea@ec-lyon.fr	M. Gérard SCORLETTI Ecole Centrale de Lyon 36 avenue Guy de Collongue 69134 ECULLY Tél : 04 72 18 60 97 Fax : 04 78 43 37 17 Gerard.scorletti@ec-lyon.fr
E2M2	**EVOLUTION, ECOSYSTEME, MICROBIOLOGIE, MODELISATION** http://e2m2.universite-lyon.fr Sec : Sylvie ROBERJOT Bât. Atrium, UCB Lyon 1 Tél : 04 72 44 83 62 Insa : H. CHARLES secretariat.e2m2@univ-lyon1.fr	M. Philippe NORMAND UMR 5557 Lab. d'Ecologie Microbienne Université Claude Bernard Lyon 1 Bât Mendel 43 bd du 11 novembre 1918 69622 VILLEURBANNE Cédex philippe.normand@univ-lyon1.fr
EDISS	**INTERDISCIPLINAIRE SCIENCES-SANTE** http://www.ediss-lyon.fr Sec : Sylvie ROBERJOT Bât. Atrium, UCB Lyon 1 Tél : 04 72 44 83 62 Insa : M. LAGARDE secretariat.ediss@univ-lyon1.fr	Mme Emmanuelle CANET-SOULAS INSERM U1060, CarMeN lab, Univ. Lyon 1 Bâtiment IMBL 11 avenue Jean Capelle INSA de Lyon 69621 Villeurbanne Tél : 04 72 68 49 09 Fax : 04 72 68 49 16 emmanuelle.canet@univ-lyon1.fr
INFOMATHS	**INFORMATIQUE ET MATHEMATIQUES** http://edinfomaths.universite-lyon.fr Sec : Renée EL MELHEM Bat Blaise Pascal 3e étage infomaths@univ-lyon1.fr	M. Luca ZAMBONI Bât. Braconnier 43 Boulevard du 11 novembre 1918 69622 VILLEURBANNE CEDEX Tél : 04 26 23 45 52 zamboni@maths.univ-lyon1.fr
Matériaux	**MATERIAUX DE LYON** http://ed34.universite-lyon.fr Sec : Marion COMBE Tél : 04 72 43 71 70 Fax : 04 72 43 87 12 Bat. Saint Direction mega@insa-lyon.fr	M. Jean-Yves BUFFIERE INSA de Lyon MATEIS Bâtiment Saint Exupéry 7 avenue Jean Capelle 69621 VILLEURBANNE Cedex Tél : 04 72 43 71 70 Fax 04 72 43 85 28 jean-yves.buffiere@insa-lyon.fr
MEGA	**MECANIQUE, ENERGETIQUE, GENIE CIVIL, ACOUSTIQUE** http://edmega.universite-lyon.fr Sec : Marion COMBE Tél : 04 72 43 71 70 Fax : 04 72 43 87 12 Bat. Saint Direction mega@insa-lyon.fr	M. Jocelyn BONJOUR INSA de Lyon Laboratoire CETHIL Bâtiment Sadi-Carnot 9, rue de la Physique 69621 VILLEURBANNE CEDEX jocelyn.bonjour@insa-lyon.fr
ScSo	**ScSo**[1] http://ed483.univ-lyon2.fr Sec : Viviane POLSINELLI, Brigitte DUBOIS INSA : J.Y. TOUSSAINT Tél : 04 78 69 72 76 viviane.polsinelli@univ-lyon2.fr	M. Christian MONTES Université Lyon 2 86 rue Pasteur 69365 LYON Cedex 07 christian.montes@univ-lyon2.fr

[1]ScSo : Histoire, Géographie, Aménagement, Urbanisme, Archéologie, Science politique, Sociologie, Anthropologie

Abstract

With the exponential growth of the Internet of Things (IoT), people now expect every household appliance to be smart and connected. At the same time, smartphones have become ubiquitous in our daily life. Their continuous performance improvement and their compatibility with a broad range of radio protocols as WiFi, Bluetooth Low Energy (BLE) or Near Field Communication (NFC) make them the most convenient way to interact with these smart objects. However, providing wireless connectivity with BLE or NFC means adding an extra chipset and an antenna, increasing the object size and price. Also, this kind of hardware modification is not without impact: even if the radio chip cost is negligible for a single unit, it may become huge when millions of products are sold.

On the other hand, previous works already have demonstrated the possibility of receiving information through visible light using an unmodified smartphone thanks to its camera. Also, LED-to-LED communication for smart devices like toys has been shown previously. However, past efforts in light-emitting diode (LED) to camera communication for IoT device-to-device communication have been limited.

In this work, we address this question and design *LightIoT*, a bidirectional visible-light communication (VLC) system between a low-cost, low-power colored LED that is part of an IoT device and an off-the-shelf smartphone. The IoT device is thus able to send and receive information through its LED, while the smartphone uses its camera to receive data and its flashlight to send information.

We implement and experimentally evaluate a LED-to-camera VLC system, designed specifically for small LEDs. The proposed solution exploits the rolling shutter effect of unmodified smartphone cameras and an original decoding algorithm, achieving a throughput of nearly 2 kb/s in ordinary illumination conditions. Based on the insight gained from an extensive experimental study, we model, for the first time in the literature, the LED-to-camera communication channel, characterized by an on/off behavior resulting in a close to regular frame error pattern. We propose a Markov-modulated Bernoulli process model, which

allows us to easily study the performance of different message retransmission strategies. We further exploit this model to implement a simulator for LED-to-Camera communications performance evaluation.

In order to achieve bi-directional communications, we evaluate flashlight-to-LED communications using non-rooted smartphones and small LEDs. With these hardware constraints, our implementation achieves a throughput of 30 bits/second. While obviously limited, this is enough for a feed-back channel coming to support the required redundancy mechanisms. Some of these redundancy mechanisms are based on random linear coding, long time considered as a possible solution in VLC, but never tested previously. Therefore, we design and implement, for the first time in the literature, a pseudo random linear coding scheme especially fitted for line-of-sight LED-to-camera conditions. Experimental evaluation highlights that this type of approach increases the goodput up to twice compared to classical retransmission strategies.

Finally, many of the small objects we address through *LightIoT* have significant energy constraints. Therefore, we compare the energy consumption of *LightIoT* with the one of a BLE module with similar activity. Our results show that using the LED for communication purposes reduces the energy consumption under a normal usage behavior.

KEYWORDS: Visible Light Communication, Internet of Things, Smartphone, Optical Camera Communication, Wireless Communication, Random Linear Coding, LED-to-Camera Communication

Résumé

Avec la croissance exponentielle de l'Internet des objets, les utilisateurs s'attendent maintenant à ce que chaque appareil ménager soit intelligent et connecté. Dans le même temps, les smartphones sont devenus omniprésents dans notre vie quotidienne. L'amélioration continue de leurs performances et leur compatibilité avec de nombreux protocoles radio tels que le WiFi, le Bluetooth Low Energy (BLE) ou le Near Field Communication (NFC) en font le moyen le plus pratique pour interagir avec ces objets connectés. Cependant, offrir une connectivité sans fil utilisant le BLE ou le NFC implique d'ajouter un chipset supplémentaire et une antenne, ce qui augmente la taille et le prix de l'objet. En outre, ce type de modification matérielle n'est pas sans impact: même si le coût unitaire de la puce radio est négligeable, il peut devenir très important lorsque des millions de produits sont vendus.

Par ailleurs, des travaux antérieurs ont déjà démontré la possibilité de recevoir de l'information par la lumière visible (VLC) à l'aide d'un smartphone non modifié grâce à sa camera. Aussi, des communications bidirectionnelles entre diodes électroluminescentes (DEL) pour les objets connectés comme les jouets ont déjà été démontrées précédemment. Cependant, les efforts afin de proposer une communication sans fil utilisant des petites DELs et la caméra des smartphones restent à ce jour inexistants.

Dans ce travail, nous abordons donc cette question et proposons *LightIoT*, un système de communication bidirectionnel par VLC entre une DEL de couleur et de faible puissance qui est intégrée à un objet connecté et un smartphone non modifié. Le dispositif est ainsi capable d'envoyer et de recevoir des informations à travers sa DEL, tandis que le smartphone utilise sa caméra pour recevoir des données et son flash pour envoyer des informations.

Nous mettons en œuvre et évaluons expérimentalement ce système VLC DEL-à-camera conçu spécifiquement pour les DEL de couleurs à faible puissance. La solution proposée exploite l'effet de l'obturateur déroulant de la caméra des smartphones associé à un algorithme de décodage original et atteint un débit

d'environ 2 kbit/s dans des conditions d'éclairage standard. En nous appuyant sur les connaissances acquises lors d'une vaste étude expérimentale, nous modélisons, pour la première fois dans la littérature, le canal de communication DEL-à-caméra, caractérisé par un schéma d'erreur de trames régulier et presque périodique. Nous proposons alors un modèle de processus de Bernoulli modulé par une chaîne de Markov, qui nous permet d'étudier facilement l'efficacité de différentes stratégies de retransmission des messages. Nous exploitons ce modèle afin de concevoir un simulateur pour l'évaluation des performances des communications DEL-à-caméra.

Afin d'obtenir un système de communication bidirectionnel, nous étudions ensuite les communications de type flash-vers-DEL entre un smartphone non-modifié et une petite DEL de couleur. En considérant les contraintes matérielles du système, notre implémentation atteint un débit de 30 bits par seconde. Bien que les performances soient limitées, elles sont suffisantes pour établir une voie retour qui permet de mettre en oeuvre des mécanismes de fiabilisation. Certains de ces mécanismes sont basés sur un codage linéaire aléatoire, considéré depuis longtemps comme une solution possible dans VLC, mais jamais testé auparavant. Par conséquent, nous concevons et mettons en œuvre, pour la première fois dans la littérature, un mécanisme de codage linéaire pseudo-aléatoire spécialement adapté aux conditions et contraintes du système DEL-à-caméra en ligne de visée directe. Notre évaluation expérimentale souligne que ce type d'approche augmente le rendement jusqu'à deux fois par rapport aux stratégies de retransmission classiques.

Enfin, la plupart de ces petits objets que nous adressons via LightIoT ont des contraintes énergétiques importantes. Par conséquent, nous comparons la consommation d'énergie de LightIoT avec celle d'un module BLE avec une activité similaire. Nos résultats montrent que l'utilisation de la DEL à des fins de communication réduit la consommation d'énergie dans le cadre d'un profil d'utilisation classique.

MOTS CLÉS: Communication par Lumière Visible, Internet des Objets, Smartphone, Optical Camera Communication, Communications Sans-Fil, Codage Linéaire Aléatoire.

Foreword

This thesis compiles my PhD work on the *Design of a bidirectionnal visible light communication system for Internet of Things devices and off-the-shelf smartphones*.

This work is a collaboration between the Rtone company and the Inria AGORA team (ex-URBANET) from the CITI laboratory (Centre d'Innovation en Télécommunications et Intégration de services) at INSA de Lyon. This cooperation happened under the ANRT CIFRE (Conventions Industrielles de Formation par la Recherche) program.

Rtone [3] is an innovative small and medium-sized enterprise (SME) funded in 2007, whose main activity is the design and the development of smart wireless IoT devices for third party companies. The team counts about thirty engineers with expertise in hardware conception, firmware, mobile phone or web application development. Thus, Rtone is able to build a whole IoT system from the specifications to the cloud.

This PhD work falls into an evolution of the company strategy started in 2015. Focusing on achieving competitive advantage with a high added value, Rtone decided to increase the research and development investments to develop an intellectual property portfolio. p In this context, the purpose of this thesis was the design and the implementation of an innovative wireless communication system using visible light for highly constrained IoT devices. One of Rtone expectations is the integration of such visible light communication system in its customer products.

Table of contents

List of figures

List of tables

List of acronyms

AOT	Ahead-Of-Time
ART	Android Runtime
ADO-OFDM	Asymmetrically Clipped DC-Biased Optical OFDM
ACO-OFDM	Asymmetrically Clipped Optical OFDM
BFSK	Binary Frequency Shift Keying
BLE	Bluetooth Low Energy
CPU	Central Processing Unit
CSK	Color Shift Keying
CC	Convolutional Codes
DCO-OFDM	Direct Current Biased Optical OFDM
DC	Direct Current
EPO	European Patent Office
FPGA	Field-Programmable Gate Array
FEC	Forward Error Correction
FSK	Frequency Shift Keying
GPIO	General Purpose Input/Output
INPI	Institut National de la Propriété Industrielle
IFG	Inter-Frame Gap
IOT	Internet Of Things

JEITA	Japan Electronics And Information Technology Industries Association
LED	Light-Emitting Diode
LOS	Line Of Sight
MMBP	Markov-Modulated Bernoulli Processes
MAC	Medium Access Control
MCU	Micro-Controller Unit
MIMO	Multiple Input Multiple Output
NDK	Native Development Kit
NFC	Near Field Communication
NC	Network Coding
NLOS	Non-Line Of Sight
OOK	On-Off Keying
OCC	Optical Camera Communication
OFDM	Orthogonal Frequency Division Multiplexing
PD	Photodiode
PHY-SDU	Physical Layer Service Data Unit
PCB	Printed Circuit Board
PRLC	Pseudo-Random Linear Coding
PWM	Pulse Width Modulation
QAM	Quadrature Amplitude Modulation
RF	Radio Frequency
RLC	Random Linear Coding
RSS	Received Signal Strength

RS	Reed-Solomon
ROI	Region Of Interest
RLL	Run-Length Limited
SNR	Signal-To-Noise Ratio
SDK	Software Development Kit
SM	Spatial Modulation
SCM	Subcarrier Multiplexing
SMD	Surface Mount Device
TIA	Trans-Impedance Amplifier
VPWM	Variable Pulse Width Modulation
VPPM	Variable Pulse-Position Modulation
VPAN	Visible Light Communication Personal Area Network
VLCC	Visible Light Communications Consortium
VLC	Visible Light Communication
WDM	Wavelength Division Multiplexing
WPAN	Wireless Personal Area Network

Chapter 1

Introduction

1.1 Visible Light Communication

Visible-light communication (VLC) is an enabling technology that exploits illumination to provide a short-range wireless communication link. VLC systems take advantage of the license-free light spectrum and their immunity to radio frequency (RF) interference. In such systems, information is often relayed by modulating the output intensity of the LEDs, whereas, at the receiver side, the data signal is recovered using simple photodiodes (PDs).

The firsts efforts of using LEDs both for illumination and communication date to the year 2000 when researchers from Keio University in Japan proposed the use of white LED in homes for building an access network [116]. The interest in the field then started growing, especially in Japan, initially aimed to build high-speed communication through visible light, with the development of VLC support for mobile devices and vehicular communications. This led to the formation of Visible Light Communications Consortium (VLCC) in Japan in November of 2003 [82]. VLCC proposed two standards by 2007 that were later accepted by Japan Electronics and Information Technology Industries Association (JEITA) as JEITA CP-1221 and CP-1222 respectively. JEITA CP-1221 only defines the requirements and the indication level that is required to avoid the interference between different VLC devices, while JEITA CP-1222 defines a Visible Light ID System.

In the meantime, the Home Gigabit Access project (OMEGA) [91] founded by the European Commission under the EU Framework Project 7 (FP7) also developed optical communication as a way to increase the RF communication network capacity, and the IEEE initiated a standardization process in 2008.

The IEEE 802.15.7 standard [62], released in 2011, was a big step towards the commercialization and widespread deployment of VLC networks [54, 55]. It defines only physical (PHY) and medium access control (MAC) layers. Three physical layer modes are proposed. PHY I is designed for outdoor scenarios, while PHY II and PHY III are designed to be used indoors. A total of thirty combinations of modulation and coding scheme are specified.

PHY I utilizes Reed-Solomon (RS) and Convolutional Codes (CC) for Forward Error Correction (FEC) , while PHY II and III mostly rely on RS codes only for FEC. Both PHY I and PHY II utilize on-off keying (OOK) and variable pulse-position modulation (VPPM), while PHY III only uses color shift keying (CSK) , meaning that PHY III is compatible only with RGB LEDs. Depending on the choice of modulation, run-length limited (RLL) code, optical clock rate, FEC code, the three PHY modes thus provide different data rates: PHY I, PHY II and PHY III, respectively support data rates of 11.67 to 266.6 kbit/s, 1.25 to 96 Mbps and 12 to 96 Mbps [100, 104].

The MAC layer addresses the following features: (1) Mobility support, (2) Dimming support, (3) Visibility support, (4) Flickering mitigation, (5) Color function support, (6) Network beacons generation if the device is a coordinator, (7) Visible light communication personal area network (VPAN) disassociation and association support, (8) Link reliability between peer MAC entities. Security considerations are delayed to the application layer. Note that the 802.15.7 MAC layer is very close to the wireless personal area network (WPAN) IEEE 802.15.4 MAC [113]. The topologies supported by the MAC layer are peer-to-peer, broadcast and star. In the star topology, all the nodes communicate with each other through a single centralized controller. In the peer-to-peer topology, the controller role is performed by one of the two nodes involved in communication with each other. Three classes of devices are considered: vehicles, mobile and infrastructure.

Nonetheless, many VLC use cases and VLC technologies are not taken into account in these standards. Also, VLC systems can use other types of light source and light sensors, according to the requirements of the application environment. That is why the IEEE Task Group 7m, firstly known as IEEE 802.17.5r1, has been initiated in late 2014 to revise the IEEE 802.15.7-2011 standard. The IEEE 802.15.7m amendment will be released by the end of 2018.

In fact, any light sources which can switch states at a high rate (i.e. transition between ON and OFF) can be used as transmitters for VLC systems. Consequently, screens can also be used as transmitters, as suggested in [72, 49, 59, 71]. These techniques use barcodes or QR codes displayed on screens to encode in-

	PD	Small Color LED	Camera
Flashlight			
Lighting Power LED			
Small Color LED			
Screen			
Passive Light			

Table 1.1: VLC systems classification and contributions presented in this thesis.

formation. HiLight [71] is a new form of a screen-to-camera link, which encodes data into pixels translucency. This technique can efficiently transmit information without being obstructive to the displayed image or video.

Any electronic device which can detect the presence or absence of visible light can be utilized as a VLC receiver. Most of the work is focused towards using PDs as receivers, because of their fast response and high bandwidth. Works like [107] show that regular LEDs can operate as a receiver in reverse bias mode, but they have limited sensitivity as compared to PDs. Smartphone cameras can also be used to detect high-frequency light patterns, as discussed in [22].

In a nutshell, VLC emitters can either be 1) modulated ceiling LED, 2) low power and often colored LED, 3) screens, and 4) passive light, such as the incandescent light bulbs and fluorescent light tubes. VLC receivers are for their part 1) photodiodes specifically built to sense light, but also 2) LED which first purpose is to emit light, and finally 3) cameras.

To get a clearer view on VLC capabilities and sort manifold VLC systems, we propose a taxonomy in Tab.1.1, where each cell corresponds to a type of emitter and receiver.

1.2 Motivations

The recent years witnessed a tremendous increase in connectivity requirements. In the age of smart-*, people now expect all of their daily usage objects to be connected, including the cheapest ones. RF technologies, such as Bluetooth Low Energy (BLE), Near Field Communication (NFC) or WiFi, represent the main choice in terms of wireless communications nowadays, and their adoption in this new context seems logical. Nevertheless, this vision is challenged by a simple cost-volume-profit analysis. At a price of around 1$ per module, BLE is already one of the cheapest RF technologies on the market, but adding a BLE module on millions of small objects would represent an important investment for manufacturers. This is especially true as most of these objects only have sporadic connectivity needs, ranging from a few times per day to a few times per year, not justifying the investment.

At the same time, most daily usage objects, e.g. household appliances, smoke and gas detectors, temperature and humidity sensors, toys, etc, already integrate small LEDs, to facilitate the regular user interactions. This enables the use of VLC, where LEDs have been shown to be effective as both transmitters and receivers [107]. Moreover, at a cost of around 0.05$ per unit, LEDs can communicate with regular, unmodified smartphones, using the smartphone camera as a receiver [22] and the flashlight as a transmitter [51].

However, LED-to-smartphone communication has only been studied in the context of relatively large lighting LEDs, for information broadcast [36] or localization [66] purposes. The feasibility of communication between small LEDs, as those integrated in most objects, and a smartphone remains an open question.

We address this question in this work and design a bidirectional VLC transmission system between a low-cost, low-power colored LED that is part of an IoT device and an unmodified and non-rooted smartphone. The IoT device is thus able to send and receive information through its LED, while the smartphone uses its camera to receive data and its flashlight to send information. The additional costs to make a traditional device part of the IoT are therefore minimal.

1.3 Contributions

Referring to the classification priorly introduced in Sec. 1.1, the focus of this thesis is on the green cells in Tab.1.1. That is to say, in the fields of low power LED-to-

camera VLC and flashlight-to-LED VLC. Our main contribution are summarized below:

- **Design and implementation of methods to enable low power LED-to-camera communication.** We introduce an original decoding algorithm that decodes every frame in real-time. By bypassing computer vision techniques, our algorithm processes every 1920x1080 picture in 18.4 ms on average on a Nexus 5 with a Java implementation. Using native C code, the whole processing takes only 3.6 ms. These results outperform previous works [68] and add robustness against user motion. This contribution is detailed in Chap. 3.

- **Evaluation of a LED-to-camera communication system for IoT devices.** We propose, for the first time in the literature, a thorough experimental evaluation of a VLC system composed of a low power colored LED and a smartphone. The experimental evaluation studies the impact of the symbol rate, the payload size, the ambient lighting conditions, the model and color of the LED, and finally the users behaviors, i.e. the motion, the angle between the LED and the smartphone, or the distance. The experimental evaluation results are given in Chap. 3.

- **Analytical study of a LED-to-camera communication system.** We propose a generic analytical model for LED-to-Camera communication systems based on the theory of Markov-modulated Bernoulli processes (MMBP). This model can be applied to all LED-to-camera communication systems Its validation with extensive experimental results proves it is very accurate. This contribution is introduced in Chap. 4.

- **Implementation of a LED-to-camera communication simulator.** We design and implement *CamComSim*, the first LED-to-camera communication simulator. The simulator can be finely tuned to evaluate not only our contributions, but also any LED-to-camera communication system. That allows fast prototyping and evaluation of LED-to-camera communication network protocols. *CamComSim* is detailed in Chap. 4 and is also available as an open source software.

- **Design and implementation of methods enabling flashlight-to-LED communication.** To make any smartphone able to send data using its flashlight to an IoT device, we propose smart mechanisms and protocols that

cope with the flashlight signal jitter and that detect the flashlight clock rate whatever the smartphone model. Our frame format includes synchronization patterns that let the device detect the beginning of a transmission, acquire the clock rate, correct both the clock jitter and bias by adapting the signal sampling itself at runtime. This flashlight-to-LED channel is suitable for feedback, wake-up or communication purposes. Thus, we were able to build *LightIoT*, the first bidirectionnal smartphone-to-LED communication system. The design of the flashlight-to-LED communication channel is presented in Chap. 5.

- **Design of a coding sheme optimized for LightIoT.** We provide the first random linear coding (RLC) coding scheme optimized for VLC and *LightIoT*, that we called *SeedLight*. In particular, we reduce the RLC overhead by using a pseudo-random linear coding (PRLC) method and keep the encoding complexity low. Our coding scheme works with small physical layer service data units (PHY-SDUs), an important constraint for the kind of small LEDs we use as emitters. We implemented *SeedLight* on cheap MCUs and show that it is perfectly fitted for devices with low memory and low computation capabilities. We detail this contribution in Chap. 6.

- **Evaluation of LightIoT energy requirements.** We studied the energy needs and evaluate through experimental measures the impact of *LightIoT* on the power consumption of an IoT device and a smartphone. We compare the results for several utilization profiles with an equivalent system based on BLE. That gives essential insights to choose the adequate communication interface during the conception of energy constrained smart devices. This last contribution is described in Chap. 7.

Chapter 2

State of the Art

In this chapter, we give further details on the background works upon which we build within this thesis. For that purpose, we follow the classification we have proposed in Tab.1.1 in the previous chapter. Section 2.1 presents the past works utilizing PDs as VLC receiver. Then, we show in Section 2.2 that LEDs can also be used as a receiver. Finally, because they are close to the scope of our work, we dedicate the last two sections to past efforts on leveraging cameras as VLC receiver. We start with screen-to-camera links in Sec. 2.3 before concluding this chapter with studies that enable LED-to-camera communications in Sec. 2.4.

2.1 Photodiode as VLC Receiver

This section details notable works that rely on photodiodes to receive a VLC signal. According to the classification shown in Sec. 1, we start focusing on studies that use lighting ceiling LED as VLC emitter in Subsec 2.1.1 and low power LED in 2.1.2. Finally, we end this section introducing research that leverages passive lights for VLC purpose.

2.1.1 Ceiling LED-to-Photodiode communication

Modulation As discussed, LED-to-photodiode is the most convenient way to enable high-speed VLC. Even if current VLC standards only cover OOK, VPPM and CSK modulations with data rates below 96 Mbps, much higher throughputs are feasible and have already been demonstrated in laboratory conditions.

A straightforward approach that has been considered to increase the data rate is to use known RF or optical fiber solutions. By following this path, a pro-

totype VLC system utilizing high-power LEDs and OOK to achieve bidirectional real-time transmission with a total rate of 230 Mbps was implemented in [121]. Inspired by optical fiber communications, a full-duplex bi-directional VLC system utilizing RGB LEDs and a commercially available phosphor-based LED is proposed in [127]. The authors used Wavelength Division Multiplexing (WDM) and Subcarrier Multiplexing (SCM) to achieve the bi-directional transmission.

Furthermore, Orthogonal Frequency Division Multiplexing (OFDM) and Quadrature Amplitude Modulation (QAM) were employed to increase the throughput [5]. OFDM has been widely used in wired and wireless communication systems due to its high spectrum efficiency and the capability of inter-symbols interferences (ISI) mitigation. Applying OFDM in VLC, the speed of the VLC system in [17] was increased to 3.75 Gbps as compared to that in [127] which was 575 Mbps downlink and 225 Mbps uplink. The potential of OFDM has been further studied in the recent years.

Considering the specific properties of VLC, several optical OFDM techniques have been developed, such as the direct current (DC) biased optical OFDM (DCO-OFDM) [64], the asymmetrically clipped optical OFDM (ACO-OFDM) [8] and the asymmetrically clipped DC-biased optical OFDM (ADO-OFDM) [27].

DCO-OFDM was first introduced for wireless infrared optical communications [64]. It adds a DC bias to the bipolar analog signal and clips any remaining negative values. If the DC bias is set a to high value, the optical signal-to-noise ratio (SNR) will become very large, leading to low optical power efficiency.

Thus, ACO-OFDM has been proposed to overcome the disadvantages of DCO-OFDM. ACO-OFDM transmits information only on odd subcarriers. Besides, the clipping noise is added only on the even subcarriers, thus it will not interfere with the information on the odd subcarriers. That improves the power efficiency of DCO-OFDM but results to a low spectrum efficiency, which is half of DCO-OFDM and quarter of RF OFDM.

In [27], ADO-OFDM has been proposed to combat ACO-OFDM and DCO-OFDM disadvantages by combining both of them. In their system, ACO-OFDM is transmitted on the odd subcarriers and DCO-OFDM is transmitted on the even subcarriers. In such way, ADO-OFDM outperforms DCO-OFDM in terms of power efficiency and outperforms ACO-OFDM in terms of spectrum efficiency.

This modulation is the background of LiFi [47] and the *PureLiFi LiFi-X* [96] ready-to-market LiFi access point and USB dongle.

MIMO For providing both illumination and uninterrupted communication coverage, the *attocell* architecture has been proposed [78]. Every LED light bulb in an attocell LiFi network is treated as an access point and a light source for covering a limited region. Based on this attocell architecture, a promising multiple input multiple output (MIMO) method called spatial modulation (SM) can be employed. This MIMO method was first proposed for RF communication in [15], where only one transmitter sends data at any point of time. RF SM was further adapted for VLC by *Fath et al.* [35]. Each transmitter LED is assigned a specific symbol and, when data to be transmitted matches the symbol, the LED is activated. The receiver estimates which LED was activated based on the received signal and uses this to decode the transmitted data. In [34], the authors show that SM is robust to correlated channels and provides high spectral efficiency. Due to the advantages showed in [34], SM has been further investigated in the recent years [83, 25].

Networking Many works address VLC through the PHY layer perspective, putting aside the networking and medium sharing aspects. Indeed, building robust and effective medium sharing protocols is necessary to enable VLC networking and VLC deployment. In the attocell architecture, users must be associated with one or some LED lamps for accessing and downlink, and they should be able to move from an attocell to another, using mechanisms for horizontal handover. The mobility models of users and performance metrics such as the download file size and average connectivity are studied in [19].

On the other hand, challenges remain regarding the VLC uplink transmission: the optical uplink may interfere with the downlink signal, the optical uplink from a device may be uncomfortable to the users, or the high power consumption may make the device integration impractical. For these reasons, LiFi leverages these concerns using IR for the uplink [47].

An alternative is the combination of VLC and the existing RF network as WiFi [97, 60, 74, 110, 19]. The indoor integration system of VLC and WiFi is proposed in [97, 60], where VLC is only used for broadcasting.

However, these hybrid schemes require that WiFi should manage the uplink acknowledgments of VLC frames, which creates frequent small-packet traffic. Their simulation results showed that this approach reduces the throughput of WiFi only devices [97]. The other insight is that these WiFi devices negatively influence the VLC latency and transmission rate. These problems need to be further investigated.

For hybrid VLC networks, vertical handover between LiFi and WiFi is the most important issue to provide continuous network access even with a moving station. *Hou and O'Brien* [53] proposed a fuzzy-logic decision-making algorithm for the vertical handover in a hybrid system in which the optical link is subject to blocking. Depending on the duration of the blocking, the system performs handover to an RF network. Similar to the RF handover protocols, received signal strength (RSS) based handover procedures are introduced by *Vegni et al.* [120]. Finally, in [137, 128], the authors introduced a novel protocol for mobile stations that enables efficient vertical and horizontal handover mechanisms in a hybrid network of LiFi and WiFi system.

Other concerns A relevant question regarding VLC is what happens to data communication when LED lights are turned OFF? A basic answer is that communication is no longer sustained. Indeed, the works discussed above have taken it for granted that lights are switched ON. Nonetheless, there are some scenarios where users do not want indoor lighting, as in a sunny day or during the night just before or during sleep, while the devices still require network connectivity. This is the case for smart home scenarios where smart thermostats are sending temperature data or IP cameras are streaming video. To address such scenarioss where VLC and no perceivable artificial indoor lighting must coexist, *Tian et al.* proposed *DarkLight* [117]. *DarkLight* relies on the fact that light beams are not perceptible by humans when LEDs are extremely dimmed. To achieve this goal, *DarkLight* modulates the LED with 500-ns light pulses and 0.007 % LED duty cycle. Then data are encoded with 12PPM modulation. Note that, to be able to receive and perceive the pulse position with a nanosecond scale, the system must have a very precise clock. For this reason, the authors implement *DarkLight* on a testbed, using FPGAs and a costly PIN-Photodiode. The experimental evaluation showed *DarkLight* achieves 1.6 kbit/s data rate and supports up to 1.3 m communication distance, while keeping the perceptible light negligible.

Ceiling LED-to-photodiode for localization A popular application field for LED-to-photodiode communication is indoor localization. With the huge deployment of IoT devices for which accurate location provisioning is often a key, the interest in location-based services is continuously growing. GPS has largely solved the problem for outdoor scenarios; however, accurate localization remains a challenge for indoor environments. WiFi-based indoor localization has attracted lots of research attention for its advantage of low deployment cost by

leveraging on-device WiFi transceivers and existing WiFi infrastructure, but its accuracy remains up to few meters [11].

However, VLC does not suffer from the drawbacks of radio-like wireless channel dynamics, fading, interference or multipath. In [70], the authors introduced *Epsilon* to provide indoor localization with a decimeter accuracy, leveraging LED luminaries and any device equipped with a photodiode. The ceiling LEDs serve as anchors that beacon an ID, their position and their duty cycle. *Epsilon* adopts binary frequency shift keying (BFSK) along with frequency hopping to enable reliable transmission from multiple and uncoordinated light sources. Then, the receiver measures the RSS from the LEDs in its field of view and computes the distance to each one using a channel model. Finally, it estimates its location with trilateration using the received beacon information and distance measurements from the light sources. Experimental results confirmed the accuracy of the system: the 90th percentile accuracies are 0.4 m, 0.7 m and 0.8 m for three typical office environments with 5 LEDs. Nonetheless, the reliability of this method is still poor and the system is not robust against the device orientation or the motion, and the distance estimation accuracy is correlated with the quality of the channel model.

Ceiling LED-to-photodiode for sensing Recent advances in augmented reality and smart home appliances have enriched user experience and entertainment in indoor environments. To sense free-hand gestures, common and commercial approaches use cameras (RGB, infrared or time of flight cameras), on-body sensors, ambient radio frequency signals, and acoustic signals. In [7], the authors proposed for the first time to leverage the light for body motion sensing and user identification. The system consists, on one side, of VLC-enabled LED lights on the ceiling that emit light beacons and, on the other, of PDs on the floor. The PDs capture a continuous stream of shadow maps, each corresponding to an LED light. These shadow maps then help to localize the user's body joints in the 3-dimensions space and recognize the user based on the estimated body parameters, such as the shoulder width or the arm length. The experimental results showed the system correctly identifies 8 participants out of 10 with a mean error of the body parameter estimation of 0.03 m.

A complementary system that uses a table lamp to reconstruct the 3D hand skeleton in real time is proposed in *Aili* [73]. *Aili* consists in a LED panel with several arrays of PDs embedded in the lamp base. To reconstruct a hand skeleton, *Aili* combines 2-dimensions light blockage maps from different PDs, which

describe if a hand blocks light rays from individual LEDs to all PDs. To identify the contribution of a specific LED, each light emits a unique flashing frequency between 20.8 kHz and 40 kHz. The authors built and evaluated a prototype of their system. The results show that their algorithm reconstructs a hand pose within 7.2 ms on average, with 10.2 °mean angular deviation and 2.5 mm mean translation deviation.

A drawback of these works is their cost and deployment complexity, even on the sensor side. This concern makes these solutions not compatible with the IoT sensors constraints for which the cost, the power consumption, and the integration complexity must remain as low as possible.

2.1.2 Low Power LED to Photodiode communication

The greater part of research on LED-to-photodiode communication aims to achieve high-speed communications. However, there are some use cases where high data rate is not needed. This is especially true with the IoT context. In fact, interacting with smart objects or gathering sensors values only requires few bits, nay kilobits. Such situation is depicted in [109, 21] where a user is interacting with smart toys using a smartphone. Since today smartphones do not embed a photodiode, the authors leverage the device audio jack to plug a small VLC board embedding a PD and an LED. This approach has been proposed first in *HiJack* [67], where the authors brought heart rate sensing capabilities to an unmodified smartphone. The device is powered from the jack output signal and no additional battery is required. The audio signals directly modulate light emissions of the LED, whereas the incoming light is detected by a photodiode. The VLC clock rate is set to 10 kHz, a convenient value for an audio sampling rate of 48 kHz. The generated electrical signals are fed into the microphone input and the audio-like signal is finally demodulated by a mobile phone application.

The authors have implemented their system on an iPhone and the experimental results showed that the throughput stays stable up to 25 cm, with a maximum of 700 bps for a packet size of 150 bytes.

With the huge dependence of the users on their smartphones and the technological advances in their design, almost all smartphones now embed a camera. As a consequence, many of them also have a flashlight, which is basically an LED. For that reason, it is of great interest to use such smartphones as an off-the-shelf VLC emitter.

In this context, the authors in [40, 38] use the smartphone LED flashlight as a secure visible light replacement of magnetic cards. They use a return-to-zero (RZ) OOK modulation after demonstrating that non-return-to-zero (NRZ) modulation scheme is inefficient for data reception since it is vulnerable to synchronization problems. The experimental evaluation using a rooted Android smartphone and a PD as receiver shows that the system can reach an error-free bitrate of up to 500 bps. However, by relying on a rooted smartphone, the authors cut down the scalability of their system. In fact, such applications that require root permissions cannot be distributed through conventional stores. Also, to gain root permissions, users need advanced knowledge of the smartphone operating-system [114], which is not the case of most of the smartphone users. Note that rooting a smartphone has many drawbacks: the device becomes more sensitive to malwares or information leaks and invalidates its warranty.

Since the bitrate is radically limited by the mobile operating systems and the API offered by the platforms software development kit (SDK), there were not so many works that try to improve flashlight based communications. Nonetheless, flashlight VLC will be integrated into the upcoming IEEE 802.15.7m standard [61].

2.1.3 Passive Lights to Photodiode communication

A new research paradigm inspired by the RF backscatter [77], is brought by *Wang et al.* [126, 125] as *Passive VLC sensing*. The authors present the remaining challenges and research direction to create a novel sensing and communication method that exploit passive light sources, such as the sun and/or passive receivers (i.e. the external surfaces of objects) such as fingers and car roofs. [126] studies the channel of such particular VLC systems, whereas [125] introduces a taxonomy and points out three main challenges: (1) there is no universal solution since the object shape is uncontrolled, (2) the need for more flexible and robust methods for reception as of the uncontrolled emitters, (3) monitoring passive objects requires high density of receivers. The authors classify such system as full-active, passive source, passive receiver and full passive.

A recent research in this field is presented in [119]. The authors built a battery-free receiver for sensing and recognizing hand gesture patterns using visible light. Their system uses energy harvesting to be totally battery-less by using solar cells enabling sub-µW power consumption. Indeed, their full passive architecture requires no modification to the existing light infrastructure. Above all,

they leverage RF backscatter to transmit the sensing and hand gesture information through RF up to distances of 330 m.

2.2 LED as VLC receiver

PDs are most commonly used as VLC receiver because they are the most appropriate electronic components to sense and receive light. They operate by converting photons to current. In this work, when we talk about *PDs*, we mean PDs specially conceived as light receivers. Nonetheless, in [26], the authors proposed for the first time in the literature to use conventional LED as photosensors in a VLC application. They rely on the fact that LEDs and PDs are intrinsically close to each other. More precisely, both LEDs and PDs are semiconductors and LEDs are technically PDs specifically manufactured to emit light.

Thus, by wiring an LED in reverse-bias, the LED produces a very weak current. Even if the current intensity is negligible compared to the one produced by a PD, it can be improved with the help of additional amplifier circuit and analogical filters. The other way to make an inexpensive photodetector out of an LED is to tie the anode to ground and connect the cathode to a micro-controller (MCU) I/O pin driven high. This reverse biases the diode and charges its internal capacitance. Then, switching the I/O pin to input mode allows the photocurrent to discharge the capacitance down to the digital input I/O pin of the MCU. Finally, by timing how long this takes, we get a measurement of the photocurrent and thus the amount of incident light.

By relying on these properties, *Dietz et al.* [26] proposed *LEDComm*, the first bidirectional LED-to-LED communication protocol. The data are encoded using pulse width modulation (PWM): 0 is a short light pulse, while 1 is a long light pulse. Their prototype achieved 250 bps both in uplink and downlink in a limited communication range of a few centimeters.

Following these first demonstrations, LED-to-LED communication has been further studied and improved by *Giustiniano et al.* [42], *Schmid et al.* [107, 108] and *Corbellini et al.* [21].

In [42], the authors give a major improvement on the seminal work [26] described above. Unlike *Dietz et al.*, they used On-Off-Keying and Manchester coding on the physical layer to encode and decode a message. On the MAC layer, the authors develop an efficient collision detection medium access (CSMA/CD) protocol [20], enhanced with mechanisms to eliminate the light flicker.

The flicker elimination scheme works as follows. The channel is partitioned into Data (D) and Energy (E) symbols. E symbols are used both by the receiver and the emitter to compensate, if necessary, the flicker: the receiver emits light only during the E period, while the emitter emits light during D, and eventually during E if the light emitted during D does not suffice. The same pattern is sent when the channel is idle. This work allows for the first time to establish a robust bidirectional LED-to-LED network with up to four LEDs.

The authors apply this to a network of toys prototype showing that their system achieves a throughput of 870 bps within a range of 90 cm. They also evaluate the impact of the LED color (yellow, green, blue, red), showing that red LEDs offer the best performances. That is because the color affects the LED internal resistance and capacitance and so the discharge cycle time. Also, yellow and green LEDs are more sensitive to the corresponding component of the visible spectrum making them more sensitive to the ambient light noise.

More recently, *Wang et al.* [124] introduced the *OpenVLC* platform. *OpenVLC* is an open-source software-defined networking platform for prototyping LED-to-LED along with LED-to-Photodiode communication. *OpenVLC* simply relies on low cost off-the-shelf electronic components and leverages the Beaglebone Black, an embedded Linux board, on which a VLC frontend is plugged through a "cape". The *OpenVLC* software provides a Linux kernel driver on which the authors have implemented signal sampling, symbol detection, coding, decoding, carrier sensing and communication with the Internet layer of the Linux operating system. The PHY layer relies on OOK modulation and Manchester coding, the same as [42, 107]. The medium access control (MAC) protocol is based either on CSMA/CD or CSMA with Collision Avoidance (CSMA/CA) [20]. The first version of *OpenVLC* [124] achieves a MAC layer data rate of 2.2 kbit/s at distances up to 1 m.

Then, further improvements and studies have been successively brought by the authors to improve the MAC layer. In [122], they propose a CSMA with Collision Detection and Hidden Avoidance (CSMA/CD-HA) MAC protocol for a network where each node solely uses one LED to transmit and receive data. The CSMA/CD-HA enables intra-frame bidirectional transmission in a network with up to 4 nodes. The key idea is to exploit the intra-frame data symbols without emission of light, the "LED OFF period", to introduce an embedded communication channel. As a result of intra-frame bidirectional transmission, a device denoted as R can send signals in parallel to receiving a frame. Since parts of this signal are "LED ON" symbols, they can be sensed by the nodes in R com-

munication range. Thus, these nodes will not send frames to device R during this period, and the potential hidden-node problem is avoided. This approach enables the transmission of additional data while receiving in the same optical channel and it makes the communication robust to different kinds of field of view (FOV). The authors implement the CSMA/CD-HA protocol on the *OpenVLC* driver, and experimental results show that collisions caused by hidden nodes can be reduced and the network capacity can be increased up to 100%.

Finally, *Galisteo et al.* [41] introduce *OpenVLC* 1.2. The main change in the platform comes from the exploitation of the Programmable Real-time Units (PRUs) of the BeagleBone Black board. The author moved the VLC demodulation implementation from the Linux kernel to the PRUs. Note that PRUs are programmed in assembler and allow to control the hardware more precisely and faster. That increases the achievable throughput of *OpenVLC* to 100 kbit/s, i.e. by a factor of 8 with respect to previous versions, without adding any hardware cost. We notice that a similar approach is proposed in *PurpleVLC* [136].

With [124, 42, 136], the FOV of the LED is limited to the direction of the LED axis. This limits its coverage to a specific direction. To overcome this limitation, the authors in [65] designed an LED front end with 20 LEDs put in a circular printed circuit board (PCB). In *Shine* [65], the LEDs are equally spaced to provide coverage in 360 °. The advantage of such front-end compared to already existing platforms is that a VLC device with many LEDs can communicate with multiples nodes in different directions, enabling a VLC multi-hop network. *Shine* front-end is based on a PCB compatible with many low-cost evaluation boards, such as BeagleBone, Raspberry Pi or Arduino.

These prototyping platforms allow fast development of VLC system at a low cost. Nonetheless, their performance remains limited by the hardware, e.g. the MCU frequency, which makes them suitable for low data-rate use cases only, with low-complexity modulations. However, such VLC systems meet the needs of IoT communications and can therefore become one of the IoT communication technologies for indoor environments.

In a similar fashion, a recent research [133] shows that a lighting power LED also works as a light sensor. Nonetheless, its sensitivity is much weaker than the LEDs used by [26, 42, 107, 108, 124]. *Yang et al.* solved this problem by using a LEDs array and adapting the method in [26]: they add an analogical amplifier and modify the circuit schematics to be able to sense the signal without the need for reverse biasing the LED. *CeilingSee* [133] leverages this sensing-capable lighting infrastructure to infer the room occupancy as well as the occupant's mo-

	Type	Modulation	Throughput
SBVLC [139]	2D B	W BC	N.A. (Sec. Framework)
COBRA [49]	2D Color BC.	OOK	172 kbit/s
PixNet [95]	LOS	OFDM	12 Mbps
SoftLight [30]	2D Color BC.	FSK	380 kbit/s
HiLight [71]	alpha changes	BFSK	11 kbit/s
Styrofoam [71]	2D Color BC	FSK	69.1 kbit/s
LightSync [58]	2D Color BC	FSK	11 kbit/s

Table 2.1: Screen-to-camera systems summary

tion. *CeilingSee* senses the light reflexion on the floor, and applies pre-processing to remove noise with smoothing, and finally uses a machine learning method based on the Support Vector Regression model to determine the occupancy. The authors have conducted extensive experiments in a laboratory area. The results showed that *CeilingSee* can determine the number of occupants in a $30\,m^2$ room with more than 80 % occupancy, even in dynamic scenarios.

2.3 Screen-to-Camera

The rising number of cameras and screens in today environment gives a great opportunity for using these devices for communication. The key feature of these screen-to-camera links is that they are highly directional enabling a form of interference-free wireless communication. Also, since both screens and cameras are 2-dimensions emitter and receiver respectively, the screen-to-camera link can be seen as a multiple-input-multiple-output (MIMO) communication channel, where each pixel is an antenna. Thus, screen-to-camera communication has the potential to provide ubiquitous broadcast communication with a higher throughput than LED-to-camera communication. Nonetheless, enabling screen-to-camera communication raises several challenges. Table 2.1 summarize the previous studies in the field that are introduced in this subsection.

In *PixNet* [95], the authors present a system for transmitting information over LCD-to-camera links. *PixNet* leverages OFDM transmission algorithms to address the characteristics of the LCD-to-camera link, namely the perspective distortion, blur, and sensitivity to ambient light. In fact, *PixNet* is a kind of 2-D barcodes but, rather than encoding data directly in the visual domain, *PixNet* encodes informa-

tion in the frequency domain. This approach is quite similar to OFDM, but unlike RF-based OFDM that encodes data in time frequencies, *PixNet* encodes data in 2-D spatial frequencies. *Perli et al.* [95] implemented *PixNet* using off-the-shelf LCDs and cameras and conducted experimental evaluation showing that their system offers a throughput of up to 12 Mbps at a distance of 10 m, and works with view angles as wide as 120 °.

Another approach for screen-to-camera communication was presented in *COBRA* [49]. *COBRA* is designed to achieve broadcast communication between small size screens and smartphone cameras using 2-D color barcodes. At the emitter, a header and the CRC checksum are added to the original data. Then, each byte of the data block is encoded by a certain color to generate a color barcode that is then displayed on the screen. The barcode is divided into the following three types of areas: 1) corner trackers, 2) timing reference blocks and 3) a code area in the center. The corner trackers (1) are used to locate the barcode on the screen using green and red blocks at the top-left and bottom-right corners respectively, and blue blocks at the other two corners. They also can be used by the smartphone to detect the black and white timing reference blocks (2). These timing blocks (2) are used as the reference for detecting the data blocks in the code area (3). In the code area, the data is encoded by a sequence of color blocks: red, green, blue and white colors represent 00, 01, 10, and 11 respectively. According to *COBRA* bit-mapping, this means that 10110001 can be encoded in the color blocks blue-white-red-green.

At the receiver, since the camera frame rate is higher than the barcode rate, *COBRA* selects the best quality for each barcode for further processing and decoding. The experimental results show that *COBRA* can achieve a throughput of up to 172 kbit/s.

The limited available throughput in screen-to-camera links was further increased in *LightSync* [58]. By improving the frame synchronization between the transmitter and receiver, *Hu et al.* [58] nearly double the achievable throughput of COBRA. *LightSync* leverages linear erasure coding to recover the lost frames. Then, color tracking is used to decode the data correctly from imperfect frames. Another coding scheme referred to as *Styrofoam* [76] addressed the problem of ISI due to lack of synchronization, by inserting empty frames in the generated set of frames.

More recently, in *HiLight*, *Li et al.* [71] introduced a new method for screen-to-camera communication without any coded images. By leveraging the properties of the transparency channel, called *alpha channel*, *HiLight* embeds the bits by

changing the pixel transparency, instead of modifying the RGB color. The transmitter modulates bits using BFSK, where bit 0 and 1 are represented by different frequencies of transparency changes over a certain number of frames. Experimental results demonstrate *HiLight* reaches up to 240 bps by using off-the-shelf smartphones.

To improve the robustness of the screen-to-camera system mentioned above, *Du et al.* [30] introduced *SoftLight*, a channel coding approach that automatically adapts the transmission data rate to the link qualities over various scenarios.

SoftLight enhances the state of the art by proposing a color modulation scheme on top of existing barcode-like modulation, providing a soft hint about the bit error rate (BER). That is to say, the receiver will discern if a bit can be considered as trustful or not. This allows establishing a bit-level VLC erasure channel instead of a packet erasure channel. Thus, the authors were able to develop a rateless coding scheme that achieves rateless transmissions at the bit level with low computation complexity. In fact, *SoftLight* is orthogonal to the previous screen-to-camera coding schemes and can be applied on top of any barcode layouts like *QR Code* or *COBRA* [49]. Their system exploits the independent components in YUV color space, along with a specific set of colors to reduce the interference between adjacent symbols for better color conservation on a picture.

The lightweight bit-level rateless coding scheme first encodes the data by a FEC code, like Reed-Solomon (RS) code or Low-Density Parity-Check (LDPC) code, before performing XOR operations at frame level to produce a set of rateless frames. On the receiver side, the smartphone decodes the frame by performing the reverse operations. Then, the soft hint provided by the color modulation allows determining the error probability for each bit position, which enables bit-level rateless coding. Also, a majority vote algorithm is used to eliminate the impact of bits with high error probability.

Experimental evaluation on Android smartphones shows that this approach can correctly transmit a 22 kB photo between two smartphones within 0.6 s and improves the average goodput of *COBRA* [49] by 2.2 times.

With the increasing popularity of screen-to-camera links for near-field communication, it has become necessary to address the security aspects of the channel. *Zhang et al.* [139] addressed screen-to-camera communication security challenges, and proposed enhancements by manipulating the screen viewing angles and user motion tracking.

	Propagation	Modulation	Use case	Shutter	Throughput	Error correction
Danakis [22]	LOS	OOK	COM	RS	148bps	N.A.
RollingLight [68]	LOS	FSK	COM	RS	90.56bps	Parity
Ceiling[Cast,Talk] [48, 132]	LOS	OOK	COM	RS	1.3kbit/s	Raptor coding
Undersampled [103]	LOS	UFSOOK	COM	GS+RS	15bps	N.A.
Luxapose [66]	LOS	OOK	LOC	RS	N.A.	N.A.
LiTell [140]	LOS	OOK	LOC	RS	N.A.	N.A.
Pulsar [141]	LOS	N.A.	LOC	RS	N.A.	N.A.
iLamp [143]	LOS	N.A.	LOC	RS	N.A.	N.A.
ColorBar [56]	LOS	CSK	COM	RS	5.2kbit/s	Reed Solomon
HybridVLC [98]	NLOS	BFSK	COM	RS	10.4bps	N.A.
Ferrandiz-Lahuerta [36]	NLOS	OOK	COM	RS	700bps	Repeat + CRC4
MARTIAN [29]	NLOS	RIM	COM	RS	1.6kbit/s	FC
ReflexCode [134]	NLOS	GSK	COM	RS	3.2kbit/s	Raptor coding
VLandmark [99]	NLOS	FSK	LOC	RS	12bps	N.A.
LiShield [142]	NLOS	OOK+CSK	PRIV	RS	N.A.	N.A.

Table 2.2: LED-to-camera systems summary

2.4 LED-to-camera

The majority of the VLC research is focused towards creating efficient and high-speed data links. For this purpose, LED-to-photodiode links are used, as PDs provide fast-response and high bandwidth, that can support complex spectrally-efficient modulation schemes. Nonetheless, smartphones can also be used as a receiver in VLC systems when high throughput is not a requirement.

Danakis et al. [22] illustrate the fact that the cameras of smartphones with the so-called rolling shutter mechanism, further detailed below, can capture patterns from fast modulating light sources. Using this mechanism, a camera can be used as a VLC receiver for data transfer and localization.

Rajagopal et al. [99] use a BFSK scheme with iOS devices as a receiver, which results in a throughput of 10 bps up to a distance of 1 m. The authors of [22] employed OOK and Manchester encoding schemes and achieved a data-rate of 1-3 kbit/s at close proximity of 9 cm by using the frame-rate of 20 fps. RollingLight [68] presents a frequency shift keying (FSK) modulation scheme and, using iPhone 4 as the receiver, they reported a throughput of 90.56 bps up to a distance of 1.6 m using a light source of 60 cm x 60 cm.

A recent work presented by *Schmid et al.* [106] achieved a throughput of more than 1.2 kbit/s with a pulse position modulation (PPM) encoding scheme. They implemented their receiver on iPhone 6s with a frame-rate of 240 fps and reported a maximum distance of 2.75 m.

All of these works use different encoding schemes and utilize different smartphones, which results in different throughputs generated by the system. Comparing these works only on the basis of throughput would not be fair, as at least four components are involved here: the data encoding method, the smartphone camera, the light propagation, i.e. line of sight (LOS) or non-line of sight (NLOS), and finally the emitter characteristics.

Such differences are illustrated by the following facts. Works [68] and [103] implemented different methods and had different camera capabilities, i.e with rolling or global shutter respectively, and managed to get similar throughput values. Also, *Schmid et al.* [106] and *Ferrandiz-Lahuerta et al.* [36] obtain two orders of magnitude higher throughput than [103] and [68] but they rely on non line-of-sight (NLOS) propagation, whereas [103] and [68] leverage a line-of-sight (LOS) channel.

We introduce a taxonomy to help the classification of the previous studies in the field. Tab. 2.2 classifies using this taxonomy the works that are further de-

tailed in the following of this section. To better understand the key mechanisms involved in LED-to-camera communication, we first give insights on the camera operations and the rolling shutter effect, and sum up the modulation schemes often used in LED-to-camera communications. Finally, we conclude this section by further describing LED-to-camera communication systems we found in the literature.

Proposed Taxonomy

To classify and better compare the different LED-to-camera implementations, we propose the following taxonomy:

- rolling vs global shutter mode: leveraging the rolling shutter effect has been proposed to improve the throughput by two orders of magnitude [22]. However, such methods have the downside to reduce the range, the reliability, and the resistance against ISI or background noise.
- LOS vs non-LOS: depending on the application requirements, LOS or NLOS communication has been considered. Because the channel properties are distinct, different loss mitigation and modulation methods have been investigated.
- modulation: a wide range of modulation schemes have been tested in LED-to-camera communication, such as OOK, FSK, undersampled frequency shift OOK (UFSOOK) or grayscale shift keying (GSK).
- LED type: three types of LED have been used for LED-to-camera communication. They are ceiling power LED (CP), low-power monochrome LED (LPM) and RGB LED.
- application: LED-to-camera communications have been proposed mainly for localization (LOC), communication (COM), and privacy protection (PRIV).
- redundancy and error correction: different mechanisms to improve the LED-to-camera communication reliability have been studied, such as Reed-Solomon (RS) code, repeat packets (R), Raptor codes, or Luby Transform (LT) codes.

We notice that the light propagation has a major impact on the LED-to-camera system design, and mainly influences the other classification criteria introduced

Figure 2.1: On the left, the picture is taken by a global shutter camera. On the right, the camera has a rolling shutter. Source: smartereveryday.com.

above, and so the global performance. Thus we start below by introducing systems that rely on LOS channel in Subsection 2.4.1, before focusing on NLOS communications in Subsection 2.4.2.

Rolling Shutter Effect

Nowadays cameras widely use two types of image sensors, Charge Coupled Devices (CCD) and Complementary Metal Oxide Semiconductors (CMOS). These two technologies have some similarities but one major distinction is the way each sensor exposes its pixels to light. CCD sensors often use the Global Shutter readout mode, where all pixels are exposed simultaneously and then each pixel is read sequentially. This mechanism helps in capturing a still image of a moving object as shown on the left of Figure 2.1.

On the other hand, CMOS sensors use the Rolling Shutter readout mode, where each row is exposed in a row-sequential way with fixed time delay. Due to this mechanism, there is a significant time difference between the beginning of the exposure of the first and the last row, making them no longer simultaneous. This leads to a distortion in the captured image of a fast moving object as depicted on the right of Figure 2.1

When an LED is modulated at a frequency higher than the rolling shutter speed, stripes of different light intensity are captured in the image, as shown in Figure 2.2. A row appears white when the LED was ON during the row exposure time. On the other hand, a row appears black when the LED was OFF during the exposure time. The intensity and width of the strip depend on the transmitter modulation frequency and the camera properties.

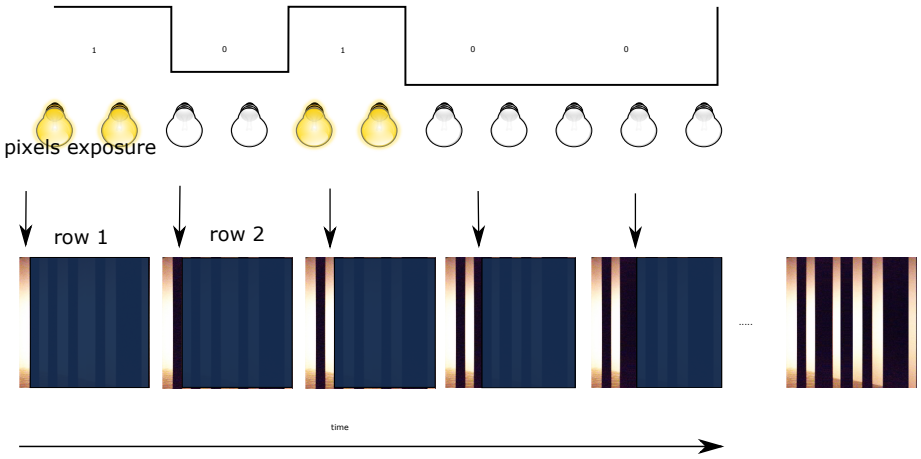

Figure 2.2: Stripe formation and LED state. The width of each strip corresponds to the duration of the LED state (ON or OFF)

A key camera property in determining the rolling shutter speed is the exposure time. Today, smartphone cameras are equipped with CMOS image sensors due to their low power consumption and low cost. Mid-range image sensors can capture a video of 25 to 30 fps in full resolution mode, while the top of the line smartphones can capture up to 940 fps. However, these cameras emphasize more towards picture quality, which leads to an unstable frame rate.

LED-to-Camera Modulations

The rolling shutter phenomenon enables communication between an LED and a camera where multiple data symbols can be transferred within one camera frame. The literature reports several modulations that have been investigated. In the following, we summarize the advantages and downsides inherent to each method.

- **On-Off Keying (OOK):** In OOK modulation, LED ON and OFF states are used to encode 1 and 0 respectively. Because OOK symbols need to be short enough to avoid a visible change in the LED brightness, which is the flickering effect, this scheme is affected by the ambient noise. OOK

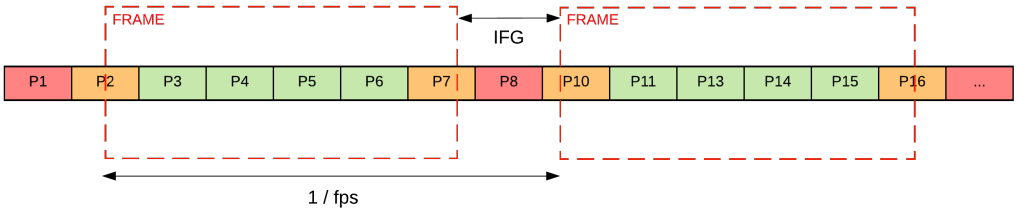

Figure 2.3: The time gap between two consecutive frames causes packet loss. The green packets are succesfully received, the orange ones are truncated and the reds are lost between two consecutive frames and so do not appear in any frame.

can also produce human perceivable LED flickering in the case of long sequences of 0 or 1 in the transmission data. That is why OOK is often combined with a duty-cycle (DC) balanced run-length limited (RLL) code like Manchester [22, 36, 48, 132, 66] or 4B6B [62].

- **Frequency Shift Keying (FSK):** To address the limitations of OOK, [66, 68, 99] have proposed to use FSK where different symbols consist of many ON-OFF bands at different frequencies. FSK reduces the demodulation error due to longer symbol duration and multiple ON-OFF bands in each symbol. That leads to a capacity an order of magnitude lower than OOK, with a throughput of 90.56 bps and 10 bps in [68] and [99].

- **Undersampled Frequency Shift ON-OFF Keying (UFSOOK):** The technique encodes the bits using a duty cycle (DC) balanced differential encoding. The modulation concept proposed in [103] is similar to FSK inasmuch as UFSOOK defines two OOK frequencies for encoding bits that are high enough to avoid flicker. Howewer, the logic 1 and logic 0 frequencies are selected in such way that when undersampled by a low frame rate camera, they alias to low pass frequencies that can then be further processed to decode the bit values. For example, if the camera has a frame rate of 30 fps, and the frequencies are 120 Hz and 105 Hz, then the aliased frequencies as seen by the camera are respectively 0 Hz and 15 Hz.

- **Color Shift Keying (CSK):** CSK was first proposed by the IEEE 802.15.7 standard [62] for visible light communication and further studied in LED-to-camera communication in [56]. CSK exploits the design of RGB LED fixtures which use three separate LEDs to generate a variety of colors, including white, using R, G and B mixture. CSK modulates the signal by modifying

the intensity of the three colors. It uses the color space chromaticity diagram defined by the International Commission on Illumination (CIE) [118]. The diagram maps colors perceivable by human eyes to a set of x and y coordinates. The IEEE 802.15.7 standard provides 8 and 16 CSK constellation symbols. The constellation symbols are chosen inside the CIE colors triangle such that the inter-symbol distance is maximized.

- **Grayscale shift keying (GSK) and relative intensity modulation (RIM)** To improve the communication throughput of OOK, modulation based on light intensity variation has been proposed in [29, 135]. Light intensity modulation is achieved in two different ways: by dimming the LED [29], or by superposing light reflexions. The main drawback of such kind of modulation is the high bit error induced by ISI: the light intensity strongly depends on the distance or the angle between the enlightened surface and the camera, the surface color, the ambient lighting conditions, etc. Also, accurately dimming an LED is not straightforward and it increases the emitter complexity.

2.4.1 Line of Sight Communications

This subsection focuses on the past efforts on leveraging a LOS LED-to-camera link. In such situation, the optical signal only occupies part of the image unless the camera is very close to the LED.

CamCom [102] uses UFSOOK by encoding data at frequencies that are harmonics of the frame rate and decoding data by processing the subsampled aliased frequencies. The key advantage of this scheme is that it works with both global as well as rolling shutter sensors. Note that UFSOOK has been contributed to the IEE 802.15.7m standard. However, it operates at frequencies around 120Hz, which can cause flickering.

In [66], the authors present *Luxapose*, a method for indoor positioning using VLC that achieves positioning and orientation with decimeter accuracy. Their system consists of a camera sensor capturing multiple LED luminaries positioned on the ceiling, which use a unique blinking frequency as the device identifier. Then, by querying a cloudlet, the mobile device translates the LED identifiers into location coordinates that can be used to derive its position via triangulation. The authors have developed a LED detection and frequency detection method using computer vision processing with the OpenCV library [63] as follows: they convert the image to grayscale, blur it, apply a binary Otsu filter [92] before find-

ing the contours for each blob. After finding each region of interest (ROI), i.e. each transmitter, they perform YIN algorithm [23] for frequency estimation on each ROI. The median processing time for the full 33 MP images captured by the smartphone is about 9s: taking pictures: 4.46 s, upload: 3.41 s, image processing: 0.3 s, estimate location: 0.87 s).

Visible light positioning with unmodulated light has been widely studied by Zhang et al. in [140, 141, 143]. LiTell [140] is a localization scheme that employs unmodified fluorescent lights (FLs) as location landmarks and relies on the observation that each FL has an inherent characteristic frequency which can serve as a discriminative feature. The authors proposed a set of sampling, signal amplification and camera optimization mechanisms that enable a smartphone to capture high frequency (> 80 kHz) features. The experimental evaluation showed that LiTell provides sub-meter accuracy and orientation with less than 3 °error.

Pulsar [141] extends LiTell by using a compact PD sensor board plugged into a smartphone. It leverages the PD to better discriminate existing ceiling lights — either fluorescent lights or conventional LEDs — based on their intrinsic optical emission features. Also, it overcomes the photodiode lack of spatial resolution by using the smartphone front-camera. Thus, Pulsar resolves the light source angle-of-arrival by photogrammetry to triangulate the device 3D location and even orientation.

iLamp [143] uses a smartphone camera to extract visual features in unmodified LED/FL lamps, i.e the color and geometric form, and identify each lamp as a landmark. iLAMP further introduces a sensor-assisted photogrammetry technique to estimate the 3D location with a small 90-percentile error of 3.5 cm (2.8 °). The authors implement iLamp on an Android smartphone and reduce both the power consumption and latency by duty-cycling the camera usage. The experimental evaluation shows their system is low-latency, between 400 and 700 ms, with a power consumption of 927 mW. That is less than 1/2 compared with LiTell [140] or Luxapose [66]).

In ColorBars [56], Hu et al. propose a system that exploits RGB LEDs to improve the LED-to-camera communication throughput. ColorBars operates as follows. On the transmitter side, the input data bit stream is divided into blocks of bits, and error correction coding is applied to each block. ColorBars uses RS coding to deal with the data loss due to the inter-frame gap (IFG) and ISI errors. The blocks of data and parity bits are then used to form packets, where a header and a packet delimiter are added to each packet. The encoded bits of the packets are then modulated into a stream of symbols using CSK modulation: the

bits are split into blocks of 3-bits and each of the blocks is mapped to a color symbol according to the CSK constellation. To keep the perceived colors always the same, dedicated illumination symbols of white light are inserted. Also, to face the color diversity on the smartphone, ColorBars emitter sends calibration packets periodically. The symbols in the calibration packets are used as reference symbols for demodulating the symbols in the data packets. Like *Lahuerta et al.* [36], the authors determine a packet size large enough to not be missed during the IFG, but small enough to fit into a single picture. The drawback of such method is that the distance between the receiver and the emitter must be constant and must be small enough to make the rolling shutter stripes cover the whole image. For demodulation on the receiver side, ColorBars uses the CIELab color space. CIELab is a three channels color space with one channel for lightness L and two channels for colors (a and b). The authors implement ColorBars in a testbed and experimental evaluation showed that it could achieve a data rate of 5.2 kbit/s on Nexus 5 and 2.5 kbit/s on iPhone 5S devices.

The authors in RollingLight [68] extensively studied the rolling shutter effect and found an unpredictable and varying idle time gap (jitter) between two consecutive frames. This idle gap leads to loss of the data packets in the captured picture, which is depicted in Figure 2.3. This packet loss was observed in the case when the frame duration was slightly greater than the packet duration. This means that one picture can capture at most one packet, but the varying idle time gap can result in random packet loss and frames without a complete packet. To face this issue, RollingLight adds a parity symbol every two packets by XORing these two symbols. Thus, if any of those two symbols is lost, it can be recovered by XORing the remaining one with the parity symbol.

More recently, Hao et al. presented CeilingCast [48]. Their VLC LED-to-camera system leverages commercial off-the-shelf (COTS) LED stripes to create a LOS broadcast channel with a Nexus 5 smartphone. Relying on the combination of OOK and Manchester coding proposed in [22], they innovated in both encoding and decoding schemes to improve link reliability and throughput. They leverage the rateless Raptor Code [112] to face the unavoidable packet erasure caused both by the camera and the distance.

Raptor Code has been proposed to improve the efficiency of the Luby Transform (LT) Code [79] using an inner and outer code as follows: a set of intermediate packets are firstly derived from the original packets so that the intermediate packets can sufficiently reconstruct the original ones. The repair packets are then produced by applying LT encoding. Each is derived by XORing some inter-

mediate packets. The final encoded packets are the combination of the original and repair ones. As for LT Code, the downside of such method is the overhead introduced by the k extra packets and the information added to each one to let the receiver correctly reconstruct the initial message.

Also, since CeilingCast relies on the full lighting system, it has the opportunity to leverage multiple LED stripes to perform MIMO communications. The system assigns the n packets to broadcast to each emitter before encoding them using Raptor Codes. Thus the receiver would successfully reconstruct the broadcast message by receiving any of the $n + k$ packets from any emitter. The experimental evaluations show that CeilingCast achieves 0.4 kbit/s with a single LED and can reach 1.35 kbit/s with 3 LEDs at 2.2 m distance. The authors indicate that the communication range is up to 5 m.

Hesselmann et al. [51] have proposed a method to enable bidirectional communication between smartphones and vision-based tabletops camera. In their approach, high and low bits are represented by turning the smartphone flash ON and OFF within a defined time frame. At the PHY layer, they use NRZ OOK modulation with 50 ms pulse length. Besides, the authors did not implement upper layer protocols for error detection or framing. On the tabletop side, because the symbol rate is lower than the camera frame rate, the signal can be recorded without loss and the demodulation is therefore straightforward. To evaluate the throughput of this flash-based communication, they implemented a test application on a non-rooted Nexus One and showed that their system could achieve a maximum throughput of approximatively 20 bps.

2.4.2 Non Line of Sight Communications

NLOS LED-to-camera communications use an illuminated surface as the source of the data, i.e. the reflected light, instead of the direct LOS, eliminating the need to point the phone directly at the LED. The NLOS LED-to-camera communication typically has higher throughput because the light signal occupies the whole image. However, RSS of a NLOS link is attenuated significantly due to the diffuse reflection, which increases the demodulation error. The following works rely on NLOS rather than LOS light propogation.

Unlike CeilingCast, *Ferrandiz-Lahuerta et al.* [36] exploits the light reflexion on a wall instead of the LED luminary thoroughly. In such situation, the rolling shutter pattern that encodes information covers the whole pictures. Thus, detecting the emitter and extracting the ROI becomes unnecessary. With respect to the

Danakis et al. [22] seminal work using OOK modulation and Manchester coding, the contributions of [36] are two-fold. First, the authors took a deep dive into the Android Camera2 API [43] in November 2014. They also gave engineering insight into the Android settings, which let them observe a rolling shutter artifact with a finer definition than the legacy camera API. This allowed increasing the emitter frequency compared to prior implementations. Secondly, the authors proposed a protocol and error detection and correction mechanisms to face asynchronous communication between the LED and the camera. The error detection and correction consist in CRC codes and is built in the following way: the input bit stream is split into 24 bits *groups* and an individual CRC4 is computed for each group. The set of 24 CRCs is appended at the end of the bitstream to be transmitted. Then, a *block* is built by prepending a 7-bits sequence number and a synchronization bit to a *group*. Finally, each *block* is sent twice, to be sure that each picture contains at least one whole block and that two consecutive blocks are captured into two successive pictures. Nonetheless, this design assumes the communication is continuous, without any loss, so missing just one block will make the transmission fail.

In [99], the authors introduce an asynchronous protocol using BFSK between an LED and an iPad. They designed a prototype able to transmit at a data rate of 10 bps on 29 channels concurrently to devices with CMOS cameras. However, they did not perform the demodulation in real-time on a smartphone, but provided a Matlab implementation on a laptop. Their method uses a sliding window followed by a 1-D Fourier transform for preamble and frequency detection. If their work mainly focuses on the communication channel, they also propose to localize devices by simply selecting the nearest landmark with the highest RSSI.

Recently, *Yang et al.* propose *ReflexCode* [135] to improve the state of the art NLOS VLC system performance by using GSK modulation. Essentially, *ReflexCode* system codes information by superposing light emissions from multiple emitters and by smartly converting destructive interferences into collaborative transmissions. The 4-GSK modulation is obtained by superposing different ON-OFF states of three synchronized LED luminaries as follows: a) all three transmitters OFF to generate a dark band for 00 (S0), b) the middle transmitter ON while others OFF for 01 (S1), c) the complement of S1 for 10 (S2), d) all ON to build S3.

On the receiver side, the decoding process combines traditional amplitude demodulation with slope detection to decode the grayscale modulated signal, and it tunes decoding thresholds dynamically depending on the spatial symbol distribution. Also, the authors design a DC-balanced code to avoid flicker from

individual transmitters. They addressed demodulation errors and intra-frame packet losses using Raptor Code, as in [132, 48]. *ReflexCode* experimental evaluation demonstrates that it can achieve a throughput up to 3.2 kbit/s at a distance of 3 m.

In [29], the authors propose a new modulation scheme and design link-layer protocols for improving the network data rate in an NLOS scenario. Their protocol, *MARTIAN*, allows broadcast communication from the light reflexion on a wall or a picture, to a group of smartphone cameras.

On the physical layer, *MARTIAN* uses Manchester coding along with light intensity based modulation. Since the light intensity strongly depends on various factors such as the distance or the angle between the lighted surface and the camera, *MARTIAN* exploits the pattern of intensity variation rather than absolute intensity value. For instance, they encode 00, 01, 10, 11 in four light intensity changes: 0% to 50%, 0% to 75%, 50% to 75% and 75% to 75% respectively, were 0% is the lowest intensity level, 50% the middle and 75% the highest.

On the link layer, to handle the IFG, the authors divide the original message into sub-messages and use Fountain Code [81] to encode the data packets. To manage the overhead introduced by Fountain Code regarding the extremely limited packet size in the light-camera communication, *Du et al.* [29] propose to minimize its encoding overhead by introducing a Fountain code table, which allows us to use only a few bits to maintain the encoding index and significantly improves the achievable data rate. *MARTIAN* can achieve a data rate of about 1.6 kbit/s even with an NLOS link.

In *LiShield*, *Zhu et al.* [142] leverage the rolling shutter artifact to protect a physical scene from photographing, by illuminating it with smart LEDs modulated by FSK modulation. This will cover any picture taken with a CMOS camera with dark vertical stripes. They also designed mechanisms to unblock authorized cameras: using a specific frequency hopping pattern on the LEDs side, a smartphone perfectly synchronized with the emitter adapts its camera exposure time to follow the emitter frequency hops, which removes the rolling shutter artifact.

2.5 Conclusion

In this chapter, we have detailed the background works upon which we build within this thhesis. For this purpose, we followed the classification we have proposed in the previous chapter. We began with the works utilizing PDs as VLC receiver in Sec. 2.1, and continued showing in Section 2.2 that LEDs can also be

used as a receiver. Then, we dedicate the last two sections to past efforts on leveraging cameras as VLC receiver.

All these works are summarized in Tab. 2.3, that is a filled version of Tab. 1.1 introduced in the previous chapter. The table brings out that past VLC research mainly focused on lighting power LED-to-photodiode communication. Also, it highlights the lack of studies related, first, to small color LED-to-camera communication, and secondly, to flashlight-to-small color LED communication. These fields, that are shown in green in Tab. 2.3, are thus addressed in this thesis.

	PD	Small Color LED	Camera
Flashlight	[40, 38]	[21]	[51]
Lighting Power LED	[138, 24, 85, 116, 82, 54] [55, 91, 100, 104, 96, 45, 131, 127, 5] [17, 127, 64, 8, 27, 27, 15] [35, 34, 47, 97, 60, 74, 110, 19, 53] [120, 137, 128, 117, 109, 70, 7, 73]	[41, 136]	[22, 68, 48, 132, 103, 66, 140, 141] [56, 98, 29, 36, 134, 142, 143, 98]
Small Color LED	[41, 124, 122, 136]	[41, 26, 42, 108, 124] [122, 107, 136, 65]	
Screen	∅	∅	[71, 58, 71, 30, 95, 49, 139]
Passive Light	[126, 123, 119]	∅	[141, 140]

Table 2.3: VLC systems classification

Chapter 3

Design and Evaluation of a LED-to-Camera Communication System

3.1 Introduction

In this chapter, we focus on the design of a VLC transmission system between a low-cost, low-power colored LED that is part of an IoT device and an unmodified and non-rooted smartphone. The IoT device is thus able to send information through its LED.

Achieving this requires us to cope with several issues unsolved so far. Our first implementation based on state of the art solutions [68, 36] gave us the following engineering insights:

1. Image processing with computer vision algorithms and filters to detect the LED position on the picture was too slow to achieve real-time processing.

2. Since the LED on the picture is small, the system becomes very sensitive to the user motion as the LED position highly varies between consecutive frames. Thus, performing LED detection only once to fasten the process cannot be envisaged [68];

We describe our LED-to-smartphone VLC system and the experimental platform in Sec. 3.2. Section 3.3 presents the design and the implementation of decoding processes that resolve the concerns pointed out above. Then, we present the testbed performance evaluation results in Sec. 3.4 Finally, we conclude this chapter summing up the lessons learned achieving this work.

Our main contributions introduced in this chapter are two-fold:

1. We deal with the LED-to-smartphone issues discussed above and introduce an original decoding algorithm that decodes every frame in real-time. By bypassing computer vision techniques, our algorithm processes every 1920x1080 picture in 18.4 ms on average, on a Nexus 5 with a Java implementation. Using native C code, the whole processing takes only 3.6 ms. These results outperform previous works [68] and add robustness against user motion.

2. We propose for the first time in the literature a thorough experimental evaluation of a VLC system composed of a low power colored LED and a smartphone. The experimental evaluation studies the impact of the PHY layer parameters, environmental factors, and also the user behavior, i.e. the motion, the angle between the LED and the smartphone, or the distance.

3.2 System description

As we have shown before in the Chap. 2, previous studies [36, 99] demonstrate the performance of LED-to-camera communication, by using commercial lighting LED bulbs. Instead, our focus is on small color LEDs, such as those integrated in most daily usage objects.

For instance, *RollingLight* [68] uses a 500 mA, 1340 lm LED (Bridgelux bxra-56c1100 LED), while the LEDs we use in our study are CREE C503B-RAN, with 20 mA and 1.2 lm. While many concepts from the literature can be reused, regardless the type of the LEDs, the reduced size and power create some new challenges, as detailed below.

We designed a 40x20 mm printed circuit board (PCB) shown in Fig. 3.1 that supports a red 5 mm LED, a micro-controller unit (MCU), a surface mount device (SMD) RGB LED whose color can be easily changed, and a temperature sensor, to broadcast the ambient temperature through the light.

To evaluate the system, we also use a Nucleo development board to interface the MCU with more convenience, and so, change without hardware modification the LED type, the General Purpose Input/Output (GPIO) mapping or the firmware implementation. The development board is depicted in Fig. 3.2.

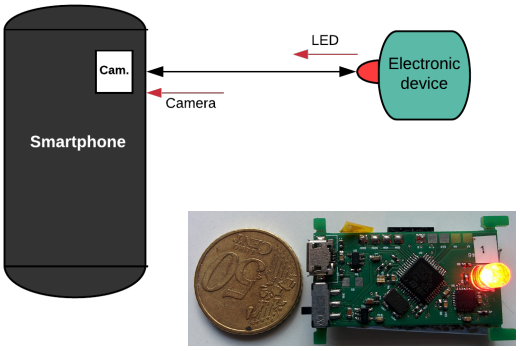

Figure 3.1: On the left, our LED-to-Camera VLC system. On the right, our smart VLC device prototype.

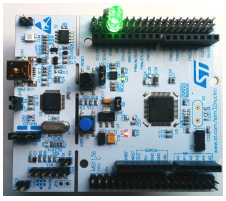

Figure 3.2: The Nucleo STM32L0 development board

3.2.1 Smart VLC device design

We choose the STM32L051 low cost and low power MCU from ST Microelectronics, an MCU similar to those already integrated in most household appliances. The core is a Cortex M0+, running up to 32 MHz, with 32 Ko Flash and 8 Ko RAM. To get a better clock accuracy and avoid clock bias due to the temperature, we use an 8 MHz high speed external crystal oscillator as the clock source and make the core run at this speed.

As proposed in [22], [68], the LED signal is modulated using the OOK modulation scheme. We consider a clock-rate varying from 2 to 10 KHz, which is a suitable bandwidth for Optical Camera Communication (OCC) using the rolling shutter effect [36]. To ensure a balanced duty cycle signal and avoid any flickering effect, we use the Manchester coding proposed in [22], [36], [90]. This RLL code ensures the signal is DC balanced and will not cause flickering.

The information is divided and transported in physical layer service data units (PHY-SDUs)[1] with a size between 20 to 128 bits. Fig. 3.3 show our PHY-SDU format with a PHY-payload of 20 bits; four bits for the SYNC sequence, prepended to this

[1]The physical layer service data unit is generally known as a *frame*. However, in order to avoid any confusions with the picture frames, we denote it as *PHY-SDU* throughout the paper.

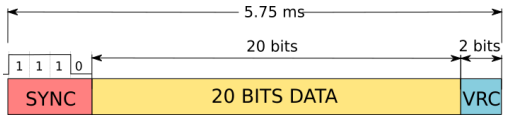

Figure 3.3: PHY-SDU format and its duration assuming a 8 KHz clock rate and 20 bits payload.

payload, finishing with two integrity bits, based on parity check. This results in a 26 bits message, whose transmission occupies the channel for 5.75 ms when using a symbol rate of 8 kHz.

3.2.2 Receiver design

On the receiver side, we use a LG Nexus 5 smartphone running Android Marshmallow version number 6.0.1. It has a Qualcomm Snapdragon 800 quad-core CPU 2,26 GHz CPU and 2 Go RAM. Its 8 megapixels 1080p 1/3.2" CMOS sensor with 1.4 μm pixel size can capture up to to 30 frames per second and supports advanced imaging application provided by the Camera2 API.

We have developed an Android application that sets up the camera parameters to observe the rolling shutter effect produced by the modulated LED. For that, based on [36], we set a very short exposure time and an increased sensor sensitivity, respectively to 100 μs and ISO 10000. As soon as a new frame is available, the application creates and starts a new thread to process and decode the picture on the background. This processing consists of a LED detection algorithm that extracts the region of interest (ROI) before performing signal processing methods to retrieve the transmitted information. This step in described more precisely in Sec. 3.3.

3.3 Enabling Real Time Computation

Computation time is an important metric in OCC, and even more when a real time transmission is required. For real time operations, all the image processing operations must be completed within the camera frame interval, i.e. the time needed to acquire a picture, which is in our case 30 ms. A higher processing duration would reduce the system throughput or introduce delay. Besides, loading the receiver with intense processing tasks will alter the capture rate regularity, and flood the RAM of the smartphone.

Figure 3.4 shows the duration distribution of the whole decoding chain on the Nexus 5 smartphone. On the left part of the figure, we show that a naive implementation, respecting the state of the art approaches [68], does not cope with real time constraints. However, as the right part of the figure highlights, after modifying the receiver Android application, we can process each frame in real time, by detecting the LED and decoding the signal, on average, in 18ms. The rest of this section describes the steps we followed to achieve this major gain in computation time.

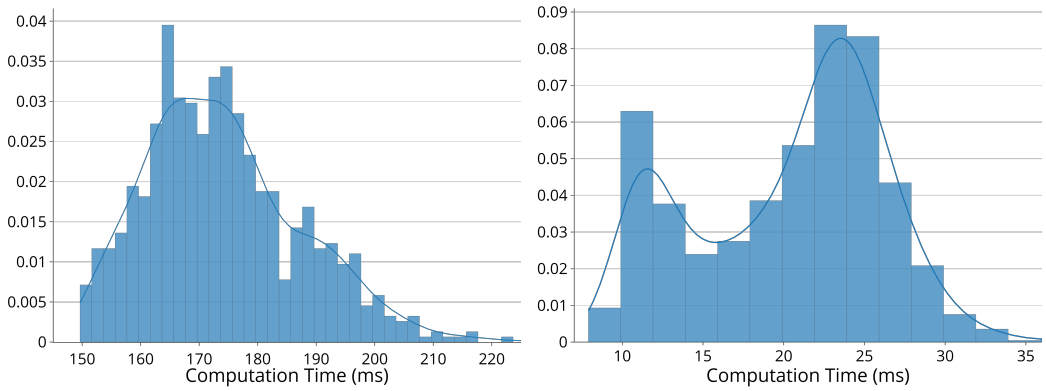

Figure 3.4: Algorithm computation time. State of the art implementation (left) and our implementation (right)

3.3.1 Algorithm presentation

The current state of the art solution, *RollingLight* [68], functions on ROIs of at least 600 pixels, achieving a throughput of 12 bytes/s. In *RollingLight*, the picture frame processing time is around 18 ms. However, this only counts the time required to extract the embedded information, and it does not take into account the ROI detection step, which is time consuming and performed only once, when the first frame is processed. Without skipping the ROI detection step, a feature which makes it deeply sensitive to motion, *RollingLight* would not be able to decode the transmitted information in real-time, i.e. it would not be able to decode information fast enough to receive the 30 frames/s allowed by most modern smartphone cameras. As a matter of fact, all the previous LED-to-camera solutions [68, 66, 48] rely on computer vision (CV) filters to detect the ROIs, which results in a frame processing time much higher than 30 ms (the time required to process 30 frames per second).

Instead, our method is specifically designed for small colored LEDs, allowing to process ROIs as small as 160 pixels and integrating a real-time decoding algorithm, able to determine the ROI position in each frame. The frame processing algorithm is decomposed in five steps:

Step 1. Image acquisition (S1): handling incoming captures in the YUV color space and extracting only the luma component Y, which allows to distinguish between dark and illuminated stripes.

Step 2. ROI detection (S2): finding the LED position on the picture, excluding LEDs that are not transmitting and reducing the length of the Y component vector.

Step 3. Signal equalization (S3): enhancing the received signal to deal with ISI and background noise, which allows to recover the signal clock rate.

Step 4. Signal quantization (S4): thresholding and binarization of the signal to recover the digital signal.

Step 5. Decoding and error checking (S5): decapsulating the PHY-SDUs, decoding the Manchester chip and checking data integrity with vertical redundancy check (VRC).

Indeed, as already explained, we first produced an implementation following the state of the art guidelines. This first implementation took up to 165 ms processing time per picture, which is not an acceptable value, as it constrains the frame-rate to avoid the system overflow, forcing the Android scheduler to randomly kill threads or drop camera captures. Thus, we investigate possible improvements at different layers: pure algorithmic, software implementation on the Android Execution Runtime (ART), and the Android Camera2 framework. The following subsections details the five algorithm steps and the successive improvements we brought.

3.3.2 S1: Image Acquisition

Image format Since the release of Android Lollipop (API 21) and the new Camera2 API, developers are now able to control the smartphone CMOS sensor and the camera subsystem without any modification in the lower layer of the operating system. Numerous parameters can be chosen that can affect the efficiency and the amount of processing performed by the Android system.

The most relevant of these parameters are the image size, format and its compression ratio. We tried different image sizes, from 800x600 up to 3264x2448 pixels. Augmenting the resolution increases the rolling shutter stripes length,

and so, reduce the ISI error [28]. However, this also increases the matrix size to process. To deal with this, we choose an intermediary value of 1920 pixels width per 1080 height: this is about 2.5 smaller than the largest possible choice, and provides sufficient precision regarding the symbol length.

Also, choosing the right color space makes the processing easier and affects the smartphone computing load. If a JPEG encoded RGB image representation has a lower memory footprint, it also requires more post-processing, due to color conversion and compression, which affects frame rate and the global system CPU and GPU load.

In contrast, RAW format contains the sensor value without any post-processing but is not exploitable as is, due to its size. To take advantage of this representation, we choose the `YUV_420_888` uncompressed format as suggested in [36]. YUV representation has three channels, the luma (Y) and the chroma (U and Y plane contains the luminosity level, while U and V represent the color information). As we modulate the LED light, only the Y plane is relevant for us, reducing the size of the required data.

Capture workflow As soon as a new frame is available, the Android application UI thread invokes the `onCaptureCompleted` callback. However, if the UI Thread is overloaded or busy, the newer frame, and as consequence data, can be lost. That means we should take care to keep the CPU and memory offloaded. To face frame loss, we can allocate an Image buffer to queue a limited number of frames upstream the callback. We follow the API recommendation setting this buffer to five.

An `Image` object in Android contains a `ByteBuffer` for each plane, and metadata, such as the capture timestamp. Accessing and manipulating big sized Java objects can be memory costly and not efficient, particularly when they contain huge buffers, most of them useless. This is typically our case, as U and V planes will be ignored. As soon as we acquire a new image, we allocate a new bytes array with the same size of the Y plane `ByteBuffer`, and fill it using the `Java NIO ByteBuffer.get` method, which generates optimized native code after Ahead-Of-Time (AOT) code compilation. We prefer moving to primitive type bytes array because read operations are notably faster.

Once this processing is done, we call the `Image.close` method to quickly release and free the memory.

Finally, we create a new Java `Thread`, passing the buffer and frame metadata to its constructor, and push it into the `ThreadPoolExecutor` set up at the application initialization.

Background processing on Android can be achieved in several ways: using Java `Thread` class implementing the `Runnable` interface, or the Android way using `AsyncTask`. Even if `AsyncTask` is generally preferred, we chose the Java Thread way. In fact, by implementing both solutions, our experiments showed that the processing is 8% slower, and with a larger distribution, when using AsyncTask (Fig. 3.5).

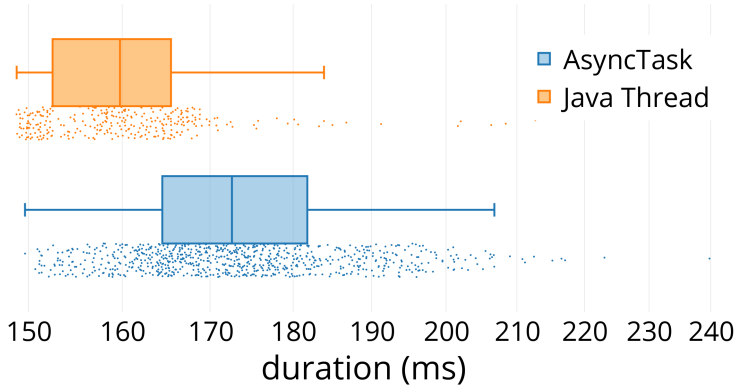

Figure 3.5: Total process duration distribution comparing AsyncTask and Thread

S2: ROI Detection

In order to detail S2 and S3, which represent advancements with respect to the state of the art, let us first introduce a series of notations, summarized in Fig. 3.6. Remember that the smartphone camera is configured to record frames of $W = 1080$ per $H = 1920$ pixels in the YUV color space.

The transmitted information is embedded only in the luma component Y, represented by a matrix I, where $I(x, y)$ is an 8-bit integer coding the pixel light intensity. Matrix I can also be represented using row and column vectors r_x and c_y, i.e. $I(x, y) = r_x(y) = c_y(x)$.

An LED captured by the camera will result in a ROI R_i in the frame. Multiple ROIs can be present in a picture frame, but some of these ROIs might be produced by regular LEDs, not carrying any information, while others, under a certain size, are too small to carry any useful information (e.g. R_2 in Fig. 3.6). As further explained in Chap. 2, rolling shutter cameras sequentially expose their

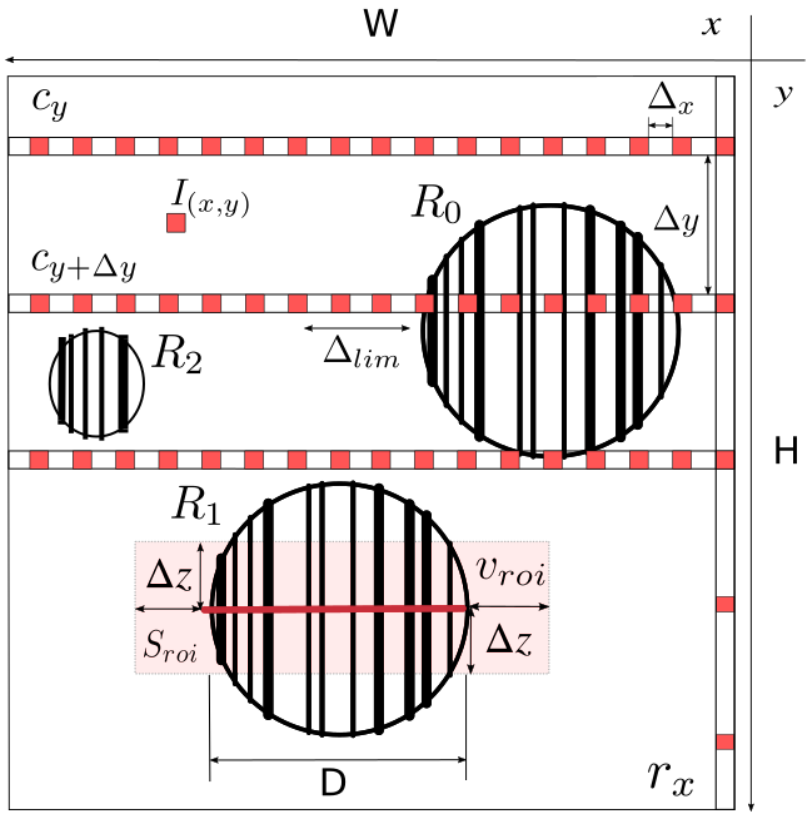

Figure 3.6: A picture frame containing three ROIs. The smaller ROI on the left will be skipped by the decoding algorithm, as it does not contain any valid PHY-SDU.

rows of pixels with a time duration δ_r. In this way, c_y is a record of the light intensity signal during a time interval of $W \cdot \delta_r$, where each OOK symbol has a width of S pixels (depending on the transmission symbol rate). Therefore, if the number of symbols included in a PHY-SDU is N_s, only ROIs with a diameter $D > S \cdot N_s$ should be considered for decoding, since smaller ROIs can not embed a full PHY-SDU.

In order to detect the beginning of a transmission, a part of the N_s symbols of a PHY-SDU are used to encode a synchronization sequence known by the receiver, which we denote as SYNC (see Fig. 3.3). This results in a SYNC sequence

Algorithm 1 Find the ROI in $Y_{W,H}$

```
V_roi ← a vector that will contain the ROIs coordinates (x, y)
I_cur ← 0
I_max ← max(I)
x, y, c ← 0
while x < W do
    while y < H do
        I_cur ← I(x, y)
        if I_cur/I_max ≥ γ then
            c ← y
            while (c − x) < Δ_lim do
                if SYNC detected then
                    V_roi[i] ← (c, y)
                    Process ROI
                end if
                c ← c + 1
            end while
        end if
        y ← y + Δy
    end while
    x ← x + Δx
end while
```

encoded over a set of Δ_{lim} pixels. The ROI detection algorithm consists in looking for a SYNC symbol in I by processing only the relevant pixels in order to keep the process duration as low as possible.

Let us denote by I_{max} the pixel with the highest luma intensity. The ROI detection algorithm starts scanning a row r_x, looking for the brightest pixels in I. However, to speed up the process, only $1/\Delta y$ pixels in r_x are considered, i.e. the red squares in Fig 3.6, with $\Delta y < D/2$ in order not to miss any meaningful ROI. More precisely, the algorithm checks whether $r_x(y) < \gamma \cdot I_{max}$, where γ is a predefined decoding threshold. Whenever this condition is true, the algorithm starts scanning the column c_y for Δ_{lim} pixels, looking for a sequence of symbols that matches the predefined SYNC pattern. In the same way as for the row scan, only $1/\Delta x$ of the Δ_{lim} pixels are considered, where $\Delta x < S$ ensures that each symbol is detected. If a SYNC pattern is found, the position is marked as ROI start. We note that a ROI will contain as many SYNC sequences as PHY-SDUs, which allows us to estimate the end of the ROI as well. Finally, by adding a margin Δz to the detected ROI, we extract the signal v_{roi}, the red line in Fig. 3.6, to be further processed in S3.

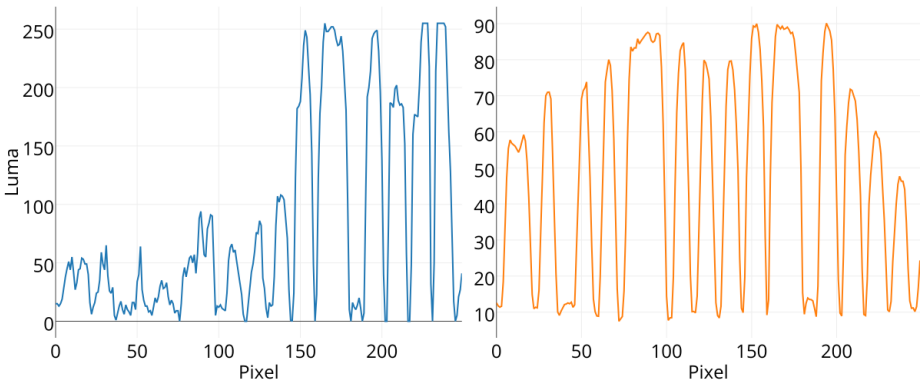

Figure 3.7: Comparison of the signal before S3 (left) and after S3 (right).

The ROI detection algorithm is summarized in Algorithm 1.

3.3.3 S3: Signal Equalization

Because of the reduced LED luminance, corroborated with the blooming effect and possibly low SNR conditions, the v_{roi} signal is difficult to quantize directly. An example is depicted on the left side of Fig. 3.7, where a sequence of falling and rising fronts can be noticed in the signal, but the luma intensity highly varies throughout v_{roi}.

To equalize the signal and prepare it for decoding, we use an averaging technique with a window Δz. More precisely, we compute the output of S3, v'_{roi} on the right side of Fig. 3.7, as:

$$v'_{roi}(x) = \frac{1}{2\Delta z} \sum_{i=y-\Delta z}^{y+\Delta z} I(x,i)$$

For performance consideration, we put these values into an integer array. In a first time, we considered using a new byte array, to reduce the memory impact and keep 8 bits representation. However, using 32 bits integers proved to be faster. The explanation is that the Nexus 5 processor has a 32-bit architecture, so the Android Runtime (ART) and its AOT does the operations using 32-bit instructions in each case. Hence, in the byte case, it executes extra instructions to convert the intermediate 32-bit value to a byte in each loop iteration. This makes performance worse in our arithmetic and compute intensive algorithm.

The cost of this operation depends on the ROI width (the column length), and averaging a lot of values can be useless or inefficient. So, after some exper-

iments, we decide to limit it to 50 points, which is enough to obtain a proper signal without impacting the system performance. The result of this step is shown in Fig. 3.7.

Since S, the duration of a symbol, slightly varies depending on the CMOS sensor [28], our algorithm first recovers the symbol rate SR by scaling the SYNC falling and rising fronts in $v'_{roi}(x)$. This steps is crucial to make the algorithm robust against the smartphone camera diversity and it is a requirement for a commercial implementation.

3.3.4 S4: Signal Quantization and Clock recovery

We next determine the threshold level by computing the mean value of the array defined in Subsec. 3.3.3, to binarize it. While we run through the array during the quantization, we count how many successive 0 or 1 pixel will follow, in order to convert them to bit symbols.

Then, we continue by dividing this counter by the number of consecutive pixels per symbol we expect, which depends on the value of SR that we have computed just before.

To avoid potential decoding errors and face the inconstant symbol size, we add an error correction mechanism based on the Manchester RLL code and the PHY-SDU format expected.

3.3.5 S5: Manchester decoding and parity check

Since we now have a short character string of bit symbols, we can take advantage of Java String API, to easily decode symbols to data information.

Then, we compute the parity checksum and compare it with the received one. Finally, we convert Manchester symbols to data symbols and return the result to the Android activity.

3.3.6 Algorithm Experimental Benchmark

Fig. 3.8 summarizes the average execution time of the processing thread functions on the Nexus 5 smartphone, with a Java implementation.

Fig. 3.8 shows that our algorithm can recover the data information in 18.4 ms on average, in various illumination conditions. Moreover, when we implement the same algorithm in a C shared library, which is then called from our Android

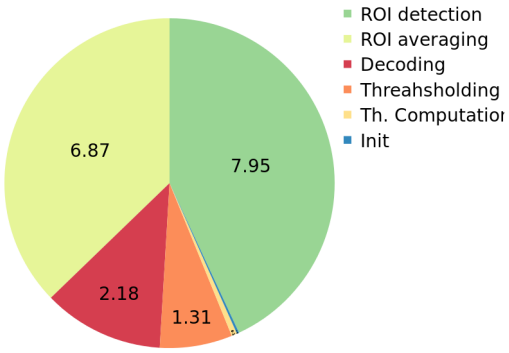

Figure 3.8: Steps duration repartition of the optimized algorithm.

application, the whole processing takes only 3.6 ms. These results outperform previous works, especially when considering that, unlike the state of the art solutions, we compute the LED position in each frame (about 30 times per second), adding robustness against user motion.

Multiple LEDs compatibility

Our first implementation, noted *no-MIMO*, assumes that the only light source in a picture is the LED used for data transmission. However, the algorithm could be easily broken if any additional LED that is not a VLC emitter, or any other intense light source like a reflective surface, appears in the same picture. Indeed, this is often the case on consumer electronics.

To solve this concern, we improve our solution to offer two additional features: first to discern emitting from non-emitting LEDs and then, to decode several LEDs embedded in the same picture. We note this version of our algorithm, at can smartly take advantage of several emitters, *MIMO*. That makes it less sensitive to noise, robust against non-emitting LEDs and enables N-LEDs-to-camera communication.

For that purpose, unlike the *no-MIMO* implementation, the *MIMO* algorithm does not stop when a ROI is detected, but continues looking for others SYNC symbols in the picture, before trying to decode each one. Also, I_{max} is not longer constant but continuously updated and smoothed using a moving average filter.

It adds a non-negligible computational overhead, but the processing delay still acceptable. That is shown in the left plot in Fig. 3.9: the computation time becomes only 20 ms for two emitters and 26 ms for four.

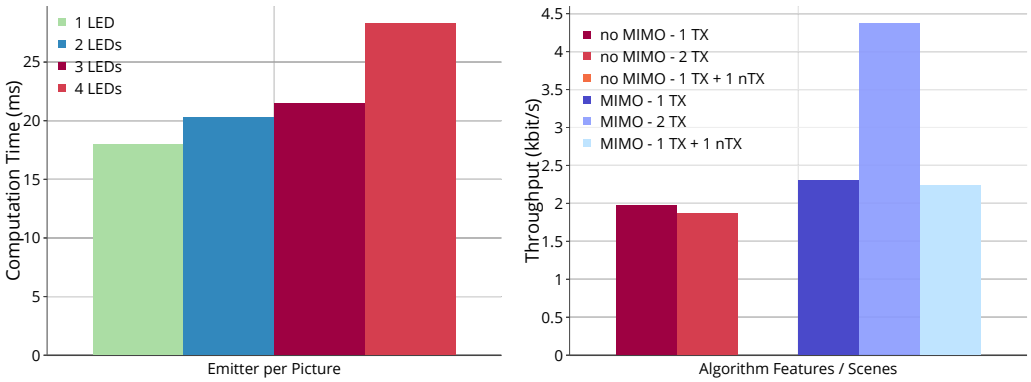

Figure 3.9: On the left, the algorithm duration for one to four emitters. On the right, the comparison of the algorithm efficiency (throughput) over different scenes.

To test the accuracy and the gain of these two additional features, we place two LEDs spaced 5cm apart and record the throughput in three different scenes using the *no-MIMO* or the *MIMO* algorithms.

In the first scene, only one LED is transmitting data (1 TX). In the second one, we leave the first LED transmitting data and lighten the second LED without VLC modulation (1 TX + 1nTX). For the last scene, we use the two LEDs as VLC emitter (2 TX).

The right plot in Fig. 3.9 compare the throughput between the *no-MIMO* (the red bars) and *MIMO* (the blue bars) algorithms, for the three different scenes introduced above.

That puts in evidence that the first method works well with just one transmitter on the image, but it completely fails, i.e. the throughput becomes 0 kbit/s when we add an other LED on the scene. On the contrary, the second version (*MIMO*) is efficient in every cases and smartly takes advantage of the two emitters to nearly double the throughput.

3.4 Experimental Results

This section presents a thorough evaluation of the LED-to-camera communication system described above.

We first evaluate the system performance as a function of the distance between the LED and the smartphone. We set the emitter symbol rate to 8 KHz and place it in standard indoor illumination conditions, that is about 650 lux. We make the distance vary from zero to the farthest one where we can still detect a

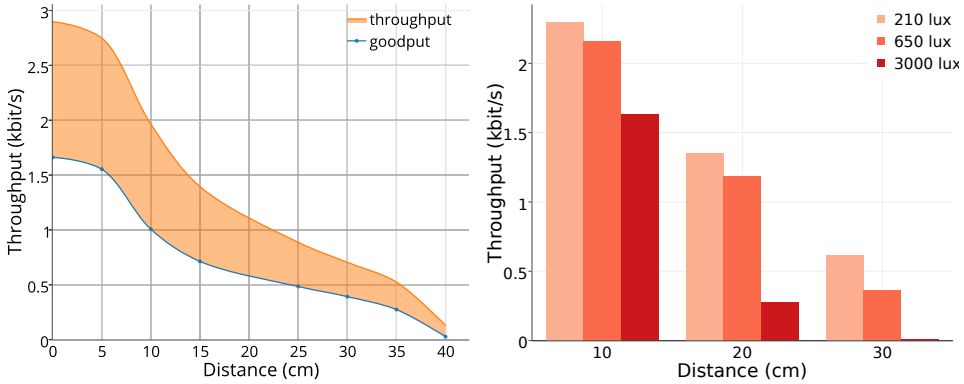

Figure 3.10: The throughput as a function of distance, compared with the goodput (left) and for various illumination conditions (right).

signal, without using a zoom or an additional lens. To avoid any external noise and ensure constant experimental conditions, we control the scene illumination with a light meter, and fix the smartphone position. We repeat each measure on different days, at the same place and same conditions, to record enough data to consider them sterling.

The emitter broadcasts continuously 50 different PHY-SDU built according to Subsec. 3.2.1.

We compute the throughput as the number of received information bits per second, error free, including duplicated PHY-SDU. The goodput is calculated removing duplicates until the smartphone receives all the 50 PHY-SDU.

3.4.1 Environment Impact

In this subsection, we propose an evaluation of the environment impact on the performance of LED-to-camera communication using small color LED. We study the effect of the ambient light and of the angle between the camera and the transmitting LED. We first evaluate the LED-to-camera communication performance as a function of the distance between the LED and the smartphone.

The left curve in Fig. 3.10 shows the goodput achieved by the system. This is around 1500 bit/s at 5 cm, but is divided by two at 15 cm and smoothly decreases with the distance up to 40 cm. On the other hand, the throughput, obtained before removing duplicate PHY-SDUs, is close to 2750 bit/s at 5 cm, corresponding to a duplicate PHY-SDUs ratio of about 45%.

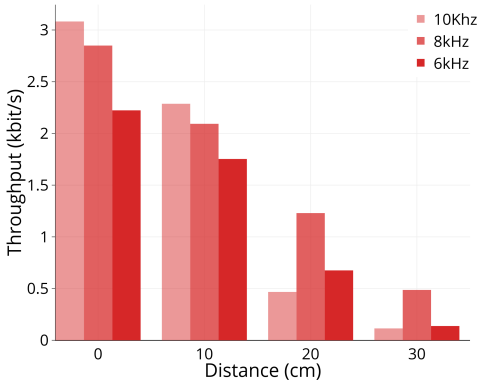

Figure 3.11: The throughput as a function of distance, for various OOK symbol rates.

While the performance of the system is clearly limited, we believe it satisfies the envisioned scenario, allowing any device to transmit information through VLC to a smartphone nearby.

3.4.2 Noise Impact

The interference of the ambient light, coming from the indoor ceiling lights or the sun, is a major issue in VLC, and even more so when using low power LEDs that are not intended for lighting purposes. In rolling shutter camera communication, this interference reflects in smaller ROIs and in a reduced difference between dark and bright stripes on the picture, which makes the decoding more error-prone. To test the noise impact, we evaluate the performance of our system in different illumination conditions, indoor and outdoor: *i)* in a room, normally lighted but not exposed to the sun through a window (210 lux); *ii)* on a laboratory work table, near a window and illuminated with neon lights (650 lux); *iii)* outdoor, during a very cloudy day (3000 lux), and *iv)* outdoor, during a sunny day (40000 lux).

In 40000 lux conditions, the signal is totally lost, even when the receiver and emitter are very close. Results for the other cases are depicted on the right side of Fig. 3.10, showing that the system is robust against interference from indoor lighting. Sunlight on the other hand has a stronger impact, but the results show that the throughput stays correct, only 25% less than in a room at 10 cm, if we avoid the direct sunlight.

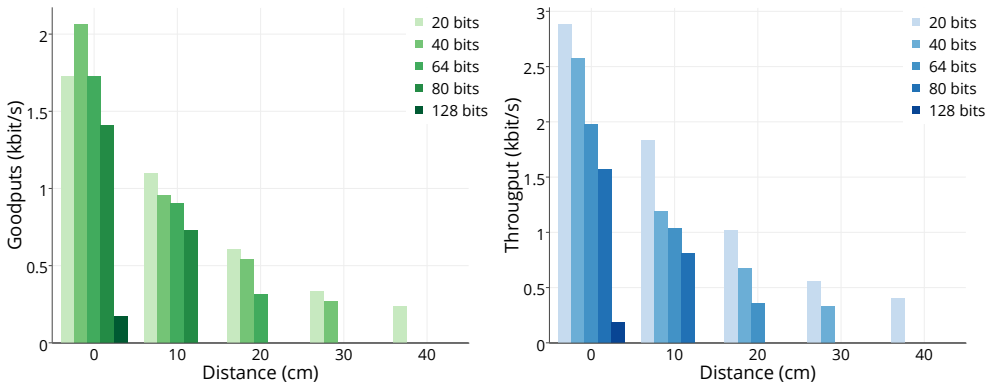

Figure 3.12: The goodput (left) an throughput (right) for different PHY-SDU sizes.

3.4.3 Symbol Rate Impact

The symbol rate of the OOK modulation used by the emitter has a direct impact on the amount of information embedded in each picture taken by the camera. Indeed, for higher transmission frequencies (i.e. for higher data rates), the width of the dark and bright stripes on the picture decreases. Similar to what happens in RF communications for more complex modulations, the bit error probability increases when the width of the stripe decreases. After testing three different symbol rates (6 kHz, 8 kHz, and 10 kHz), the best results, shown in Fig.3.11 are obtained for a symbol rate of 8 KHz, which we use in the rest of this work.

3.4.4 PHY-SDU Size Impact

Finally, an important property of the LED-to-camera communication is the fact that information losses are unavoidable and almost regular. Indeed, even when functioning at the maximum frame rate of 30 frames per second, the pictures taken by the smartphone do not represent a continuous view of the LED, as between any two pictures there is a camera inactivity time. To cope with these losses, the data needs to be segmented into PHY-SDUs, which can be easily reassembled and retransmitted.

However, the size of the PHY-SDU has a significant impact on the PHY-SDU loss probability. This is shown in Fig. 3.12, for a PHY-SDU size varying between 20 bits and 128 bits. We notice that the PHY-SDU loss ratio becomes unmanageable as the PHY-SDU size increases, e.g. almost all the 128-bits PHY-SDUs are lost, even at a few centimeters from the LED. Even for small 20-bits PHY-SDUs, the loss ratio is important (around 25% close to the transmitter); this is the con-

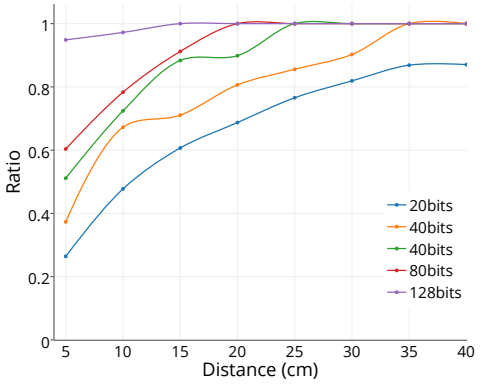

Figure 3.13: The PHY-SDU loss ratio as a function of distance, for different PHY-SDU sizes.

sequence of the PHY-SDUs that are transmitted between the camera captures, which do not fully appear in any picture. Overall, the results in Fig. 3.12 show a clear advantage for small PHY-SDUs of 20-40 bits.

3.4.5 User and Angle Impact

During our experiments, we noticed the user can have a significant impact on the system performance. This can be observed by comparing the results obtained when the smartphone was held by the user and those obtained when using a fixed support. As Fig. 3.14 shows, the average throughput is the same in both cases, but a wider distribution can be noticed for a handheld device. To understand the factor that causes this phenomenon, we make the LED direction and its angle with the camera change from -10° to +10°, when at 0° they are perfectly aligned with both plans.

Fig. 3.14 brings out the sharp dropping of the throughput with the variation of the angle. In fact, due to the directivity of the LED used (C503B-GAN), which is 30°, the angle has a huge influence on the ROI size on the picture. This can be avoided by choosing a LED with a large angle of view, but the side effect is that the light reflection on the camera lens will be weaker.

3.4.6 LED Model Impact

Finally, we studied the impact of the LED model and the LED color on the communication performance. We test eight LED with different characteristics,as summarized in Tab. 3.1. The LEDs #1, #2 and #3 are equivalent models for which only the color changes. Note that they have a clear transparent lens. The LED

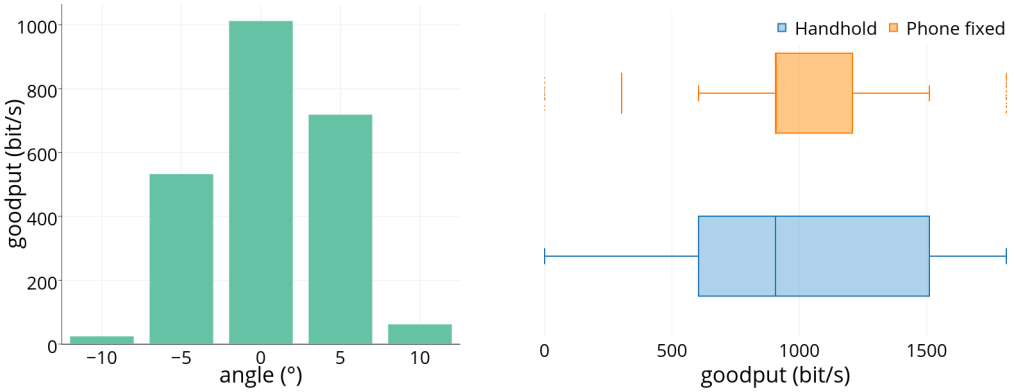

Figure 3.14: On the left, the impact of the angle between the LED and the camera at 10 cm. On the right, the throughput when he user is holding the smartphone compared to a mechanical support.

Table 3.1: Characteristics of different LEDs we used

#	Color	AOV	Intensity	Ref	Size	DC
1	R (630nm)	30°	5.1cd	C503B-RCN-CW0Z0AA1	5mm	20mA
2	G (535nm)	30°	34cd	C503B-GAN-CB0F0791	5mm	20mA
3	B (480nm)	30°	4.1cd	C503B-BCN-CV0Z0461	5mm	20mA
4	G (584nm)	25°	0.64cd	TLLG440	3mm	7mA
5	G (530nm)	115°	1.6cd	ASMB-MTB1-0A3A2	3mm	25mA
6	R (640nm)	60°	1cd	L-154A4SUREQBFZGEW	5mm	30mA
7	G (520nm)	60°	1.7cd	L-154A4SUREQBFZGEW	5mm	30mA
8	B (461nm)	60°	0.5cd	L-154A4SUREQBFZGEW	5mm	30mA

#2 is the one we use in our testbed above. For the LEDs #6, #7 and #8, the same RGB LED is used to produce red, green and blue light respectively. Unlike the LEDs #1, #2, and #3, the lens is white and diffused. Also, LED #5 is a surface mount device LED, while all the others are through-hole LEDs. Fig 3.15 shows the average number of PHY-SDU received per frame for the 8 LEDs depicted in Tab. 3.1. The LED and the smartphone are 5 cm far from each other. The LED #2 obtains the best performance, with up to 4 PHY-SDU received in a frame. Also, the green LEDs, i.e. LED #2 and #7, outperform the same LEDs in other colors. That is a consequence of the Bayer RGB color filter used in the smartphone camera CMOS sensor. The Bayer filter array has alternating rows of red-green and green-blue filters. Since the human eye is more sensitive to green than to the other two colors, the Bayer array has twice as many green

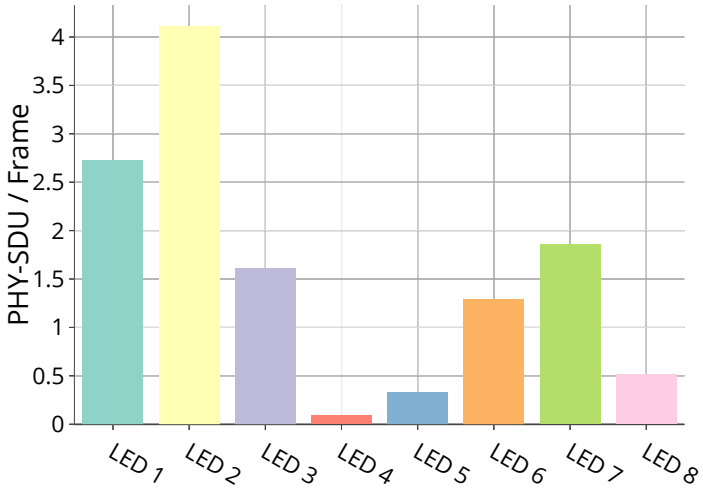

Figure 3.15: The average number of PHY-SDU received per camera frame for several LED models.

color filters [10]. Putting these result alongside the LED characteristics, we can see that the performance is directly linked to the LED intensity in candela.

3.5 Lessons Learned

Several observations can be made following this experimental campaign. First of all, although the throughput of our system, in the order of a few kb/s, is obviously limited by the low quality receiver and transmitter, this is still better than the performance of competing solutions, e.g. *RollingLight* [68] obtains a throughput in the order of 100 bit/s.

A second observation consists in the small PHY-SDU size used. Our results indicate that the PHY-SDU size must be in the order of tens of bits. Therefore, we propose the PHY-SDU format introduced in Sec. 3.2.1 and depicted in Fig. 3.3.

Because of the significant PHY-SDU losses in the system, a sequence number is still needed, in order to recompose the correct message sequence and allow retransmissions. While representing a significant overhead, this sequence number is essential for the retransmission strategies, further discussed in Chap. 4.

This small PHY-SDU size implies that standardized message formats with a size of hundreds of bytes, such as those proposed by the IEEE 802.15.7 or 802.15.7m standards, cannot be applied for small LED-to-camera communication. The same is true for previous works using lighting LED bulbs [36], which

propose a message format that includes a header, sequence numbers and tail bits for error detection. However, lighting LEDs installed on the ceiling, and even more in a non-line of sight configuration, produce a larger ROI on the picture, allowing the design of these larger messages, which is not the case for our small LED, as shown above.

This reduced payload is problematic for the networking layer, where packets usually add their own header, containing addressing information and a sequence number for reassembly purposes. However, with 20 available bits, the required control overhead would be too important, especially as some of this information is not always needed. This is the case, for example, for addressing: the communication range in our system is very limited, in terms of both distance and directivity. Therefore, a receiver does not necessarily needs MAC or network layer identifiers to distinguish the different transmitters; an application layer identifier is sufficient for this purpose.

However, the data fragmentation issue is still open, that is why, in the following Chap. 4, we propose analytical and simulation models to design and evaluate the performance of several redundancy mechanisms.

Chapter 4

Performance Evaluation of LED to Camera Communication

The use of LED-to-camera communication opens the door to a wide range of use cases and applications, with diverse requirements in terms of quality of service. However, while analytical models and simulation tools exist for all the major RF technologies, the only way of currently evaluating the performance of a network mechanism over LED-to-camera is to implement and test it. Our work described in this chapter aims to fill this gap by proposing models and tools that help in the assessment of LED-to-camera communication network mechanisms. Our contributions are threefold:

1. We propose an analytical model for LED-to-camera communication systems in Sec. 4.1.
2. We design and implement a LED-to-camera communication simulator in Sec. 4.3.
3. The model and the simulator allow us to benchmark several redundancy mechanisms proposed in the literature. By checking against experimental results, we are able to confirm the correctness of our models in this context.

4.1 Modeling LED-to-camera communication

In this section, we describe, for the first time in the literature from our knowledge, a LED-to-camera communication channel model. Based on the theory of

Markov-modulated Bernoulli processes (MMBP) [93], discussed in Sec. 4.1.1, this model can be applied to all LED-to-camera communication systems, not only to the particular case of ours. The proposed model is not only generic, but also very accurate, as demonstrated by its validation with extensive experimental results in Sec. 4.1.4. As an example, we use the analytical model to compare two simple redundancy mechanisms, required to cope with the inherent losses of LED-to-camera communication and described in Sec. 4.1.2 and Sec. 4.1.3.

4.1.1 Model design

In a LED-to-camera system, data is received as a series of dark and illuminated stripes in a picture frame captured by the camera. In the following, we note by f_i the i-th frame captured by the camera and by δ_f the time between the beginning of two consecutive frames. Obviously, even at the highest frame rate allowed by the camera, data is not continuously received, as a minimum time δ_g exists between two frames. This is denoted as the inter-frame gap (IFG). Moreover, as depicted in Fig. 4.1, the distance between the LED and the camera also has an impact: when the camera is farther away, the LED transmission is captured for a shorter time, resulting in a smaller ROI.

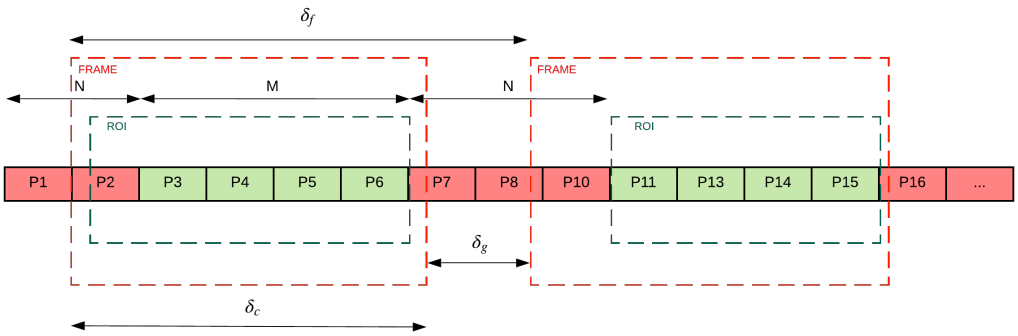

Figure 4.1: Frame capture time and inter-frame interval, and their relation with the MMBP parameters.

Gilbert-Elliot model: The Gilbert-Elliot model is widely used to model bursty losses [32]. This unique type of channel can intuitively be modeled by the two states Markov chain depicted in Fig 4.2, where the states S_1 and S_2, correspond to a packet reception time slot.

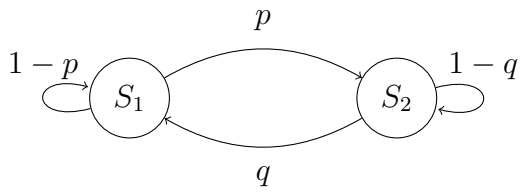

Figure 4.2: The *Gilbert-Elliott* model of the LED-to-camera channel.

In the S_1 state, the system is capturing a frame. The camera is receiving packets, and the reception probability is $1 - p_e$ where p_e is the packet decoding error probability. In the S_2 state, the camera is not capturing any pictures, therefore we consider the reception probability is 0. The transition probability between S_1 and S_2 (respectively S_2 and S_1) is denoted p (respectively q). We assume that p, q, p_e are independent and constant.

The probability of being in state S_1 under the steady-state regime can be easily computed as $p_{s_1} = \frac{p}{p+q}$. The probability of being in state S_2 is so $p_{s_2} = \frac{q}{p+q}$.

The values of p and q are function of the duration of the IFG, δ_g, the frame duration, δ_f, and the camera capture time δ_c, depicted in Fig. 4.1, and linked as the following:

$$p = 1 - q = \frac{\delta_f - \delta_g}{\delta_f} = \frac{\delta_c}{\delta_f} \tag{4.1}$$

MMBP model: If the model introduced just before is straightforward and widely used, it is not realist since, in our case, the transitions between ON and OFF states are almost deterministic. Practically, in our system, the transition probability from a state to another depends on the residence time in this state.

To improve this approach, we model the LED-to-camera channel using a Markov-modulated Bernoulli process (MMBP, represented in Fig. 4.3. In this figure, we depict a Markov chain with a total number of $M + N$ states. Each of these states represents a packet reception time slot, i.e. the time duration needed in order to receive one PHY-SDU (5.75 ms using the values discussed in Chap. 3). The transition between a state representing a given slot and the state representing the following slot is automatic, i.e. it happens with a probability 1.

Practically, the $M + N$ states in Fig. 4.3 represent a δ_f time interval, and they are divided in two groups: M states corresponding to the camera capture time δ_c (S_{ON} states), and N states corresponding to the inter-frame time δ_g (S_{OFF} states). A Bernoulli arrival process is associated with each of these $M + N$ states, representing the reception of a packet.

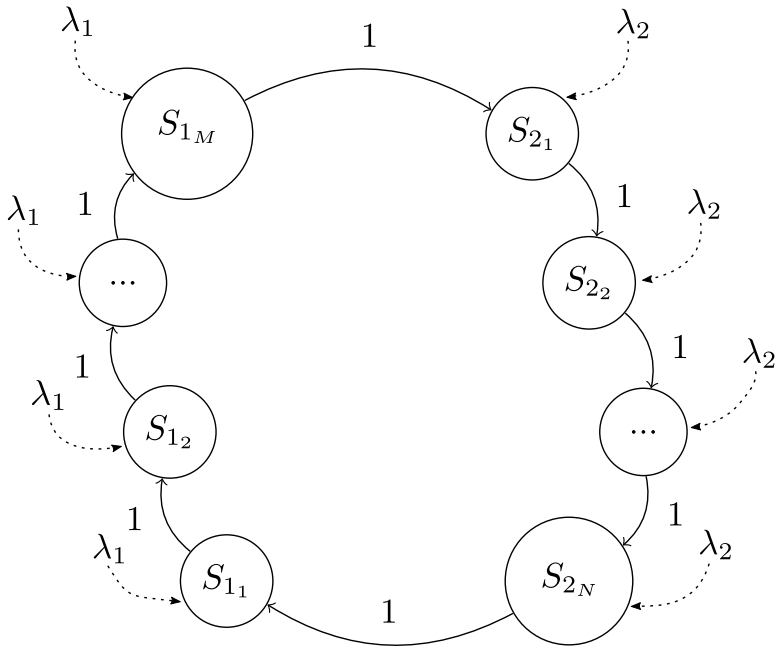

Figure 4.3: The MMBP model of the LED-to-camera channel.

In S_{ON} states, the camera is receiving packets, and the arrival rate is $\lambda_1 = (1 - p_e)$, where p_e is the packet decoding error probability. In S_{OFF} states, the camera is not capturing any pictures, therefore we consider the arrival rate $\lambda_2 = 0$.

We denote as s a state in the Markov chain and we define state $s + j$ as the state reached after j transitions, starting from state s. The probability of being in state s under the steady-state regime can be easily computed as $\pi_s = \frac{1}{N+M}$.

At the same time, the probability of noticing no arrivals (i.e. no packet reception) in state s is $p_0(s)$.

This can be written as:

$$p_0(s) = \begin{cases} 1, & if \quad s \in N \\ p_e, & if \quad s \in M \end{cases} \tag{4.2}$$

As it can be seen from both models, the relatively high packet loss probability (compared with RF technologies) is an intrinsic property of the LED-to-camera communication channel. To overcome this problem, redundancy mechanisms are needed.

In the following, we show that the classical Gilbert-Elliot model is inaccurate, which highlights the need to rely on the MMBP theory. Then, we use the MMBP

channel model to compare two simple, but widely used redundancy solutions: repeating a packet or repeating a sequence of packets.

4.1.2 Repeat Packet

The first strategy to cope with the inherent losses in the LED-to-camera communication system, used for example by *Ferrandiz-Lahuerta et al.* [36], is to send each packet twice in a row, to increase the probability that at least one of the transmissions will be fully captured by the smartphone camera. We generalize this approach in the Repeat Packet (RP) strategy, where each packet is repeated r times, one after the other. In this case, the r value needs to be chosen in order to attain a desired reception probability, its optimal value depending on the inter-frame time and on the packet size.

In the following, we study the probability of receiving a packet at least once when considering the RP strategy, for the two models introduced above.

Gilbert-Elliott Model: If we consider the Gilbert-Elliott model, the probability of receiving a packet at least once can be written as $p_s^{RP} = 1 - p_0^{RP}$, where p_0^{RP} represents the probability of failing to receive a packet r times in a row, written as:

$$p_0^{RP} = (P_{S_1} \cdot p_e + P_{S_2})^r = \left(\frac{p \cdot p_e + q}{q + p}\right)^r \tag{4.3}$$

MMBP Model: If we consider the MMBP model, the probability of failing to receive a packet r times in a row p_0^{RP} can be written as:

$$p_0^{RP} = \frac{\sum_s \left(p_0(s) \cdot p_0(s+1) \cdot p_0(s+2) \cdot \ldots \cdot p_0(s+r-1)\right)}{N+M} \tag{4.4}$$

The value of p_0^{RP} will depend on m_s, defined as the number of S_{ON} states during the r retransmissions, when the first packet transmission is in state s:

$$p_0^{RP} = \frac{1}{N+M} \cdot \sum_{s=1}^{M+N} (p_e)^{m_s} \tag{4.5}$$

Depending on the values of r, M and N, several cases can be distinguished. We present results for the two most current cases:

- If $r < M, N$: This means that the number of retransmissions does not always cover the δ_g period (which counts N states). In this case, p_0^{RP} can be written a follows.

$$\begin{aligned} p_0^{RP} &= P\left[s \in N \wedge (s+r) \in N\right] + P\left[s \in M \wedge (s+r) \in M\right] + P\left[s \in M \wedge (s+r) \in N\right) \\ &\quad + P\left[s \in N \wedge (s+r) \in M\right] \\ &= \frac{N-r+1}{M+N} + \frac{M-r+1}{M+N} \cdot {p_e}^r + \frac{1}{M+N}\sum_{i=1}^{r-1} {p_e}^i + \frac{1}{M+N}\sum_{i=1}^{r-1} {p_e}^{r-i} \\ &= \frac{N-r+1}{M+N} + \frac{M-r+1}{M+N} \cdot {p_e}^r + \frac{2}{M+N} \cdot \frac{p_e - {p_e}^r}{1-p_e} \end{aligned} \tag{4.6}$$

- If $N < r < M$: This is the most common case, where the number of repetitions is chosen to cover the entire inter-frame period. However, a reception is still not certain in this case, because of the decoding error p_e.

$$\begin{aligned} p_0^{RP} &= P\left[s \in M \wedge (s+r) \in M)\right] + P\left[s \in M \wedge (s+r) \in N\right] + P\left[s \in N \wedge (s+r) \in M\right] \\ &= \frac{M-r+1}{M+N} \cdot {p_e}^r + \frac{1}{M+N}\sum_{i=1}^{r-1} {p_e}^i + \frac{1}{M+N}\sum_{i=1}^{N} {p_e}^{r-i} \\ &= \frac{M-r+1}{M+N} \cdot {p_e}^r + \frac{1}{M+N} \cdot \frac{p_e - {p_e}^r}{1-p_e} + \frac{1}{M+N}\sum_{i=1}^{N} {p_e}^{r-i} \\ &= \frac{M-r+1}{M+N} \cdot {p_e}^r + \frac{1}{M+N} \cdot \frac{p_e - {p_e}^r}{1-p_e} + \frac{1}{M+N} \cdot \frac{{p_e}^{r-N}(p_e - {p_e}^N)}{1-p_e} \end{aligned} \tag{4.7}$$

We compare the analytical results given by the two aforesaid models to experimentation results in Fig. 4.4. This figure shows the success probability of receiving a message of $N_p = 50$ packets of data, as a function of the number of retransmissions r.

The very different results between the Gilbert-Elliott model and the experimentation confirms that the stochastic transition assumptions of this model are too unrealistic. On the other hand, MMBP approximates quite well the experimental behavior. This highlights the need for the more complex, but finer grained, MMBP model.

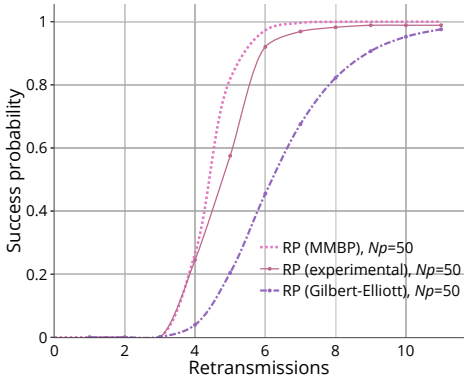

Figure 4.4: Probability to successfully receive N_p packets for the RP strategy. Dotted-lines show analytical results for the Gilbert-Elliott and MMBP models, while plain lines represent experimental results.

4.1.3 Repeat Sequence

A different approach to improve reliability is the Repeat Sequence (RS) strategy, consisting in the transmission of a sequence of N_p packets, repeated r times. In contrast with the previous mechanism, RS does not try to cover the inter frame time at the packet level, and it does not ensure that a packet is received before sending the next one. Instead, the reliability and presumed efficiency is based on the fact that the probability of losing the same packets over different transmitted sequences is low.

In the case of an RS strategy with a sequence of N_p packets retransmitted r times, the probability of receiving a packet at least once can be written as $p_s^{RS} = 1 - p_0^{RS}$. Using the MMBP model, the probability of failing to receive a packet r times in a row, p_0^{RS}, can be written as:

$$p_0^{RS} = \frac{1}{M+N} \cdot \sum_{s=1}^{M+N} \prod_{j=0}^{r-1} p_0(s + rN_p) \tag{4.8}$$

4.1.4 Evaluation results

We use our MMBP analytical model to study the RP and RS strategies by focusing on the probability of delivering the entire quantity of information in a given number of transmissions. We provide both analytical and experimental results, allowing us to validate the proposed MMBP model.

Fig. 4.5 shows, for the two mechanisms, the probability of integrally receiving N_p packets of data as a function of the number of retransmissions r. In this figure, we set $M = 5$ and $N = 2$; these values are in line with the packet length,

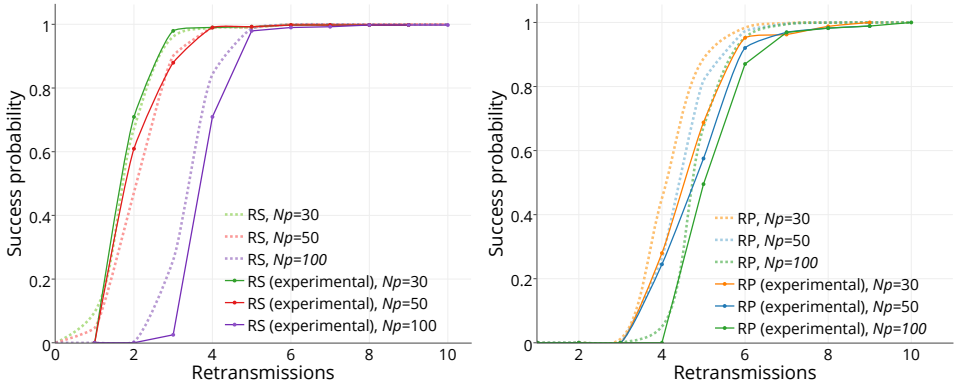

Figure 4.5: Probability to successfully receive N_p packets for the RS (left) and RP (right) strategies as a function of the number of retransmissions r. Dotted-lines show analytical results while plain lines represent experimental results.

the transmitter frequency and the camera capture interval discussed in Chap. 3, for a distance of 5 cm between LED and camera. The results show quite a nice fit between the analytical and experimentation results, despite the assumptions required by our MMBP model.

To better understand the performance of the two retransmission strategies, we compare them in Fig. 4.6. This figure shows that, for the RS strategy, 3 retransmissions are needed to achieve a reception probability higher than 0.9, while this value raises to 6 for the RP strategy. On the right side of the figure, we show that the performance of the two strategies depends on the ratio between the number of S_{ON} and S_{OFF} states, $M : N$. When this ratio changes from $5 : 2$ to $2 : 5$, which practically corresponds to increasing the distance between the LED and the camera, RP gives better results than RS. Indeed, for the RS method, the success probability sharply decrease when $M < 3$ and stays below 0.6 even for 10 retransmission.

Practically, this means that RP is more suitable when the distance between the LED and the camera is higher, while RS is better for short communication distances. This phenomenon was previously unknown in the research community, but it is straightforward to study with our analytical model

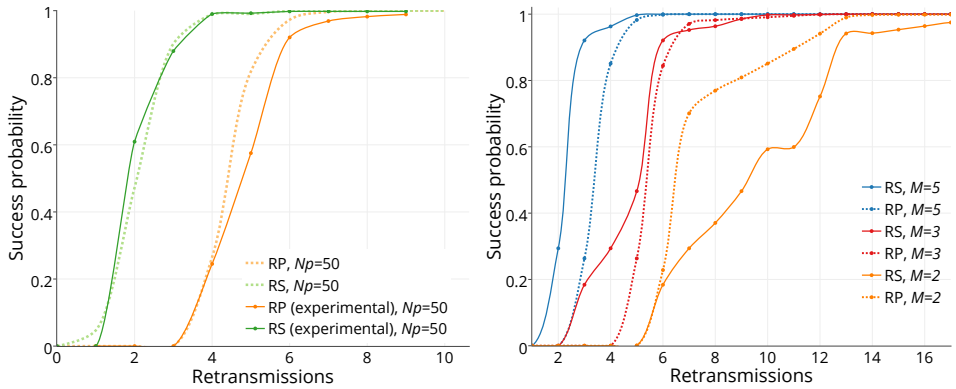

Figure 4.6: Comparison between RS and RP. On the left, analytical and experimental results for $M = 5$ and $N = 2$. On the right, analytical results when $M + N = 7$, but the $M : N$ ratio changes. In both cases, $N_p = 50$.

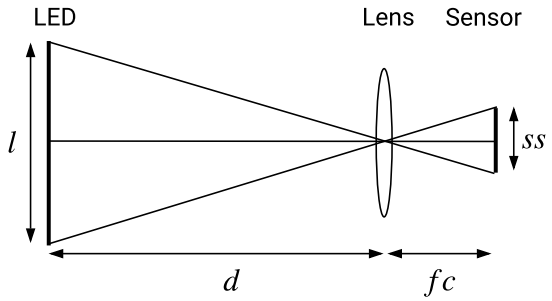

Figure 4.7: Formation of an image on a sensor by a converging lens.

4.2 ROI model

As shown in Chap 3, the distance between the LED and the camera reduces the ROI size and, as a consequence, cuts down the number of PHY-SDU that the camera can receive per frame, i.e. the M states in Fig. 4.1.

To include this performance factor into our model, we propose an analytical function that gives the ratio between the ROI and the picture size. In the model discussed in Subsec. 4.1.1, this is the ratio of M states in the $M + N$ states.

We apply photogrammetry rules to give the ROI ratio as a function of the distance d, the LED size l, the camera CMOS sensor size ss, the image size on the sensor i and the camera focal distance fc.

According to the optical system depicted in Fig 4.7, basic lens optic rules give the following Eq. 4.9.

$$\frac{i}{fc} = \frac{l}{d} \iff i = \frac{l \cdot fc}{d} \tag{4.9}$$

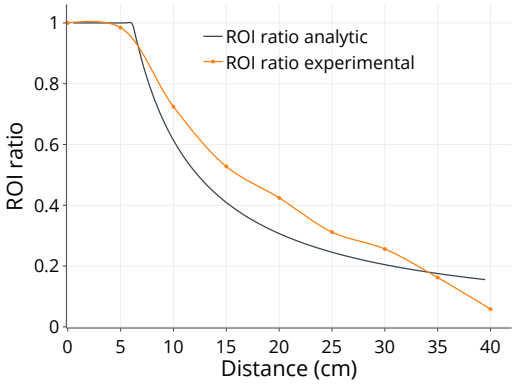

Figure 4.8: ROI as a function of distance. The orange line shows experimental results, while the green line represents analytical results given by Eq. 4.10.

We introduce the CMOS sensor size ss to obtain the ROI as the ratio of the total number of pixels in the picture. We apply the $\min$ function to have $ROI \in [0, 1]$ even if the image size on the sensor, i, is larger than the sensor size, ss. The result is given in the following Eq. (4.10).

$$ROI = \min\left(1, \frac{l \cdot fc}{d \cdot ss}\right) \tag{4.10}$$

To validate the results given by Eq. 4.10, we measure the ROI experimentally, using the testbed introduced in Sec. 3.2 for distances from 0 to 40 cm. Fig. 4.8 plots in orange the ROI ratio we observed during our experiments and in green the analytical results computed with the Nexus 5 sensor characteristics: $fc = 35$, $ss = 5.7$ and $l = 10$. This shows that the analytical curve approximates quite well the experimental ROI ratio. However, we notice that the experimental results are better for a distance between 10 and 30 cm, and they become worse than the model at 35 cm. In fact, the light radiance on the camera lens, that our model does not take into account, artificially increases the LED size on the picture when the camera is close to the LED. The difference at larger distance is a consequence of the ambient light which was measured at 650 lux during the experiments, also neglected in Eq. (4.10).

This ROI model and the MMBP reception model described in Sec. 4.2 are the basis of the simulator implementation discussed in the following section.

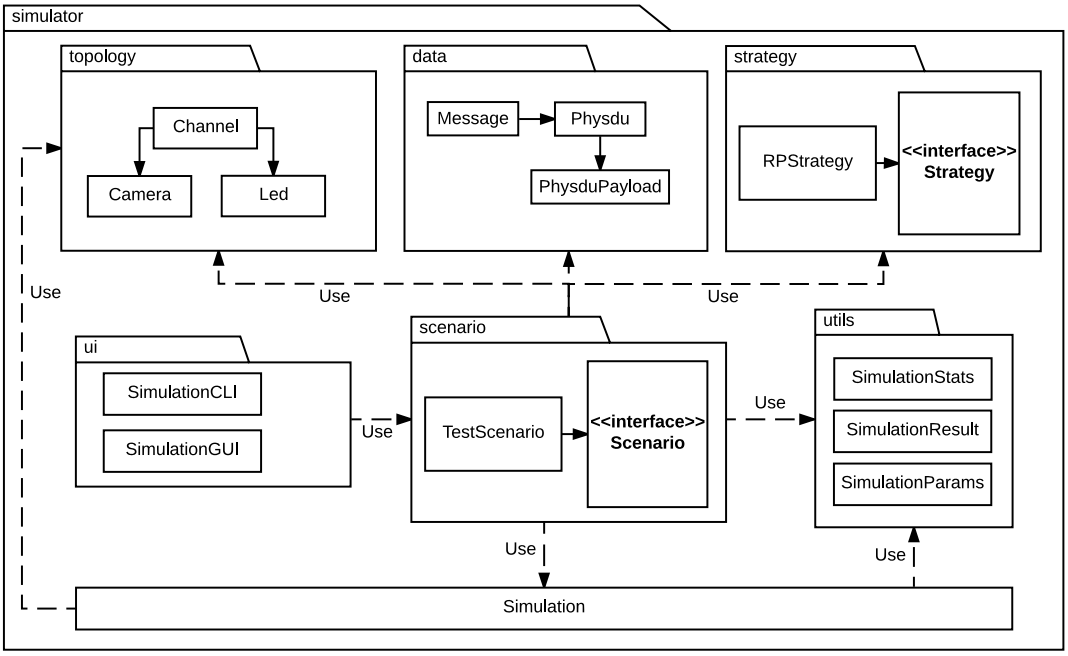

Figure 4.9: The *CamComSim* software architecture and packages dependency graph.

4.3 The CamComSim Simulator

If network simulation has been largely studied in the field of wireless communication [111], VLC simulation remains poorly investigated and VLC simulation tools are still missing.

The main efforts on simulating VLC systems have focused on the indoor channel simulation [115], or on the IEEE 802.15.7 PHY [6, 69] and MAC [86] layers. These approaches rely on classical network simulation frameworks such as the ones used in the context of wireless and ad hoc networks, e.g. NS-2 [86], NS-3 [6], OMNET++[69] or MATLAB [115].

Since all these works consider LED-to-Photodiode communication, LED-to-camera communication remains completely unexplored. Our work is the first such effort reported in the literature, making *CamComSim* the first implementation of a LED-to-camera VLC simulator.

4.3.1 Simulator Implementation

Software architecture

CamComSim is an event-driven LED-to-camera simulator we developed in Java. Java makes it easy to maintain and distribute code, and it provides built-in multi-platform compatibility for systems with a Java Virtual Machine. Fig. 4.9 shows the *CamComSim* software architecture that consists of a simulator kernel class and four core packages.

The `topology` package groups classes that describe the system components: `Led`, `Camera` and `Channel`.

The classes in the `data` package implement the data encapsulation `Message` is a set of PHY-SDU that encapsulates a `PhysduPayload`. A `Packet` is a `PhysduPayload` child class, with a sequence number as header and a payload that contains data. Before each simulation, a `Message` is created according to the user settings. The resulting set of `Physdu` is initialized with `Packet` filled with arbitrary data in the payload and a unique sequence number in the header.

The broadcast strategy abstraction is implemented in the `strategy` package. The `Strategy` interface lets the users implement their transmission strategy. This package also contains the straightforward `ReapeatPhysduStrategy` (RP) implementation that consists in repeating each PHY-SDU r times, one after the other. When the last PHY-SDU of the message is reached, the process is repeated from the beginning.

Finally the `scenario` package proposes an interface to build a simulation, wiring altogether the `Message`, the `Channel`, the `Led`, the `Camera` and the `Strategy` with the `Simulator` kernel.

Besides, the package `utility` provides helper classes used to compute the simulation results statistics, format and save the results as a JSON file and load or save the simulation parameters. The `ui` package contains a command-line interface (CLI) used to run a simulation scenario.

Simulator parameters

Our simulator exposes a set of finely grained parameters to describe the LED-to-camera communication system behavior. Table 4.1 shows the parameters we use in this work and expose through *CamComSim* CLI. As we discuss in Sec. 3.5, the performance of the LED-to-camera communication is significantly affected by the distance d, the IFG noted δ_g in Fig. 4.3, and the LED size l.

Param.	Description	Default Value
d	The distance between the camera and the LED (cm)	5
l	The LED size (mm)	4
d_g	The camera inter-frame gap (IFG) ratio	0.1
p_e	The decoder PHY-SDU Error Rate (PER)	0.001
f	The modulation frequency (Hz)	8000
P	The `PhysduPayload` header length (bit)	8
H	The `PhysduPayload` payload length (bit)	16
sc	The scenario implementation class name	`TestScenario`
r	The PHY-SDU repeat number	1
G	The message size (bytes)	50
X	The stop condition (number of PHY-SDU) or stop when received (R)	50000R
n_r	The number of simulation runs	300

Table 4.1: Simulator parameters.

Further parameters, introduced in Sec. 4.2, that refer to the CMOS sensor characteristics, are optional but can be considered to refine the channel model, as they impact the ROI. However, smartphone manufacturers rarely provide them. They are, for example, the sensor size ss and the focal distance fc.

The PER p_e is the consequence of the errors occurring in a M state when a PHY-SDU is well included in a picture but is wrongly decoded by the smartphone. These errors are bits substitutions induced by interference, low SNR, and artifacts on the picture.

The `PhysduPayload` payload size P and the `PhysduPayload` header size H configure the data encapsulation, as Fig. 3.3 shows. Given these two settings, the PHY-SDU size is computed as $P + H + SYNC + PB$, where $SYNC$ is the PHY-SDU delimiter symbol, which size is 4 bits, and PB consists of 2 parity bits. This PHY-SDU size, along with d, l, δ_f and the modulation frequency f, determines the number of the M time slots in Fig. 4.3.

Parameter sc lets us choose the transmission strategy among the `Strategy` interface implementations. We have implemented and considered only the RP strategy, for which the parameters are the size of the message to broadcast, G, the number of consecutive PHY-SDU emissions, r, and the simulation stop condition X.

Three possible stop conditions are defined: (1) a limit on the total number of PHY-SDU sent during the simulation, (2) stopping the simulation when all the PHY-SDU are received (R), and (3) one of the two conditions above. Finally, the CLI parameter n_r gives the number of simulation runs.

Kernel Implementation

The *CamComSim* kernel is implemented in the `Simulator` class. Its role is to produce PHY-SDU emission events (TX) and manage their result.

The number of events, i.e. the number of PHY-SDU sent, is noted c and is updated at runtime. At each clock tick, c is incremented, a TX event is created and processed as follow: (1) the next PHY-SDU in the transmission strategy queue is associated with this event; (2) considering p_e, f, P, H, c, the channel response function gives the event result. The result is one among *reception success*, *reception with errors* or *loss during IFG*; (3) this result is stored in a list to further determine if all the PHY-SDU that form the message are received.

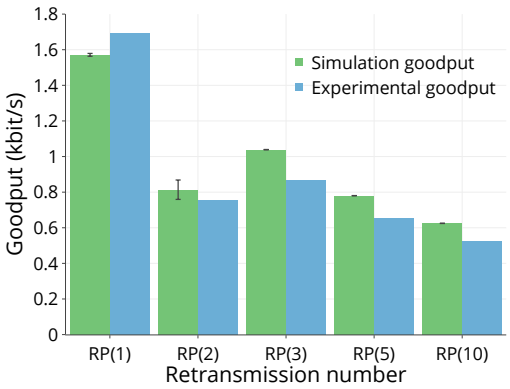

Figure 4.10: The experimental goodput (blue) compared with the simulation goodput (green) as a function of the number of consecutive PHY-SDU emissions. The bars on top are 95% confidence intervals.

The simulator loops over (1), (2) and (3) until the stop condition X happens, i.e. c has reached the maximum number of PHY-SDU emissions or the complete message is received.

The simulation is repeated n_r times using the Java `ThreadPoolExecutor` for multi-threading purpose. Finally, the simulations results and statistics are saved in a JSON file for further processing.

4.3.2 CamComSim validation

In this section, we present LED-to-camera simulation results given by *CamComSim*. To assess the correctness of our simulator, we conduct a series of experiments with the testbed introduced in Sec. 3.2. We set the emitter symbol rate to 8 kHz and place it in standard indoor illumination conditions, near a window and illuminated with neon lights. The illuminance has been measured with a luxmeter at around 650 lux. We compare the testbed performance with the results given by *CamComSim* for a set of key parameters: the message size G, the number of consecutive PHY-SDU emissions r, the distance d and the PHY-SDU payload length P.

PHY-SDU Retransmission

As discussed in Sec. 4.1.2, to face the IFG bits erasure and ensure that all the packets are well received, [36] proposes to transmit consecutively each PHY-SDU r times in a row. The authors proved experimentally that, for their system, $r = 2$

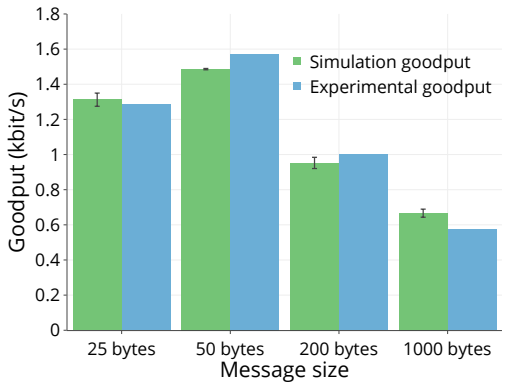

Figure 4.11: The experimental goodput (blue) compared with the simulation goodput (green) for different message size (G, bytes).

gives the best performance. We name this strategy RP and implement it in the `ReapeatPhysduStrategy` class.

Fig. 4.10 shows the goodput at 5 cm for different values of PHY-SDU consecutive retransmissions r, with $G = 50$, $P = 19$, $H = 5$. The stop condition X is set to $50000R$, meaning that the message transmission restarts when each PHY-SDU has been transmitted r times, until the message is received. To avoid infinite loops, we stop the simulation when 50000 PHY-SDU are sent even if the message is not received. In such case, the goodput is 0. The results highlight that the simulation and testbed goodput follow the same tendency when r varies. The best case is when $r = 1$ for which the goodput is 1.6 kbit/s according to *CamComSim* and 1.7 kbit/s for the testbed, an estimation error of only 6%.

Based on these results, note that all the simulations that follow use, unless mentioned otherwise, the RP strategy implementation with $r = 1$, which is equivalent to the RS strategy discussed in Sec. 4.1.3.

Message size

We now consider the impact of the message size G on the goodput at a 5 cm distance, with $r = 1$ and a 24 bits length PHY-SDU. Fig. 4.11 shows that *CamComSim* results are very close to those of the testbed, confirming that the simulator well considers the impact of the message size. We notice that the goodput reduces when the message size increases, as the RP strategy leads to a large number of useless transmissions: the simulator gives 1.6 kbit/s of goodput for $G = 50$ bytes, while this falls to 670 bit/s when $G = 1000$ bytes. These results differ from the testbed in no more than 7%.

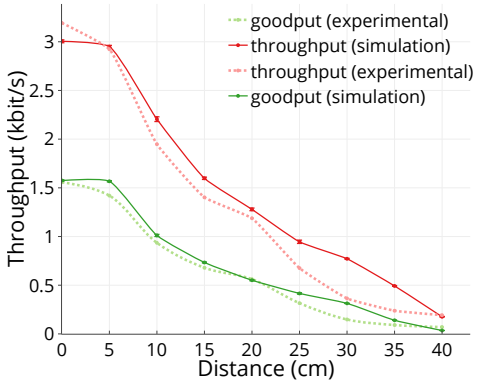

Figure 4.12: The throughput (red) as a function of the distance, compared with the goodput (green). Dotted-lines show experimental results while plain lines represent simulation results.

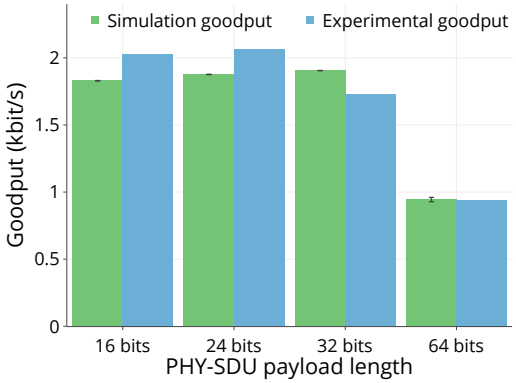

Figure 4.13: The experimental goodput (blue) compared with the simulation goodput (green) for different PHY-SDU payload size (bit).

Distance

Fig. 4.12 shows the goodput and the throughput as a function of the distance, when the LED broadcasts a 50 bytes message. The PHY-SDU payload is set to 24 bits, with $P = 19$ and $H = 5$. The results show a good match between the simulation and real life results. At 10 cm, *CamComSim* gives 2.2 kbit/s of throughput when this is 1.94 kbit/s experimentally. The results are closer for the goodput: 0.94 and 1.0 kbit/s respectively for simulation and experimentation, that is only 6% of difference.

PHY-SDU length

Fig. 4.13 shows the impact of the PHY-SDU payload size on the goodput at 5 cm, with $G = 50$. The packets are built using the best value for P and H, that is to say

with just enough bits in the header to label each packet with a unique sequence number.

Both experimental and simulation results show that the optimal PHY-SDU payload size is 24 bits: *CamComSim* gives a goodput of 1.9 kbit/s, while the testbed reaches 2.1 Kbit/s. As Sec 3.5 explains, large PHY-SDU reduce the encapsulation overhead, but they increase the probability that the IFG and a small ROI truncate the PHY-SDU.

Fig. 4.13 brings out that *CamComSim* well considers this behavior: the goodput becomes 0.9 kbit/s when the PHY-SDU payload size is 64 bits, very close to the experimental results.

This section highlights that *CamComSim* gives results very close to the testbed for all the parameters we have studied. The difference is around 10% and often less. For all the cases we consider, *CamComSim* respects the behavior of the LED-to-camera communication system implemented by the testbed.

4.3.3 Use case

In this section, we detail a case study for *CamComSim*, applied to a real life scenario. Then, we compare simulations with experimental results for the given application. These results will help us design and optimize the broadcast protocol used in the application.

Use case description

A common issue with cheap consumer electronics is the lack of diagnostics when a dysfunction happens. Manufacturers often blink the state LED with a pattern and color that match with an error code. Such mechanism is easy to implement but leads to inaccurate diagnostics. In such cases, we propose to benefit from this LED to perform LED-to-camera communication and broadcast a log file that would include helpful information to diagnose a dysfunction. We consider a worst case file size of 1 kbyte that is large enough for events history or debug traces.

Running the Simulation

As Sec. 4.3.1 explains, the simulation has to be described in a class implementing the *Scenario* interface. More precisely, the function `runScenario(SimulationParams params)` that takes as arguments the simulation settings must be implemented.

The implementation must perform the following operations:
(1) Set the transmission strategy to be used during the simulation. That is the RP strategy for this use case. If the simulation aims to evaluate another communication protocol, it must be defined in a class that implements the *Strategy* interface.
(2) Build the `Message` choosing a kind of `PhysduPayload`, for example a `Packet`.
(3) Create the `Led` and the `Camera` objects to build the `Channel` next.
(4) Initialize the *Simulator* kernel with the `Channel`, the `Message`, and the `Strategy`.

After implementing the simulation scenario in *CamComSim*, we run the simulator with the CLI through the following command: `java -jar camcomsim-1.0.jar -v=1 -i=100 -r=1 -P=14 -H=10 -d=5 -F=8000 -G=1000 -e=0.0001 -X=50000R -dg=0.1 -sc=TestScenario -output=path`.

Note that the parameter `-v=1` sets the verbosity level and `-output=path` the folder where the results will be saved.

Results

Fig. 4.14 compares the goodput given by *CamComSim* with the goodput that our testbed achieved for the transfer of a 1 kbyte log file as a function of the number of PHY-SDU retransmissions r. Note that this is equivalent to $G = 1000$ bytes in Fig. 4.11. The transmission restarts until the message is received. The left side plot shows the results when the LED and the smartphone are 5 cm apart, while the distance is 10 cm on the right side figure.

At 5 cm, the simulation brings out that, to obtain the higher goodput, the emitter should send each PHY-SDU one or three times consecutively, i.e. $r = 1$ or $r = 3$. The goodput is respectively 680 and 720 bit/s in these cases. This finding is the same on the testbed for which the goodput is 570 bit/s when $r = 1$ and 540 bit/s when $r = 3$.

Because the ROI decreases with the distance, the behavior is different when the smartphone is 10 cm far from the LED. In this situation, $r = 4$ stands out clearly to be the best choice both for the simulation and the experiments. The goodput then becomes 620 bit/s on the testbed and 540 bit/s with *CamComSim*.

Since the results are very close to the reality, using *CamComSim* highly reduces the search space for the experimental optimization of a system. As shown

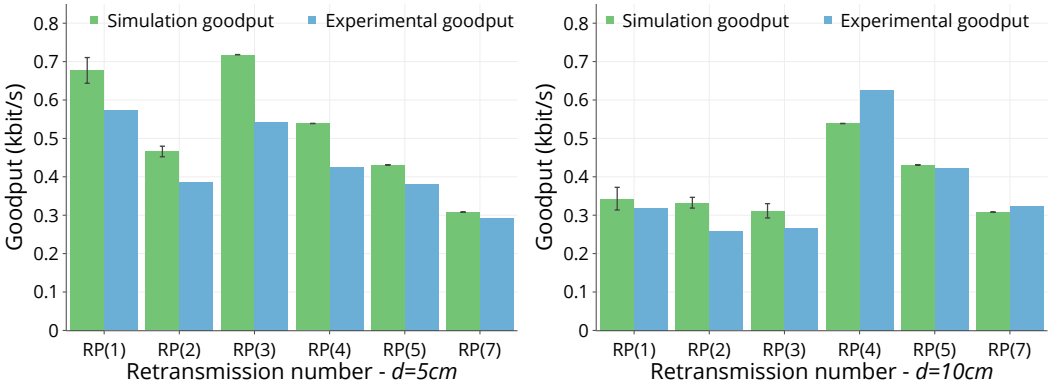

Figure 4.14: The experimental goodput (blue) compared with the simulation goodput (green) for the use case as a function of the number of consecutive PHY-SDU emission at 5 cm (left) and 10 cm (right).

by these results, the best value for r can be decided using simulations only, removing the need for a lengthy experimental campaign.

4.4 Lessons learned

Modeling the LED-to-camera communication using a Markov-modulated Bernoulli process allows us to easily study the performance of the RP and RS strategies. For both cases, experimentation validates analytical and simulation results and outlines that RP is more suitable when the distance between the LED and the camera is higher, while RS is better for short communication distances. Nonetheless, having an adaptive strategy that switches between RS and RP depending on the distance between transmitter and receiver is not possible without a feedback from the smartphone side. Also, these strategies inefficiently exploit the channel capacity by replicating information and dot not guarantee the successful data delivery, since the emitter never receives acknowledgement from the smartphone. We further discuss these issues in Chap. 5 and Chap. 6 by studying first the addition of a feedback channel and then a coding approach to avoid duplication.

Relying on this model, we were able to design the first LED-to-camera communication simulator. We have validated *CamComSim* comparing simulation results with the performance reached by a real life testbed. Then, we illustrated with a practical use case the complete usage of *CamComSim* to tune a broadcast protocol that implements the transmission of a 1 kbyte log file. The results highlight that our simulator is very precise and can predict the performance of a

LED-to-camera system with less than 10% of error in most cases. The availability of very accurate simulators offers a great ease of use and the opportunity to tune protocols without the burden of realizing experiments on a testbed.

We hope these tools will allow researchers to focus on the design of network solutions for LED-to-camera communication, without going each time through cumbersome experimental campains. For that purpose, we made *CamComSim* available as an open-source software under Apache license at http://vlc.project.citi-lab.fr/camcomsim.

Chapter 5

Flashlight-to-LED Communication

In Chap. 4, we demonstrated that our LED-to-camera communication system would benefit a lot from a feed back channel. Also, many smartphones already have a flashlight that can be used as VLC emitter [40]. On the smart device side, conventional LEDs can be leveraged as VLC receivers and so, are the only requirement to enable bidirectional visible light communication [42]. We gave further details on flashlight-to-photodiode and LED-to-LED communication in Chap. 2.

Therefore, in this chapter, we improve the system we depicted in Chap. 3 and introduce *LightIoT*. *LightIoT* enables bidirectional VLC transmission between a low-cost, low-power colored LED that is part of an IoT device and an unmodified and non-rooted smartphone.

The IoT device is thus able to send and receive information at the same time through its LED, while the smartphone uses its flashlight and camera, respectively to send and receive data. The *LightIoT* system is depicted in Fig 5.1.

This bidirectional system allows the design of more complex and robust redundancy mechanisms. Nonetheless, achieving this requires us to cope with several issues unsolved so far:

- Smartphone manufacturers do not allow applications to control the flashlight at the kernel level without root access, enforcing the use of their high-level API.
- The flashlight clock rate varies between smartphones, and even on the same device, depending on the system load.

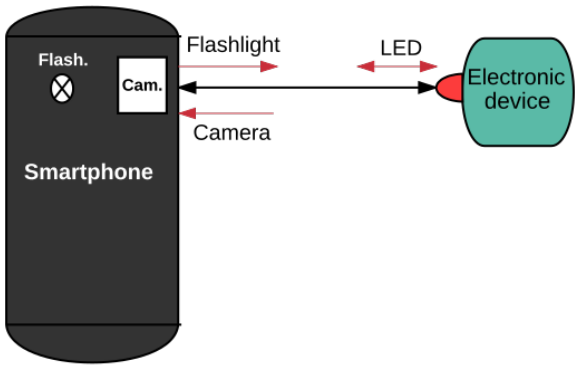

Figure 5.1: Our bidirectional VLC system.

- The flashlight API is not time accurate, which adds latency and heavy jitter to the signal.

This chapter presents the design and the implementation of decoding processes that resolve these concerns. Our contribution is the proposal of smart mechanisms and protocols that cope with the flashlight signal jitter and that detect the flashlight clock rate whatever the smartphone. We designed a modulation scheme based on a variable pulse width modulation (VPWM), that we adapted abetted by our experimental observations of the smartphones flashlight behavior. Our frame format includes synchronization patterns that let the device detect the beginning of a transmission, acquire the clock rate, correct both the clock jitter and bias by adapting the signal sampling itself at runtime. These key features harden the flashlight-to-LED channel.

The rest of this chapter is organized as follow. We describe the design of the flashlight-to-LED communication in Sec. 5.1, and we evaluate the performance of our implementation in Sec. 5.2. Finally, we present uses-cases for this flashlight-to-LED communication channel in Sec. 5.3.

5.1 Enabling Smartphone-to-LED Communication

A feed-back channel from the smartphone to the object is required if we want to transform the LED from a simple *beacon*, continuously broadcasting the same information, in a real network device, capable of transmitting different types of information in unicast communication. Moreover, this smartphone-to-LED communication might prove useful not only for control purposes, but even for

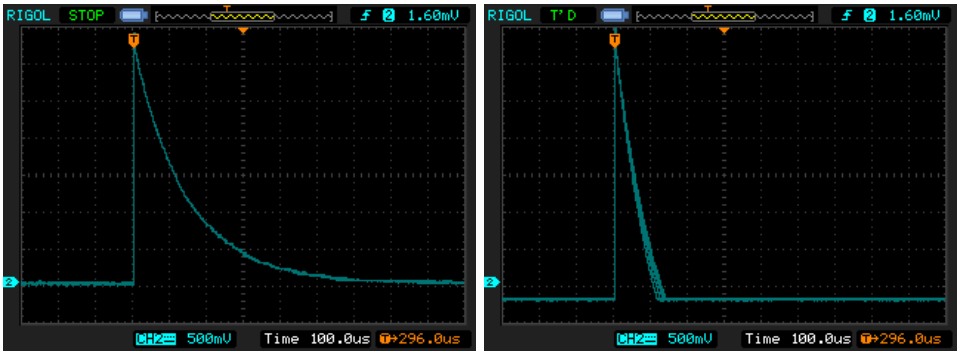

Figure 5.2: On the left, the LED discharge tension when it is lighted by ambient light only. On the right, the same LED lighted with the smartphone flashlight

some applications, e.g. modifying the default settings of a device using the user smartphone.

In this section, we present the main design principles of the communication link from the smartphone, more precisely from the flashlight, to the LED. PDs are often considered as receivers for VLC, as they are the appropriate and most efficient electronic components for light sensing, by converting the incoming light into electrical current. However, to be efficiently exploited, PDs require a fitted piece of hardware, based on a trans-impedance amplifier (TIA) and analogical filters, making this solution not suitable for our system, which aims to be as unobtrusive and cheap as possible. Therefore, in our design, we exploit the fact that a regular LED can also be used as a photo-receptor, by reverse-biasing it.

5.1.1 Communication principles

Although they possess a low light sensitivity, LEDs have already been used as receiver in VLC, as detailed in [42, 26]: by charging a reversed biased LED, and measuring its electric tension after a short time interval following this charging, the variations in the incoming light intensity can be detected. This allows us to easily detect the on/off periods of an incoming OOK modulated beam of light. However, previous works [42, 26] only investigated LED-to-LED communication; our implementation of smartphone-to-LED communication is therefore a first in the literature.

As explained, a regular LED can also be used as a photoreceptor, by reverse-biasing it, as further detailed in Chap. 2. Nonetheless, wired in this way, the light generates a very weak reverse current. The light sensitivity of the LED is low

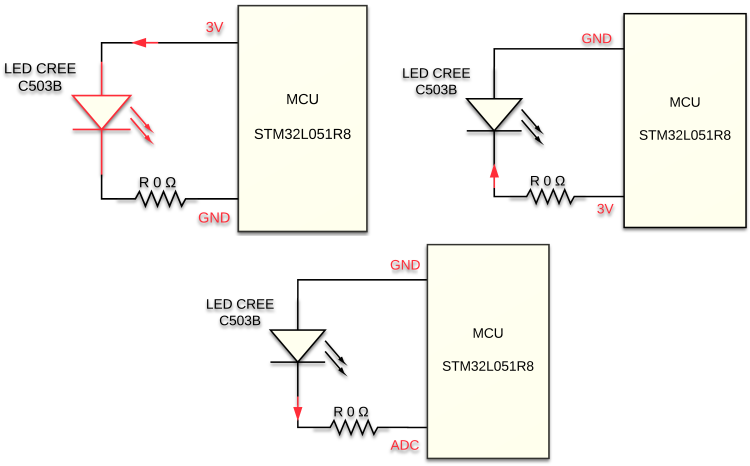

Figure 5.3: From the top left to the bottom, the LED is wired in *transmitter*, *charging* and *receiver* operating mode respectively.

and can not be directly measured by the MCU ADC. Even if this method avoids the use of a photodiode, an amplification circuit is needed to achieve correct performances.

In [42] and [26], the authors propose a smart way to solve this issue, by taking advantage of the intrinsic capacitance characteristic of the LED and the fact that the incoming light creates a current flow that can make the discharge duration shorter.

This is what is shown in Figure 5.2. After charging the LED internal capacitance during 1 µs, the voltage at its bounds is measured with an oscilloscope 100 µs later. When the LED is lighted by the ambient light only, the full discharge takes 600 µs (the left curve), while it is less than 100 µs with a supplementary lighting source that is the smartphone flashlight in our case.

This design only necessitates MCU software changes and does not repose on additional hardware: the LED anode is directly wired to a digital GPIO pin of the MCU and its cathode to an ADC alternate function capable pin.

On the smartphone side, two possibilities exist: using the LCD screen of the smartphone by controlling its back-light intensity [129], or triggering the flashlight built into most current mobile phones [51]. Since using the screen makes it impossible to use the back camera as a receiver at the same time, we adopt the flashlight as VLC transmitter in *LightIoT*.

The listings in Appendix A show the two methods we can use to switch the flashlight ON or OFF using the Android Software Development Kit (SDK). The first one in List. A.1 with an API version lower than 23 and the second in List. A.2 with an API higher than 23. The first one is much slower because it relies on the

Camera `CaptureSession` settings. Thus, turning the flashlight ON or OFF is done by updating the `CaptureSession` configuration and reloading it, which induces latency. The second one is faster, but it needs a full and exclusive access to the camera. That means no capture can be done at the same time, so the smartphone can not receive data anymore.

5.1.2 Modulation characteristics

Controlling hardware components, such as an LED generating a signal, is generally achieved by a real time system, such as an MCU or a field-programmable gate array (FPGA), using hardware oscillators and timers to achieve time synchronization and turn on or off the light at precise time intervals. However, this precision is not possible for application-based implementations on a mobile operating system, e.g. an application controlling the smartphone flashlight, where these tasks are performed by the central processing unit (CPU).

An alternative would be to build a driver at the operating system kernel level, to take advantage of its ISR, and thus get a better time accuracy, such as in the OpenVLC platform [124]. Nonetheless, this strategy is not feasible in our case, since mobile application developers can not access the lower levels of the operating system and they are constrained to use the programming interfaces offered by the native development kit (NDK), in the best case, or the SDK, otherwise. For instance, flashlight control is not implemented in the Android NDK; therefore, changing the flashlight state through its file descriptor is not allowed on a non-rooted smartphone, imposing the SDK implementation introduced in Sec. 5.1.1.

The clock jitter is therefore avoidable in our case. To add robustness against it, we propose a modulation scheme based on VPWM, where a pulse is a period during which the flashlight is turned ON. In this scheme, the data bits are mapped on VPWM symbols, as shown in Fig. 5.4, where each symbol is built using an initial pulse duration P, and a variable offset of the flash-ON duration, V.

The two modulation parameters, P and V, should be determined based on some properties of the smartphone in order to alleviate the clock jitter problem. First of all, the smartphone must be able to turn on and off the flashlight in a time interval P. Ideally, the smartphone needs to be able to correctly measure the period P and the execution of the switch on/off command should take a constant time. The clock jitter is produced by imprecisions on all these points,

Symbol	Data	Pulse width
S_0	0	P + 0V
S_1	1	P + 1V

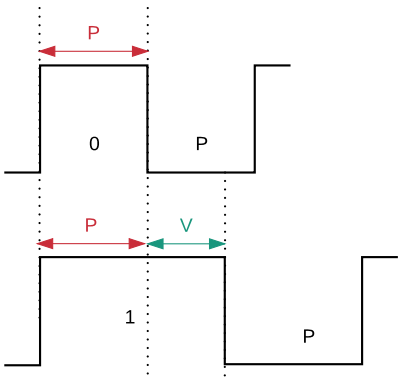

Figure 5.4: Symbol data mapping and diagram.

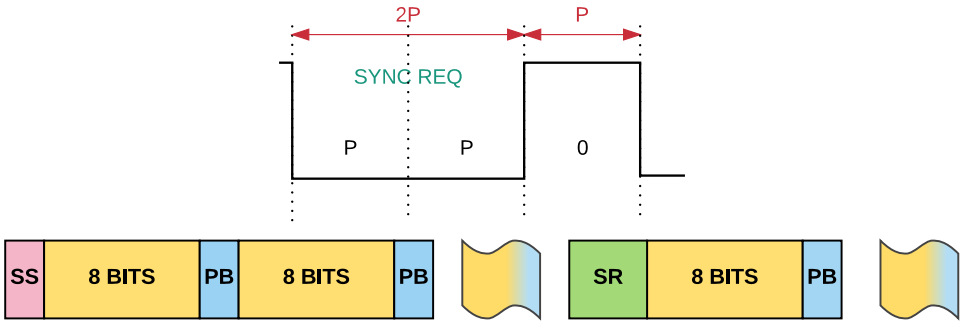

Figure 5.5: On the top, the SR symbol. On the bottom, a frame that includes a SR symbol sent to initiate the synchronization process.

hence the use of parameter V, which should be larger than the worst case jitter. Since the optimal values of P and V depend on the smartphone, the LED needs to estimate their duration at the beginning of the reception. We cope with the jitter of the flashlight signal and with heterogeneous clock rates between devices with a smart synchronization and clock recovery method that we detail in Sec. 5.1.3.

5.1.3 Receiver synchronization

As explained before, the optimal values of the modulation parameters vary depending on the smartphone and the timing inaccuracy of the mobile OS, producing signal pulses of inconsistent width during a transmission. To address this point, the receiver must synchronize with the transmitter and retrieve the signal characteristics, P and V, to correctly demodulate the signal. For that purpose, the smartphone emits two kind of synchronization symbols as shown in Fig. 5.5: 1) a start sequence (SS) at the beginning of each transmission and 2) a SYNC REQ

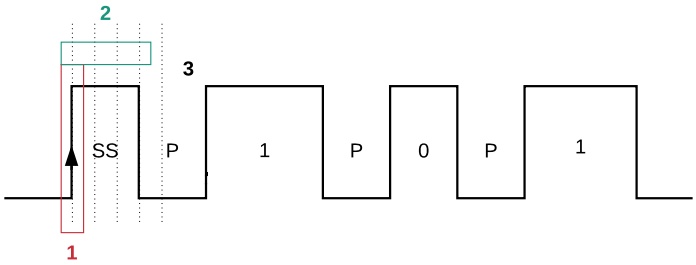

Figure 5.6: Receiver synchronization process.

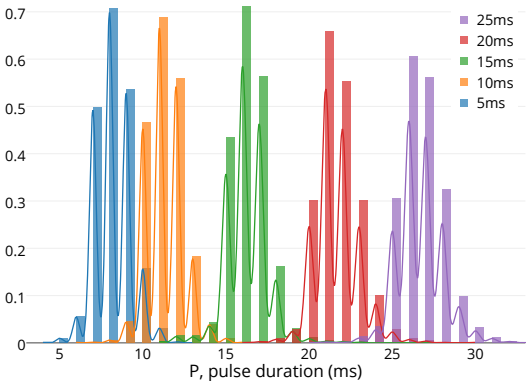

Figure 5.7: Distribution of effective pulse duration for different configured pulse duration values.

symbol (SR) periodically sent to recalibrate the smartphone symbols table. This is the key point to recover from clock jitter and harden the demodulation.

This mechanism is depicted in Fig. 5.6: the receiver detects a rising edge (1), that signals the start of a transmission; it then measures the pulse width (2) by sampling the signal until a falling edge is detected. Thus, the receiver recovers the pulse width period P and it can continue receiving data symbols to retrieve V on the fly, by measuring the longest next pulse. To compensate for a potential loss of synchronization and recover from a link loss, an SR control symbol is inserted after the parity bit, if the next symbol is S_0, as shown in Fig. 5.5. Thus, if a *light off* pulse of duration $2P$ occurs, the receiver knows that the next *light on* pulse represents the beginning of a byte, and it can recalibrate.

5.2 Experimental evaluation

To evaluate the flashlight-to-LED communication system, we implement a smartphone application that randomly generates series of 1000 bytes and transmits them to the LED using the flashlight. The application toggles the flashlight using

the API 23 method shown in List. A.2. We maintain the smartphone in a fixed position, while controlling the angle it makes with the LED axis. On the smartphone side, we record the transmitted symbols and timing information, while on the VLC module we log the LED discharge voltage and the received symbols, as computed by the MCU.

We begin by evaluating the smartphone ability to transmit symbols at a fixed frequency, in order to deduce the optimal modulation parameters, P and V. Fig. 5.7 shows the distribution of the effective pulse duration for several configured values, from 5 ms to 25 ms.

We note that the system can not achieve a pulse duration as low as 5 ms; this can be noticed from the blue curve, where the effective pulse duration is around 8 ms. Another important observation is that the distribution of the effective pulse duration is rather wide and the results obtained for two configured values separated by less than 10 ms are overlapping. Based on these results, we consider $V = 1.5P$ safe to avoid ISI in the worst case scenario and let the receiver synchronization algorithm, described in Sec. 5.1.3, retrieve the correct clock rate whatever the smartphone.

Next, we evaluate the communication range of the flashlight-to-LED technology and its potential limitations, especially the ambient light interference and the LOS condition. For this, in Fig. 5.8, we show the signal-to-noise ratio (SNR), measured at the LED receiver, for different communication distances and angles. Regarding the communication throughput, we notice an on/off behavior: for an SNR above 1 dB, a throughput of 30 bit/s is achieved, regardless the quality of the channel, while the throughput quickly drops to zero for an SNR under 1 dB. This 1 dB SNR threshold is represented by the red line in Fig. 5.8. We observe in these results a maximum communication range of 110 cm and a very constrained communication angle between the LED and the smartphone, lower than 15°. These results can not be compared with previous works on flashlight usage [39, 33], as these studies use a photo-diode as a receiver, and not a small LED as we do in our work.

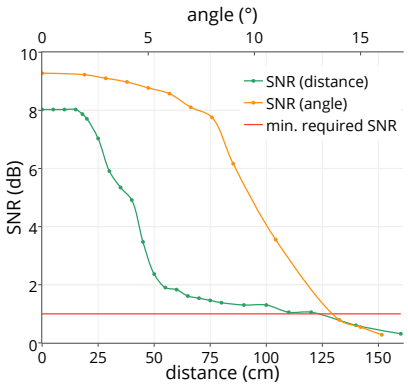

Figure 5.8: SNR when varying the flashlight distance (green curve) and the smartphone-LED angle (orange curve). The red line represents the minimum required SNR for correct decoding.

5.3 Flashlight-to LED Usages

5.3.1 Flashlight-to-LED for communication

The experimental evaluation discussed in Sec 5.2 shows that the Flashlight-to-LED communication channel achieves 30 bit/s of throughput, at a maximum communication range of 110 cm.

Although the throughput achieved for the flashlight-to-LED communication is very low, this channel can still be useful for some applications: 1) for control purposes, e.g. switching a device ON-OFF or sending a short command, 2) for configuration, e.g. modifying the default settings of a device, and 3) for security purpose, e.g. enhancing BLE security using light as a side channel for key provisioning or proximity assessment.

5.3.2 Flashlight-to-LED as wake-up channel

Beyond communication, the Flashlight-to-LED channel might be used for other purposes. The first example is the use of flashlight-to-LED as a wake-up channel. Indeed, the use of a low-energy wake-up side channels has gained a lot of popularity in energy-constrained RF networks [46]. This allows switching off the main radio transceiver, with a significant impact on the overall power consumption.

We notice that this wake-up feature is innate in our system, since the LED does not need to be powered in order to detect an incoming transmission from the smartphone. Coming back to the LED-to-camera communication in Chap. 3, this means that the LED does not need to act like an RF beacon, continuously

transmitting the information in a loop. Instead, by sending a short light message, the smartphone can wake up the object from stand-by mode and initiate a unicast communication. This mode of functioning has important advantages, not only from an energy point of view, but also for security reasons: the object will transmit the information only when it knows that the smartphone is in its proximity.

5.3.3 Flashlight-to-LED as feedback channel

A third usage of the flashlight-to-LED system is its integration in a bidirectional communication technology between a small colored LED and an unmodified smartphone. However, our results indicate that the flashlight-to-LED average throughput is 30 bit/s, while it can be about 2 kbit/s for the LED-to-camera link. With such an asymmetry between the channels, acknowledging every packet reception on the smartphone side does not seem realistic. Nevertheless, this feedback capacity still allows us to refine the redundancy mechanisms introduced in Chap. 4.

For the case of the RS mechanism, we propose two acknowledgment strategies. In the first one, denoted as RSc, the LED module continuously transmits the entire sequence of packets until the smartphone receives it completely and sends an acknowledgment. In the second strategy, named RSs, after sending the whole sequence of packets, the LED module stops and waits as feedback from the smartphone the list of missing sequence numbers, thus retransmitting only the missing packets in the following iteration. This is repeated until the smartphone notifies the reception of all the packets. These two solutions can also be combined with the RP strategy, by sending a packet i consecutive times instead of only once. This results in two other redundancy mechanisms, denoted as RSc(i) and RSs(i).

Results in Fig. 5.9 show the goodput of these four strategies, as well as the number of packets that need to be sent in order to recover 600 bits of data split into 50 packets. The results on the left side of the figure evidence that, in terms of packet transmissions, the RSs strategy takes advantage of the feedback channel and performs better than all the others approaches. However, on the right side of the figure, we notice that the goodput for the RSs and RSs(i) strategies falls dramatically. This performance drop is produced by the time spent sending the flashlight-to-LED feed-back, which is about 50 times higher that the transmission on the LED-to-camera link. On the other hand, the RSc

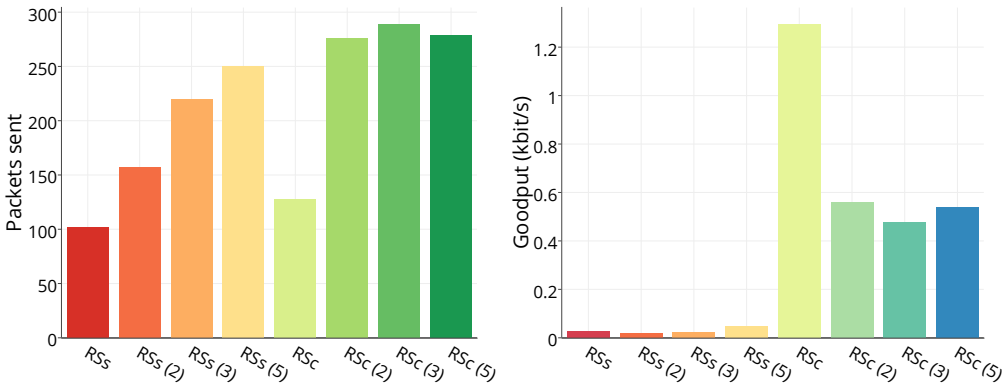

Figure 5.9: On the left, the number of transmissions needed in order to receive the information for the different redundancy mechanisms when feed-back on the flashlight-to-LED channel is used. On the right, the goodput achieved in the same conditions. The distance between object and smartphone is 5 cm.

strategy offers a goodput twice higher than all the RSc(i) mechanism, with a moderate overhead.

We note that, with all the retransmission and feed-back overhead, we obtain a goodput of 1.2 kbit/s between an LED-equipped object and an unmodified smartphone. Moreover, this communication can be launched on demand by the smartphone and makes use of unicast messages, allowing to transmit different types of information to different users (e.g. usage statistics to the owner and troubleshooting logs to a customer service employee).

These results show that using the flashlight as a feedback channel helps hardening the communication system. Nonetheless, because the flashlight-to-LED throughput is only 30 bit/s, these feedback strategies add a non-negligible overhead making their use not realist.

Chapter 6

SeedLight

6.1 Introduction

As we have shown in Chap. 3, with a LOS-Channel like in our system, the ROI is only a part of the picture and decreases rapidly with the distance between the LED and the camera [132, 68]. Losses are important and intra-frame losses are added to these caused by the IFG. This highly constrains the PHY-SDU size since the PHY-SDU must fit into the ROI to be received. To get around these issues, we show in Chap. 3 that our system basically retransmits packets continuously. This leads to useless transmissions and makes the sending of large messages arduous. Therefore, this chapter introduces an appropriate mechanism to overcome these drawbacks by developing *SeedLight*, a coding scheme based on random linear coding (RLC) to improve the goodput and reduce the number of retransmissions in LED-to-Camera broadcast.

Several similar approaches are presented in Sec. 6.2 below but our work is distinguished through the following contributions:

- We provide the first RLC coding scheme optimized for VLC and OCC.
- We reduce the RLC overhead by using a pseudo-random linear coding (PRLC) method.
- Our coding scheme works with smaller PHY-SDUs, an important constraint for the kind of small LEDs we use as emitter.
- The code complexity enables its implementation on cheap MCUs.

The rest of this chapter is organized as follow. Sec. 6.2 presents previous works and spotlights the remaining challenges this work addresses. We then

detail the coding scheme that is the main contribution of this chapter in Sec. 6.3, before evaluating its performances by experimentations in Sec. 6.4. Sec. 6.5 focuses on the algorithm implementation and its performances evaluation on low-end IoT devices, before ending with the lessons learned in Sec. 6.6.

6.2 Related work

To overcome the loss and asynchronous communications between emitter and receiver in LED-to-Camera broadcast, several methods have been studied in the literature. [36] proposes to repeat consecutively every packet r times to ensure that each of them can be captured, regardless of the misalignment between the LED and the camera. It is clear that this scheme leads to a reduced data rate. Moreover, the smartphone must receive all the packets independently to decode the frame. A single missing or erroneous packet hence imposes to wait for a whole frame retransmission.

More recently, [89] proposes a combination of Hamming codes and bit interleaving to avoid packet retransmission. Similarly, [57] uses Reed-Solomon encoding to recover both IFG losses and inter-symbol interferences. These approaches assume that missing bits occur only during the IFG, and that the whole picture contains information. These all require a channel estimation, to set the appropriate number of retransmissions r, Hamming blocks or redundancy bits. As a consequence, this makes them unsuitable for LOS scenario, where the bit loss highly increases with the distance between the emitter and the receiver.

Recovering the loss in LOS channel is more challenging. Fountain codes [81] have been employed to improve the data rate and the link reliability in LED-to-Camera broadcast. [132] leverages Raptor Codes [112] and proposes an experimental evaluation using ceiling LEDs. Their approach reduces the latency at the cost of 25% overhead. In a close way, [29] studies LT-codes [79] and proves their implementation can reach 1.6 kbit/s of goodput. Nonetheless, the overhead induced by these approaches would be unacceptably large for the extremely small PHY-SDU size we use. Another flaw of LT-codes and Raptor codes is the coding complexity at the emitter side, which is why studies like [29, 132] perform the encoding step on a computer.

Another class of rateless codes, called RLC, are widely used in the context of network coding (NC) [88]. However, RLC has not been considered yet for VLC. RLC can be applied within a NC framework, but also as a good packet-level error correction solution: they are simple to implement at the emitter side and per-

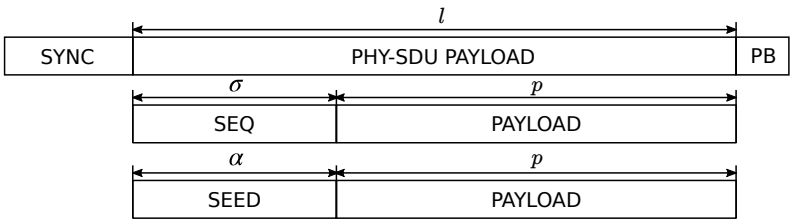

Figure 6.1: Packet sent without coding (middle) and chunk format generated by PRLC (bottom) encapsulated into a PHY-SDU (top).

form similarly to optimal erasure codes for a sufficiently large finite field used for creating the linear combinations of source symbols. The major obstacle in the application of RLC is the decoding complexity of Gaussian elimination decoding. In [87] the authors compare RLC and Raptor code performance, showing that, for moderate loss values, the performance of both codes is almost the same but, with higher loss rates, RLC has a better performance.

6.3 SeedLight design

These solutions already proposed for VLC in the literature can hardly apply to our system, since we use small color LEDs that constrain the PHY-SDU length, as explained in Chap. 3. Also, implementing these codes on a low-resources MCU, as those we use is not feasible. Therefore, this section presents a coding scheme that overcomes these limitations and improves LED-to-camera communication reliability and goodput compared to a systematic retransmission method.

SeedLight implements a PRLC approach and considers the following requirements: (a) the code overhead must remain as low as possible, (b) the complexity should be low on the emitter side, since the end-devices are low-cost embedded devices with limited computational, memory and energy capabilities, (c) application layer message size cannot be larger than a few hundreds bytes as a consequence of (b).

We begin by introducing RLC, which is the root of *SeedLight*, and a few notations used in this chapter, before proposing our alternative implementation to reduce the code overhead.

6.3.1 Random Linear Coding

We assume that an emitter Tx wants to send a message of G bytes to a receiver Rx, split into N blocks $b_1, \ldots, b_N$ that will be encapsulated in a PHY-SDU with payload size p. In the RLC terminology, G is usually termed as the generation size and chunks designate the coded blocks. RLC performs the coding over blocks using a Galois Field $GF(q)$ where $q = 2^\gamma, \gamma \in \mathbb{Z}^+$, is the field size. GF is a key parameter in RLC since it significantly impacts the code performance: a small field size will result in linear dependent chunks, lower overhead and lower computation cost. Because linearly dependent chunks contain the same information, their number must remain as low as possible. However, [80] demonstrated that a field size of 2^8 is large enough to have a nearly non-linear dependency.

Following the RLC method further detailed in [88], a set of N coefficients of γ bits $c = (c_1, \ldots, c_N), c_i \in GF(q)$, is used to generate the coding vector v by linearly combining uncoded blocks as follow:

$$v = \sum_{i=1}^{N} (c_i \cdot b_i) \tag{6.1}$$

For successfully decoding and solving the linear equations, Rx must know the coding coefficient c for each coding vector v. This is usually achieved by building a chunk $C = [c, v]$, appending c to v, which results in an overhead of $N * \gamma$ bits. Considering GF(256), a 50B message and a 2B payload, this adds an overhead of $25 \cdot 8$ bits coefficients for each chunk. That does not fit into a PHY-SDU in our system, where the size l is only 24 bits.

6.3.2 Compressing the coding vector

To avoid transmitting the coding coefficients c, our method is inspired by [9] and uses a seed of a pseudo-random generator (PRNG) to determine c. Our PRLC scheme works as follows.

Encoding: For each chunk C, E randomly generates a seed $s \in [0 \ldots 2^\alpha]$, where α is the seed length. Then, a Mersenne-Twister PRNG that follows the implementation proposed in [84] and initialized with s is used to produce the N coefficients $c_1, \ldots, c_N$.

Since $c_i \in GF(q)$ and $\gamma \leq \alpha$, c_i is built with the γ lowest significant bits of the PRNG results. The coding vector v is then computed using Eq. (6.1) and sent with

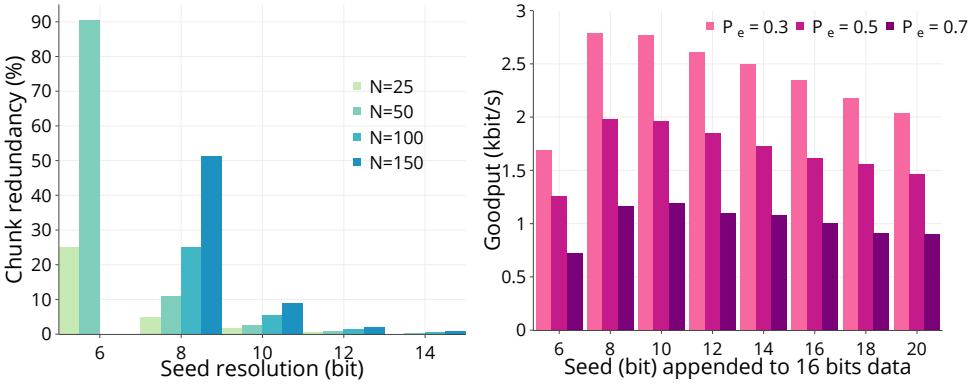

Figure 6.2: The chunk redundancy introduced by duplicated chunks vs the seed length and N. Goodput with G=100B for some PHY-SDU error probability vs the seed length (left).

the seed s in a single chunk $C = [s, v]$, following the format shown on the bottom of Fig. 6.1.

In this way, the overhead remains low: only α bits to describe the seed, which replaces a sequence number no further needed with *SeedLight* approach.

Decoding: The decoding step on the receiver side is straightforward and works iteratively. Each time a chunk is received, Rx reconstructs the coding vector c using the seed s included in C: the same PRNG described above is initialized with s and used to produce the N coefficients that build c. Then, c and v are added to the decoding unit matrix and used to solve the linear equations by Gaussian elimination: $\begin{bmatrix} c & v \end{bmatrix} \begin{bmatrix} c^{-1} \end{bmatrix} = \begin{bmatrix} 1 & b \end{bmatrix}$.

Rx can then successfully decode G if it has received at least N linear independent packets. Complexity and implementation are discussed in Sec. 6.5.

6.4 SeedLight evaluation

In this section, we evaluate and present the achievable performance of *SeedLight* LED-to-camera broadcast under realistic conditions and compare this with a conventional retransmission method. We mainly focus on two aspects: 1) the coding parameters, namely the seed length α and the generation size G, and 2) the distance between Tx and Rx.

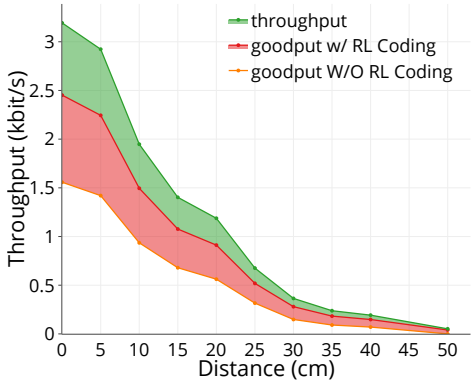

Figure 6.3: Throughput compared with the goodput vs the distance, with and without coding for G=100B (right).

6.4.1 Seed and Generation size

An important property of the LED-to-camera communication is the fact that information losses are unavoidable and almost regular. Indeed, even when functioning at the maximum frame rate of 30 frames per second, the pictures taken by the smartphone do not represent a continuous view of the LED, because of the IFG. To cope with these losses, the data needs to be segmented into packets, which can be easily reassembled and retransmitted. However, the size of the packet has a significant impact on the packet loss probability as discussed before in Sec. 3.4.4. These results expose that the seed length and the overhead ratio have to be carefully designed and are major influence factors of *SeedLight* performance.

Therefore, we evaluate by simulation with *CamComSim* the impact of the seed length (α) on the ratio of linearly dependent chunks produced when the minimum number of blocks N, required to recover the message, increases. Results in the left plot in Fig. 6.2 highlight that, to get less than 10% duplicated chunks, 8 bits are necessary for $N = 50$ while 10 bits are needed for N between 100 and 150. We also see that less than 8 bits are not enough, even when N is small. The duplicate ratio becomes negligible for seeds larger than 10 bits and for values of N higher than 100.

Right plot in Fig. 6.2 shows the goodput when varying the seed length for a PHY-SDU error rate P_e of 0.3, 0.5 and 0.7, respectively at 5, 10 and 20 centimeter, taking a generation size $G = 100B$. The best goodput is always obtained with a 8 bits seed and decreases progressively when the seed length increases. This is mostly because the overhead rises, even if the amount of linearly dependent

blocks highly reduces. As the duplicated chunks ratio with a 6 bits seed is larger than that of 8 bits, the goodput is in such case lower.

6.4.2 Communication Performance

We now evaluate by experimentation the LED-to-camera communication performance as a function of the distance between the LED and the smartphone. We use a slightly modified version of the testbed we described in Sec. 3.2. We set the emitter symbol rate to 8 KHz and place it in standard indoor illumination conditions, near a window and illuminated with neon lights. The illuminance has been measured with a luxmeter, around 650 lux. The PHY-SDU size is 24 bits and is build as depicted in Fig. 6.1, with a 8 bits seed or a 8 bits sequence number. The emitter broadcasts continuously a 100 B message, which is transformed with *SeedLight* in a set of chunks, built with a 8 bits seed or, without *SeedLight*, in 50 different packets containing a 8 bits sequence number. For the case with *SeedLight*, the encoding of the set of chunks is done before the transmission starts. Thus, the computation does not affect the communication performance.

We compute the throughput as the number of received bits per second, error free, including duplicated packets or linearly dependent chunks. The goodput is calculated by removing duplicates and transmitting until the smartphone receives all the 100B message.

Fig. 6.3 compares the throughput and the goodput for transmissions with and without the benefits of *SeedLight*. The three curves follow the same tendency: the throughput drops from 3 kbit/s at 5 cm to less than 0.2 kbit/s at 40 cm. We notice two sharp decreases, respectively at 5 cm and 15 cm. From 0 to near 5 cm, the LED lightens the full surface of the CMOS sensor and the loss is mainly due to the SNR decrease and the blooming effect [18]. The second, between 15 to 20 cm, is less straightforward and we make some assumptions to explain the phenomena. First, at this distance, the ambient light, that has been measured at 650 lux, starts interfering with the LED signal. Then, the ROI becomes small enough that no more than one packet is received at once and a non negligible amount of frames do not contain any information. The red curve shows that, at 5 cm, the goodput with *SeedLight* is 2.25 kbit/s while it is 1.4 kbit/s without, bringing out a gain of 60%. Also, we can see that the relative gain grows with the distance: a gain of 90% is noticed at distance of 30 cm.

6.5 Implementation discussion

We show that *SeedLight* improves the LED-to-Camera transmission goodput compared to a systematic retransmission method. However, this is achieved at the cost of an additional coding step before transmission. Nonetheless, as explained in Sec. 6.3, *SeedLight* focuses on low-end IoT devices for which available resources are very limited. In such situation, software development and computation intensive algorithm optimization is complicated. Therefore, in this section, we discuss *SeedLight* implementation problems on aforesaid embedded platforms and evaluate the coding impact on two distinct MCUs and smartphones.

6.5.1 PRLC implementation on IoT devices

Several RLC algorithm benchmarks have been implemented previously. These studies were performed on computers [130], smartphones [37] or high-end embedded devices, at least able to run a UNIX operating system [50]. There is not a known implementation on low powerful platform like ours. As a comparison, a Raspberry Pi has 256 Mo RAM and 700 MHz CPU while the STM32L0 is only 32 MHz and 8 Ko RAM. That spots the large gap in the expected performances.

As shown in Sec. 6.3, PRLC relies on the widely used GF(256): its optimization has been discussed in [44], where the authors propose several solutions optimized for speed or memory. Tab. 6.1 shows the number of operations and memory requirement for three of those solutions that comply with our system: (1) uses full multiplications LTs, is the fastest method but its LTs are larger than our MCU flash size; (2) uses log and anti-log LTs, is well balanced between memory requirement and complexity; (3) requires the largest number of operations and will be restricted to cases in which the low available RAM forbids the use of lookup tables (LTs).

Comparing these solutions with the constraints of the emitter and the receiver of our system, using method (1) is preferred for the decoding considering the 1 GB non-volatile memory available on the Nexus 5, while method (2) is preferred for the encoding, considering the low memory and the low computation resources available on the MCU we use.

According to Tab. 6.1 for (2) and the PRLC scheme proposed in Sec. 6.3, the code complexity to produce N chunks of P symbols is then $O(N^2P)$. In the same way, the complexity to decode a matrix of $N \cdot (N + P)$ bytes is given by $O(N^3 + N^2P)$.

Method	Operation	Memory
(1) Mult. LT	1 LK	$2^\gamma \cdot 2^\gamma$
(2) Log. LT	3 LK + 2 BR + 1 MOD + 1 ADD	$2 \cdot 2^\gamma$
(3) No LT	γ . (2 XOR + 2 SHIFT + 1 AND)	0

Table 6.1: Several multiplication implementations in GF(256). The operations are abbreviated LK for table lookup, BR for branch and MOD for modulus. γ is the GF size.

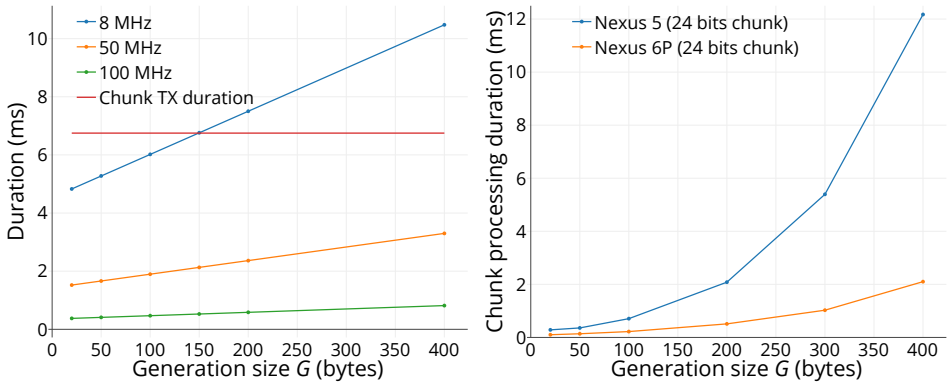

Figure 6.4: Computation time on a MCU to build 1 chunk (left). Computation time on the Nexus 5 and 6P to process 1 chunk and perform decoding (right).

6.5.2 Evaluation

We now evaluate the PRLC algorithm introduced in Sec. 6.3 and implementations discussed in Sec. 6.5.1 in our testbed. We benchmark the algorithm on the 8 MHz Cortex-M0+ MCU and the more powerful Cortex-M4 with a clock configured at 50 and 100 MHz. Both MCUs are often used in consumer IoT. For the receiver, we use an LG Nexus 5 with a Snapdragon 800 4-Cores 2,26 GHz CPU and the newer Huawei Nexus 6P with a Snapdragon 810 8-Cores 2.0 GHz CPU. The LG Nexus 5 runs Android Marshmallow version number 6.0.1 while the Nexus 6P runs Android Nougat version number 7.1.2. Both operating system are unmodified.

Algorithm benchmark: Considering the optimizations discussed in the previous section, Tx uses the method (2), i.e. using the log and anti-log LTs to perform GF(256) arithmetic operations, while Rx uses the method (1), i.e. using a full multiplication LT.

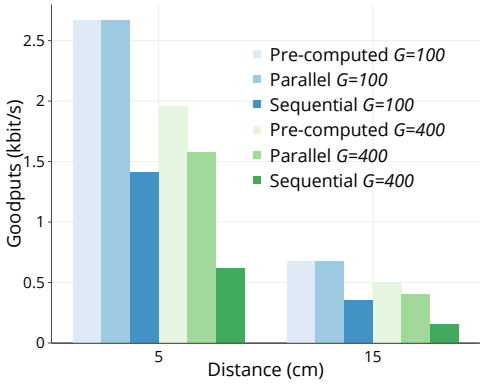

Figure 6.5: Goodput using different implementations for G=100B and G=400B.

The left plot in Fig.6.4 shows that the encoding computation time per chunk, when the MCU is used for communication only, increases linearly with the generation size. For the 8 MHz MCU (blue line), the time remains below the chunk transmission time when the generation size is smaller than 150 B and is about 5 ms with a 25 B generation, doubling when $G = 400$. This emphasizes that, with an appropriate mechanism, performing coding in parallel with modulating the LED is feasible, even for a low MCU frequency, when less than 150 B are transmitted.

The right plot in Fig. 6.4 shows the decoding and Gaussian elimination computation time for each chunk. The evaluation was performed while the smartphone was capturing and processing camera frames using the VLC decoding algorithm introduced in Sec.3.2. Unlike the encoding algorithm, the increase is exponential. The benefits of the Nexus 6P, that has a 8-cores CPU while the Nexus CPU has only 4 cores, is high and the chunk processing remains below 2 ms even with a generation size of 400 B. Since the camera frame rate is 30 fps and about 6 chunks are decoded in each, this shows the operations can be done in real-time. Also, further improvements are left pending, as developing the decoding algorithm in a C library to replace the current Java implementation.

MCU mode: The results discussed above were obtained when the MCU was used for communication only. However, the main functionality of consumer IoT MCU is not usually communication. This can be gathering data from sensors or performing main application logic algorithms. Radio transmission and signal generation can be realized by a dedicated chip but that is not always the case and, as a consequence, the MCU resources must be shared. In such cases, communication happens during reserved time slots or concurrently in a multi-

tasking approach. To conform with these needs, we propose three scheduling strategies for PRLC and transmission:

1. **Pre-computed** mode computes a set of chunks before transmitting them in a row, as in Sec. 6.4. This increases the needed RAM.

2. **Sequential** mode computes a chunk and transmits it right away. This increases the delay, since the transmission is interrupted between each chunk.

3. **Parallel** mode takes advantage of multi-tasking to compute a chunk when the previous one is transmitting. If the chunk computation time is larger than the transmission time, the transmission will stop during this time gap. This solution has the highest impact on then CPU load.

The pre-computed mode (1) is relevant when the data source, e.g. a sensor, stores the data immediately as encoded chunks, the sequential mode (2) when a prioritized task is running at the same time and the parallel mode (3) when the communication is prioritized regarding the main application task or when the MCU speed is hight enough to run both. We now focus on the end-to-end goodput evaluation, considering these three implementations to depict real use cases of IoT devices.

Fig. 6.5 shows the goodput transmitting a message of 100 B (blue) and 400 B (green) with *SeedLight* ($\alpha = 8$) using the software implementations discussed above. We place Tx and Rx at 5 cm and 15 cm distance. We note that the goodput falls as the message size increases, mainly because of the amount of linear dependent chunks transmitted. The plot highlights that, when working with a generation size of 100 B, the chunk computation can be achieved in real time even at 8 MHz and both transmission and chunk computation can be paralleled without throughput loss compared to the *Pre-computed* implementation. The *Sequential* mode for the 100 B message is half the values of the others since it adds a delay to compute the chunk before its transmission. With a generation size of 400 B, the goodput decreases between the *Pre-computed* and *Parallel* modes. This highlights that, in such cases, the MCU reaches its limit and a small gap between chunk transmission appears. Nonetheless, *SeedLight* still boosts the goodput observed with a systematic retransmissions scheme.

These results conform quiet well with Fig. 6.4 and bring out that *SeedLight* can be implemented even on the cheapest MCU with a minor impact on the communication performance. Note that reducing the GF size would alleviate

the load on the MCU at the cost of few linearly dependency overhead. Furthermore, *SeedLight* will not increase the MCU power consumption, since this can be done without any modification on the MCU clock and during the same time interval the device modulates the LED. Moreover, since it reduces the number of transmissions that are necessary to receive the full message, *SeedLight* hence reduces the energy consumed.

6.6 Lessons Learned

In this chapter, we have introduced *SeedLight*, a novel lightweight erasure coding method that leverages RLC theory to face the inevitable and large packet losses in LOS LED-to-Camera broadcast. The key idea of *SeedLight* is to take advantage of a PRNG to avoid the systematic transmission of the mandatory coding header introduced by RLC.

This work provides practical engineering insights and puts in evidence two points. First, experimental evaluation shows that this approach increases the goodput from 1.5 to 2.5 kbit/s compared to a packet retransmission mechanism. Compared to the erasure coding scheme recently proposed for OCC [132, 29], *SeedLight* introduces a much smaller overhead that makes it suitable when the PHY-SDU can not be larger than few tens of bits.

Secondly, the major benefit of *SeedLight* is the low computational overhead on the emitter side. A careful implementation of *SeedLight* on low-cost MCUs, like those often used in IoT devices and smartphones, brings out that our solution does not affect the communication performance and is compatible with such applications.

Chapter 7

Insight on Energy Consumption

Many of the objects we wish to see connected are generally powered by batteries, which are limited in size and, therefore, capacity. Also, wireless modules are known to be important power consumers, and energy considerations like the product autonomy are major constraints during the design and the conception phase of smart objects. Thus, choosing the appropriate communication technology, best fitting the user expectations in term of data rate, availability and battery life, is often challenging. In this context of personal objects, BLE is usually used when power efficiency is preferred to data rates.

Similar energy considerations stand true for mobile phones, where software, battery or radio optimizations are major research topics and crucial problems for phone manufacturers [101]. Therefore, in this chapter, we measure the *LightIoT* energy requirements and compare them with an equivalent system based on BLE, more precisely, on the Nordic Semiconductor nRF51 module. We measure the energy consumption on the object side in Sec. 7.1, and the one on the smartphone side in Sec. 7.2, before studying in detail representative use cases in Sec. 7.3.

7.1 Wireless module battery drain

We measure the power consumption of both BLE and VLC modules, using the Keysight N6705B Power Analyzer, that we also used as a power source to supply a 3.3 V voltage. The measurements are recorded on a computer to be further analyzed. The power measurement testbed in shown in Fig. 7.1.

These measurements were done in different conditions. In **BLE Wait**, the object is in the peripheral mode defined by the Bluetooth 4.0 specifications, with

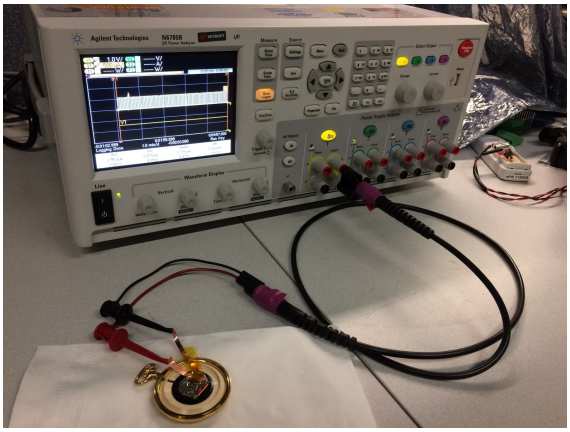

Figure 7.1: Our power measurement testbed. The smart device is directly powered by the Keysight Power Analyzer.

an advertising interval of 750 ms and waiting for a connection request. Advertising packets are sent without an extra payload on the three advertising channels with a radio output power of -12 dBm. In **BLE Scan**, the object is in master mode and it listens for advertising peripherals (e.g. sensors or beacons) in range, while in **BLE Tx/Rx** the object is connected to the smartphone, and the peripheral (the object in BLE Tx, the smartphone in BLE Rx) is sending a 20 B notification packet every 30 ms. The leakage current profile for the **BLE Wait**, **BLE TX/Rx**, and **BLE Scan** is shown in Fig. 7.2.

On the VLC side, the energy consumption is measured under the following settings: in **VLC Wait**, the module is placed in low power mode and wakes up every 750 ms to detect the beginning of a transmission, using an LED receiver as described in Chap. 5; in **VLC Tx**, the module is continuously sending packets using the LED; finally in **VLC Rx**, the module is actively sampling the LED and receives data from the smartphone flash. The leakage current profile for the **VLC Wait**, **VLC Tx**, and **VLC Rx** is shown in Fig. 7.3.

Then, using the leakage current measurements from the Keysight Power Analyzer, we compute the power consumption P and the consumed energy E.

The results in Fig. 7.4 show the average power consumption for the six scenarios defined above. Consumption in the wait state for the BLE and VLC devices are similar, 0.9 and 0.11 mW respectively. We can see that VLC Tx consumption is 39 mW, while it is only 18.4 mW for the BLE. This can be explained by the short duration of the RF energy consumption peaks compared to the time the LED needs to be on for a transmission, although the current intensity is around 20 mA in both cases. On the contrary, the VLC receiver is power efficient, for it

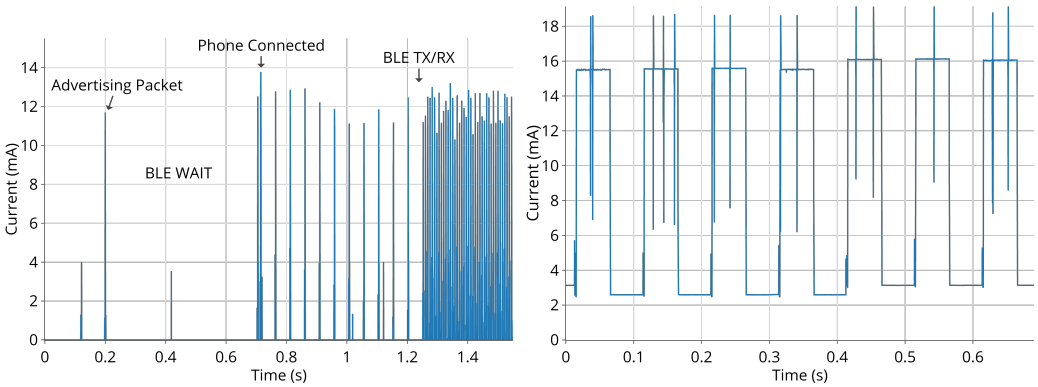

Figure 7.2: The BLE module leakage current in slave mode (left) or master mode (right). In master mode, the BLE device is scanning for peripherals.

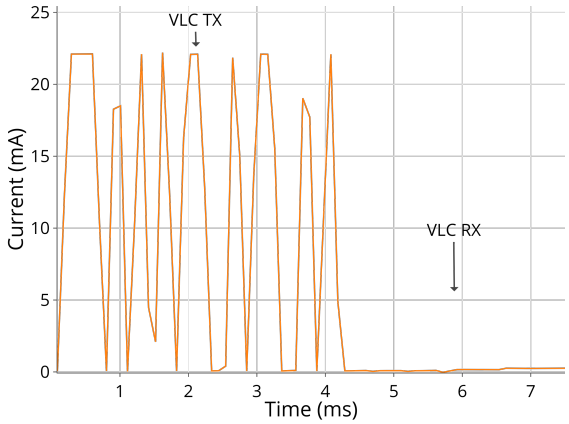

Figure 7.3: The VLC module leakage current in TX mode and RX mode.

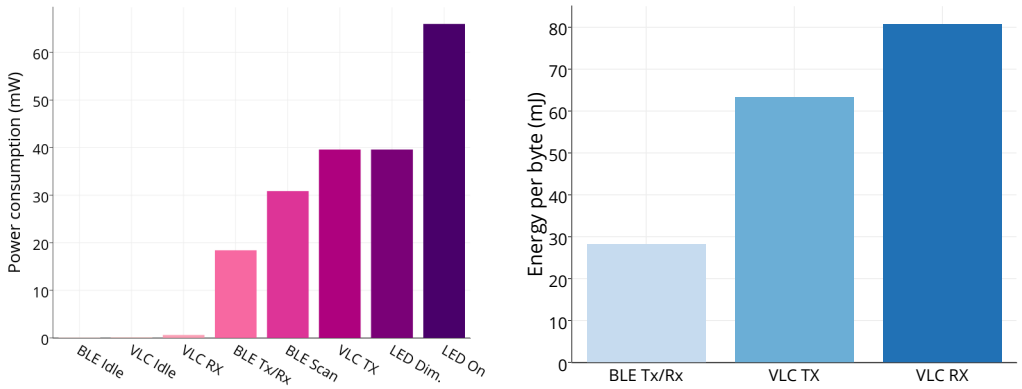

Figure 7.4: The communication module power consumption for BLE and VLC.

only requires MCU consumption and the brief LED charge steps, taking only 1 µs. Therefore, the VLC Rx consumption is negligible (0.61 mW) in contrast to the BLE reception (18.41 mW).

We note that the consumption of the VLC Tx module, while superior to its BLE counterpart, is actually produced by the energy required to turn on the LED. Since the modulation scheme and the driving circuit have a low complexity, the processing and signal generation contribution to the energy consumption is very low, quite similar to the contribution of the module in wait mode (0.61 mW). This means that, for devices that already keep LEDs turned on during their usage, no extra energy is consumed for transmission. Moreover, since the LED-to-camera transmission is based on alternating on and off periods, the communication module practically reduces the energy consumption of the LED, as if the intensity of the LED was dimmed.

This can be noticed in Fig. 7.4, by taking as reference the consumption in the **LED On** state, where the LED is simply on, without transmitting any information.

These results show that VLC becomes advantageous compared to BLE when the device is inoperative most of the time or if it has to wait a master request before transmitting. In that way, the LED can be kept off and the device in a very low power mode.

7.2 Smartphone battery drain

Measuring the power consumption on a smartphone is not obvious, as accessing its motherboard or removing the battery may affect its integrity. Another issue is to isolate the contribution of a single component, or that of an application, from the global power consumption. Different approaches based on hardware or software solutions have been proposed and compared. Relying on the study in [13], we use the Qualcomm Trepn Profiler application and compare these results with the battery drain estimation provided by the Android OS. During the tests, we shut down all the useless radio interfaces and we set the screen brightness to its minimal value following the recommendations in [14, 13, 52, 12].

Fig. 7.5 shows that VLC highly increases the leakage current both in RX and TX modes. On the other side, the impact of BLE remains low for both cases.

As expected, the results in Fig. 7.6 highlight the high power consumption of the VLC application, mainly due to the flashlight (1.35 W) when transmitting, and to the camera (1.92 W) when receiving. As a consequence, smartphone auton-

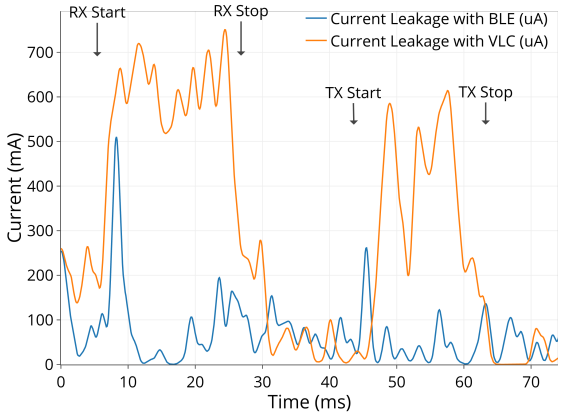

Figure 7.5: The leakage current on the Nexus 5 using BLE (blue) or VLC (orange) for communication relative to the baseline current.

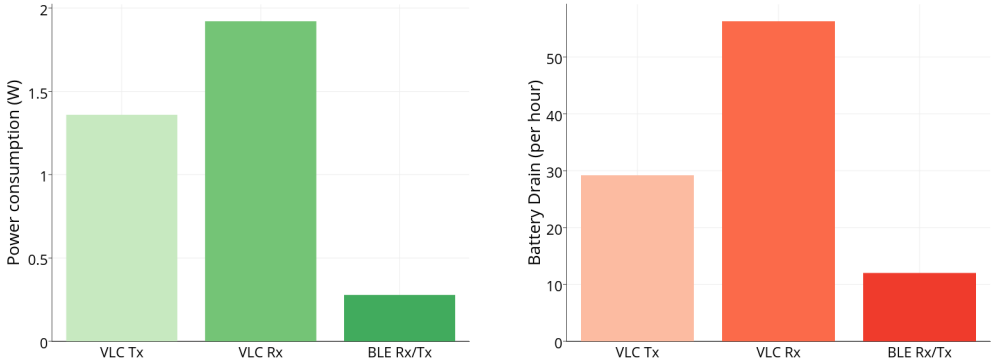

Figure 7.6: Nexus 5 power consumption (left) and battery drain (right) in different VLC and BLE states.

omy for a continuous use of *LightIoT* is less that 2 hours, while it is 9 hours using BLE.

Tab. 7.1 summarizes the *LightIoT* energy consumption per byte of information transmitted. As a consequence of its low data rate, communication using *LightIoT* is overall more energy consuming compared to BLE, especially because of the increased consumption on the smartphone side. However, taking into account that most of the targeted products already keep an LED turned on during functioning, the VLC transmission practically becomes energy-free in this case. Also, most of the energy consumption is moved towards the smartphone, which can be much easily recharged compared with battery-powered objects.

	Smartphone	Wireless Module
LED-to-Camera	9.31340	0.06327
Flash-to-LED	362.4	0.080773
BLE RX/TX.	0.09266867	0.028278

Table 7.1: Energy consumption per byte using VLC and BLE (mJ)

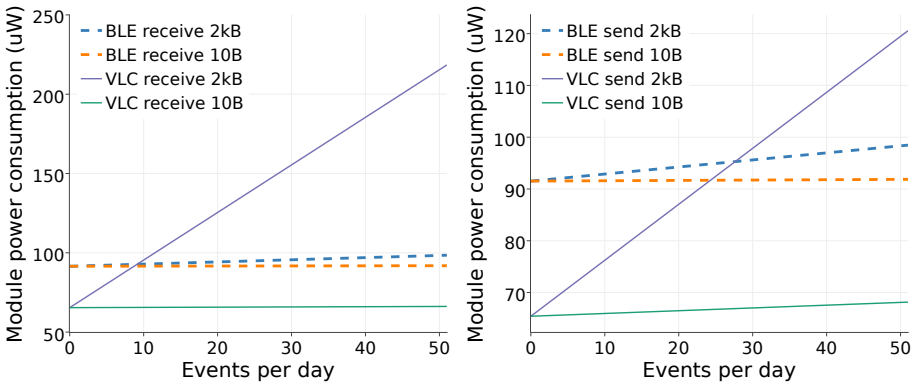

Figure 7.7: Wireless module average power consumption when it is used to receive data from the smartphone (left) or to send (right) for the four scenarios described below.

7.3 Use cases energy balance

We now compare the average VLC module power consumption along the product life with its BLE equivalent for several scenarios and usage profiles.

Scenario 1 assumes the wireless connectivity is used for a firmware Over-The-Air (OTA) update, resulting in a 2 kB file transfer from the smartphone to the object. On the contrary, **Scenario 2** is a 2 kB troubleshooting log file transfer from the object to the smartphone. **Scenario 3** and **Scenario 4** represent much smaller transmissions, respectively a 10 B configuration command from the smartphone to the object or a 10 B sensor value reading from the object to the mobile phone. On the *LightIoT* side, we use the flash-to-LED wake-up feature, meaning that the object consumes energy only for transmission and reception of data. On the BLE side, we consider the object is in peripheral mode, broadcasting an advertising packet every 750 ms, as defined by the Bluetooth 4.0 standard.

In Fig. 7.7, we show the object average power consumption as a function of the number of uses per day, for the two scenarios where the object is a receiver (i.e. scenarios 1 and 3) on the left side, and for the two where the object is the

	Description	Data Size	Direction
Scenario 1	firmware OTA update	2 kB	smartphone to object
Scenario 2	log file transfer	2 kB	object to smartphone
Scenario 3	configuration command	10 B	smartphone to object
Scenario 4	sensor value reading	10 B	object to smartphone

Table 7.2: Summary of the use cases scenarios.

transmitter (i.e. scenarios 2 and 4) on the right. Note that, for BLE, both plots show identical results, as the consumption in BLE Rx and BLE Tx is the same.

For Scenario 1, the results show that using VLC brings an advantage if the OTA update does not occur more than ten times a day, which is well below the expected firmware update frequency. For the transfer of a log file from the object to the smartphone, the VLC solution consumes less energy than its RF counterpart if we remain under 28 transmissions per day. Finally, for the short message exchanges, the results show that *LightIoT* saves energy even for a high number of events.

7.4 Lessons Learned

Results in this chapter underline that VLC improves the product battery life when the connectivity function is an auxiliary one, by decreasing the average power consumption of the communication module. Of course, this conclusion can be challenged by totally turning off the BLE radio and turning it on only when needed, for instance with a wake-up receiver, or with a power switch. Nonetheless, this would be either impractical or expensive to integrate in daily usage objects.

Chapter 8

Conclusion and Perspectives

8.1 Conclusion

Visible Light Communication is unquestionably a great opportunity for the future of wireless communication. This is particularly true in the context of smart cities, where the number of devices connected to Internet or a local network is continuously and quickly growing. In such cases, leveraging visible light spectrum allows radio communication offloading and thus results in an overall communication capacity and bandwidth increase. We show in Chap. 1 and Chap. 2 that this is the main focus of VLC research nowadays.

On a smaller scale, with the rise of Internet of Things, today consumers expect every object to be smart, allowing wireless interaction with them through their smartphone. Radio solutions already exist, like BLE or WiFi, but they have many drawbacks that make their use sometimes unwanted, and even impossible. Companies such as Rtone have widely observed this fact, and have faced this issue many times during the design of IoT devices with strict customers constrains. Looking for a solution raises the following question: since these IoT devices already integrate LEDs and people already own smartphones with a camera, can we leverage them for communication using VLC? That is this question that has driven this thesis and which we answered with this work. Thus, the main purpose of this thesis focuses on the use of VLC to enable bidirectional communication between small IoT devices and off-the-shelf smartphone.

To reach this goal, we have solved many issues which improve the baseline state of the art. As a matter of fact, we summarize the state of the art through a taxonomy of VLC systems in Chap. 2. We divide previous works into three

different groups according to the receiver they are using: photodiodes, LED or camera.

This study highlights the lack of contributions and solutions for low-power LED-to-camera on one side, and for flashlight-to-LED communication systems on the other side.

We begin our work by enabling low power LED-to-camera communication in Chap. 3. We implement and experimentally evaluate this LED-to-camera VLC system, designed specifically for small LEDs. Our solution exploits the rolling shutter effect of off-the-shelf smartphone cameras and an original decoding algorithm, achieving a throughput of nearly 2 kbit/s in ordinary illumination conditions. Our extensive evaluation studies many performance factors and brings out the high loss rate that is inherent to such channel and the need to add redundancy mechanisms to make the communication robust.

Based on these insights, we model the LED-to-camera communication channel, characterized by an on/off behavior resulting in a close to regular frame error pattern. In Chap. 4, we propose a MMBP model, which allows us to easily study the performance of different message retransmission strategies. Relying on this model, we also implement a simulator for LED-to-camera communication performance evaluation. Then, we use both tools to benchmark two trivial redundancy methods we found in the literature. The similar results between experimental, analytical and simulation approaches prove the accuracy both of our analytical model and our simulator.

To make our system bidirectional, in Chap. 5 we leverage the flashlight of unrooted smartphone and use a small LED as receiver. The LED is exactly the same as those we used as an emitter in Chap. 3. We design smart mechanisms and protocols that cope with the flashlight signal jitter and that detect the flashlight clock rate whatever the smartphone type. With these hardware constraints, our system, *LightIoT*, achieves a throughput of 30 bps, from the phone to the object. With such an asymmetry between the channels, acknowledging every packet reception on the smartphone side does not seem realistic. Instead, the flash can be efficiently used as a feed-back channel by acknowledging the reception of the whole message. Thus, in combination with an erasure code on the emitter side, such as random linear coding, the flash-to-LED channel would significantly improve the system performance and robustness.

Therefore, in Chap. 6, we design and implement *SeedLight*, a pseudo random linear coding scheme especially fitted for LOS LED-to-camera communication. Experimental evaluation highlights that this type of approach increases

the goodput from 1.5 to 2.5 kbit/s with respect to packet retransmission mechanisms.

Finally, many of the small objects we address through *LightIoT* have significant energy constraints. Therefore, in Chap. 7 we compare the energy consumption of *LightIoT* with the one of a BLE module with similar activity. Our results show that using the LED for communication purposes reduces the energy consumption under a normal utilization profile.

8.1.1 Industrial Ouputs

This research work lead to a patent application filed by Rtone and INSA de Lyon with the Institut National de la Propriété Industrielle (INPI) in February 2017 and with the European Patent Office (EPO) in January 2018. The patent application has been approved and will be delivered soon by the French patent office. The EPO is currently processing and reviewing our deposit.

Furthermore, *LightIoT* is exploited commercially by Rtone since December 2017 under the registered trademark *Kiwink*®. Several companies have shown a high interest in the technology and are now developing proof of concept products using *Kiwink*®.

8.2 Short Term Perspective

8.2.1 RGB LEDs as emitter

As a continuation of our work, we propose to study the feasibility of replacing the small monochrome LEDs we used in this thesis with an equivalent RGB LED. In fact, RGB LEDs are often preferred to monochrome LEDs in commercial electronics, since their color can be changed at runtime. The use of RGB LEDs for LED-to-Camera communication may help increase the system throughput or its reliability. After some investigations, we show bellow our first findings and discuss the remaining challenges.

An RGB LED is composed in fact of three LEDs: red, green, and blue, inside one package. Typically an RGB LED has four pins: one common pin that is connected to ground and one for each of the three LEDs. This allows a program to vary both the color and the color intensity level of the RGB LED.

While OOK combined with the Manchester line code is usual for monochrome LEDs [22, 31], Wavelength Division Multiplexing (WDM) and Color-Shift-Keying

(CSK) have been proposed for RGB LEDs respectively by *Liang et al.* [75] and by *Chen et al.* [16]. WDM schemes modulate the three color channels independently with distinct input data. On the contrary, in the CSK scheme, the transmitted bits match specific color coordinates in the color space.

We experimentally found the color we can obtain with our low-cost hardware to establish a suitable CSK modulation scheme and we defined a constellation of symbols. This allowed us to identify the following challenges:

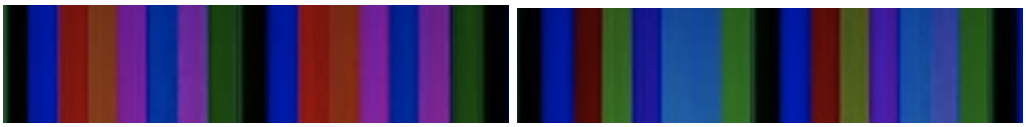

Figure 8.1: The same symbols observed with different ISO and white balance camera settings.

Figure 8.2: The same 8-CSK symbols sent consecutively and repetitively are perceived differently in the picture.

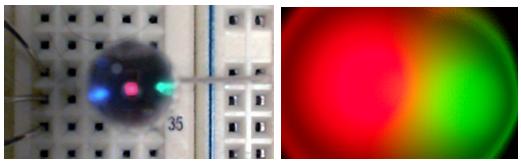

Figure 8.3: The RGB LED without modulation. On the left, the LED is strongly dimmed with the 3 channels ON. On the right, the LED is normally dimmed with only the red and green channels ON.

Limited number of symbols that the RGB LED can produce: the RGB LED can produce only a few colors. Some of these symbols seem to be very close and so, are often difficult to distinguish. The experiments bring out that the green channel state is not easily determinable for the symbols for which the green is mixed with another color.

Color diversity across devices, camera settings and ambient light: various smartphones will render the colors differently and that behavior also happens on the same device when the ambient lighting conditions change or by modifying the ISO, exposure and white balance settings. Fig 8.1 shows the color rendered when changing the ISO setting of the camera.

Intra-frame color diversity: this phenomenon is shown in Fig.8.2, where the symbols are completely disparate across the regions considered in the picture.

Color mixing issue with the cheap low-power RGB LEDs: as shown in Fig. 8.3, even without modulation and with proper use of the RGB LED, the colors are not perfectly mixed so that two or three color beams are visible. However, to exploit the rolling shutter effect, we expect that the color remains the same along a row of pixels when the LED is not modulated.

Real-time implementation: color detection requires complex computer vision algorithms that need a three-dimensional matrix as input. That is three times larger than the one for a monochrome LED. Previous works [57, 16, 105] all present a compute-intensive demodulation method using Matlab or OpenCV on a computer. A real-time algorithm implementation on a smartphone has not been unveiled yet.

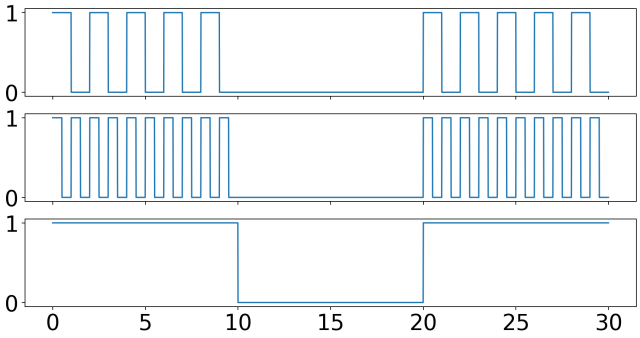

Figure 8.4: On the bottom, the red channel input with the regular f_{csk} CSK modulation. On the middle, the CSK modulation is combined with a f_{pwm} PWM and a $f_{pwm}/2$ PWM on the top, so that dark and light stripes will appear within the CSK symbol period.

Figure 8.5: The 8-CSK modulation symbols combined with a 20 kHz PWM on the green channel only.

In this RGB context, we propose a few methods which we are currently testing to overcome these issues:

Demodulation methods: a demodulation method within the LAB color space would be preferred, as only two vectors (*A* and *B*) describe the chromaticity, instead of all the three for the RGB color space. Because the computer vision algorithms are particularly adapted to the GPU architecture, using the smartphone

GPU can considerably fasten the demodulation process compared to a CPU implementation. This can be done using *Renderscript* [2] framework on Android, and the *Apple Metal 2* [1] framework on iOS.

Use the Pulse Width Modulation (PWM): by modulating a PWM output, configured with a frequency f_{pwm}, higher than the CSK frequency f_{csk}, but below the CMOS sensor sampling frequency, we can induce stripes more or less spaced within a CSK symbol. That method may extend the range of symbols as it is shown in Fig. 8.4.

Constellation choice: we propose to use a combination of M-CSK and N-PWM modulation schemes. Samples of symbols are shown in Fig 8.5. Based on our observations, we found that the M frequencies f^i_{pwm} must be $16 \leq f^i_{pwm} \leq 22$ kHz to comply with the camera shutter bandwidth. To avoid ISI, i.e. to be able to determine the difference on the stripes spacing, two adjacent PWM frequencies f^i_{pwm} and f^{i+1}_{pwm} must be spaced with $\Delta_f \geq 2$ kHz.

Thus, each channel has 6 states, e.g. for the green channel: $G_{f^1_{pwm}}$, $G_{f^2_{pwm}}$, $G_{f^3_{pwm}}$, $G_{f^4_{pwm}}$, G_{ON}, G_{OFF} with $f^i_{pwm} = \{16, 18, 20, 22\}$ kHz, $i = 1, \ldots, 4$. However, we make the hypothesis that choosing symbols where f_{pwm} varies more than one channel will cause unmanageable ISI.

Furthermore, using the PWM modulation on only the green channel will help the color identification process so that the green channel state can be inferred from the presence of stripes.

Until today, we pointed out through a series of experiments the challenges of RGB LED to camera communication and proposed several ideas that we expect to ease the decoding process. These algorithms and methods are left under development. In a future work, we plan to conduct an extensive experimentation campaign to evaluate and (in)validate the hypothesis we presented just before.

8.2.2 Multiple LEDs to Smartphone[s]

We have shown in Chap. 3 that our algorithm can discriminate the LEDs used for VLC, between many LEDs used only to emit light. Also, our algorithm can decode data from several LEDs captured within a same picture, in real time.

Nonetheless, we never took advantage of this property, nor study MIMO communication. Because consumer electronics often have many LEDs, N-LEDs to camera communication is a realistic scenario that allows the following improvement to *LightIoT*:

- increasing the communication system throughput in the N-LEDs-to-smartphone topology, with the benefits of *SeedLight* to smartly combine packets from several emitters.
- sending different information at the same time, e.g. data with localization information or data with signaling.
- enabling multi-user communication in the N-LEDs-to-N-smartphones scenario.

8.3 Open Perspectives

This thesis and our results also open long term research perspectives. In this section, I will introduce two research topics on which I personally would enjoy working toward.

8.3.1 Coexistence of VLC and RF on IoT Devices

In the next generation of wireless communication systems, or 5G, the dense deployment of IoT devices will add an increasing amount of RF congestion. New wireless access techniques are required to maximize the use of the available RF bandwidth and increase aggregate throughput in these environments. Indeed, these RF standards such as 802.11n [94] offer many features (i.e., channel width, number of streams, length of guard interval, and modulation and coding scheme index) that can be selected to improve the communication performances in dense environments. In fact, 128 and 640 combinations exist for 802.11n and 802.11ac respectively [4]. Nonetheless, since neither 802.11n nor 802.11ac have a signaling channel, parameters selection algorithms remains inefficient [4]. Thus, using a low data rate VLC for out-of-band signaling channel for RF parameters optimization can improve the utilization of the RF resources.

Also, IoT devices must be energy efficient and the wireless connection must provide security and privacy. Low data rate VLC can thus serve as a light based wake-up receiver to improve the IoT energy use on the first hand, or the IoT security using light as a side channel for secrets or certificates provisioning on the other hand.

8.3.2 Adaptative Protocols and Mechanisms for Smartphone to IoT Network Communication

Our work allows short range bidirectional communication and interaction between user and consumer electronics. Since this network topology only has two nodes and the communication range is very limited, we never considered nor studied networking scenarios.

Besides, today consumers often have many smart devices at home, and *Schmid et al.* [108] demonstrated that LED-to-LED communication between IoT devices is feasible within a range up to 2m, allowing IoT networking.

Combining our approach with the one proposed in [108] to make a smartphone able to communication with any device within a multi-hop IoT network raises the following questions:

- *How to solve the devices addressing issue?*
- *Can we increase the smartphone-to-device communication range by relaying packets?*
- *How to route packets efficiently?*
- *Since the smartphone-to-device and device-to-device channel characteristics are totally different, how can we automatically adapt them?*
- *What about the security and confidentiality consideration?*

Enabling smartphone to multi-hop IoT network definitely opens many challenges and interesting research opportunities. Evaluating and modeling such a system will be a shining example of that.

To sum up, our future works are mainly related to studies of more complex systems: first by considering different system topologies, i.e. different kind of emitters or devices, and then studying the networking aspects.

References

[1] (2018). Metal | Apple Developer Documentation.

[2] (2018). RenderScript | Android Developers.

[3] (2018). Rtone - IOT Makers.

[4] Abedi, A. and Brecht, T. (2016). Examining Relationships Between 802.11n Physical Layer Transmission Feature Combinations. In *Proceedings of the 19th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems - MSWiM '16*, pages 229–238, New York, New York, USA. ACM Press.

[5] Afgani, M., Haas, H., Elgala, H., and Knipp, D. (2006). Visible light communication using OFDM. In *2nd International Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities, TRIDENTCOM 2006*, volume 2006, pages 129–134.

[6] Aldalbahi, A., Rahaim, M., Khreishah, A., Ayyash, M., Ackerman, R., Basuino, J., Berreta, W., and Little, T. D. (2016). Extending ns3 to simulate visible light communication at network-level. In *2016 23rd International Conference on Telecommunications (ICT)*, pages 1–6. IEEE.

[7] An, C., Li, T., Tian, Z., Campbell, A. T., and Zhou, X. (2015). Visible Light Knows Who You Are. In *Proceedings of the 2nd International Workshop on Visible Light Communications Systems - VLCS '15*, pages 39–44, New York, New York, USA. ACM Press.

[8] Armstrong, J. and Lowery, A. (2006). Power efficient optical OFDM. *Electronics Letters*, 42(6):370–372.

[9] Astudilloy, D., Chaput, E., and Beylot, A.-l. (2014). PRAVDA: Pseudo random network coding in vanet for data download. In *2014 IFIP Wireless Days (WD)*, pages 1–3. IEEE.

[10] Axis Communications (2011). CCD and CMOS sensor technology. Technical report.

[11] Bahl, P., Padmanabhan, V. N., Bahl, V., and Padmanabhan, V. (2000). Radar: An in-building rf-based user location and tracking system. Institute of Electrical and Electronics Engineers, Inc.

[12] Bakker, A. (2014). Comparing Energy Profilers for Android. *21st Twente Student Conference on IT*, 21.

[13] Brouwers, N., Zuniga, M., and Langendoen, K. (2014). NEAT. In *Proceedings of the 12th ACM Conference on Embedded Network Sensor Systems - SenSys '14*, pages 16–30, New York, New York, USA. ACM Press.

[14] Carroll, A. and Heiser, G. (2010). An analysis of power consumption in a smartphone. *Proceedings of the 2010 USENIX conference on USENIX annual technical conference*, pages 21–21.

[15] Chau, Y. and Yu, S.-H. (2001). Space modulation on wireless fading channels. *IEEE Vehicular Technology Conference*, 3(54ND):1668–1671.

[16] Chen, S.-H. and Chow, C.-W. (2014). Color-Shift Keying and Code-Division Multiple-Access Transmission for RGB-LED Visible Light Communications Using Mobile Phone Camera. *IEEE Photonics Journal*, 6(6):1–6.

[17] Chi, N., Wang, Y., Wang, Y., Huang, X., and Lu, X. (2014). Ultra-high-speed single red-green-blue light-emitting diode-based visible light communication systemutilizing advanced modulation formats. *Chin. Opt. Lett.*, 12(1):010605.

[18] Chow, C.-W., Chen, C.-Y., and Chen, S.-H. (2015). Enhancement of Signal Performance in LED Visible Light Communications Using Mobile Phone Camera. *IEEE Photonics Journal*, 7(5):1–7.

[19] Chowdhury, H. and Katz, M. (2014). Cooperative data download on the move in indoor hybrid (radio-optical) WLAN-VLC hotspot coverage. *Transactions on Emerging Telecommunications Technologies*, 25(6):666–677.

[20] Colvin, A. (1983). CSMA with collision avoidance. *Computer Communications*, 6(5):227–235.

[21] Corbellini, G., Aksit, K., Schmid, S., Mangold, S., and Gross, T. R. (2014). Connecting networks of toys and smartphones with visible light communication. *IEEE Communications Magazine*, 52(7):72–78.

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

[23] De Cheveigné, A. and Kawahara, H. (2002). YIN, a fundamental frequency estimator for speech and music. *The Journal of the Acoustical Society of America*, 111(4):1917–30.

[24] Dehghani Soltani, M., Wu, X., Safari, M., and Haas, H. (2016). Access point selection in Li-Fi cellular networks with arbitrary receiver orientation. In *2016 IEEE 27th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, pages 1–6. IEEE.

[25] Di Renzo, M., Haas, H., Ghrayeb, A., Sugiura, S., and Hanzo, L. (2014). Spatial Modulation for Generalized MIMO: Challenges, Opportunities, and Implementation. *Proceedings of the IEEE*, 102(1):56–103.

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

[27] Dissanayake, S. and Armstrong, J. (2013). Comparison of ACO-OFDM, DCO-OFDM and ADO-OFDM in IM/DD systems. *Journal of Lightwave Technology*, 31(7):1063–1072.

[28] Do, T.-h. and Too, M. (2015). Analysis on Visible Light Communication using Rolling Shutter CMOS Sensor. In *International Conference on Information and Communication Technology Convergence (ICTC)*, number 1, pages 755–757. IEEE.

[29] Du, H., Han, J., Jian, X., Jung, T., Bo, C., Wang, Y., and Li, X.-Y. (2017). Martian: Message Broadcast via LED Lights to Heterogeneous Smartphones. *IEEE Journal on Selected Areas in Communications*, 35(5):1154–1162.

[30] Du, W., Liando, J. C., and Li, M. (2016). SoftLight: Adaptive visible light communication over screen-camera links. In *IEEE INFOCOM 2016 - The 35th Annual IEEE International Conference on Computer Communications*, pages 1–9. IEEE.

[31] Duque, A., Stanica, R., Rivano, H., and Desportes, A. (2016). Unleashing the power of LED-to-camera communications for IoT devices. In *Proceedings of the 3rd Workshop on Visible Light Communication Systems - VLCS '16*, pages 55–60, New York, New York, USA. ACM Press.

[32] Elliott, E. O. (1963). Estimates of Error Rates for Codes on Burst-Noise Channels. *Bell System Technical Journal*, 42(5):1977–1997.

[33] Fan, L., Liu, Q., Jiang, C., Xu, H., Hu, J., Luo, D., He, Z., and Huang, Q. (2016). Visible light communication using the flash light LED of the smart phone as a light source and its application in the access control system. In *2016 IEEE MTT-S International Wireless Symposium (IWS)*, pages 1–4. IEEE.

[34] Fath, T. and Haas, H. (2013). Performance comparison of mimo techniques for optical wireless communications in indoor environments. *IEEE Transactions on Communications*, 61(2):733–742.

[35] Fath, T., Haas, H., Di Renzo, M., and Mesleh, R. (2011). Spatial Modulation Applied to Optical Wireless Communications in Indoor LOS Environments. In *2011 IEEE Global Telecommunications Conference - GLOBECOM 2011*, pages 1–5. IEEE.

[36] Ferrandiz-Lahuerta, J., Camps-Mur, D., and Paradells-Aspas, J. (2015). A Reliable Asynchronous Protocol for VLC Communications Based on the Rolling Shutter Effect. In *2015 IEEE Global Communications Conference (GLOBECOM)*, pages 1–6. IEEE.

[37] Fitzek, F., Pedersen, M. V., Heide, J., and M?dard, M. (2010). Network Coding: Applications and Implementations on Mobile Devices. In *Proceedings of the 5th ACM workshop on Performance monitoring and*

measurement of heterogeneous wireless and wired networks - PM2HW2N '10, page 83, New York, New York, USA. ACM Press.

[38] Galal, M. M., El Aziz, A. A., Fayed, H. A., and Aly, M. H. (2014). Employing smartphones Xenon flashlight for mobile payment. In *2014 IEEE 11th International Multi-Conference on Systems, Signals & Devices (SSD14)*, pages 1–5. IEEE.

[39] Galal, M. M., Fayed, H. A., Aziz, A. A. E., and Aly, M. H. (2013a). Smartphones for Payments and Withdrawals Utilizing Embedded LED Flashlight for High Speed Data Transmission. In *2013 Fifth International Conference on Computational Intelligence, Communication Systems and Networks*, pages 63–66. IEEE.

[40] Galal, M. M., Fayed, H. a., Aziz, A. A. E., Aly, M. H., and El Aziz, A. A. (2013b). Smartphones for Payments and Withdrawals Utilizing Embedded LED Flashlight for High Speed Data Transmission. *Computational Intelligence, Communication Systems and Networks (CICSyN), 2013 Fifth International Conference on*, pages 63–66.

[41] Galisteo, A., Juara, D., Wang, Q., and Giustiniano, D. (2018). OpenVLC 1.2 : Achieving Higher Throughput in Low-End Visible Light Communication Networks.

[42] Giustiniano, D., Tippenhauer, N. O., and Mangold, S. (2012). Low-complexity Visible Light Networking with LED-to-LED communication. In *2012 IFIP Wireless Days*, pages 1–8. IEEE.

[43] Google (2018). Android Developers Camera2 API Guide.

[44] Greenan, K. M., Miller, E. L., and Thomas J. E. Schwarz, S. (2008). Optimizing Galois Field Arithmetic for Diverse Processor Architectures and Applications. In *2008 IEEE International Symposium on Modeling, Analysis and Simulation of Computers and Telecommunication Systems*, pages 1–10. IEEE.

[45] Grobe, L., Paraskevopoulos, A., Hilt, J., Schulz, D., Lassak, F., Hartlieb, F., Kottke, C., Jungnickel, V., and Langer, K.-D. (2013). High-speed visible light communication systems. *IEEE Communications Magazine*, 51(12):60–66.

[46] Gu, L. and Stankovic, J. A. (2005). Radio-Triggered Wake-Up for Wireless Sensor Networks. *Real-Time Systems*, 29(2-3):157–182.

[47] Haas, H., Yin, L., Wang, Y., and Chen, C. (2016). What is LiFi? *Journal of Lightwave Technology*, 34(6):1533–1544.

[48] Hao, J., Yang, Y., and Luo, J. (2016). CeilingCast: Energy efficient and location-bound broadcast through LED-camera communication. In *IEEE INFOCOM 2016 - The 35th Annual IEEE International Conference on Computer Communications*, volume 2016-July, pages 1–9. IEEE.

[49] Hao, T., Zhou, R., and Xing, G. (2012). COBRA: color barcode streaming for smartphone systems. *Proceedings of the 10th international conference on Mobile systems, applications, and services - MobiSys '12*, page 85.

[50] Hernández Marcano, N., Sørensen, C., Cabrera G., J., Wunderlich, S., Lucani, D., and Fitzek, F. (2016). On Goodput and Energy Measurements of Network Coding Schemes in the Raspberry Pi. *Electronics*, 5(4):66.

[51] Hesselmann, T., Henze, N., and Boll, S. (2010). FlashLight - Optical Communication Between Mobile Phones and Interactive Tabletops. *ACM International Conference on Interactive Tabletops and Surfaces*, pages 135–138.

[52] Hoque, M. A., Siekkinen, M., Khan, K. N., Xiao, Y., and Tarkoma, S. (2015). Modeling, Profiling, and Debugging the Energy Consumption of Mobile Devices. *ACM Computing Surveys*, 48(3):1–40.

[53] Hou, J. and O'Brien, D. (2006). Vertical handover decision-making algorithm using fuzzy logic for the integrated radio-and-OW system. *IEEE Transactions on Wireless Communications*, 5(1):176–185.

[54] Hranilovic, S., Lampe, L., and Hosur, S. (2013). Visible light communications: the road to standardization and commercialization (Part 1) [Guest Editorial]. *IEEE Communications Magazine*, 51(12):24–25.

[55] Hranilovic, S., Lampe, L., Hosur, S., and Roberts, R. (2014). Visible light communications: the road to standardization and commercialization (Part 2) [Guest Editorial]. *IEEE Communications Magazine*, 52(7):62–63.

[56] Hu, P., Pathak, P. H., Feng, X., Fu, H., and Mohapatra, P. (2015a). ColorBars. In *Proceedings of the 11th ACM Conference on Emerging Networking Experiments and Technologies - CoNEXT '15*, pages 1–13, New York, New York, USA. ACM Press.

[57] Hu, P., Pathak, P. H., Feng, X., Fu, H., and Mohapatra, P. (2015b). ColorBars. In *Proc. 11th ACM Conf. Emerg. Netw. Exp. Technol. - Conex. '15*, pages 1–13, New York, New York, USA. ACM Press.

[58] Hu, W., Gu, H., and Pu, Q. (2013). LightSync. In *Proceedings of the 19th annual international conference on Mobile computing & networking - MobiCom '13*, page 15, New York, New York, USA. ACM Press.

[59] Hu, W., Mao, J., Huang, Z., Xue, Y., She, J., Bian, K., and Shen, G. (2014). Strata: Layered coding for scalable visual communication. *20th ACM Annual International Conference on Mobile Computing and Networking, MobiCom 2014*, pages 79–90.

[60] Huang, Z. and Ji, Y. (2013). Design and demonstration of room division multiplexing-based hybrid VLC network. *Chinese Optics Letters*, 11(6).

[61] IEEE (2015). IEEE 802.15 WPAN 15.7 Amendment Study Group.

[62] IEEE Computer Society (2011). *IEEE Standard for Local and metropolitan area networks - Part 15.7: Short-Range Wireless Optical Communication Using Visible Light*. Number September.

[63] Itseez (2015). Open source computer vision library. https://github.com/itseez/opencv.

[64] Kahn, J. (1997). Wireless infrared communications. *Proceedings of the IEEE*, 85(2):265–298.

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

[66] Kuo, Y.-S., Pannuto, P., Hsiao, K.-J., and Dutta, P. (2014). Luxapose. In *Proceedings of the 20th annual international conference on Mobile computing and networking - MobiCom '14*, pages 447–458, New York, New York, USA. ACM Press.

[67] Kuo, Y.-S., Verma, S., Schmid, T., and Dutta, P. (2010). Hijacking power and bandwidth from the mobile phone's audio interface. *Proceedings of the First ACM Symposium on Computing for Development - ACM DEV '10*, page 1.

[68] Lee, H.-Y., Lin, H.-M., Wei, Y.-L., Wu, H.-I., Tsai, H.-M., and Lin, K. C.-J. (2015). RollingLight. In *Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services - MobiSys '15*, pages 167–180, New York, New York, USA. ACM Press.

[69] Ley-Bosch, C., Medina-Sosa, R., Alonso-González, I., and Sánchez-Rodríguez, D. (2015). Implementing an IEEE802.15.7 Physical Layer Simulation Model with OMNET++. In *Distributed Computing and Artificial Intelligence, 12th International Conference*, volume 373, pages 251–258. Springer Verlag.

[70] Li, L., Hu, P., Peng, C., Shen, G., and Zhao, F. (2014a). Epsilon: A Visible Light Based Positioning System. *11th USENIX Symposium on Network Systems Design and Implementation*, (1):1–13.

[71] Li, T., An, C., Campbell, A., and Zhou, X. (2014b). HiLight. *Proceedings of the 1st ACM MobiCom workshop on Visible light communication systems - VLCS '14*, pages 45–50.

[72] Li, T., An, C., Xiao, X., Campbell, A. T., and Zhou, X. (2015a). Real-Time Screen-Camera Communication Behind Any Scene. In *Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services - MobiSys '15*, pages 197–211, New York, New York, USA. ACM Press.

[73] Li, T., Xiong, X., Xie, Y., Hito, G., Yang, X.-D., and Zhou, X. (2017). Reconstructing Hand Poses Using Visible Light. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 1(3):1–20.

[74] Li, X., Zhang, R., and Hanzo, L. (2015b). Cooperative load balancing in hybrid visible light communications and WiFi. *IEEE Transactions on Communications*, 63(4):1319–1329.

[75] Liang, K., Chow, C.-W., and Liu, Y. (2016). RGB visible light communication using mobile-phone camera and multi-input multi-output. *Optics Express*, 24(9):9383.

[76] LiKamWa, R., Ramirez, D., and Holloway, J. (2014). Styrofoam. In *Proceedings of the 1st ACM MobiCom workshop on Visible light communication systems - VLCS '14*, pages 27–32, New York, New York, USA. ACM Press.

[77] Liu, V., Parks, A., Talla, V., Gollakota, S., Wetherall, D., and Smith, J. (2013). Ambient backscatter: Wireless communication out of thin air. In *Computer Communication Review*, volume 43, pages 39–50.

[78] Liya Yi and Tao Cui (2011). Interference mitigaion between femtocell and macrocell. In *Proceedings of 2011 International Conference on Electronics and Optoelectronics*, volume 2, pages V2–102–V2–104. IEEE.

[79] Luby, M. (2002). LT codes. In *The 43rd Annual IEEE Symposium on Foundations of Computer Science, 2002. Proceedings.*, pages 271–280. IEEE Comput. Soc.

[80] Lucani, D. E., Medard, M., and Stojanovic, M. (2009). Random Linear Network Coding for Time-Division Duplexing: Field Size Considerations. In *GLOBECOM 2009 - 2009 IEEE Global Telecommunications Conference*, pages 1–6. IEEE.

[81] MacKay, D. (2005). Fountain codes. *IEE Proceedings - Communications*, 152(6):1062.

[82] Masao Nakagawa (2007). Visible Light Communications Consortium (VLCC).

[83] Masuda, K., Kamakura, K., and Yamazato, T. (2016). Spatial modulation in layered space-time coding for image-sensor-based visible light communication. In *2016 IEEE 27th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, pages 1–6. IEEE.

[84] Matsumoto, M. and Nishimura, T. (1998). Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator. *ACM Transactions on Modeling and Computer Simulation*, 8(1):3–30.

[85] Mostafa, A. (2017). Physical-layer security for visible-light communication systems. *Physical-layer security for visible-light communication systems*, (April).

[86] Musa, A., Baba, M. D., and Haji Mansor, H. M. A. (2014). The design and implementation of IEEE 802.15.7 module with ns-2 simulator. In *2014 International Conference on Computer, Communications, and Control Technology (I4CT)*, number I4ct, pages 111–115. IEEE.

[87] Nazir, S., Vukobratovic, D., and Stankovic, V. (2011). Performance evaluation of Raptor and Random Linear Codes for H.264/AVC video transmission over DVB-H networks. In *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2328–2331. IEEE.

[88] Nguyen, D., Tran, T., Nguyen, T., and Bose, B. (2009). Wireless broadcast using network coding. *IEEE Transactions on Vehicular Technology*, 58(2):914–925.

[89] Nguyen, D. T. and Park, Y. (2017). Data rate enhancement of optical camera communications by compensating inter-frame gaps. *Optics Communications*, 394:56–61.

[90] Nguyen, T. and Jang, Y. M. (2015). High-speed asynchronous Optical Camera Communication using LED and rolling shutter camera. In *2015 Seventh International Conference on Ubiquitous and Future Networks*, pages 214–219. IEEE.

[91] Omega (2010). Home Gigabit Access (OMEGA).

[92] Otsu, N. (1979). A Threshold Selection Method from Gray-Level Histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(1):62–66.

[93] Ozekici, S. (1997). Markov modulated Bernoulli process. *Mathematical Methods of Operations Research*, 45(3):311–324.

[94] Paul, T. and Ogunfrunmiri, T. (2008). Wireless LAN Comes of Age: Understanding the IEEE 802.11n Amendment. *IEEE Circuits and Systems Magazine*, 8(1):28–54.

[95] Perli, S. D., Ahmed, N., and Katabi, D. (2010). PixNet. In *Proceedings of the sixteenth annual international conference on Mobile computing and networking - MobiCom '10*, page 137, New York, New York, USA. ACM Press.

[96] PureLifim (2018). pureLiFi.

[97] Rahaim, M., Vegni, A., and Little, T. (2011). A hybrid Radio Frequency and broadcast Visible Light Communication system. In *2011 IEEE GLOBECOM Workshops, GC Wkshps 2011*, pages 792–796.

[98] Rajagopal, N., Lazik, P., and Rowe, A. (2014a). Hybrid visible light communication for cameras and low-power embedded devices. In *VLCS '14 Proceedings of the 1st ACM MobiCom workshop on Visible light communication systems*, pages 33–38.

[99] Rajagopal, N., Lazik, P., and Rowe, A. (2014b). Visual light landmarks for mobile devices. In *IPSN-14 Proceedings of the 13th International Symposium on Information Processing in Sensor Networks*, volume 32, pages 249–260. IEEE.

[100] Rajagopal, S., Roberts, R., and Lim, S.-K. (2012). IEEE 802.15.7 visible light communication: modulation schemes and dimming support. *IEEE Communications Magazine*, 50(3):72–82.

[101] Ravi, N., Scott, J., Han, L., and Iftode, L. (2008). Context-aware Battery Management for Mobile Phones. In *2008 Sixth Annual IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pages 224–233. IEEE.

[102] Roberts, R. D. (2013a). A MIMO protocol for camera communications (CamCom) using undersampled frequency shift ON-OFF keying (UFSOOK). *2013 IEEE Globecom Workshops, GC Wkshps 2013*, pages 1052–1057.

[103] Roberts, R. D. (2013b). Undersampled frequency shift ON-OFF keying (UFSOOK) for camera communications (CamCom). In *2013 22nd Wireless and Optical Communication Conference*, pages 645–648. IEEE.

[104] Roberts, R. D., Rajagopal, S., and Lim, S.-K. (2011). IEEE 802.15.7 physical layer summary. In *2011 IEEE GLOBECOM Workshops (GC Wkshps)*, pages 772–776. IEEE.

[105] Ryu, W. J. and Shin, S. Y. (2017). RGB MIMO optical camera communication with Histogram equalization. In *2017 International Conference on Signals and Systems (ICSigSys)*, pages 303–307. IEEE.

[106] Schmid, S., Arquint, L., and Gross, T. R. (2016). Using smartphones as continuous receivers in a visible light communication system. *Proceedings of the 3rd Workshop on Visible Light Communication Systems - VLCS '16*, pages 61–66.

[107] Schmid, S., Corbellini, G., Mangold, S., and Gross, T. R. (2012). An LED-to-LED Visible Light Communication system with software-based synchronization. In *2012 IEEE Globecom Workshops*, pages 1264–1268. IEEE.

[108] Schmid, S., Corbellini, G., Mangold, S., and Gross, T. R. (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*, page 1, New York, New York, USA. ACM Press.

[109] Schmid, S., Schwyn, D., Aks, K., Corbellini, G., Gross, T. R., and Mangold, S. (2014). From Sound to Sight : Using Audio Processing to enable Visible Light Communication. *5th IEEE Workshop on Optical Wireless Communications (OWC'14)*, (Section III).

[110] Shao, S., Khreishah, A., Rahaim, M., Elgala, H., Ayyash, M., Little, T., and Wu, J. (2015). An indoor hybrid WiFi-VLC internet access system. In *Proceedings - 11th IEEE International Conference on Mobile Ad Hoc and Sensor Systems, MASS 2014*, pages 569–574.

[111] Sharif, M. and Sadeghi-Niaraki, A. (2017). Ubiquitous sensor network simulation and emulation environments: A survey. *Journal of Network and Computer Applications*, 93:150–181.

[112] Shokrollahi, A. (2006). Raptor codes. *IEEE Transactions on Information Theory*, 52(6):2551–2567.

[113] Society, I. C. (2003). Ieee standard for information technology - telecommunications and information exchange between systems - local and metropolitan area networks specific requirements part 15.4: Wireless medium access control (mac) and physical layer (phy) specifications for low-rate wireless personal area networks (lr-wpans). *IEEE Std 802.15.4-2003*, pages 1–670.

[114] Sun, S.-T., Cuadros, A., and Beznosov, K. (2015). Android Rooting. In *Proceedings of the 5th Annual ACM CCS Workshop on Security and Privacy in Smartphones and Mobile Devices - SPSM '15*, pages 3–14, New York, New York, USA. ACM Press.

[115] Tagliaferri, D. and Capsoni, C. (2015). Development and testing of an indoor VLC simulator. In *2015 4th International Workshop on Optical Wireless Communications (IWOW)*, pages 122–126. IEEE.

[116] Tanaka, Y., Haruyama, S., and Nakagawa, M. (2000). Wireless optical transmissions with white colored led for wireless home links. 2:1325–1329 vol.2.

[117] Tian, Z., Wright, K., and Zhou, X. (2016). The darkLight rises. In *Proceedings of the 22nd Annual International Conference on Mobile Computing and Networking - MobiCom '16*, pages 2–15, New York, New York, USA. ACM Press.

[118] Tkalcic, M. and Tasic, J. (2003). Colour spaces: perceptual, historical and applicational background. In *The IEEE Region 8 EUROCON 2003. Computer as a Tool.*, volume 1, pages 304–308. IEEE.

[119] Varshney, A., Soleiman, A., Mottola, L., and Voigt, T. (2017). Battery-free Visible Light Sensing. In *Proceedings of the 4th ACM Workshop on Visible Light Communication Systems - VLCS '17*, pages 3–8, New York, New York, USA. ACM Press.

[120] Vegni, A. and Little, T. (2012). Handover in VLC systems with cooperating mobile devices. In *2012 International Conference on Computing, Networking and Communications, ICNC'12*, pages 126–130.

[121] Vučić, J., Kottke, C., Nerreter, S., Habel, K., Buettner, A., Langer, K.-D., and Walewski, J. W. (2010). 230 mbit/s via a wireless visible-light link based on ook modulation of phosphorescent white leds. In *Optical Fiber Communication Conference*, page OThH3. Optical Society of America.

[122] Wang, Q. and Giustiniano, D. (2014). Communication Networks of Visible Light Emitting Diodes with Intra-Frame Bidirectional Transmission. *Proceedings of the 10th ACM International on Conference on emerging Networking Experiments and Technologies - CoNEXT '14*, pages 21–28.

[123] Wang, Q. and Giustiniano, D. (2016). Intra-Frame Bidirectional Transmission in Networks of Visible LEDs. *IEEE/ACM Transactions on Networking*, 24(6):3607–3619.

[124] Wang, Q., Giustiniano, D., and Puccinelli, D. (2014). OpenVLC. In *Proceedings of the 1st ACM MobiCom workshop on Visible light communication systems - VLCS '14*, pages 15–20, New York, New York, USA. ACM Press.

[125] Wang, Q. and Zuniga, M. (2017). Passive Sensing and Communication Using Visible Light: Taxonomy, Challenges and Opportunities.

[126] Wang, Q., Zuniga, M., and Giustiniano, D. (2016). Passive Communication with Ambient Light. In *Proceedings of the 12th International on Conference on emerging Networking EXperiments and Technologies - CoNEXT '16*, pages 97–104, New York, New York, USA. ACM Press.

[127] Wang, Y., Wang, Y., Chi, N., Yu, J., and Shang, H. (2013). Demonstration of 575-mb/s downlink and 225-mb/s uplink bi-directional scm-wdm visible light communication using rgb led and phosphor-based led. *Opt. Express*, 21(1):1203–1208.

[128] Wang, Y., Wu, X., and Haas, H. (2017). Load Balancing Game With Shadowing Effect for Indoor Hybrid LiFi/RF Networks. *IEEE Transactions on Wireless Communications*, 16(4):2366–2378.

[129] Wilson, A. D. and Sarin, R. (2007). BlueTable: Connecting Wireless Mobile Devices on Interactive Surfaces Using Vision-Based Handshaking. *Proceedings of Graphics Interface 2007 - GI '07*, pages 119–125.

[130] Xing, H., Qu, R., Bai, L., and Ji, Y. (2014). On minimizing coding operations in network coding based multicast: an evolutionary algorithm. *Applied Intelligence*, 41(3):820–836.

[131] Yang, C., Wang, Y., Wang, Y., Huang, X., and Chi, N. (2015). Demonstration of high-speed multi-user multi-carrier CDMA visible light communication. *Optics Communications*, 336:269–272.

[132] Yang, Y., Hao, J., and Luo, J. (2017a). CeilingTalk: Lightweight Indoor Broadcast Through LED-Camera Communication. *IEEE Transactions on Mobile Computing*, (April):1–1.

[133] Yang, Y., Hao, J., Luo, J., and Pan, S. J. (2017b). CeilingSee: Device-free occupancy inference through lighting infrastructure based LED sensing. In *2017 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, number March, pages 247–256. IEEE.

[134] Yang, Y., Nie, J., and Luo, J. (2017c). ReflexCode. In *Proceedings of the 23rd Annual International Conference on Mobile Computing and Networking - MobiCom '17*, pages 193–205, New York, New York, USA. ACM Press.

[135] Yang, Y., Nie, J., and Luo, J. (2017d). ReflexCode. In *Proceedings of the 23rd Annual International Conference on Mobile Computing and Networking - MobiCom '17*, pages 193–205, New York, New York, USA. ACM Press.

[136] Yin, S., Smaoui, N., Heydariaan, M., and Gnawali, O. (2018). Purple VLC: Accelerating Visible Light Communication in Room Area through PRU Offloading. In *EWSN '18: Proceedings of the 2018 International Conference on Embedded Wireless Systems and Networks*, pages 67–78, Madrid, Spain.

[137] Yunlu Wang and Haas, H. (2015). Dynamic Load Balancing With Handover in Hybrid Li-Fi and Wi-Fi Networks. *Journal of Lightwave Technology*, 33(22):4671–4682.

[138] Zeng, L., O'Brien, D., Minh, H., Faulkner, G., Lee, K., Jung, D., Oh, Y., and Won, E. (2009). High data rate multiple input multiple output (MIMO) optical wireless communications using white led lighting. *IEEE Journal on Selected Areas in Communications*, 27(9):1654–1662.

[139] Zhang, B., Ren, K., Xing, G., Fu, X., and Wang, C. (2014). SBVLC: Secure barcode-based visible light communication for smartphones. In *IEEE INFOCOM 2014 - IEEE Conference on Computer Communications*, pages 2661–2669. IEEE.

[140] Zhang, C. and Zhang, X. (2016). LiTell. In *Proceedings of the 22nd Annual International Conference on Mobile Computing and Networking - MobiCom '16*, pages 230–242, New York, New York, USA. ACM Press.

[141] Zhang, C. and Zhang, X. (2017). Pulsar : Towards Ubiquitous Visible Light Localization. In *Proceedings of the 23rd Annual International Conference on Mobile Computing and Networking - MobiCom '17*, pages 208–221. ACM Press.

[142] Zhu, S., Zhang, C., and Zhang, X. (2017). Automating Visual Privacy Protection Using a Smart LED. In *Proceedings of the 23rd Annual International Conference on Mobile Computing and Networking - MobiCom '17*, pages 329–342, New York, New York, USA. ACM Press.

[143] Zhu, S. and Zhang, X. (2017). Enabling High-Precision Visible Light Localization in Today's Buildings. In *Proceedings of the 15th Annual International Conference on Mobile Systems, Applications, and Services - MobiSys '17*, pages 96–108, New York, New York, USA. ACM Press.

Appendix A

Controlling the Flashlight of Off-the-Shelf Android Smartphones

Controlling the smartphone flashlight with the Android SDK below 23

```
1  // The session must be already configured and started
2  CameraCaptureSession session;
3  CaptureRequest.Builder builder;
4  public void off() {
5     mBuilder.set(CaptureRequest.FLASH_MODE,
         CameraMetadata.FLASH_MODE_TORCH);
6     mSession.setRepeatingRequest(mBuilder.build(), null,
          null);
7  }
8  public void on() {
9     mBuilder.set(CaptureRequest.FLASH_MODE,
         CameraMetadata.FLASH_MODE_OFF);
10    mSession.setRepeatingRequest(mBuilder.build(), null,
          null);
11 }
```

Listing A.1: Turning the flashlight ON/OFF with the API below 23

Controlling the smartphone flashlight with the Android SDK API 23

```
1  // Flash_API23
2  String mCameraId;
3  CameraManager mCameraManager;
4  public void on() {
5     mCameraManager.setTorchMode(mCameraId, true);
6  }
7  protected void off() {
8     mCameraManager.setTorchMode(mCameraId, false);
9  }
```

Listing A.2: Turning the flashlight ON/OFF with the API 23

FOLIO ADMINISTRATIF

THÈSE DE L'UNIVERSITÉ DE LYON OPÉRÉE AU SEIN DE L'INSA LYON

Nom : **DUQUE** Date de soutenance : **09/10/2018**

Prénoms : **Alexis**

Titre :

BIDIRECTIONAL VISIBLE LIGHT COMMUNICATIONS FOR THE INTERNET OF THINGS

Nature : **DOCTORAT** Numéro d'ordre : **2018LYSEI072**

Ecole doctorale : **InfoMaths**

Spécialité : **Informatique**

RÉSUMÉ :

Dans cette thèse, nous concevons et étudions un système de communication bidirectionnel par lumière visible (VLC) entre une diode électroluminescente (DEL) de couleur, intégrée à un objet connecté, et un smartphone. Le dispositif est ainsi capable d'envoyer et de recevoir des informations à travers sa DEL, tandis que le smartphone utilise sa caméra pour recevoir des données et son flash pour envoyer des informations.

Nous mettons en œuvre et évaluons expérimentalement ce système VLC DEL-à-camera conçu spécifiquement pour les DELs de couleur à faible puissance. En nous appuyant sur les résultats d'une vaste étude expérimentale, nous modélisons, pour la première fois dans la littérature, le canal de communication DEL-à-caméra. Nous proposons alors un modèle de processus de Bernoulli modulé par une chaîne de Markov, qui nous permet d'étudier facilement l'efficacité de différentes stratégies de retransmission des messages. Nous exploitons ce modèle afin de concevoir un simulateur pour l'évaluation des performances des communications DEL-à-caméra.

Afin d'obtenir un système de communication bidirectionnel, nous étudions ensuite les communications de type flash-vers-DEL entre un smartphone non-modifié et une petite DEL de couleur. Les performances, bien que limitées, sont suffisantes pour établir une voie retour qui permet de mettre en oeuvre des mécanismes de fiabilisation.

Nous proposons alors un mécanisme de codage linéaire pseudo-aléatoire, spécialement adapté aux conditions et contraintes du système DEL-à-caméra en ligne de visée directe. Notre évaluation expérimentale souligne que ce type d'approche augmente le rendement jusqu'à deux fois par rapport aux stratégies de retransmission classiques.

Enfin, la plupart des objets que nous adressons ont des contraintes énergétiques importantes. Par conséquent, nous comparons la consommation d'énergie de notre système avec celle d'un module Bluetooth Low Energy avec une activité similaire. Nos résultats montrent que notre système réduit la consommation d'énergie dans le cadre d'un profil d'utilisation classique.

MOTS-CLÉS : **Communication par Lumière Visible, Internet des Objets, Smartphone, Optical Camera Communication, Communications Sans-Fil, Codage Linéaire Aléatoire**

Laboratoire (s) de recherche : **CITI**

Directeur de thèse: **Hervé RIVANO**

Président de jury : **Luc CHASSAGNE**

Composition du jury :

Luc CHASSAGNE

Anne JULIEN-VERGONJANNE

Josep PARADELLS ASPAS

Emmanuel CHAPUT

Valeria LOSCRI

Hervé RIVANO

Razvan STANICA