Celso Daniel Santos Pereira

Comunicações com Câmara e Aprendizagem Automática para Posicionamento por Luz Visível em Espaços Interiores
Optical Camera Communications and Machine Learning for Indoor Visible Light Positioning

Celso Daniel Santos Pereira

Comunicações com Câmara e Aprendizagem Automática para Posicionamento por Luz Visível em Espaços Interioress
Optical Camera Communications and Machine Learning for Indoor Visible Light Positioning

Celso Daniel Santos Pereira

Comunicações com Câmara e Aprendizagem Automática para Posicionamento por Luz Visível em Espaços Interiores
Optical Camera Communications and Machine Learning for Indoor Visible Light Positioning

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia Eletrónica e Telecomunicações, realizada sob a orientação científica do Doutor Pedro Nicolau Faria da Fonseca, Professor Auxiliar do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro, e do Doutor Luís Filipe Mesquita Nero Moreira Alves, Professor Auxiliar do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro.

o júri / the jury

presidente / president

Professor Doutor António José Ribeiro Neves
Professor Auxiliar da Universidade de Aveiro

vogais / examiners committee

Doutor Luís Manuel de Sousa Pessoa
Investigador Sénior do Instituto de Engenharia e Sistemas de Computadores, Tecnologia e Ciência do Porto

Professor Doutor Pedro Nicolau Faria da Fonseca
Professor Auxiliar da Universidade de Aveiro

agradecimentos / acknowledgements

Agradeço ao meu orientador, Professor Pedro Fonseca, pelo incentivo e orientação prestada durante a realização desta dissertação. Quero também agradecer ao Instituto de Telecomunicações e à Universidade de Aveiro pelas condições oferecidas. Por fim, um agradecimento especial aos meus pais e ao meu primo por me terem apoiado nesta jornada.

Palavras Chave

Posicionamento por luz visível, Sistema de posicionamento em espaços interiores, Posicionamento 3D, Sensor de imagem CMOS, Obturador rolante, Transformação projetiva, Aprendizagem automática, Rede neural convolucional, YOLO

Resumo

Nesta dissertação, é apresentado um sistema de posicionamento 3D por luz visível, baseado em aprendizagem automática e comunicações com câmara. O sistema foi desenvolvido para espaços interiores e utiliza luminárias LED (diodo emissor de luz) como pontos de referência e um sensor CMOS (complementary metal-oxide semiconductor) como receptor. As luminárias LED são moduladas utilizando OOK (On–Off Keying) com frequências únicas. O algoritmo YOLOv5 (You Only Look Once version 5) é utilizado para classificar e estimar a posição de cada luminária LED visível na imagem. A posição e orientação do receptor é estimada utilizando um algoritmo de geometria projetiva. O sistema foi validado utilizando um setup em tamanho real com 8 luminárias LED, e obteve um erro de posicionamento médio de $3.5\,\mathrm{cm}$. O tempo médio para obter a posição e orientação da câmara é de aproximadamente $52\,\mathrm{ms}$, o que torna o sistema adequado para posicionamento em tempo real. Tanto quanto sabemos, esta é a primeira aplicação do algoritmo YOLOv5 para localização por luz visível em espaços interiores.

Keywords Visible light positioning, Indoor positioning system, Three-dimensional positioning, CMOS image sensor, Rolling shutter, Perspective-n-Point, Machine Learning, Convolutional Neural Network, YOLO

Abstract In this dissertation, a 3D indoor visible light positioning system based on machine learning and optical camera communications is presented. The system uses LED (light-emitting diode) luminaires as reference points and a rolling shutter complementary metal-oxide semiconductor (CMOS) sensor as the receiver. The LED luminaires are modulated using On-Off Keying (OOK) with unique frequencies. You Only Look Once version 5 (YOLOv5) is used for classification and estimation of the position of each visible LED luminaire in the image. The pose of the receiver is estimated using a perspective-n-point (PnP) problem algorithm. The system is validated using a real-world sized setup containing eight LED luminaires, and achieved an average positioning error of $3.5\,\mathrm{cm}$. The average time to compute the camera pose is approximately $52\,\mathrm{ms}$, which makes it suitable for real-time positioning. To the best of our knowledge, this is the first application of the YOLOv5 algorithm in the field of VLP for indoor environments.

Contents

List of Figures

List of Tables

Chapter 1

Introduction and Motivation

This chapter provides a background to the performed work. It presents an introduction to the topic, and describes the reasons for exploring it. The structure of the dissertation is described at the end.

Nowadays, in the era of the Internet of Things (IoT) and robotics, many applications need precise location and real-time tracking in indoor environments [1, 2]. The current global positioning system (GPS) is unreliable for indoor spaces, as signal strength and multipath problems require complex algorithms and longer times to successfully track the position [3]. Therefore, the demand for indoor localization services is growing.

There are a variety of indoor positioning solutions, such as WiFi-based, infrared, and Ultra-Wideband (UWB) already well developed [4, 5, 6]. These techniques achieve good positioning accuracy, especially in simulation, but they all have certain limitations in practice that reduce their accuracy to around 15 cm in the case of UWB signals, which is the best of the previously mentioned techniques [7, 8]. For instance, radio frequency (RF) signals are affected by multipath effects in indoor spaces that increase localization errors and are constrained by the available spectrum, which is heavily congested [9]. For comparison, an overview of indoor positioning technologies is provided in Table 1.1 including their accuracy and major limitations. By contrast, the positioning technique based on light emitting diodes (LEDs), visible light positioning (VLP), is getting more attention due to the tremendous advantages of LEDs, such as high bandwidth, high security, long life expectancy, electromagnetic interference free, low implementation cost, and high energy-efficiency [10]. Moreover, VLP has a positioning accuracy that is higher than those non-VLP-based methods due to its precise angular resolution [11, 12]. Since VLP systems do not generate electromagnetic interference, they can be implemented in signal-sensitive areas such as hospitals and airports, unlike RF-based positioning systems [13].

VLP-based indoor positioning systems can be divided into two categories, related to the type of sensor in the receiver: photodiode-based (PD-based) and image sensor-based (IS-based). The PD-based positioning system has been widely studied. However it has several drawbacks, such as high sensitivity to the direction of the light beam and poor performance on dynamic positioning, making the image sensor-based positioning system a promising alternative [14]. Moreover, due to the incorporation of image sensors in various commercial mobile devices, IS-based VLP systems can be used in real-life scenarios without the need for peripherals. It is important to note that IS-based VLP requires a clear line-of-sight between

the LED luminaires and the receivers and does not work when the LED luminaires are off. For reference, Table 1.2 gives a summary of the distinctions between a VLP system using a photodiode and a camera as receivers.

The deployment of IS-based VLP systems does not involve many infrastructure changes. If the indoor environment already has LED illumination, it is only necessary to add a specific driver to control each LED luminaire.

In recent years, machine learning (ML) has become a powerful tool and has been used in various research fields, solving classification and regression problems, such as, data mining and pattern recognition. As stated in [15], ML has a high potential to contribute to the development of flexible and robust vision algorithms. Furthermore, learning-based vision systems are expected to provide a higher level of competence and greater generality. Recently, ML-based techniques have been adopted in optical camera communications (OCC) in applications such as image classification and feature identification [16, 17]. Deep learning has also provided some methods for the OCC systems, namely to detect and recognise the LED luminaires [18].

In this dissertation, we present an indoor 3D VLP system based on the YOLOv5 algorithm to enable accurate real-time 3D positioning. Firstly, LED luminaires are modulated using On-Off Keying (OOK) with unique frequencies to simplify their recognition. After that, the image acquired by the CMOS sensor in the receiver goes through YOLOv5, and the classification and position of each visible LED luminaire in the image are acquired. Employing the pinhole camera model, we estimate the camera pose in 3D space using three OpenCV algorithms. This problem is usually formulated as finding the extrinsic parameters of a calibrated camera, given a set of 2D-to-3D point correspondences. Furthermore, and since any ML-based system is heavily dependent on datasets, an automated procedure for collecting and labeling images is presented in order to create a labeled dataset for VLP. The goal of this work is to implement and demonstrate a novel and complete VLP system with six degrees of freedom (DOF) using ML and optical camera communications (OCC), validating 3 of the 6 DOF in a room with typical dimensions. This system can be applied in a large number of scenarios, for example to manage a fleet of multiple mobile IoT devices in a hospital, to provide information to visitors in a museum based on their current location or for the shopping center to offer advertising to its customers based on their location.

The rest of this dissertation is organized as follows. The state of the art and past contributions are detailed in chapter 2. In chapter 3, the system architecture and methodology are explained. The implementation of the VLP system is shown in chapter 4. Finally, chapter 5 presents the experimental results and discussion, followed by the conclusions and future work.

Technology	Accuracy	UC	IC	Weaknesses
Infrared	57 cm - 2.3 m	H	L	Sunlight interference
Ultrasonic	1 cm - 2 m	H	H	Cost and interference
Audible sound	Meters	L	L	Low precision
Wi-Fi	1.5 m	L	L	Vulnerable to access point changes
Bluetooth	30 cm - meters	L	L	Intrusive; Needs signal mapping
ZigBee	25 cm	L	H	Low precision; User needs special equip.
RFID	1 - 5 m	H	L	Very low precision
UWB	15 cm	H	H	High cost

Table 1.1: Comparison of main indoor positioning technologies. UC: end user cost; IC: installation and maintenance cost; H: high; L: low [19].

Receiver	Photodiode	Camera
Interference	High	Low
SNR	Low	High
MIMO Multiplexing	Easy to implement	Difficult to implement
Decoding (complexity)	Signal processing (low)	Image processing (high)
Data rate	High	Low
Range	Near	Far

Table 1.2: Comparison between photodiode-based and camera-based communication systems [7].

Chapter 2

State of the art and Past Contributions

This chapter describes the state of the art and presents previous contributions to camera-based VLP systems.

Over the past few years, several contributions regarding camera-based VLP systems have been proposed in the literature. For instance, in [20], a 3D camera-based positioning system was proposed using visible light sources as beacons and a smartphone camera as the receiver, taking advantage of the rolling shutter effect. According to the authors, the beacons were modulated using On-Off Keying (OOK) and Manchester encoding, transmitting an unique identifier (ID). An image processing algorithm was proposed for the ID decoding and the fixture localization. The authors stated that they obtained an average position error of 7.4 cm using a setup with 1 m long, 1 m wide and with 2.7 m high. In [21], a 2D positioning system using a smartphone camera and two LED luminaires was proposed. According to the authors, Multiple Frequency-Shift Keying (MFSK) modulation was used to transmit the coordinate information of each LED luminaire. The authors performed image processing to detect the LED-IDs and obtained an average position error of 3.25 cm for the flat situation and 5.1 cm for the 10-degree tilt of the smartphone. In another work [22], according to the authors, an indoor VLP system based on the Robot Operating System (ROS) was presented for the first time. They used the OOK-PWM modulation method to differentiate the LED-IDs, thus introducing different features into the image captured by the rolling shutter CMOS sensor. The proposed VLP algorithm consisted of three parts: the dynamic tracking algorithm for LED-ROI, the LED-ID detection scheme, and the positioning algorithm using 2 LEDs. The authors implemented a prototype system on a Turtlebot3 Robot [23]. The experimental results presented by the authors show that the proposed system was able to provide robot indoor positioning accuracy within 1cm and an average computational time of 80 ms.

Machine learning (ML) approaches are normally divided into three categories, supervised learning, unsupervised learning and reinforcement learning. Supervised learning is defined by the use of labeled datasets to train artificial intelligence (AI) algorithms that can classify data (classification) or accurately predict the results (regression). Unsupervised learning refers to the use of AI algorithms to identify patterns in datasets containing data points that are neither classified nor labelled. Reinforcement learning is an AI algorithm based on rewarding desired behaviors and/or punishing undesired ones. For these algorithms, it is important

to collect representative data of good quality and in large quantity. It is also important to avoid overfitting or underfitting the data. As a reference, supervised learning algorithms are used in this dissertation. In [24], a VLP system was introduced using a hierarchical k-means clustering approach and two LED luminaires. The experimental results showed that the average accuracy of 31 cm was achieved for a room of 4.3×4×4 m^3. According to the authors, this was the first publication to use ML for VLP.

In [16], a ML-based indoor positioning system was introduced that takes advantage of the rolling shutter effect of the smartphone's CMOS image sensor. To recognise the LED-IDs, the authors used Linear Support Vector Machine (SVM). Furthermore, an image processing method was used for the image features extraction and selection. In addition, they used the smartphone's built-in fusion sensors to improve the LED-ID detection and positioning accuracy. According to the authors, the system was able to provide an accuracy of 2.49, 4.63, 8.46 and 12.20 cm with angles of 0, 5, 10, and 15°, respectively, within a 2×2×2 m^3 positioning space. Finally, it was also indicated that the total image processing/classification time was 17.36 ms.

Deep learning (DL) is one of the ML techniques that outperform traditional methods in various applications especially when a large amount of data is available [25]. A Convolutional Neural Network (CNN) is a DL algorithm and was first introduced to the field of visible light communication (VLC) in [26], according to the authors. In this paper, an online-to-offline (O2O) method based on VLC was proposed, which implemented, instead of the traditional modulation and demodulation, modulation and recognition. The authors used RGB light-emitting diode (RGB-LED) as the transmitter, and used Pulse Width Modulation (PWM) to modulate the signal to make it flicker at high frequency. At the receiver, the CMOS image sensor was used in the system to capture LED images with stripes. A CNN was introduced in the proposed system as a classifier. According to the authors, this scheme greatly simplifies the complexity of the VLC system.

In 2020, just days after the release of YOLOv4, YOLOv5 was launched by Glenn Jocher, founder and CEO of Utralytics [27] and, so far, is the latest product of YOLO. YOLOv5 is a family of object detection architectures and models pre-trained on the COCO (Common Objects in Context) dataset [28]. The object detection in YOLO is done as a regression problem and provides the class probabilities of the detected bounding boxes. It is available in four models, namely s, m, l and x, each one of them offering different levels of detection accuracy and performance. Larger models like YOLOv5x can, in some cases, achieve better results, but have more parameters, require more GPU memory to train and are slower to run. For reference, YOLOv5s has 7.2 Million parameters and YOLOv5x has 86.7 Million parameters. A model with more parameters is more prone to overfitting.

The network architecture of YOLOv5 is shown in Figure 2.1. The architecture is composed of three main parts: backbone, neck and output. The backbone part extracts feature information from the input images, the neck part aggregates the extracted feature information and generates three scales of feature maps, and the final part detects/classifies the objects and outputs the results. As stated in [29], the backbone part is a convolutional neural network which extracts feature maps of different sizes from the input image by multiple convolution and pooling. As can be seen in Figure 2.1, there are four layers of feature maps generated in the backbone network whose sizes are: $\frac{Image\ width}{4} \times \frac{Image\ height}{4}$ pixels, $\frac{Image\ width}{8} \times \frac{Image\ height}{8}$ pixels, $\frac{Image\ width}{16} \times \frac{Image\ height}{16}$ pixels and $\frac{Image\ width}{32} \times \frac{Image\ height}{32}$ pixels. Having these feature maps of different sizes, the neck part merges the feature maps

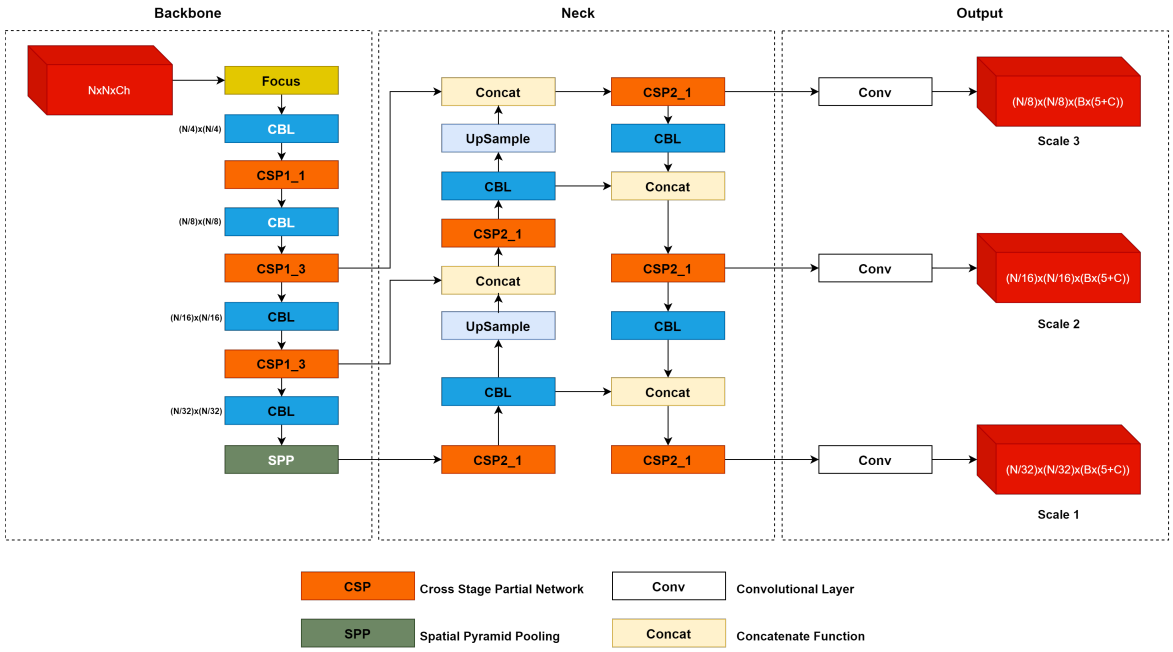

Figure 2.1: Network architecture of YOLOv5 [29].

of different levels to reduce information loss and get more contextual information. In the merging process, the feature pyramid structures of the feature pyramid network (FPN) [30] and the path aggregation network (PAN) [31] are used. As mentioned in [32], the FPN structure transmits strong semantic features from the higher feature maps to the lower feature maps. At the same time, the PAN structure transmits strong localization features from lower feature maps to the higher feature maps, which increases the object localization accuracy. Similarly to YOLOv3, YOLOv5 outputs predictions at three different scales, which are given by downsampling the dimensions of the input image by 32, 16 and 8 respectively. Thus, the shape of each detection feature map is given by the following expressions:

$$\begin{aligned} &\left(\frac{N}{32} \times \frac{N}{32} \times [B \times (5+C)]\right) \\ &\left(\frac{N}{16} \times \frac{N}{16} \times [B \times (5+C)]\right) \\ &\left(\frac{N}{8} \times \frac{N}{8} \times [B \times (5+C)]\right) \end{aligned} \tag{2.1}$$

In this expressions, B is the number of bounding boxes a cell on the feature map can predict, "5" is for the 4 bounding box attributes and one object confidence and C is the number of classes. In YOLOv5 trained on COCO, B is equal to 3, using 3 anchors, and C is equal to 80. As stated in [33], anchor boxes are a set of predefined bounding boxes of a certain height and width. These boxes are defined to capture the scale and aspect ratio of the objects to be detected and are usually chosen based on the dimensions of the objects in the training dataset. If we have an image of 1632×1632 pixels, the detection feature maps would be $(51 \times 51 \times 255,\ 102 \times 102 \times 255,\ 204 \times 204 \times 255)$. This technique enables multi-scale object detection and classification, allowing the model to handle small, medium and large objects.

Multi-scale detection ensures that the model can follow the size changes of the LED patterns in the image. A detailed illustration of this process is presented in Figure 2.2.

In the architecture, the focus module (seen in Figure 2.1) slices the image and fully extracts the features to retain more information, the goal of which is to reduce the amount of model calculations and speed up model training [34]. The CBL module is the smallest component in the YOLO network structure and consists of the modules of convolution, batch normalization and Leaky_relu activation function [32]. As mentioned in [29], there are two types of cross-stage partial network (CSP) in YOLOv5, one used in the backbone part and the other in the neck. The CSP network in the backbone consists of several residual units, while the CSP network in the neck replaces the residual units with CBL modules. Being the residual units a block that uses a feedfoward technique to reduce the vanishing gradient problem. The CSP network is designed to improve the inference speed while maintaining precision by reducing the size of the model. The spatial pyramid pooling (SPP) module performs maximum pooling with different kernel sizes and merges the features by concatenating them. Pooling mimics the human visual system by performing downsampling operations to represent image features at a higher level of abstraction. It makes the feature map smaller and simplifies the computational complexity of the network by extracting the main features. The Concat module represents the tensor concatenation operation.

Finally, YOLOv5 applies Non-Maximum Suppression (NMS) to discard all bounding boxes that have a low probability (lower than the confidence threshold) or that contain the same object as other bounding boxes with higher confidence scores. By choosing a threshold value, NMS discards all overlapping bounding boxes that have an Intersection over Union (IoU) value greater than the threshold value.

Recently, the aforementioned object detection architectures have been used in multiple applications, namely to supervise human behaviour. In [36], a safety helmet monitoring system based on YOLOv5 was proposed. According to the authors, 6045 annotated images were used to train the four models. In each image, each head without a helmet was annotated as "Alarm" and each head with a correctly positioned helmet was annotated as "Helmet". In the end, YOLOv5s achieved an average detection speed of 110 FPS and a mean average precision (mAP) of 93.6% and YOLOv5x achieved an average detection speed of 21 FPS and a mAP of 94.7%, being respectively the fastest model and the most precise. Another example is presented in [37], in which a deep learning algorithm based on YOLOv5s was proposed to replace manual inspection of mask wearing in the fight against the COVID-19 pandemic. According to the authors, the algorithm was trained on 7,959 annotated images and obtained an experimental success rate of about 97.9%. More similar to our work, in [18], a LED detection and recognition system based on Deep Learning was proposed for Vehicle to Vehicle (V2V) communications. It uses a newly developed network based on the YOLOv5 object detection model to determine the position and status of the LED in the vehicle OCC system. The performance of the developed network was compared with other state of the art methods such as YOLOv3, YOLOv4, YOLOv5s, YOLOv5m, YOLOv5l and YOLOv5x. The results were quite acceptable (average precision higher than 71.28%) which demonstrates the potential of these methods for LED identification and recognition. According to the authors, the whole system achieved a processing frame rate of 36 FPS.

In 2017, at the Maxima Medisch Centrum, an autonomous indoor drone, developed by Blue Jay, was demonstrated using VLC technology from Philips Lighting to navigate the indoor space [38]. The autonomous drone is able to determine its position in the indoor space in real time, allowing it to play a game of tic-tac-toe with children and collect and deliver

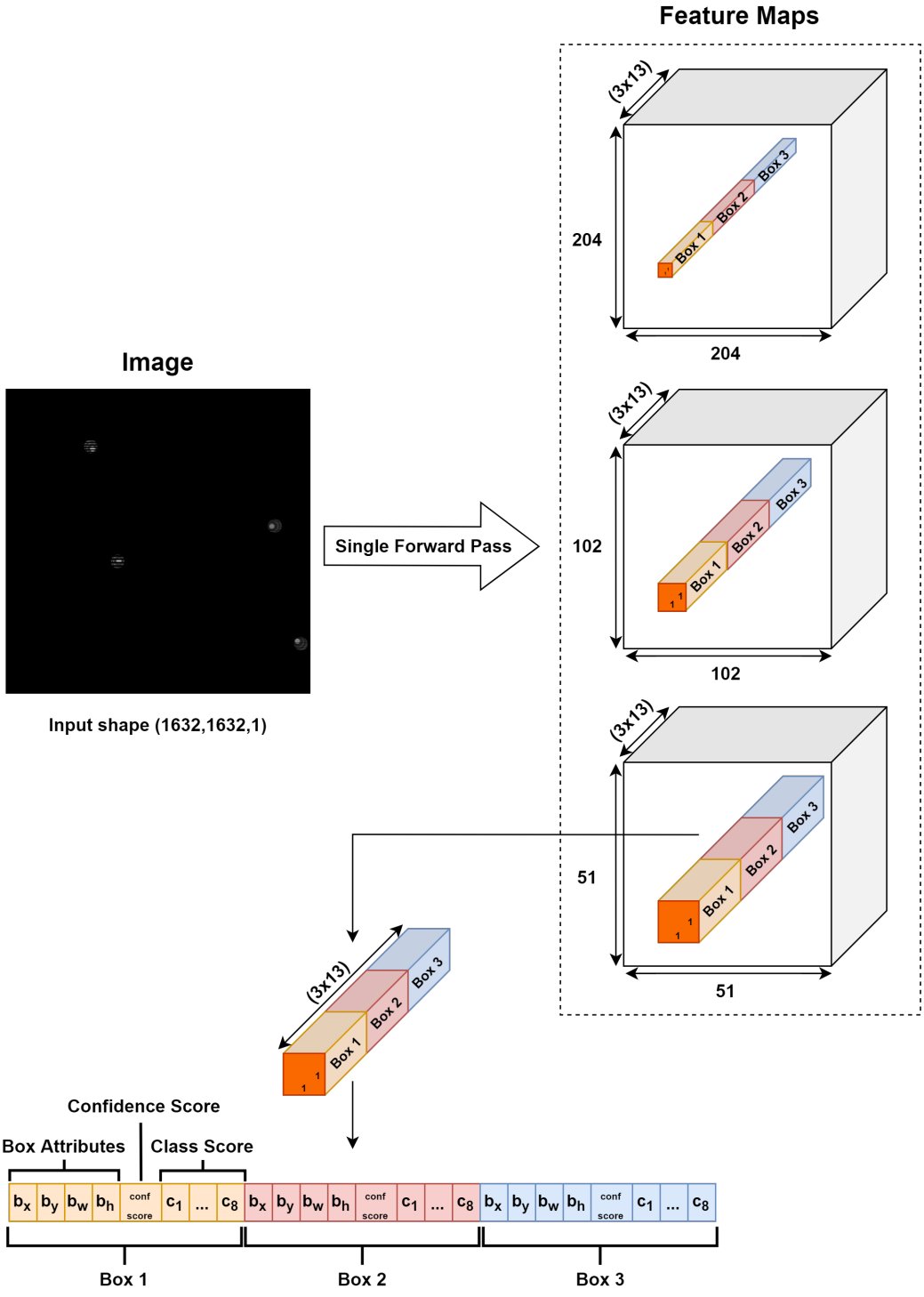

Figure 2.2: Illustration of the basic principle of YOLOv5 [35].

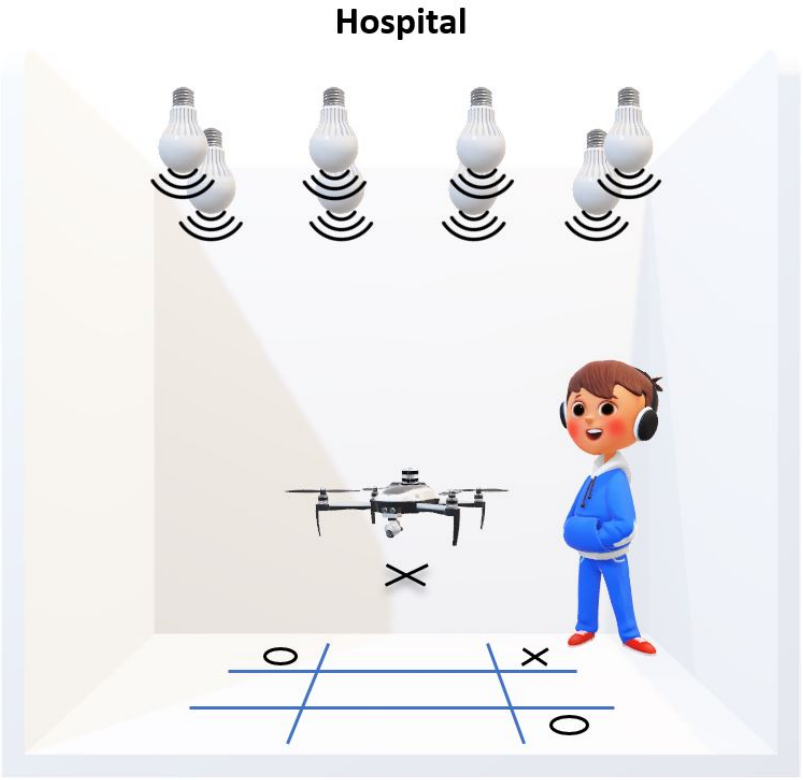

Figure 2.3: Representation of the autonomous indoor drone developed by Blue Jay.

items to a location to help the less mobile. According to the authors, the project aims to help in healthcare by assisting caregivers. An illustration of the innovative project is shown in the Figure 2.3.

Also in 2017, Philips Lighting installed the first indoor-positioned supermarket in Germany [39]. This system joins Bluetooth Low Energy (BLE) as well as smartphone sensor-based positioning with VLC technology. According to the authors, Philips Lighting introduced a new smartphone app that gives shoppers access to location-based services, such as finding items in the shop with an accuracy of 30 cm. The team that developed this system indicates that they are convinced that this system is future-proof, especially for larger shops, and is not only targeted at the younger clientele, as customers without smartphones can also benefit from the service, as staff can use the app to search for goods more quickly and reliably. A similar technology has also been implemented in a supermarket in Dubai in the United Arab Emirates [40].

2.1 Summary

This chapter introduced the results presented in past contributions for camera-based VLP systems to serve as a comparison for the results obtained in this work. Furthermore, the YOLOv5 architecture was detailed to explain how the state-of-the-art algorithm works. Finally, two implementations of VLP systems in real-life scenarios were also analysed.

Chapter 3

System Architecture and Methodology

This chapter focuses on the system architecture and the methodology for developing the complete system. It also presents the working principles of a camera from a geometrical point of view, emphasising the mathematical expressions behind it.

The proposed system is designed to be used in an indoor environment (e.g. at an airport, hospital or shopping center) with illumination LEDs on the ceiling and a mobile device with a camera and positioning needs. By using the LED luminaires as emitters and the mobile device as a receiver, the system provides the receiver's position and orientation in the 3D space. An illustration of the system usage environment is presented in Figure 3.1. The black box in the illustration can represent a robot, a smartphone or any other mobile device that has a camera as an optical receiver.

Figure 3.2 presents the overview of the positioning system architecture. It consists of two main parts, the emitter side and the receiver side, separated by the optical channel.

On the emitter side, each LED luminaire is modulated with OOK at a unique frequency, high enough that the human eye cannot detect any fluctuation in the light intensity – essentially flicker-free. Thus, each LED luminaire has a distinctive characteristic. Generally, 200 Hz is accepted as the minimum frequency to avoid this phenomenon in VLC systems [41]. Since we are interested in developing a positioning system that also serves for general-purpose illumination, the luminous intensity of the various LED luminaires should, ideally, be constant and equal between them.

On the receiver side, a Tagged Image File Format (TIFF) image is acquired with the CMOS sensor exposure time set to the minimum possible value and the maximum resolution, in order to maximize the detectable blinking frequency of the LED luminaires. Assuming a constant frame rate, a higher image resolution means a higher data rate. After the image passes through the 3D VLP algorithm, the position and orientation of the receiver are estimated.

The 3D VLP algorithm is mainly composed of two processes: LED-ID recognition and the positioning algorithm. The LED-ID recognition is performed by taking advantage of the rolling shutter mechanism of the CMOS image sensor. YOLOv5 is used to classify the pattern of each LED luminaire and obtain their corresponding coordinates in the image. A bounding box is generated around each detected LED pattern, which allows the central

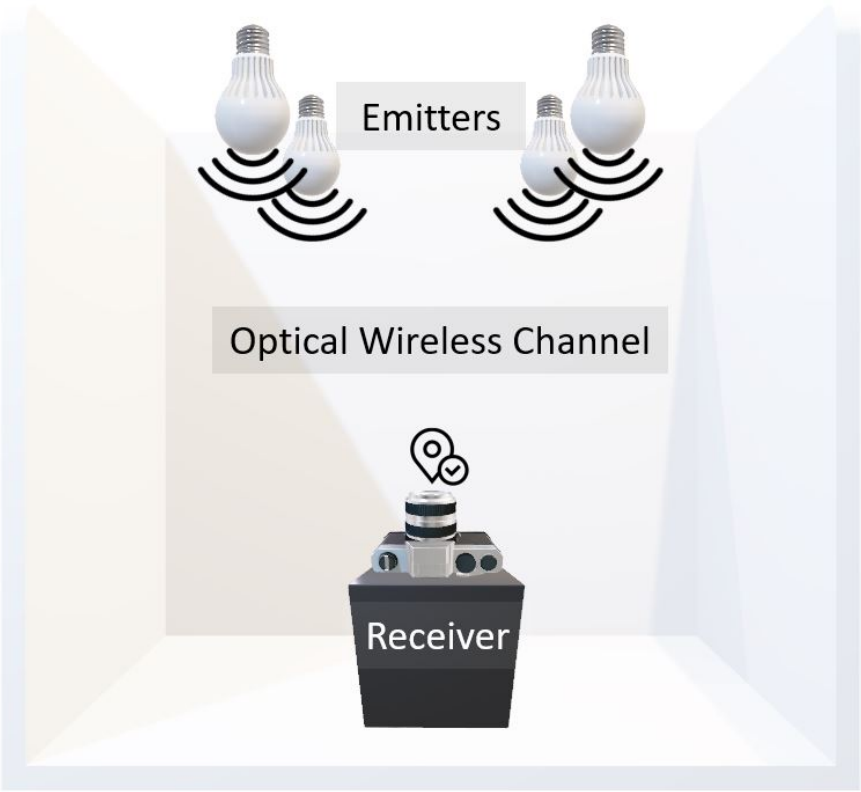

Figure 3.1: System usage environment.

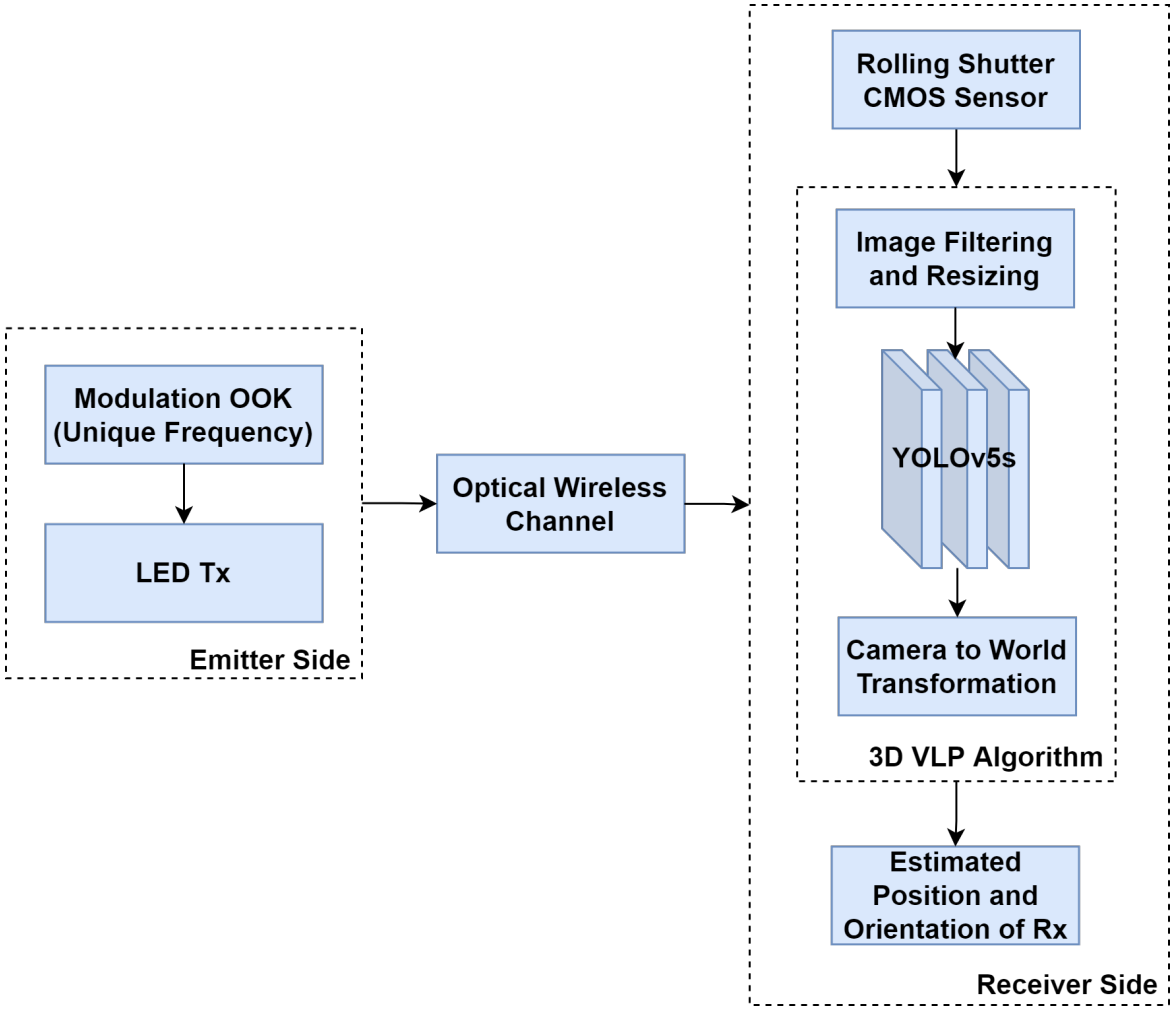

Figure 3.2: Block diagram of the VLP system.

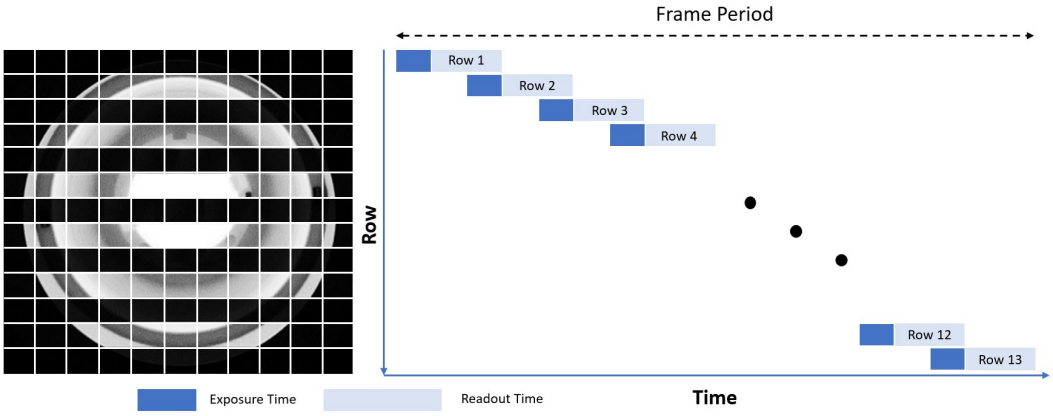

Figure 3.3: Diagram demonstrating the time delay between each row of pixels in a rolling shutter CMOS image sensor.

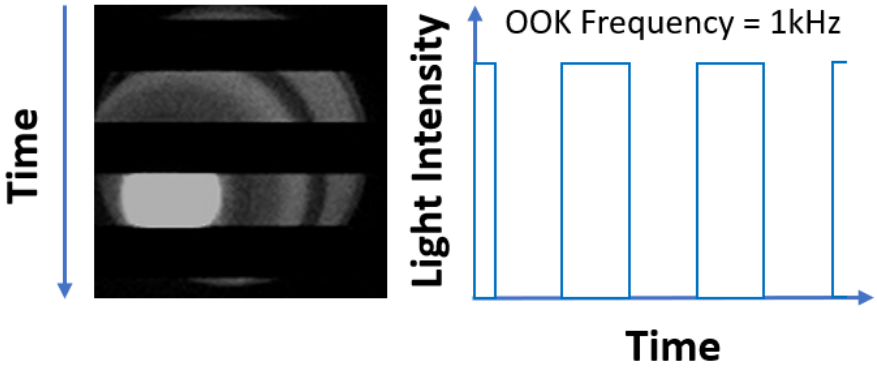

Figure 3.4: Illustration of the rolling shutter effect.

position of each pattern in the image to be obtained. YOLOv5 has state-of-the-art results in several benchmarks, such as COCO [42], and due to the image only going through the CNN once, this process is considerably faster than solutions such as Region Based Convolutional Neural Network (R-CNN) [43]. Therefore, the algorithm is expected to be fast and robust in identifying the LED-IDs. Since this framework provides different models with different numbers of network parameters and FLOPs (Floating Point Operation Per Second), model s was selected in order to minimise the probability of overfitting the data.

For a rolling shutter CMOS image sensor, the data reading is performed row by row. Figure 3.3 shows a diagram demonstrating the time delay between each row of pixels in this sensor. Thus, the captured image displays bright and dark stripes while the LED luminaire is turning on and off during the period of exposure. Changing the distance of the CMOS sensor from the LED luminaires does not affect the width of the stripes, because the scanning frequency of the CMOS sensor is fixed. However, the area of the LED luminaires in the image will decrease as the distance between the LED luminaires and the camera increases, and this will result in a reduction in the number of stripes. Increasing the modulation frequency causes a decrease in the width of the stripes. An illustration of this phenomenon is shown in Figure 3.4.

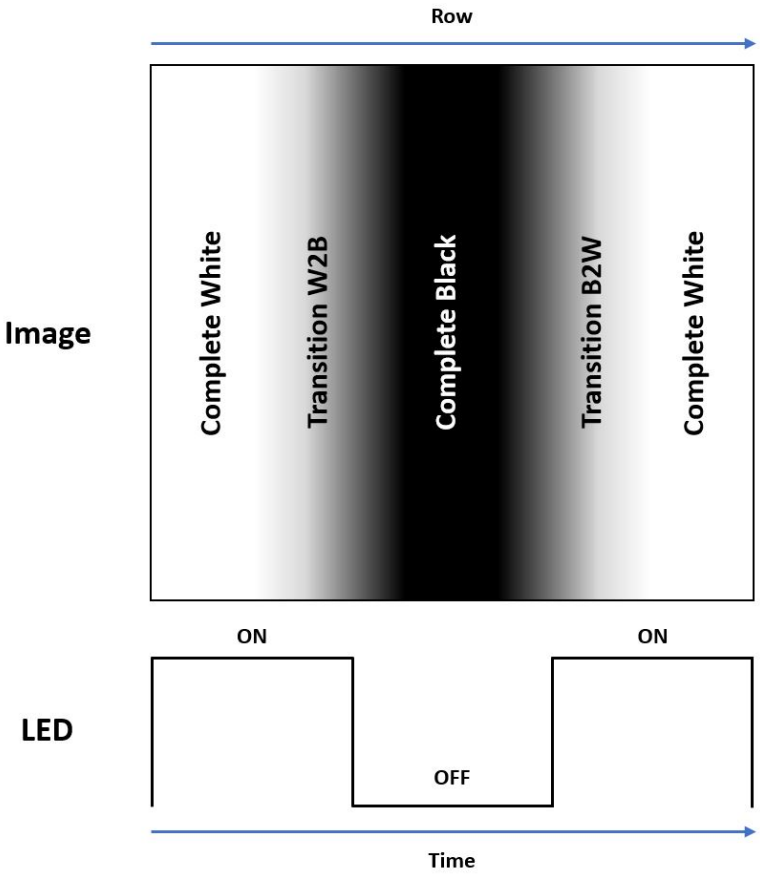

Figure 3.5: Demonstration of the transition between stripes.

Depending on the exposure and readout times, white to black and black to white transition bands appear in the image. These transition bands, illustrated in Figure 3.5, can be considered as intersymbol interference (ISI) [44]. The height of the transition bands (in pixels) is calculated using the following expression [45]:

$$h_{trans} = t_{exp} \times \frac{h_{im}}{t_{fr}} = \frac{t_{exp}}{t_{rr}} \tag{3.1}$$

where t_{exp} is the exposure time, h_{im} is the height of the image, t_{fr} is the frame readout time and t_{rr} is the row readout time.

In a VLC system, the transition bands should be as narrow as possible. The transitions increase when the exposure time increases or when the row readout time decreases.

Regarding the positioning algorithm, the camera pose is made up of 6 degrees of freedom (DOF) (x, y, z, roll, pitch and yaw). Therefore, to obtain the receiver's position and orientation, it is necessary to get information of at least three image points (2D) and their corresponding world coordinates (3D). In our setup, the 3D points correspond to the world coordinates of the center of each visible LED luminaire in the image, and the 2D points correspond to the image coordinates of the center of each LED projection. It is also necessary to have the intrinsic matrix and the distortion coefficients of the camera. These parameters are obtained after performing the camera calibration. Thus, and although it is not evidenced

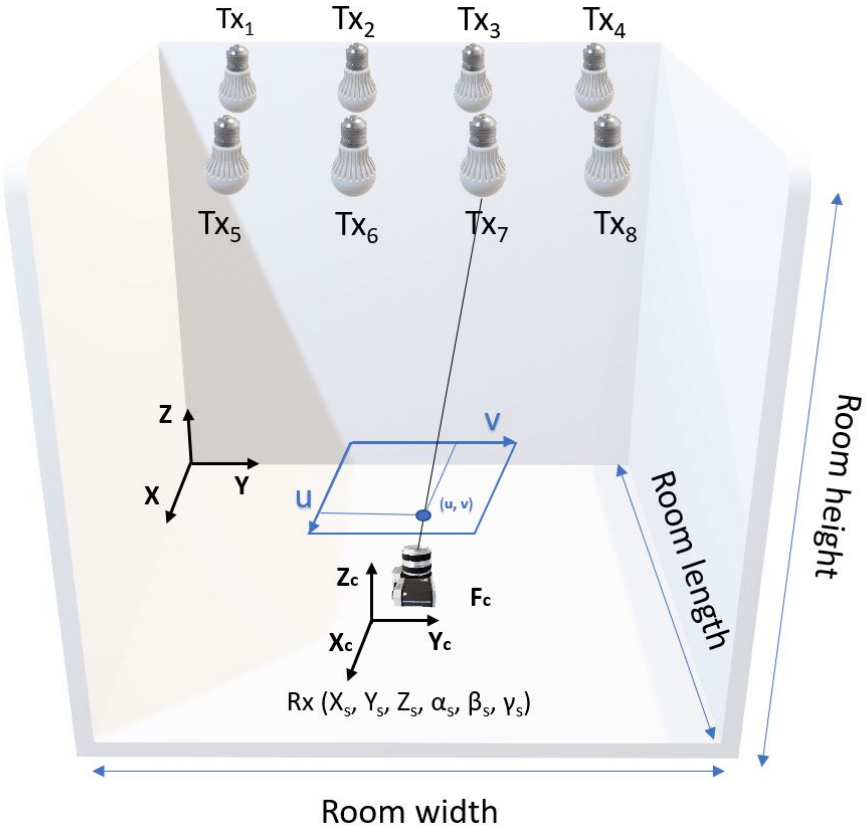

Figure 3.6: Example of the geometrical setup diagram of the VLP system.

in diagram 3.2, it is assumed that the system knows: the camera parameters, the LED luminaires coordinates in the world coordinate system (WCS) and the association between the frequency modulated by each LED luminaire and the corresponding coordinates. Subsection 3.1 presents the working principles of a camera and the mathematical expressions behind the camera-to-world transformation. To solve the Perspective-n-Point (PnP) problem, there are already several algorithms implemented in the literature.

The image filtering and resizing block has been added since we are not interested in color information and do not need to differentiate between a large number of LED luminaires for the system demonstration.

Figure 3.6 shows an example of the geometrical setup diagram of the indoor VLP system, in this case composed of 8 emitters (i.e., LED luminaires) and one receiver (i.e., a rolling shutter CMOS sensor) positioned on the ceiling and at the floor level. The diagram shows the world coordinate system, the camera coordinate system and the image plane. The k_{th} emitter has a known set of coordinates (x_{Tk}, y_{Tk}, z_{Tk}), which is associated with the WCS. The image plane is parallel to the X_c and Y_c axes and the camera's viewing direction (optical axis) is aligned with the Z_c axis. The main goal is to obtain accurately and in real time the position and orientation (x_s, y_s, z_s, $\alpha_s, \beta_s, \gamma_s$) of the receiver in the WCS from a single input photograph.

To implement the YOLO-based VLP system it is essential, besides implementing the PnP algorithm, to first train the object detection algorithm. Thus, it is necessary to have a labeled

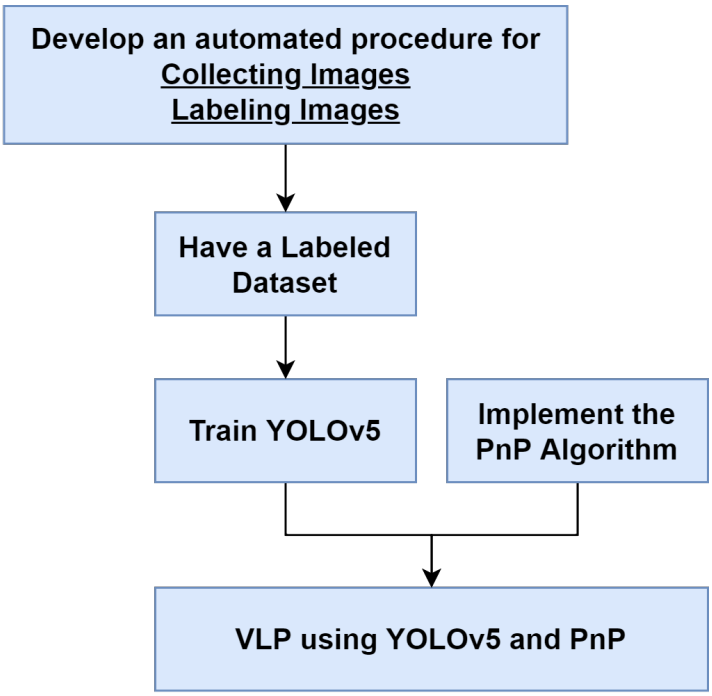

Figure 3.7: Flow diagram to implement the VLP system using YOLOv5 and PnP.

dataset composed of images, similar to the test images, and their annotations. To obtain the labeled dataset quickly and reliably, an automated image acquisition and labeling procedure was developed. This sequence is shown in the block diagram in Figure 3.7.

To have an automated image collection procedure, an image acquisition module was developed and built using 3D printing. This module aims to streamline the process of collecting images for datasets, offering a practical and intuitive graphical interface. To this end, the module operates automatically and autonomously, allows quick adjustments to the camera orientation and is easy to position precisely at a given point in space.

In order to generate the labeled dataset to train YOLOv5 to detect LED patterns, it is necessary to acquire a large number of images similar to the test images, i.e. acquire images with all modulated LED luminaires active, at various points in the indoor environment, and create the annotation file for each image. For this process to be fast and accurate, it is important that it is fully automated. The process of annotating the images consists of delimiting all the visible luminaires in the image with a bounding box, assigning each of them a class label. To perform this task automatically, a ROI algorithm is required to generate the bounding boxes and isolate each visible LED luminaire in the image and a CNN to assign the correct label to each isolated LED luminaire. For this purpose, it is essential to first train the CNN to classify the LED patterns.

In order to generate the dataset to train the CNN, it is necessary to acquire a large number of images with only one modulated LED luminaire active at various points in the indoor environment. In this way, it is possible to be certain of the LED-ID of each set of images. By using a ROI algorithm and grouping the cropped LED images by class, the CNN can learn to differentiate between the various individual LED patterns. This entire sequence can be seen in the block diagram in the Figure 3.8.

The datasets developed in this dissertation can be used to train and validate other algorithms with different characteristics. These datasets can be found at `https://www.kaggle.com/celsopereira1/visible-light-positioning-dataset`.

3.1 Camera-to-World Transformation

A camera is an optical instrument that allows incident light in to capture an image on a light-sensitive surface (image sensor), resulting in a 2D projection of the 3D world in front of it. This 2D-to-3D mathematical relationship can be described, in an ideal scenario, by the pinhole camera model. In this model, the camera aperture is described as a point (infinitely small hole) and no lenses are used to focus the light. The light from the scene passes through the aperture and projects an inverted image on the image plane. The pinhole camera diagram is presented in Figure 3.9.

Given these characteristics, a pinhole camera has an infinite depth of field, since everything in the image is projected in focus. Like a real camera, the pinhole camera has a focal length (F) that corresponds to the distance between the focal center (F_c) and the image plane. The focal length influences the field of view (FOV) and, in a real camera, the amount of perspective distortion present in the image. A shorter focal length gives a wider FOV with more perspective distortion and a longer focal length gives a narrower FOV with less perspective distortion. The geometrical model for the pinhole camera is presented in Figure 3.10.

The camera axis follows the standard camera coordinate system (CCS), with the $\mathrm{X_c}$ axis pointing to the right, the $\mathrm{Y_c}$ axis downwards and the $\mathrm{Z_c}$ axis pointing to the front. An arbitrary point P in 3D space is projected onto the image plane at (u,v), as can be seen in Figure 3.10.

In this subsection, all operations will be represented by matrices, and the multiple points in both 2D and 3D space will be represented in homogeneous coordinates.

In the pinhole camera model, the distortion-free projective transformation that maps 3D points in world coordinates ($\mathrm{X_w}$,$\mathrm{Y_w}$,$\mathrm{Z_w}$) into 2D points in image coordinates (u,v) is given by:

$$s \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \tag{3.2}$$

which is equivalent to:

$$s \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = A \begin{bmatrix} R|t \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \tag{3.3}$$

with

$$A = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{3.4}$$

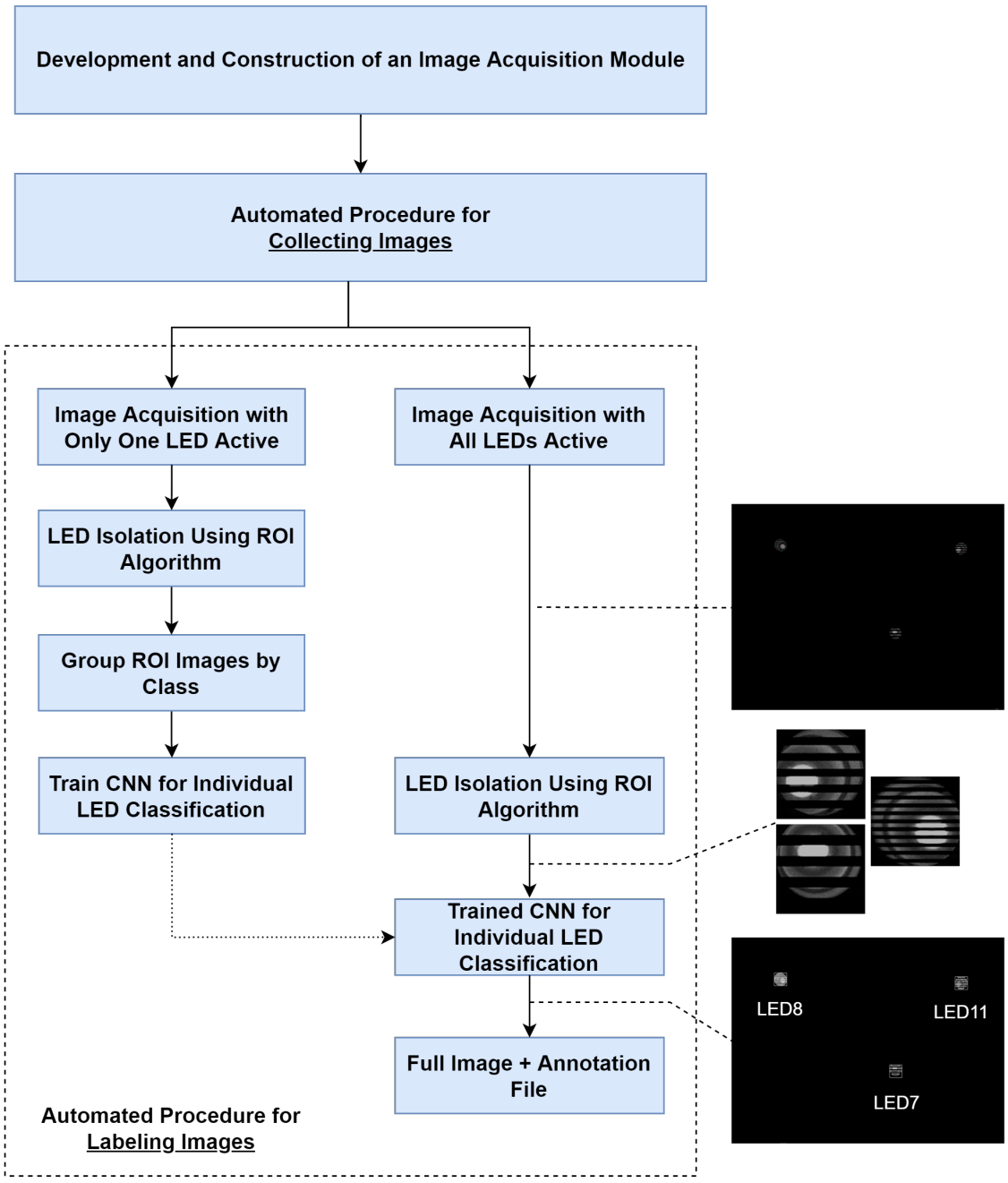

Figure 3.8: Flow diagram to implement an automated procedure for image collection and labeling.

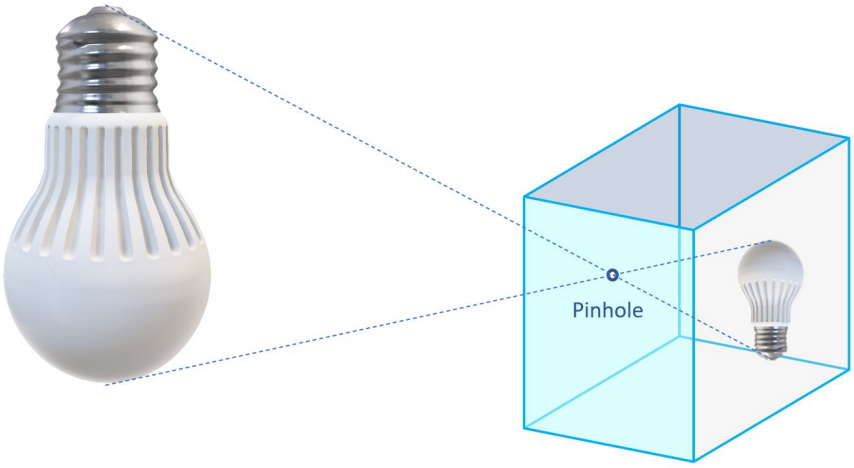

Figure 3.9: Pinhole camera diagram.

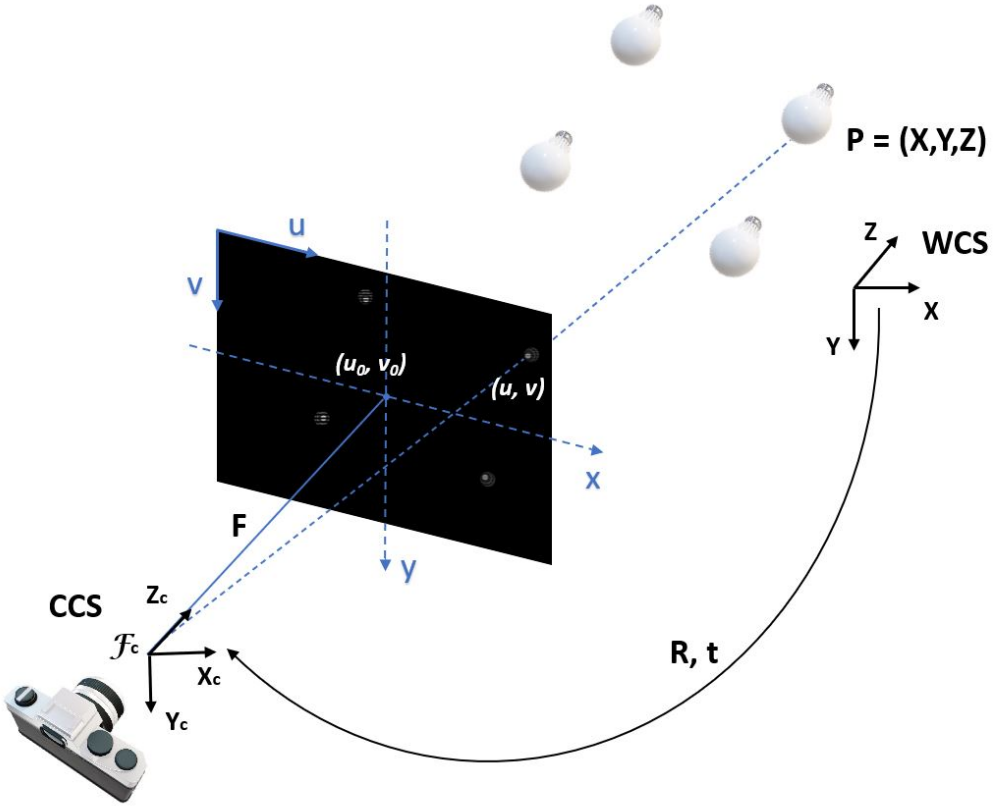

Figure 3.10: Geometrical model for the pinhole camera.

$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \tag{3.5}$$

$$t = \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} \tag{3.6}$$

where s is the projective transformation's arbitrary scaling, A is the camera intrinsic matrix composed of the focal lengths f_x and f_y which are expressed in pixels, and the principal point $(c_x,\ c_y)$, R is the rotation matrix and t is the translation vector. The focal lengths are converted from meters to pixels with the following expressions:

$$f_x = F\frac{N_x}{W} \tag{3.7}$$

$$f_y = F\frac{N_y}{H} \tag{3.8}$$

where F is the focal length, in meters, W and H are the sensor width and height also in meters, and N_x and N_y are the number of pixels in both axes. In this scenario, the translation vector (t_x,t_y,t_z) corresponds directly to the camera's position in the world coordinate system (WCS). To obtain the orientation of the camera, R can be expressed by three matrices representing the rotations around each axis.

$$R = R_z R_y R_x \tag{3.9}$$

$$R_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta_x) & -\sin(\theta_x) \\ 0 & \sin(\theta_x) & \cos(\theta_x) \end{bmatrix} \tag{3.10}$$

$$R_y = \begin{bmatrix} \cos(\theta_y) & 0 & \sin(\theta_y) \\ 0 & 1 & 0 \\ -\sin(\theta_y) & 0 & \cos(\theta_y) \end{bmatrix} \tag{3.11}$$

$$R_z = \begin{bmatrix} \cos(\theta_z) & -\sin(\theta_z) & 0 \\ \sin(\theta_z) & \cos(\theta_z) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.12}$$

The angles $(\theta_x,\ \theta_y,\ \theta_z)$ are the Euler angles and directly represent the camera's orientation in the WCS, in radians.

To portray a real camera, the camera model should incorporate the lens distortions. Thus, the above model is extended and the 2D points in the image plane (u,v) are updated to compensate for these distortions.

$$
\begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} s \Leftrightarrow
$$

$$
\Leftrightarrow \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} X_c/Z_c \\ Y_c/Z_c \end{bmatrix} \Leftrightarrow
$$

$$
\Leftrightarrow \begin{bmatrix} x'' \\ y'' \end{bmatrix} = \begin{bmatrix} x' \frac{1+k_1r^2+k_2r^4+k_3r^6}{1+k_4r^2+k_5r^4+k_6r^6} + 2p_1x'y' + p_2\left(r^2+2x'^2\right) + s_1r^2 + s_2r^4 \\ y' \frac{1+k_1r^2+k_2r^4+k_3r^6}{1+k_4r^2+k_5r^4+k_6r^6} + p_1\left(r^2+2y'^2\right) + 2p_2x'y' + s_3r^2 + s_4r^4 \end{bmatrix}, \; r^2 = x'^2 + y'^2 \Leftrightarrow
$$

$$
\Leftrightarrow \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} f_x x'' + c_x \\ f_y y'' + c_y \end{bmatrix} \tag{3.13}
$$

where $k_1, k_2, k_3, k_4, k_5,\ k_6$ are the radial coefficients, p_1, p_2 are the tangential distortion coefficients and s_1, s_2, s_3, s_4 are the thin prism distortion coefficients.

Then, equation (3.2) is used to obtain the extrinsic camera parameters.

3.2 Summary

This chapter presented the system architecture and the methodology for developing the complete system. Moreover, it presented the rolling shutter effect and emphasised the working principle of a camera from a geometrical point of view. The mathematical process introduced at the end of this chapter is implemented by the PnP algorithm used in this work.

Chapter 4

Implementation

This chapter describes in detail the implementation process of the complete system. The developed software is presented and an explanation for each step is given. It also provides additional information about the emitters and the receiver.

Figure 4.1 shows a photograph and description of the experimental setup, as well as the module developed for automatic image acquisition. The setup consists of eight emitters (LED luminaires) modulated with OOK with frequencies ranging from 1 kHz to 4.5 kHz with a 500 Hz interval. Full details of the transmitters and receiver are given in sections 4.1 and 4.2 respectively.

With its distinctive feature, each LED luminaire presents a different pattern in the image acquired by the receiver. An example of each LED pattern is shown in Figure 4.2.

During the implementation process, the CMOS image sensor was placed parallel to the floor at a height of 25.6 cm. All images were acquired with full resolution (3264×2464 pixels) and with the exposure time set to the minimum possible value. This value was chosen to maximise the detectable blinking frequency of the LED luminaires, which can be calculated using the following expression [45]:

$$h_{comp} = \frac{\frac{1}{f_{led}} - t_{exp}}{t_r} \tag{4.1}$$

where h_{comp} is the the height (in pixels) of the complete white (or black) strip, f_{led} is the blinking frequency of the LED luminaire, t_{r} is the row readout time, and t_{exp} is the exposure time.

Due to the camera characteristics (the minimum t_{exp} is equal to 9 µs and the t_{r} is equal to 18 µs), the maximum detectable blinking frequency is approximately 37 kHz. This range shows an indication that more LED luminaires with unique frequencies could be placed ensuring positioning in larger indoor environments. For this, it is also important to take into consideration the ability to distinguish between close frequencies.

With the presented camera characteristics it is also possible to calculate the height of the transition bands in the image, in pixels. Using the expression presented previously:

$$h_{trans} = \frac{9\mu s}{18\mu s} = 0.5px \tag{4.2}$$

As the result is less than 1, it is considered negligible, and it can be assumed that there is no transition bands for the emitter frequencies used.

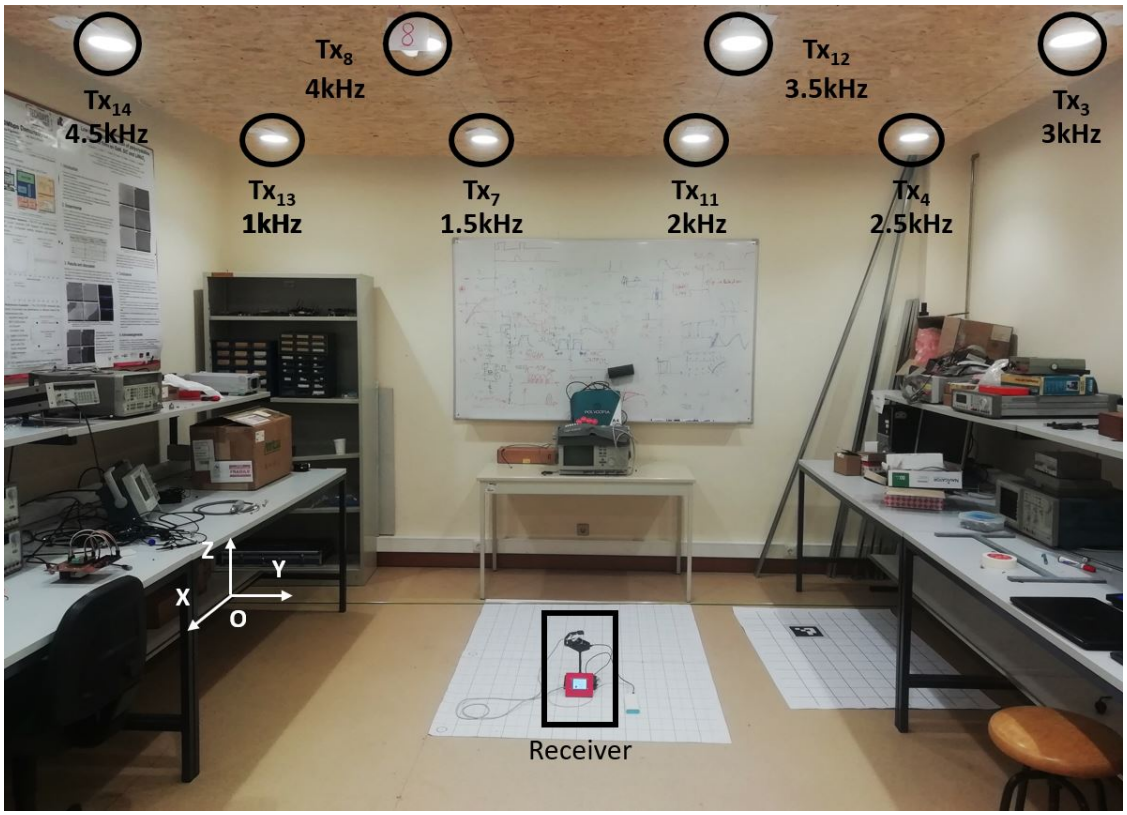

Figure 4.1: Experimental setup.

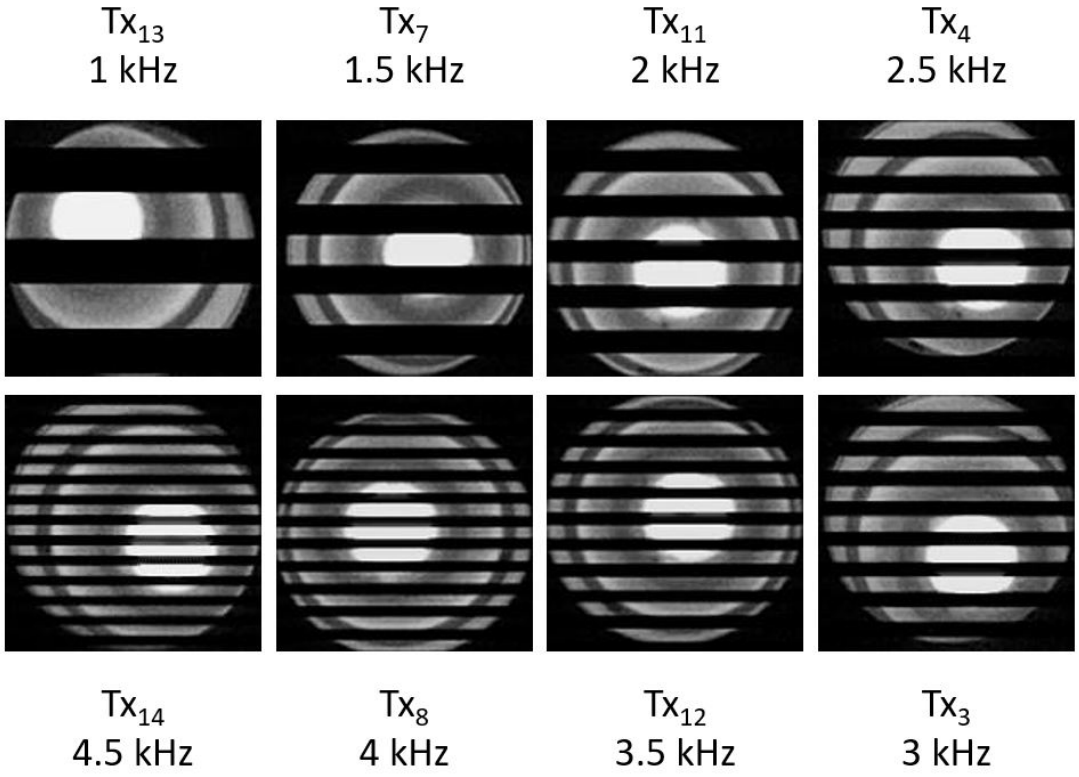

Figure 4.2: Example of the pattern in the image of each modulated LED (in grayscale).

Input image size	150x150
Initial learning rate	0.0001
Optimizer	Adam
Epoch	30
Batch size	64

Table 4.1: The training parameters related to the CNN.

In order to implement the complete system, we started by acquiring the images needed to train the CNN using the image acquisition module. Thus, 2849 images were acquired with only one known LED luminaire active. Then, all images were processed using the developed ROI algorithm, in order to isolate each LED in an image with 150×150 pixels, and grouped by class. In this process, the LED luminaires that are fully visible and the LED luminaires that appear in the image with a size greater than $\frac{1}{3}$ of the total LED are isolated. This processed image dataset is balanced, with approximately 355 images of each LED-ID. All the details of the developed ROI algorithm are presented in section 4.3.

Taking into consideration the nature of the dataset, a CNN was developed with the aim of precisely classifying each LED pattern. CNNs perform well in image classification because, rather than pre-processing the data to derive features such as textures and shapes, CNNs take image's raw pixel data as input and "learn" to extract these features, and ultimately infer what object they constitute [46]. The CNN architecture is described in section 4.4.

To train and test the developed CNN, the dataset composed of the cropped images was augmented to a total of 63876 data-augmented images using horizontal and vertical flip augmentation. The various classes have approximately the same number of images (8000 images). This artificial expansion of the dataset allows to increase the performance and generalizability of the model. The dataset was divided into 80% for training and 20% for testing. The training hyperparameters related to the developed CNN are shown in Table 4. The Adam (Adaptive Moment Estimation) [47] optimizer was used for training. Optimizers are algorithms used to change the attributes of the neural network, such as learning rate and weights, in order to reduce losses and provide the most accurate results possible. The authors of the Adam algorithm indicate that it is computationally efficient, has low memory requirements and is suitable for problems with very noisy/or sparse gradients.

To evaluate the CNN performance quantitatively, we used Precision, Recall and F1-Score. The calculation formulas are given as follows:

$$\text{Precision} = \frac{TP}{TP + FP} \tag{4.3}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{4.4}$$

$$\text{F1} = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4.5}$$

where, for each class, TP (True Positives) is the number of correctly detected LEDs of that same class, FP (False Positives) is the number of LEDs of other classes treated as LEDs of that class, and FN (False Negatives) is the number of LEDs of that class treated as LEDs of the other classes.

	Precision	Recall	F1-Score	Support
LED11	1.0000	1.0000	1.0000	1611
LED12	0.9988	0.9988	0.9988	1648
LED13	0.9960	1.0000	0.9980	1511
LED14	1.0000	1.0000	1.0000	1535
LED3	0.9988	1.0000	0.9994	1615
LED4	1.0000	1.0000	1.0000	1634
LED7	1.0000	0.9957	0.9978	1611
LED8	0.9994	0.9988	0.9991	1611
Accuracy			0.9991	12776
Macro Avg	0.9991	0.9991	0.9991	12776
Weighted Avg	0.9991	0.9991	0.9991	12776

Table 4.2: CNN classification report.

Using the macro-average metric, we obtained: Precision score equal to 99.91%, Recall score equal to 99.91% and F1 score equal to 99.91%. The macro-average computes the metric independently for each class and then takes the average, thus treating all classes equally. The CNN classification report, with the main classification metrics, is presented in Table 4.2. In the weighted-average metric, the contribution of each class to the average is weighted by its size. This metric is most often used when the dataset is unbalanced. The support column is the number of samples for each class in the test dataset.

To check where the model failed, we plotted the normalized confusion matrix seen in Figure 4.3. It compares each predicted class with the associated label. The diagonal is where the model performed accurately. The higher the diagonal values of the confusion matrix the better, indicating many correct predictions. When the classification is incorrect we can see exactly where the model has failed. Looking at the confusion matrix, we can verify that practically all the predicted labels are the true labels. Occasionally, classification errors may occur between LED luminaires with close frequencies, namely between LED13 (1 kHz) and LED7 (1.5 kHz) and between LED8 (4 kHz) and LED12 (3.5 kHz).

With the classification of the LED patterns being done with high precision, 820 images with all LED luminaires active were acquired and converted to grayscale. In each of them, at least 3 LED luminaires are always visible. Using the ROI algorithm and CNN, an annotation file is generated for each image, thus creating the labeled dataset required to train YOLOv5. Note that the annotations were generated in YOLO and PascalVOC formats. The Pascal VOC format generates an XML file for each image, highlighting each bounding box with its class and pixel coordinates (x_{min}, y_{min}, x_{max}, y_{max}). This process is demonstrated in Figure 4.4.

To train and validate the YOLOv5s, the labelled dataset was augmented to a total of 2630 annotated images, applying the same data augmentation technique used previously. For this, the Albumentations library [48] was used to adjust the annotations accordingly. The augmented dataset was divided into 1980 annotated images for training and 650 for validation. The training hyperparameters related to the YOLOv5s are shown in Table 4. The SGD (Stochastic gradient descent) optimizer was used for training. The main advantages of this iterative method are its efficiency and ease of implementation. After training the

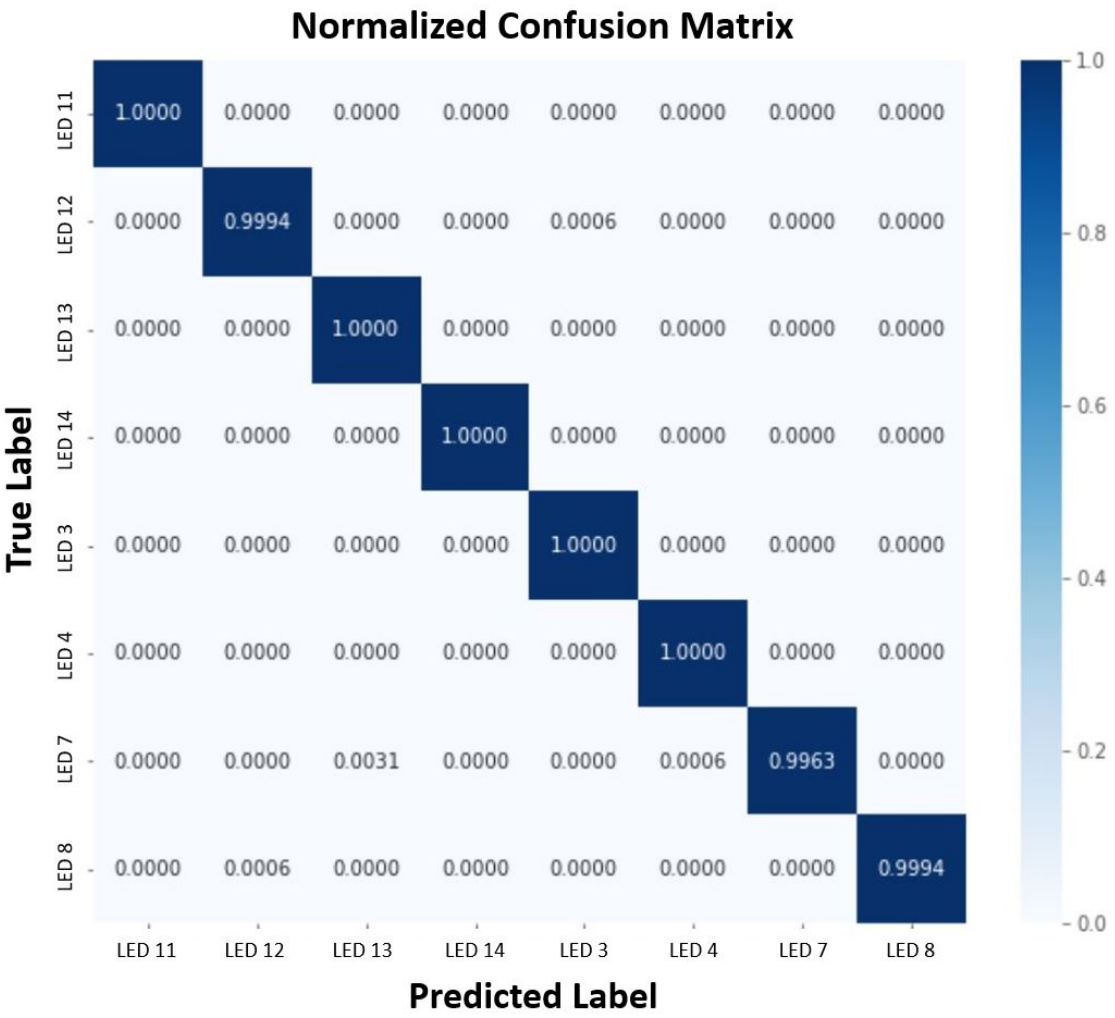

Figure 4.3: CNN confusion matrix.

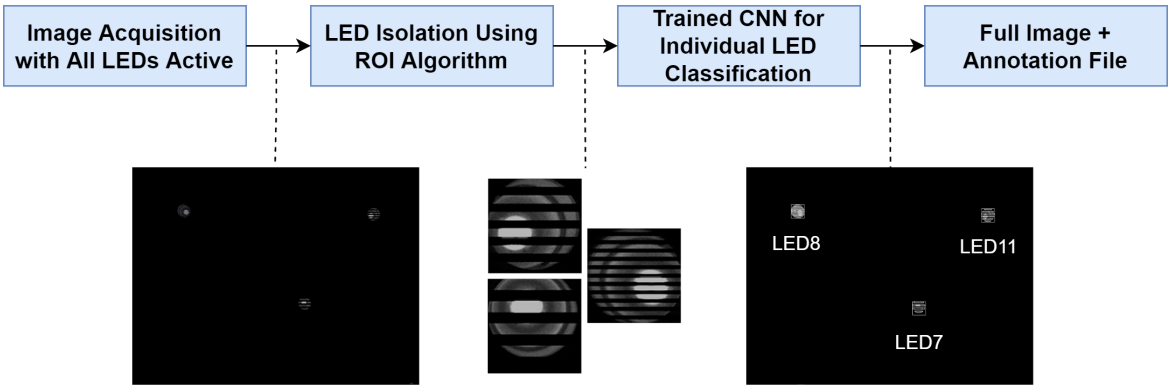

Figure 4.4: Process for generating the image annotations with the CNN already trained (with visual representation).

Input image size	1632x1632
Initial learning rate	0.01
End learning rate	0.2
Optimizer	SGD
Epoch	100
Batch size	2
IoU threshold	0.2
Momentum	0.937

Table 4.3: The training parameters related to the YOLOv5s.

algorithm, the best weights were saved.

To measure the accuracy of object detectors such as YOLO or R-CNN the average precision (AP) metric is typically used. AP computes the area under the precision-recall (PR) curve. The PR curve is a plot of precision as a function of recall. It shows the trade-off between precision and recall for different thresholds [49]. A high area under the curve represents both high recall and high precision, where high recall is related to a low false negative rate, and high precision is related to a low false positive rate. Mathematically, AP is defined as:

$$\text{AP@IoUThreshold} = \int_0^1 p(r)\, dr \tag{4.6}$$

The average of this value, taken over all classes, is called mean Average Precision (mAP).

In our validation dataset, a mAP of 0.992 was obtained for an Intersection over Union (IoU) threshold of 0.5. The fact that this value is close to 1 indicates that the system is quite precise. The PR curve is shown in Figure 4.5.

To check where the YOLOv5s failed, we plotted the normalized confusion matrix seen in Figure 4.6. Observing the confusion matrix, we can check that almost all the predicted labels are the true labels. Eventually, classification errors may occur between LED luminaires with close frequencies, namely between LED13 (1 kHz) and LED7 (1.5 kHz) and between LED3 (3 kHz) and LED4 (2.5 kHz).

With YOLOv5s trained and validated, the next step was to implement the PnP algorithm. For that, the position of the LED luminaires on the ceiling was measured using a tape. The position of the LED luminaires in the room is presented in Table 4.4. The point with coordinates (0,0,0) corresponds to the bottom left corner of the room, seen in Figure 4.1. Furthermore, the calibration of the camera was also performed to obtain the intrinsic parameters and the distortion coefficients. Since we have full control over the imaging process, the checkerboard based method was used to calibrate the camera. The checkerboard patterns are commonly used in calibration because of their distinctiveness and ease of recognition in an image. The corners of the squares on the checkerboard are quite easy to detect, as they show sharp gradients in two directions. The calibration process is presented by a flowchart in Figure 4.7. After applying this calibration method, the following results were obtained:

$$\text{Camera intrinsic matrix: } \begin{bmatrix} 1288.6255 & 0 & 813.2959 \\ 0 & 1290.6448 & 819.7536 \\ 0 & 0 & 1 \end{bmatrix}$$

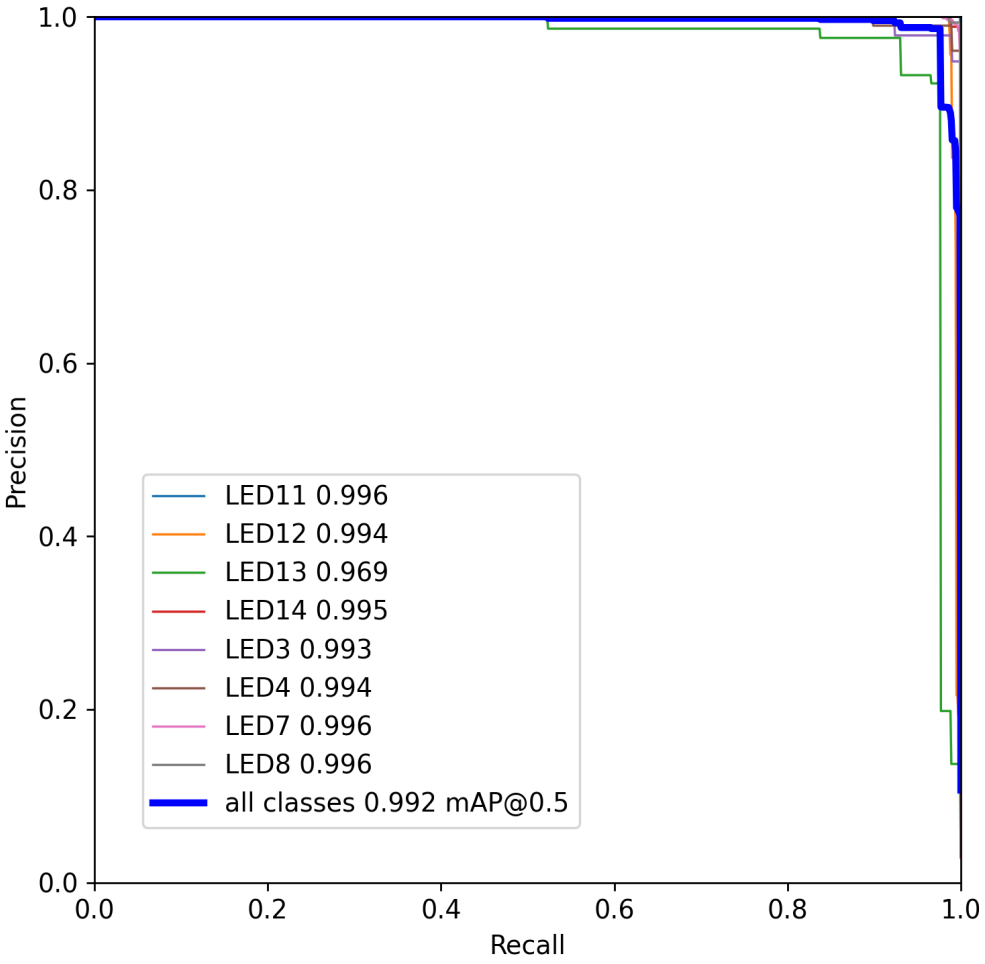

Figure 4.5: YOLOv5s Precision-Recall curve.

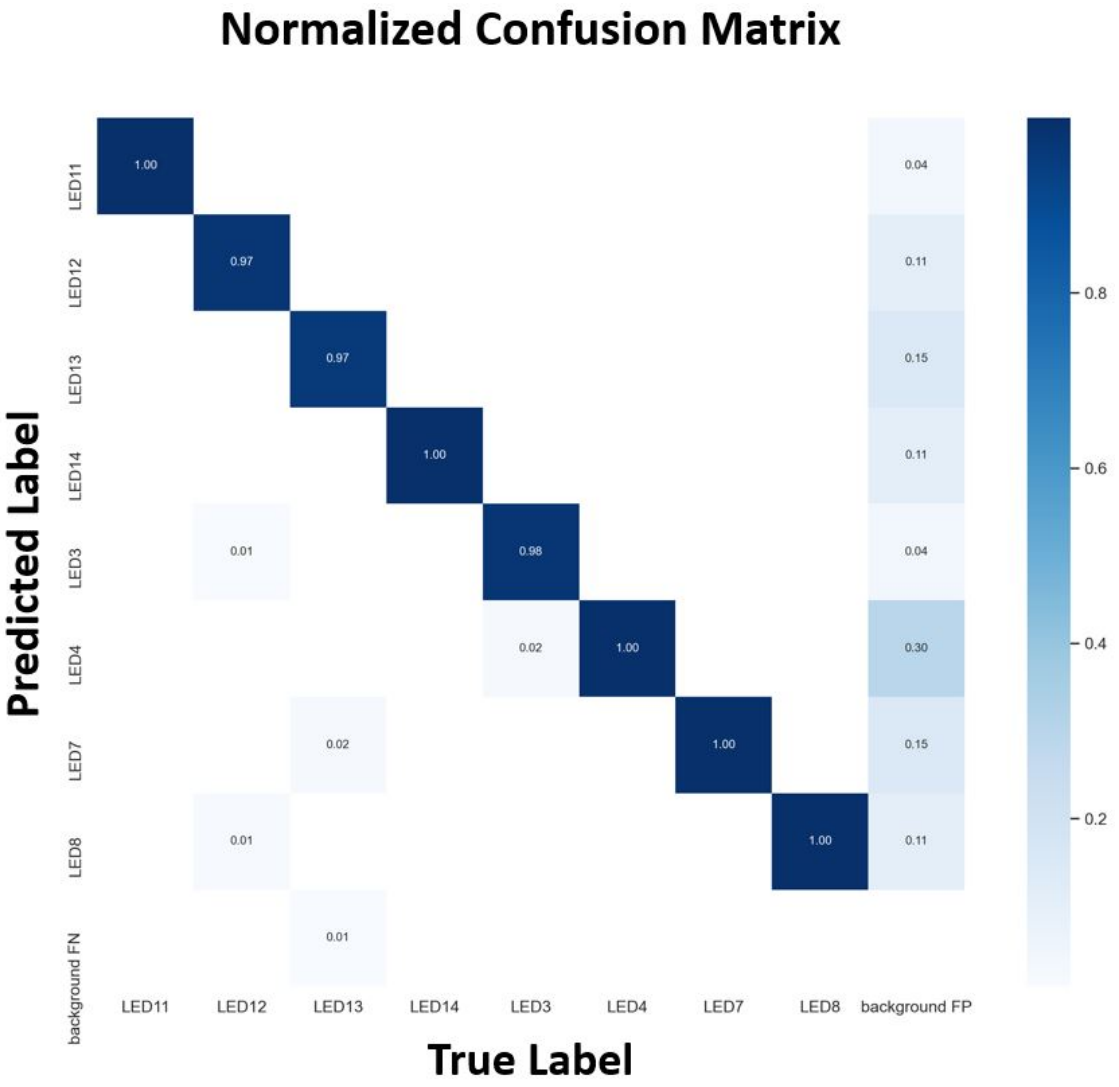

Figure 4.6: YOLOv5s confusion matrix.

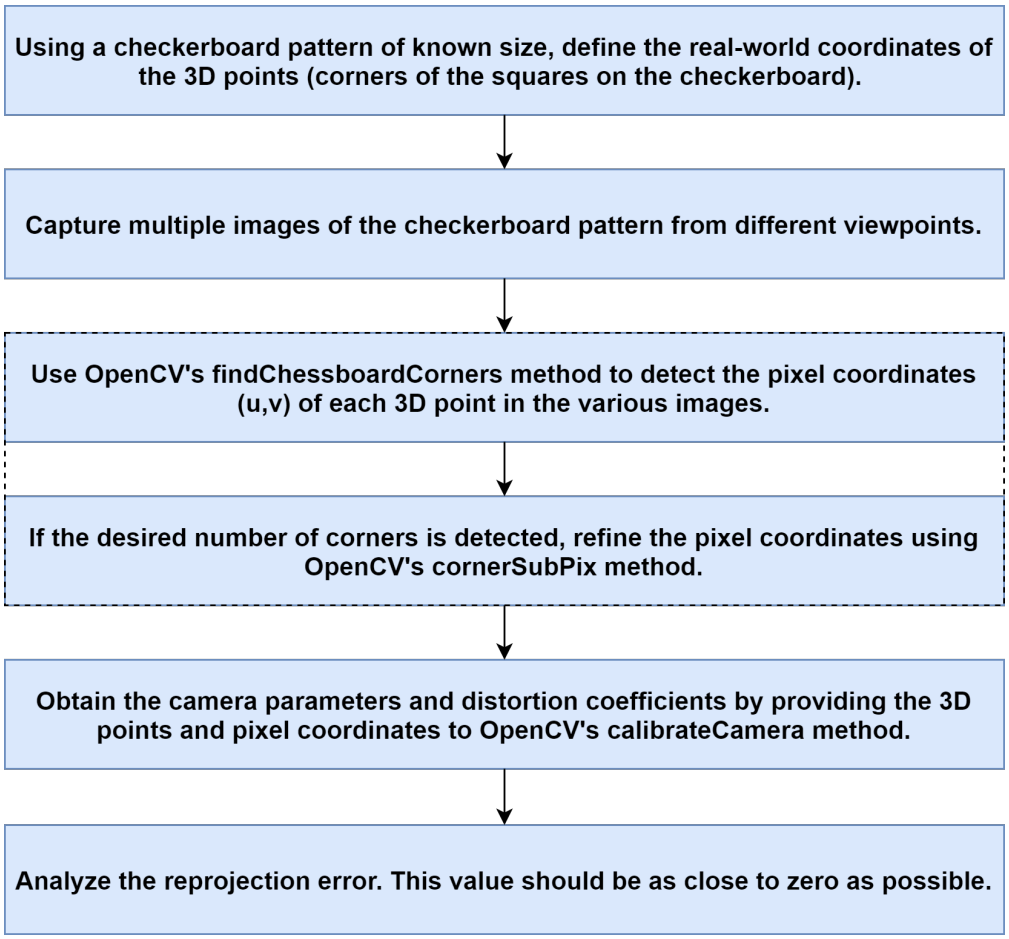

Figure 4.7: Camera calibration flowchart.

Distortion coefficients $(k_1, k_2, p_1, p_2, k_3)$: $\begin{bmatrix} 0.2172 & -0.6233 & -0.0008 & -0.0004 & 0.5242 \end{bmatrix}$

With this data, it was possible to implement the camera-to-world transformation algorithm and thus finalise the system implementation. The camera-to-world transformation algorithm is detailed in section 4.5.

As stated in chapter 3, the image filtering and resizing block was added since we are not interested in color information and do not need to differentiate between a large number of LED luminaires for the system demonstration. Thus, in this block, the input images are converted to grayscale, resized in order to make the system faster, and padded to 1632x1632 pixels while maintaining the aspect ratio.

For the system to be implemented in another similar indoor setup, it is only necessary to perform the calibration of the new rolling shutter camera in order to obtain the camera parameters and measure the real position of the LED luminaires in the new indoor space. The new camera must generate images with similar characteristics to those presented in this dissertation, this is, this new camera must present the same readout and exposure times. With different values for readout and exposure times, the same LED luminaire frequency

Parameter	Symbol	Value
Room size	(w, l, h)	4.200 × 2.700 × 2.710 (m)
The coordinates of		
Tx_{13}	(x_{T13}, y_{T13}, z_{T13})	0.330,0.485,2.710 (m)
Tx_{7}	(x_{T7}, y_{T7}, z_{T7})	1.523,0.485,2.710 (m)
Tx_{11}	(x_{T11}, y_{T11}, z_{T11})	2.713,0.485,2.710 (m)
Tx_{4}	(x_{T4}, y_{T4}, z_{T4})	2.908,0.492,2.710 (m)
Tx_{14}	(x_{T14}, y_{T14}, z_{T14})	0.335,2.080,2.710 (m)
Tx_{8}	(x_{T8}, y_{T8}, z_{T8})	1.528,2.087,2.710 (m)
Tx_{12}	(x_{T12}, y_{T12}, z_{T12})	2.713,2.081,2.710 (m)
Tx_{3}	(x_{T3}, y_{T3}, z_{T3})	2.908,2.085,2.710 (m)
P6	(x_{P6}, y_{P6}, z_{P6})	1.623,1.010,0.000 (m)
P7	(x_{P7}, y_{P7}, z_{P7})	1.873,1.010,0.000 (m)
P8	(x_{P8}, y_{P8}, z_{P8})	2.123,1.010,0.000 (m)
P9	(x_{P9}, y_{P9}, z_{P9})	2.373,1.009,0.000 (m)
P11	(x_{P11}, y_{P11}, z_{P11})	1.623,1.260,0.000 (m)
P12	(x_{P12}, y_{P12}, z_{P12})	1.873,1.260,0.000 (m)
P13	(x_{P13}, y_{P13}, z_{P13})	2.123,1.260,0.000 (m)
P14	(x_{P14}, y_{P14}, z_{P14})	2.373,1.259,0.000 (m)
P16	(x_{P16}, y_{P16}, z_{P16})	1.623,1.510,0.000 (m)
P17	(x_{P17}, y_{P17}, z_{P17})	1.873,1.510,0.000 (m)
P18	(x_{P18}, y_{P18}, z_{P18})	2.123,1.510,0.000 (m)
P19	(x_{P19}, y_{P19}, z_{P19})	2.373,1.509,0.000 (m)
Transmit power of each Tx	Pt	18 W
Transmitter size (diameter)		150 (mm)
Receiver's field of view	FoV	62.2 degrees (horizontal) and 48.8 degrees (vertical)
Focal Length	f	3.04 mm
Image Resolution		3264x2464 pixels

Table 4.4: Key system parameters.

would correspond to a different strip pattern. If the system is to be implemented in a setup with more LED luminaires, the whole procedure presented in this dissertation has to be repeated, since the object detection algorithm was only trained for 8 LED luminaires with the mentioned frequencies.

4.1 Emitter Details

Each emitter is composed of an LED luminaire and its driver. The driver used in this dissertation was developed by Pedro Rodrigues (former student of the University of Aveiro) in his dissertation project [50] and it drives a DLA G2 luminaire from Tridonic with a nominal power of 18 W and a diameter of 15 cm.

At the beginning of this dissertation project, four emitters were already mounted on the ceiling, and in preparation for this dissertation, twelve more units were assembled and mounted. Figure 4.8 shows an emitter mounted on the ceiling in the Integrated Circuit Systems laboratory of Instituto de Telecomunicações in Aveiro.

To better understand the behavior of the emitter in the system, the transient voltage in the sensing resistor R_s (LED signal) was measured using an oscilloscope. The driver was programmed to switch on and off at 3 kHz. Figures 4.9 and 4.10 show the response of the current regulator in transient regions, namely on the rising edge and the falling edge,

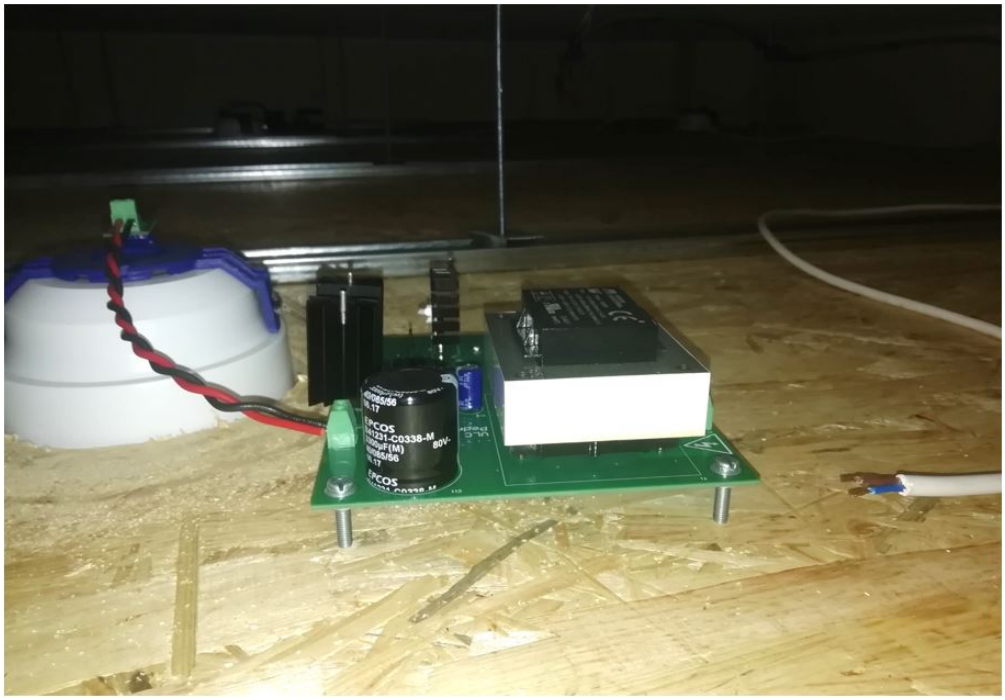

Figure 4.8: Ceiling-mounted emitter.

respectively.

As we can see, the settling times of the rising and falling edges of the LED signal are 342 ns and 446 ns, respectively. These non-ideal on-off/off-on transition times affect the white-to-black and black-to-white transitions in the image, but not significantly.

For VLC, the maximum frequency of the emitters is an important characteristic. In this case, it can be calculated using the following expression:

$$f_{\max} = \frac{1}{t_{\text{sup}} + t_{\text{sdown}}} = \frac{1}{342ns + 446ns} = 1.27MHz \tag{4.7}$$

where t_{sup} is the settling time of the rising edge and t_{sdown} is the settling time of the falling edge in the sensing resistor.

Note that this frequency is sufficient to test all the capabilities of our system. On the other hand, the minimum frequency at which these emitters can operate is approximately 150 Hz, which is limited by the counter present in the microcontroller.

The driver is remotely controlled using an nRF24L01+ transceiver. The graphical user interface (GUI) developed by Pedro Rodrigues in python was used to control the functionalities of the emitters. In addition to the three existing modes for controlling the emitters (DC, OOK+Manchester and VPPM), a new mode, OOK, has been added, allowing the LED luminaires to be switched on and off at different frequencies. The GUI is presented in Figure 4.11.

Figure 4.9: LED signal (settling time of the rising edge highlighted).

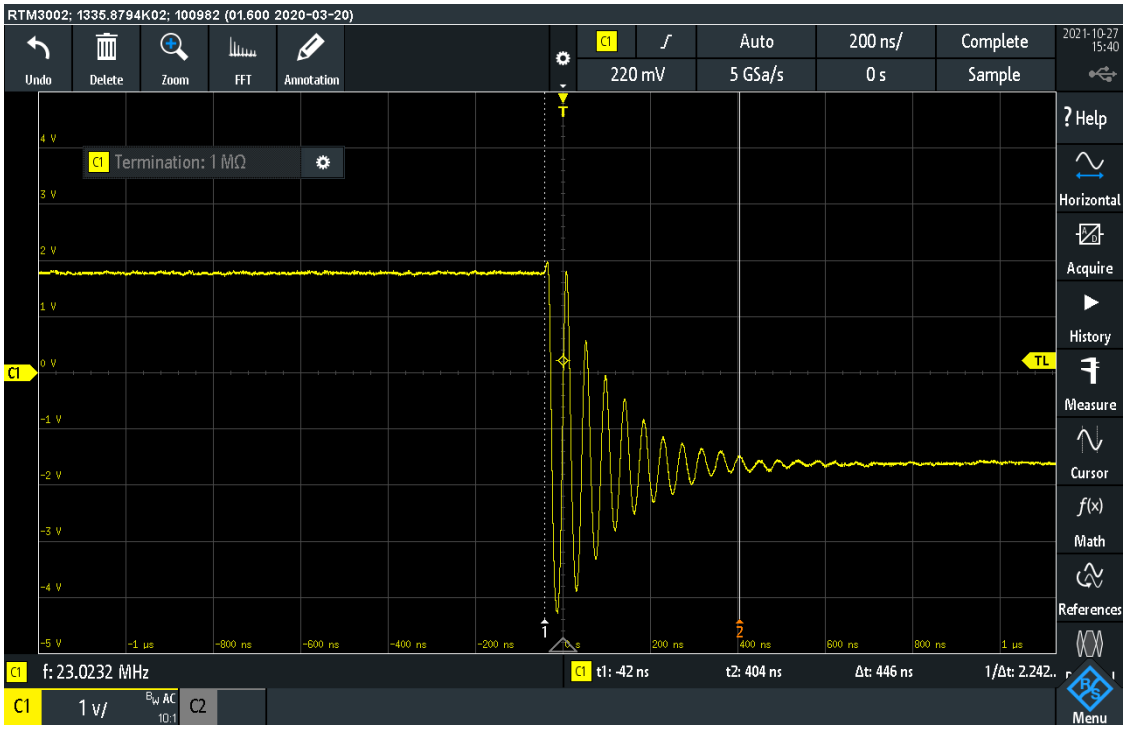

Figure 4.10: LED signal (settling time of the falling edge highlighted).

Figure 4.11: GUI to control the emitters.

4.2 Receiver Details

To acquire the images in a controlled manner, the receiver module shown in Figure 4.12 was developed and constructed. It is composed of a Raspberry Pi 3B, a Raspberry Pi V2 camera module, a HerkuleX DRS-0101 servo, an Arduino Uno and a Nextion NX3224T028 display. The camera module uses the 8 MP Sony IMX219 image sensor. The 3D printed structure allows the roll, pitch and height of the camera to be varied. The roll variation is performed automatically and the pitch and height adjustments are performed manually. This module is powered by a battery (5 V - 15.6 Ah) and operates autonomously.

During the course of the dissertation, this module has undergone several updates to make image acquisition a faster, more user-friendly, versatile, and accurate process. This iterative process can be seen in Figure 4.13.

Taking advantage of the touchscreen present in the final version, the GUI allows the user to enter the reference point number where the images are being acquired and the degrees of rotation between the acquisition of each image. In this work, the reference point number was useful since we collected images on a grid fixed on the floor surface. In addition, the module's operating state is also displayed (standby or running). The GUI present in the receiver module is shown in Figure 4.14.

The sequence of operation of the receiver is shown in the block diagram in Figure 4.15. The receiver was programmed to acquire images including the raw Bayer data with a resolution of 3264×2464 pixels and an exposure time of $9\,\mu s$. Then, to make the images more "normal", demosaicing was performed. The name of each image file follows the structure

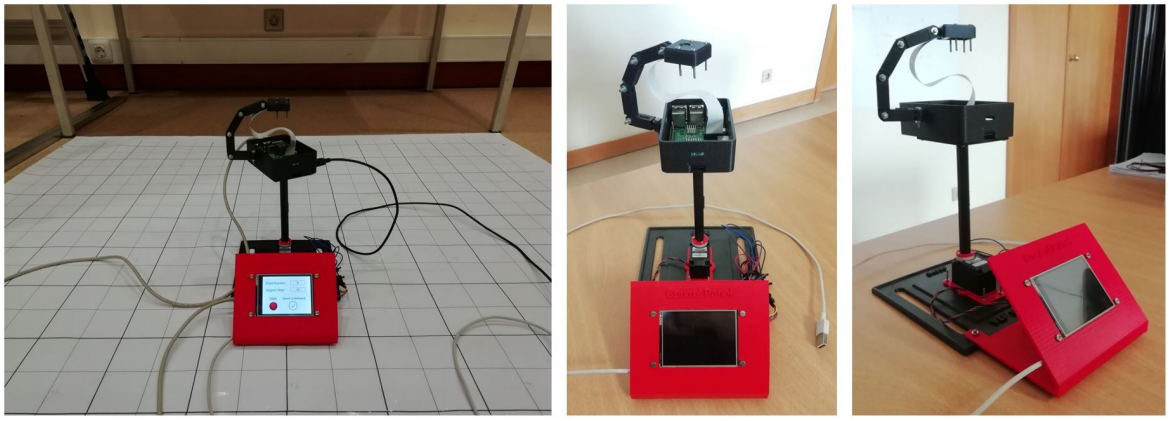

Figure 4.12: Receiver module.

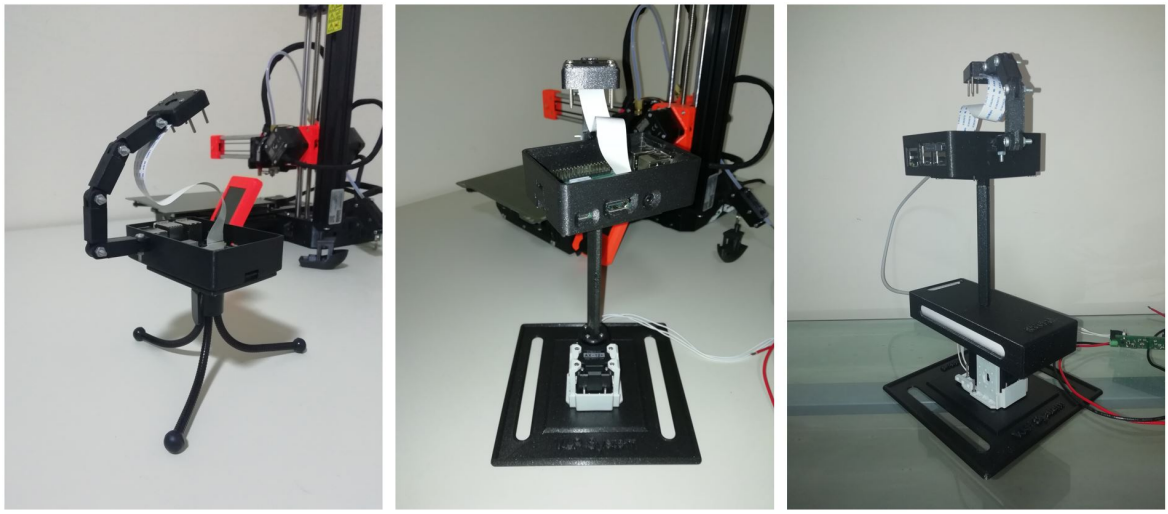

Figure 4.13: Earlier versions of the receiver module.

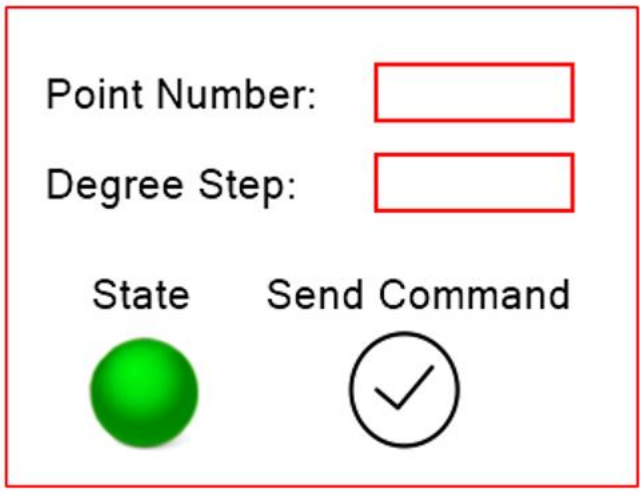

Figure 4.14: GUI present in the receiver module.

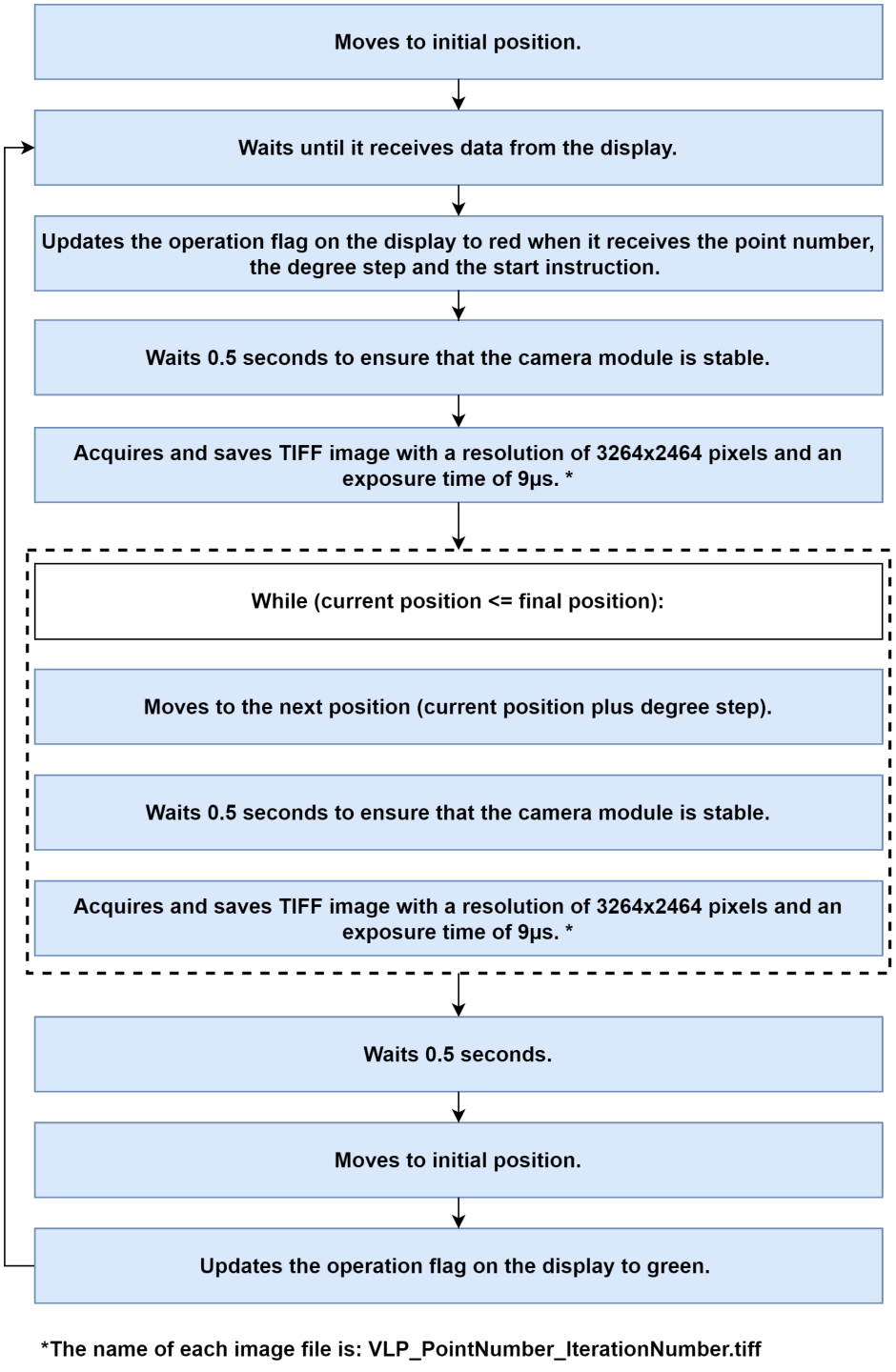

Figure 4.15: Sequence of operation of the receiver module.

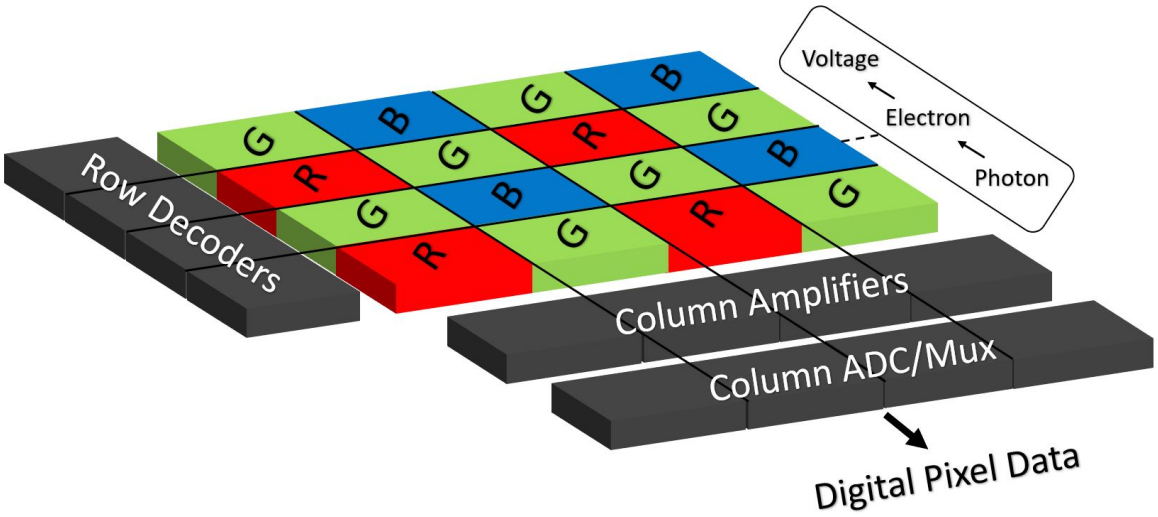

Figure 4.16: CMOS camera architecture with BGGR pattern [51].

`VLP_PointNumber_IterationNumber.tiff`, so for example, if the receiver is positioned at point 6 on the grid and the user has requested 2 degrees of rotation between each image, the file names will be `VLP_6_1.tiff` to `VLP_6_161.tiff`. Note that the Herkulex servo can only rotate 320 degrees.

Raw Bayer data is the data recorded by the camera's sensor before any GPU processing. Thus, the Bayer data is always full resolution and is organized in a BGGR pattern. An illustration of the CMOS camera architectute with this pattern is presented in Figure 4.16.

Finally, Figure 4.17 shows the 3D model of each constituent part of the receiver. These models were designed using the Fusion360 software [52]. The HerkuleX DRS-0101 bracket, the Raspberry Pi 3B case and the camera case were adapted from the models shown in [53], [54] and [55], respectively.

4.3 ROI Algorithm

The ROI algorithm extracts the LED luminaires from the original image. Taking advantage of the background of the images being almost black and the LED luminaires appearing in almost pure white, several algorithms present in the OpenCV library were used.

Figure 4.18 shows the block diagram of the ROI algorithm, and Figure 4.19 presents the images that result from the different steps of the process. The first steps in the ROI algorithm are the conversion to grayscale and the smoothing of the image (median blur). This filter is intended to reduce the noise from the original image. Then, an adaptive threshold and two consecutive morphological operations (erosion and dilation) are applied. This allows the bright stripes to stand out from the background. Using contour detection, the borders of the bright stripes are detected.

The contours that are close to each other, which correspond to the stripes of the same LED luminaire, are grouped together to generate a bounding box adapted to the dimensions of the LED luminaires in the image. If the bounding box is larger than $\frac{full\ LED\ width}{3} \times \frac{full\ LED\ height}{3}$ pixels, this means that in these conditions at least $\frac{1}{3}$ of the total LED luminaire

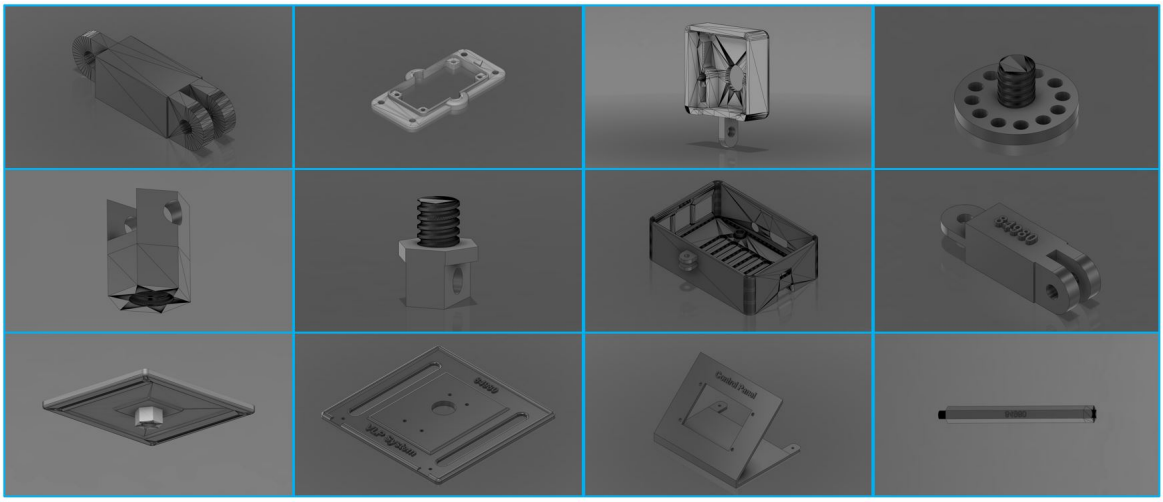

Figure 4.17: 3D models of the receiver parts.

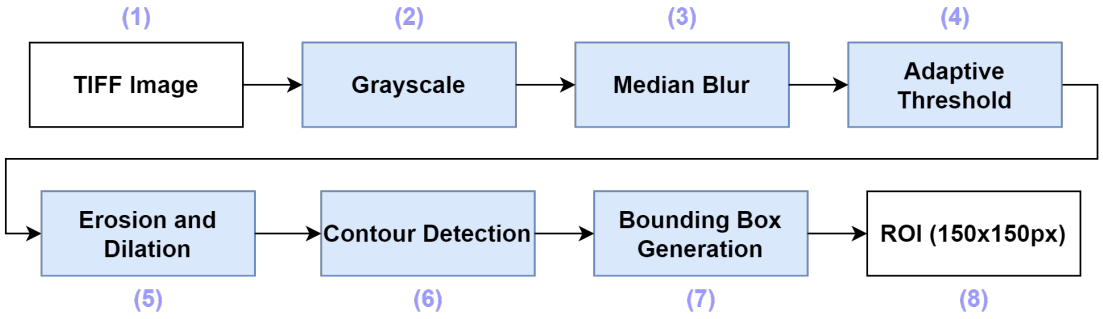

Figure 4.18: Block diagram of the ROI algorithm.

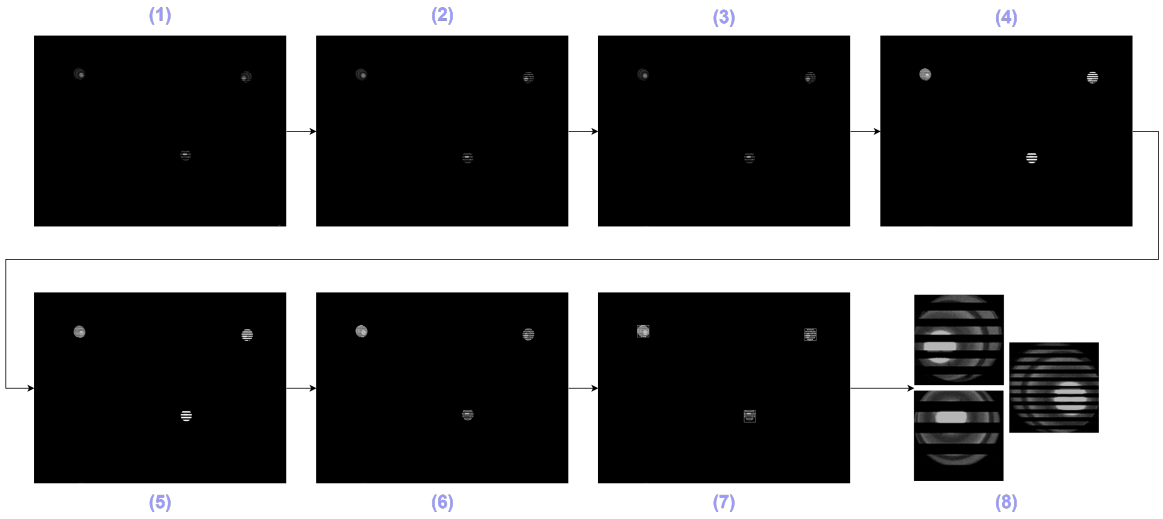

Figure 4.19: Illustration of the resulting images from each block of the ROI algorithm.

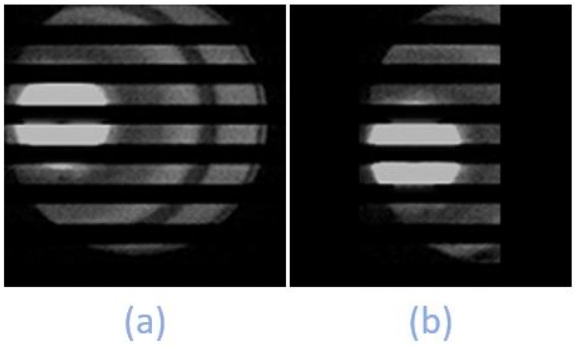

Figure 4.20: Example result of the LED isolation, (a) full LED, (b) partially visible LED.

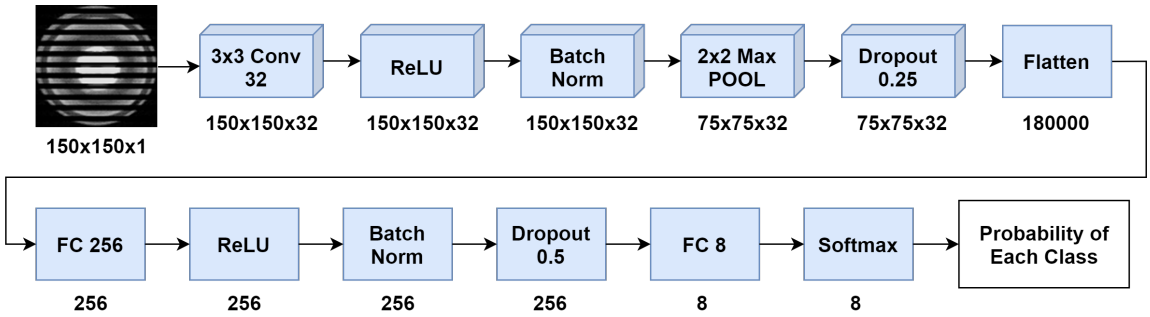

Figure 4.21: Visual Representation of CNN's architecture.

is visible, then the bounding box is resized and padded to 150×150 pixels, keeping the aspect ratio. If this condition is not met, the bounding box is discarded. This way, the LED luminaires that are fully visible and the LED luminaires that appear in the image with a size greater than $\frac{1}{3}$ of the total LED are considered. We chose to include the partially visible luminaires in order to increase the robustness of the system in cases that few luminaires are visible. However, if the portion of the partially visible luminaire was too small, an identification error in YOLOv5 could arise, for that, a minimum of $\frac{1}{3}$ of the total LED luminaire was chosen as a rejection threshold. The original grayscale image is then cropped according to these regions. An example result of the LED isolation is shown in Figure 4.20.

4.4 CNN Structure

To accurately classify each individualised LED pattern, the CNN architecture shown in Figure 4.21 was implemented.

The model consists of 12 layers. All model parameters were chosen to avoid overfitting or underfitting the data.

The model starts with a 2D convolutional layer. In a 2D convolution, the kernel (small 2D matrix of weights), slides over the 2D input data performing an element-wise multiplication and summing the results into a single output pixel. The kernel replicates this process for each location it slides over, thus the output features are the weighted sums of the input features [56]. Since the images are visually simple, with little irregular shapes, just a convolutional layer was

implemented to highlight the lines in the image. This layer allows the model to learn position and scale invariant structures in the data [57]. The implemented 2D convolutional layer has 32 kernels with a kernel size of (3×3), which defines the dimensions of the output space. Furthermore, the stride was set to 1 and zero padding was added evenly to the left/right or up/down of the input. Zero padding adds zeros outside the image border. Padding is added to the image frame to allow more space for the kernel to cover the image. It also allows more accurate analysis of the image and preservation of the original input size. Stride refers to the number of pixels that the filter shifts over the input matrix, both horizontally and vertically.

The activation function used in this architecture is the rectified linear activation function (ReLU). As stated in [58], the goal of an activation function is to introduce non-linearity in the output of a neuron. These functions allow for back-propagation, the process of performing a backward pass while adjusting the model's parameters (weights and biases), since the gradients are provided together with the error to update the weights and biases. The ReLU outputs the input directly if it is positive, otherwise, it will output zero. It overcomes the vanishing gradient problem, allowing our model to learn faster and perform better. Furthermore, the ReLU is trivial to implement and has a linear behavior [59].

Then, Batch Normalization is used to normalize to zero mean and unit variance each element of a layer in the network [60]. Normally, this normalization step is applied right after (or right before) the non-linear activation function. As stated in [61], it is very effective at reducing the number of epochs required to train a CNN as well as stabilizing the training itself. Just for reference, the batch normalization is calculated in a different way during the training and testing phases.

Pooling layers have a primary function of progressively reducing the amount of parameters in the network, thus reducing the dimensions of the feature maps generated by a convolution layer [62]. Max pooling is a pooling operation that selects the maximum element from the region of the feature map covered by the two-dimensional filter, in this case by a (2×2) filter. Thus, the output of the max pooling layer is a feature map containing the most prominent features of the previous feature map. This makes the model more robust to changes in the position of the features in the input image.

In order to make the network more robust we applied dropout, the process of disconnecting random neurons (along with their connections) between layers during the training. It is reported in the literature that this process reduces overfitting, increases accuracy, and allows our network to generalize better to unknown images [63]. It is also a very computationally efficient regularization method. The first dropout layer was set with a rate of 0.25 and the second with a rate of 0.5. The dropout rate is a value between 0 and 1 and represents the fraction of the input units being dropped.

In the last stages of the CNN, we have the fully-connected (Dense) layers. The input to the first fully-connected layer is the output from the dropout layer, which is flattened. The flattening layer converts the data into a long feature vector [64]. In the fully-connected layers, all the neurons in one layer are connected to the neurons in the next layer. In the background, the fully-connected layer performs a matrix-vector multiplication [65]. The values used in the matrix are parameters trained and updated using back-propagation. The final fully-connected layer has eight outputs, since we have eight distinct LED-IDs (classes).

Normally, activation functions and dropout layers are used between two consecutive fully-connected layers to introduce non-linearity and reduce overfitting respectively [66]. In this model, besides these two types of layers, a batch normalization layer was also added.

At the end, the softmax layer returns the class probabilities for each class. It converts a

Model: "sequential"		
Layer Type	**Output Shape**	**Param #**
Conv2D	(150, 150, 32)	320
Activation	(150, 150, 32)	0
BatchNormalization	(150, 150, 32)	128
MaxPooling2D	(75, 75, 32)	0
Dropout	(75,75,32)	0
Flatten	180000	0
Dense	256	46080256
Activation	256	0
BatchNormalization	256	1024
Dropout	256	0
Dense	8	2056
Activation	8	0

Total Params:	46083784
Trainable Params:	46083208
Non-trainable Params:	576

Table 4.5: CNN model summary.

vector of numbers into a vector of normalized probabilities, where the probabilities of each class are proportional to the exponentials of the input numbers. The sum of the probabilities is 1 [67]. Finally, the normalized probability distribution is displayed.

The CNN model summary is presented in Table 4.5. In addition to the output shape of each layer, the number of network parameters is also shown.

4.5 Camera-to-World Transformation Algorithm

The problem of finding the pose (position and orientation) of a given camera, given a set of 2D-3D point correspondences, is called the Perspective-n-Point (PnP) problem. To obtain the camera pose in the WCS, the process presented in Figure 4.22 was implemented. Having as inputs the camera parameters and the coordinates of the identified LED luminaires, the camera pose was obtained using three OpenCV algorithms [68]. These algorithms implement the mathematical process described in section 3.1. Since it is beyond the scope of this dissertation to study in depth the algorithms used to solve the PnP problem, an already developed solution was used. The position of the camera was obtained using the algorithms solvePnP [69], with the solver introduced in [70], and the Rodrigues' algorithm [71]. To obtain the Tait–Bryan angles, in degrees, the RQDecomp3x3 algorithm [72] was used. For reference, an illustration of the receiver with the Tait-Bryan angles (roll, pitch and yaw) is shown in Figure 4.23. As a brief context, the Rodrigues algorithm converts a rotation vector to a rotation matrix, or vice versa, and the RQDecomp3x3 algorithm converts a rotation matrix to Tait–Bryan angles. The solvePnP algorithm outputs rotation and translation vectors, since it uses the Rodrigues algorithm internally.

As explained in Chapter 3, in our setup, the 3D points correspond to the world coordinates of the center of each visible LED luminaire in the image, and in the same order, the 2D points

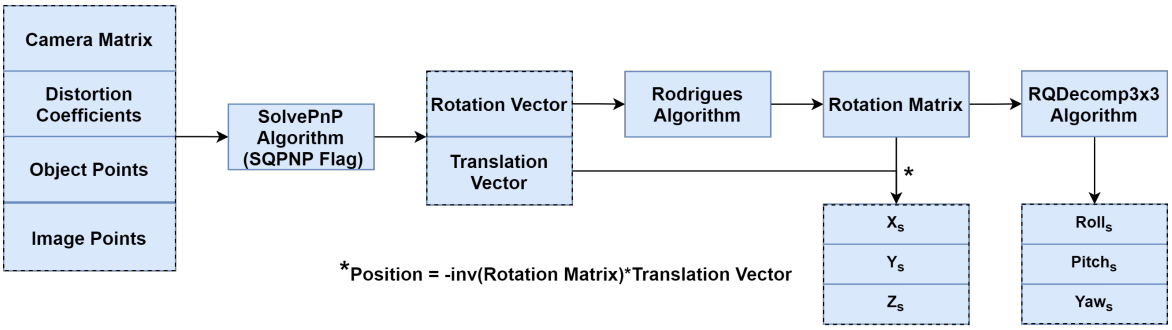

Figure 4.22: Camera-to-world transformation algorithm.

correspond to the image coordinates of the center of each LED projection.

For the solving method, there are three main options, namely: Iterative, P3P and SQPnP. Starting with the P3P method, it requires exactly four point correspondences, which is not ideal since we want to use as many LED luminaires as possible for the position estimation. The Iterative method requires four or more point correspondences. Finally, the SQPnP method, presented in 2020, requires at least three point correspondences and allow for a large number of points. As stated in [51], the Iterative method is based on the Levenberg-Marquardt optimization. This method finds a pose that minimizes reprojection error, given by the sum of the squared distances between the initial coordinates of each image point and the reprojected coordinates obtained with each guessed pose. According to the authors, the SQPnP method formulates the problem as a nonlinear quadratic program and identifies regions in the parameter space that contain unique minima with assurances that at least one of them will be the global minimum, regardless of possible coplanar arrangements of the imaged 3D points.

The SQPnP method was chosen for two main reasons. The first is that by requiring only 3 point correspondences to obtain the pose, it offers superior robustness to the system. The second is that, through testing, the SQPnP method efficiently produced accurate results, similar to those obtained with the Iterative method.

4.6 Summary

In this chapter the implementation process of the complete system has been described in detail. Furthermore, the developed software has been explained step by step with some illustrations. Finally, additional information about the transmitters and receiver has been provided.

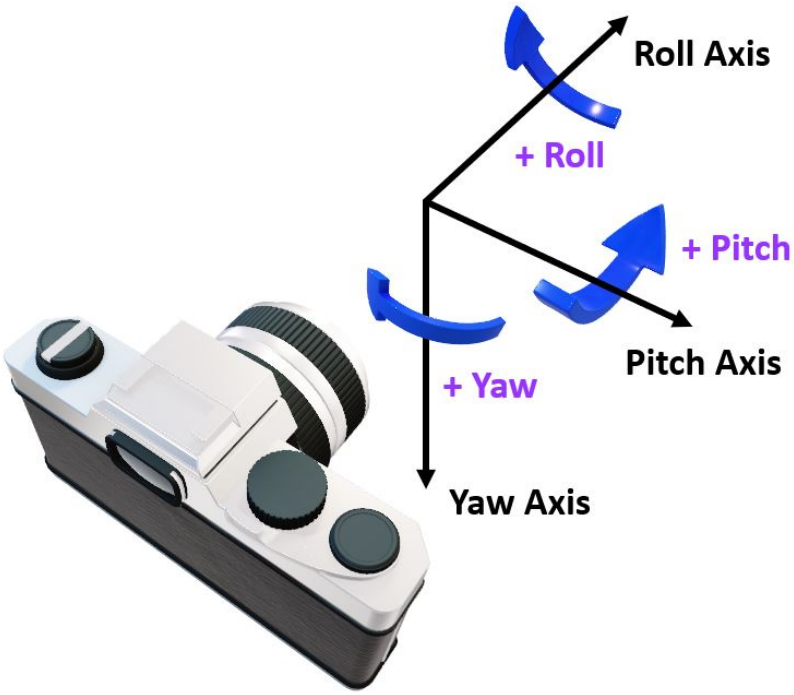

Figure 4.23: Illustration of the Tait-Bryan angles (roll, pitch and yaw) on the receiver.

Chapter 5

Experimental Results and Discussion

This chapter presents the procedure carried out to test the proposed system and the results obtained.

Given that the system is complete, we can analyse the behaviour of the object detection algorithm. An example of the operation of the YOLOv5s is shown in Figure 5.1. As we can see, the input image shows 5 LED luminaires. After passing through the algorithm, the LED luminaires identified in the image are classified and the information regarding the bounding boxes (x, y, w, h, confidence) is displayed. The coordinates (x, y) represent the center of the boxes, relative to the size of the image. The normalized parameters w and h represent the width and the height of the bounding boxes. The confidence score indicates the probability that the predicted LED-ID is actually present in that bounding box. In this example, LED luminaires number 7, 8, 11, 12 and 13 have been correctly identified. It can be seen that the dimensions of the bounding boxes are quite optimised for the size of the LED luminaires in the image.

During the experimental process, the CMOS image sensor was kept parallel to the floor at a height of 25.6 cm. All images were acquired as TIFF with full resolution (3264×2464 pixels) and with the exposure time set to the minimum possible value.

To analyse the positioning error and evaluate the performance of the system, a grid with 12 regularly spaced reference points was defined on the floor surface. The position of these points relative to the referential defined in the room was measured using a tape, see Table 4.4. Figure 5.2 provides an illustration of the position of the grid in the room.

At each point a set of 161 images were taken with 2 degrees of rotation between each one. The 1932 images were then passed through the 3D VLP algorithm shown in Figure 3.2, with the YOLOv5s algorithm set with the parameters Confidence threshold equal to 0.35 and IoU threshold equal to 0.45. An example of the results of the VLP system is shown in Table 5.1, in this case at reference point number 6. The coordinates of this reference point are (1.623, 1.010, 0.000) in meters. As indicated earlier, the camera was positioned parallel to the floor at a height of 25.6 cm facing upwards, i.e. with pitch and yaw angles equal to 0. From photograph to photograph, the camera was rotated by 2 degrees. This behaviour is visible in the fifth column of Table 5.1. Overall, the results are in line with expectations. Note that, estimates that provided considerably different values in one or more components,

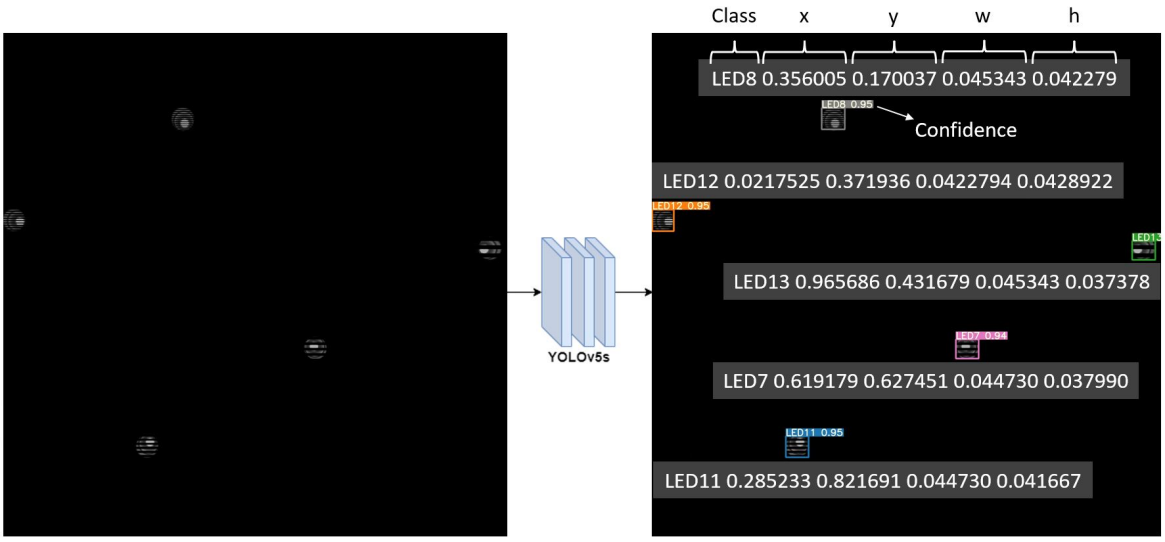

Figure 5.1: Input and output examples of the YOLOv5s algorithm.

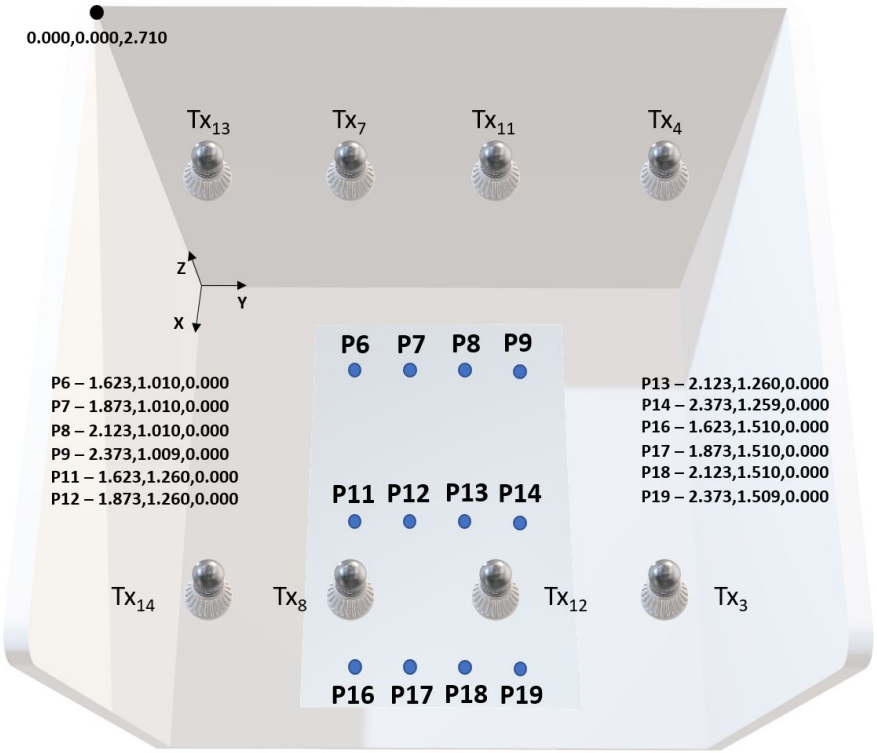

Figure 5.2: Illustration of the position of the reference points in the room (values are given in meters).

P6						
Elapsed Time (s)	**X_s (m)**	**Y_s (m)**	**Z_s (m)**	**$Roll_s$ (°)**	**$Pitch_s$ (°)**	**Yaw_s (°)**
0.017	1.588	0.962	0.252	140.335	-1.519	0.898
0.018	1.594	0.959	0.253	142.384	-1.454	0.879
0.017	1.600	0.985	0.251	144.134	-1.345	0.346
0.017	1.614	0.969	0.258	145.948	-1.048	0.647
0.019	1.608	0.999	0.257	148.305	-1.167	0.013
0.017	1.595	1.040	0.258	149.957	-1.366	-0.782
0.017	1.608	1.015	0.260	152.000	-1.152	-0.350
0.017	1.595	1.040	0.256	153.925	-1.472	-0.958

Table 5.1: Example of the VLP system outputs at floor point number 6.

P6						
Elapsed Time (s)	**X_s (m)**	**Y_s (m)**	**Z_s (m)**	**$Roll_s$ (°)**	**$Pitch_s$ (°)**	**Yaw_s (°)**
0.018	1.559	2.655	1.450	-154.688	-1.913	-49.049
0.017	1.306	0.776	0.385	-91.753	-7.717	3.407
0.017	1.200	0.964	0.177	-58.670	-8.965	-1.478
0.017	1.543	0.017	1.073	103.828	-1.934	28.194

Table 5.2: Outliers obtained at floor point number 6.

such as those obtained in point 6, seen in Figure 5.2, were discarded manually. These outliers are due to errors in the LED pattern classification and in the projective geometry algorithm.

Figures 5.3 and 5.4 show the position estimates and the original reference points in 2D and 3D, respectively. The arithmetic mean of the 2D and 3D positioning error was approximately 3.5 cm. The 95th percentile was approximately 4.1 cm and 4.2 cm in 2D and 3D, respectively. Figure 5.5 shows the bar charts of the average error at each floor point in 2D and 3D. The cumulative distribution function (CDF) plot of the positioning error is presented in Figure 5.6. From these results, we can observe that the error remains below 5.5 cm, which is considered a good accuracy for an indoor positioning system, using as reference the systems already presented in the literature. Moreover, it is also possible to conclude, by comparing the two bar charts (seen in Figure 5.5), that the Z component has a very low error.

There are several factors that contribute to errors in the positioning system estimates. The following possible sources of error have been identified:

- The camera matrix: to obtain the camera's intrinsic parameters, the camera has to be calibrated. Errors in this process translate directly into errors in the position estimation. This is one of the most critical processes for obtaining consistent and accurate estimates.

- The measurement of the LED luminaires position: as explained above, the location of each LED luminaire in the image is related to its position in the world. If the world position is measured incorrectly, the final position estimate will be affected. As mentioned previously, a tape was used to measure these positions, which has an uncertainty of 0.5mm.

- The object detection algorithm: to estimate the position of the camera from an image, the multiple LED luminaires need to be correctly identified. If this is not the case, the

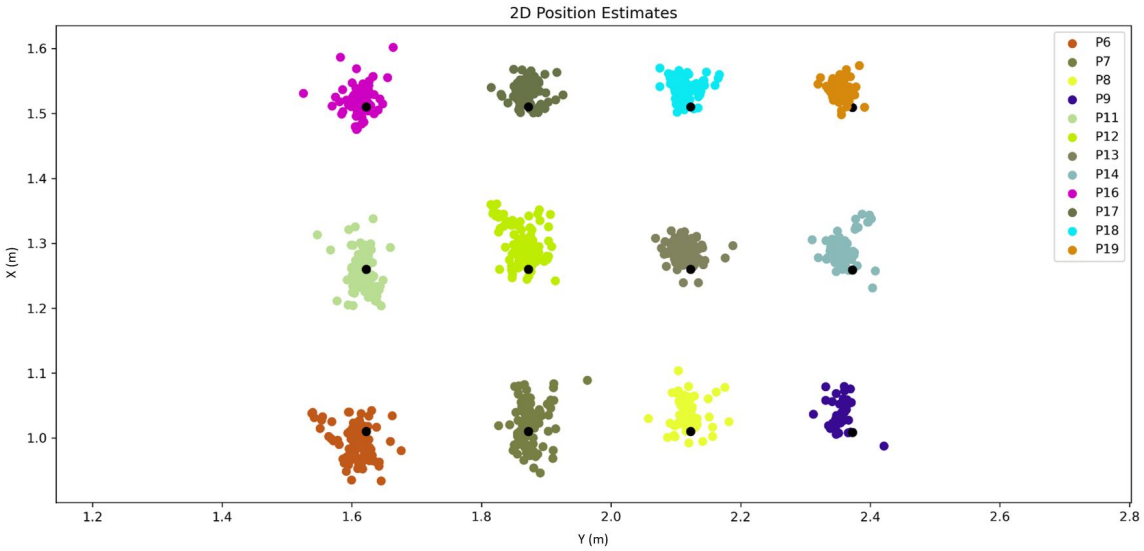

Figure 5.3: Experimental results for 2D positioning estimation.

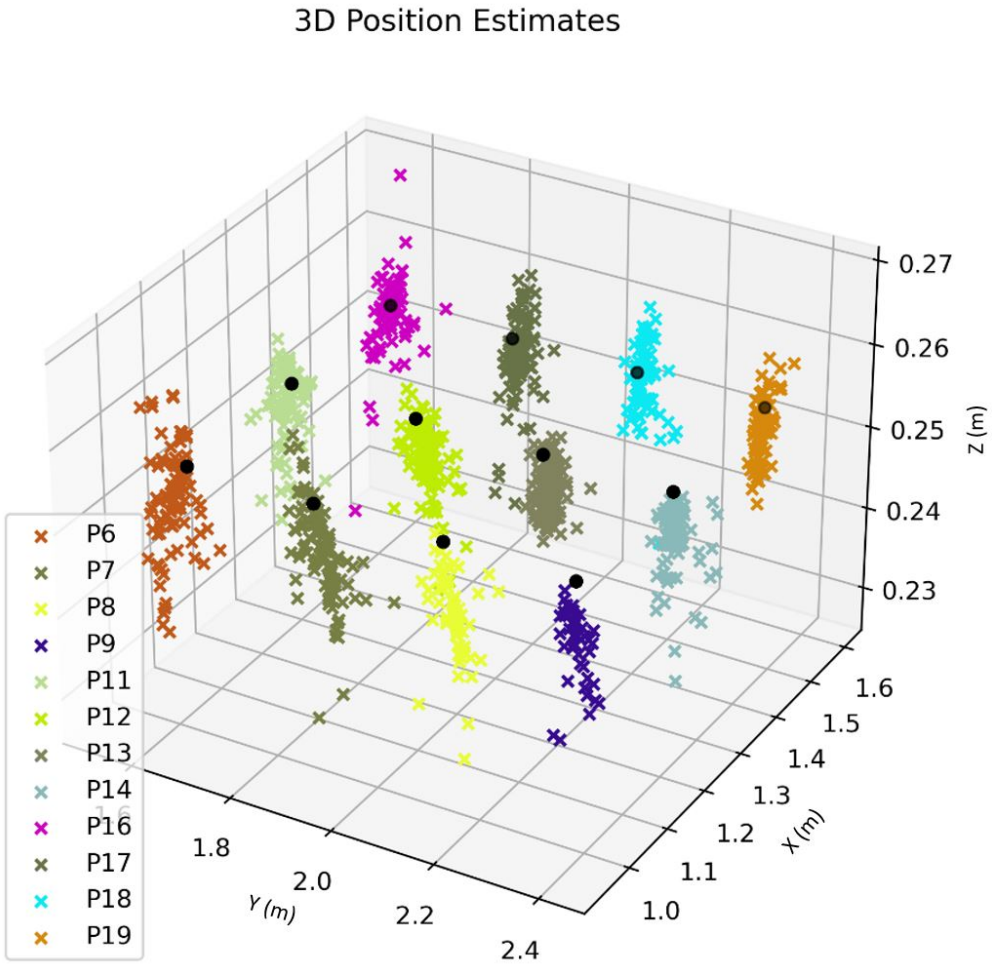

Figure 5.4: Experimental results for 3D positioning estimation.

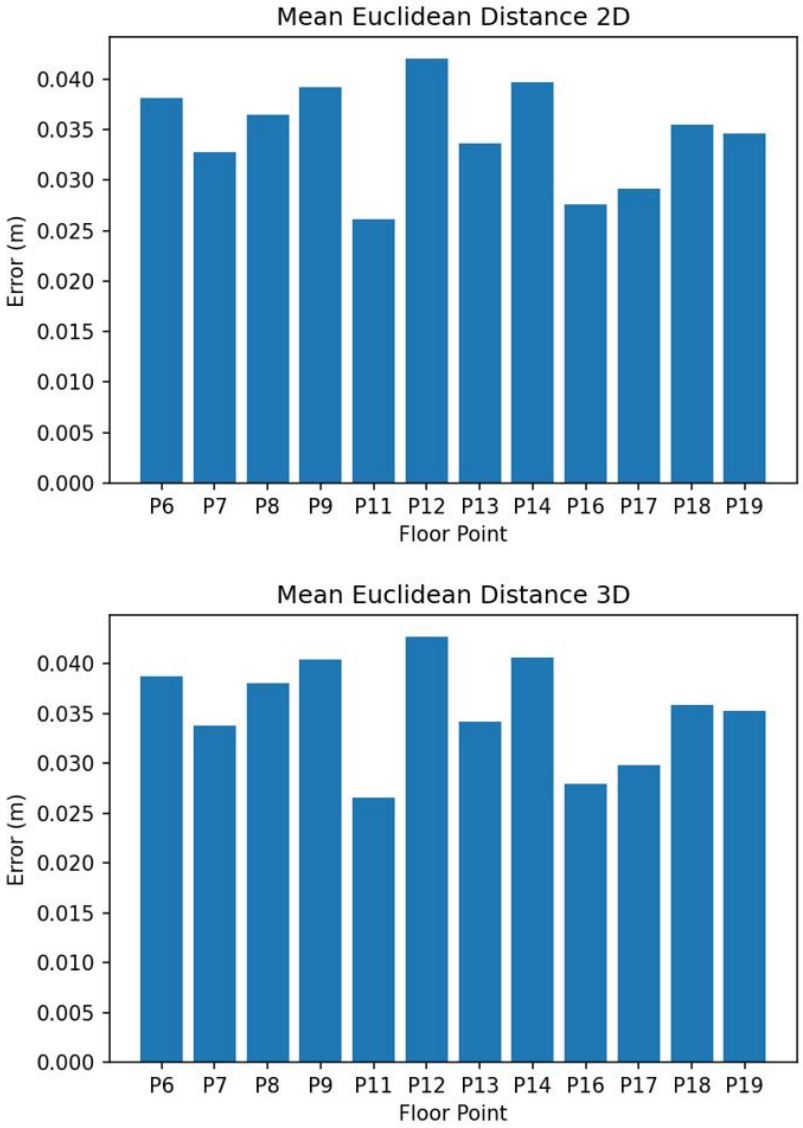

Figure 5.5: Bar charts of the average positioning error at each floor point in 3D and 2D.

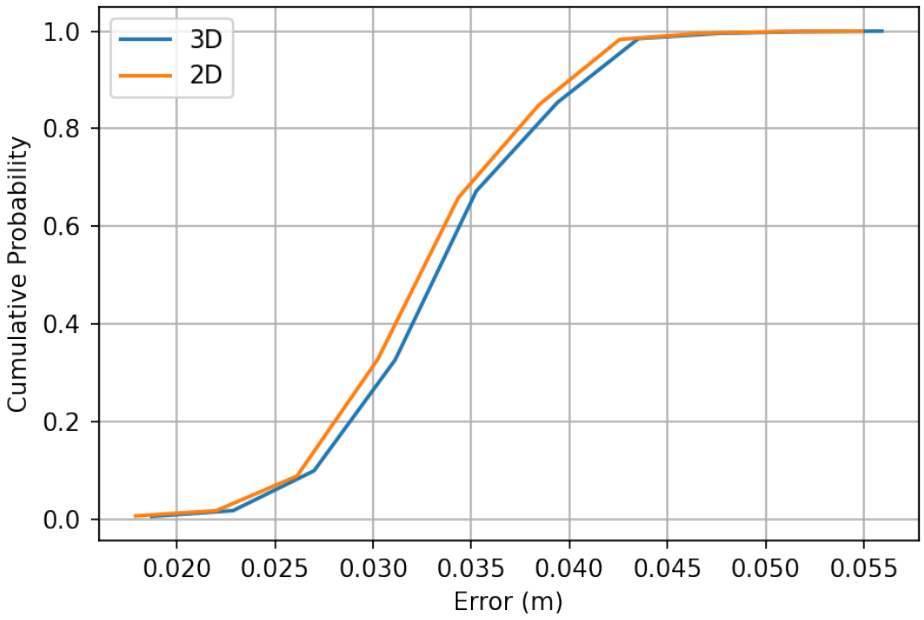

Figure 5.6: CDF plot of the positioning error (3D and 2D).

position estimate will have a very large error. In this work, the recognition algorithm is extremely accurate, however, it can sometimes misidentify the luminaires. Furthermore, the algorithm can detect LED luminaires that are partially visible in the image. This feature makes the system more robust in certain situations, but less accurate. This is because the center of the bounding box moves away from the center of the complete LED pattern.

- The resolution of the camera: as the image is formulated by a set of pixels, the number of pixels in the image is directly related to the final error. The higher the number, the lower the error. Images with 1632×1632 pixels were used in the test procedure, also taking into account that the processing time is affected by the image size.

Finally, the time efficiency of the system was measured on an computer Asus Zephyrus G14 with NVIDIA GeForce RTX 2060 with Max-Q Design 6GB and AMD Ryzen 9 4900HS with 32GB of RAM. The average computational time of the VLP system is approximately 52 ms and the maximum is 54 ms, with YOLOv5s taking approximately 34 ms on average to display the classification of the detected LED luminaires. The histograms of the elapsed times for the two individual algorithms and for the complete system are displayed in Figures 5.7, 5.8 and 5.9. To generate these graphs, the elapsed times for processing 161 representative images in the two algorithms were computed.

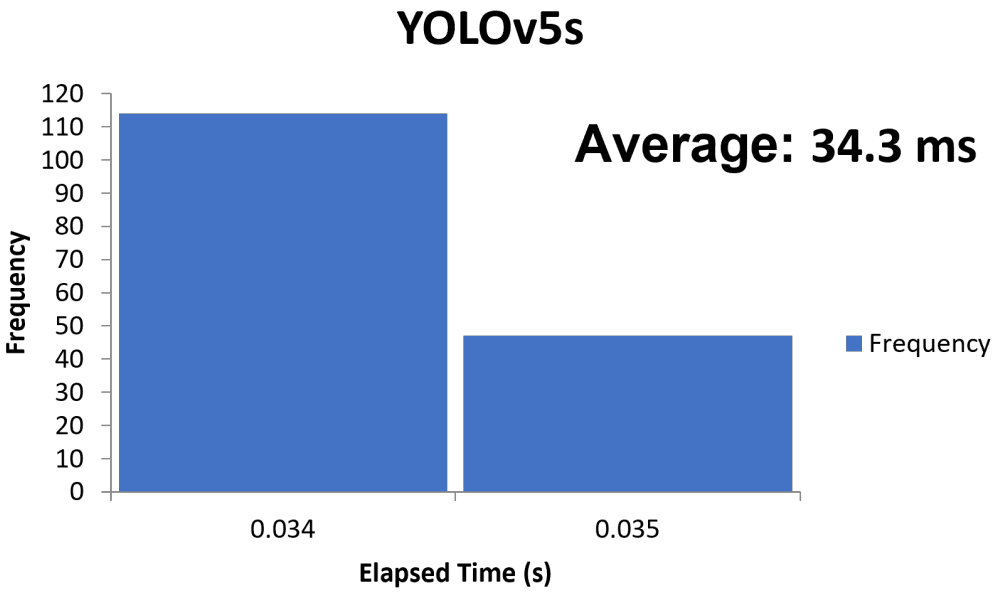

Figure 5.7: Histogram of the elapsed time for the YOLOv5s.

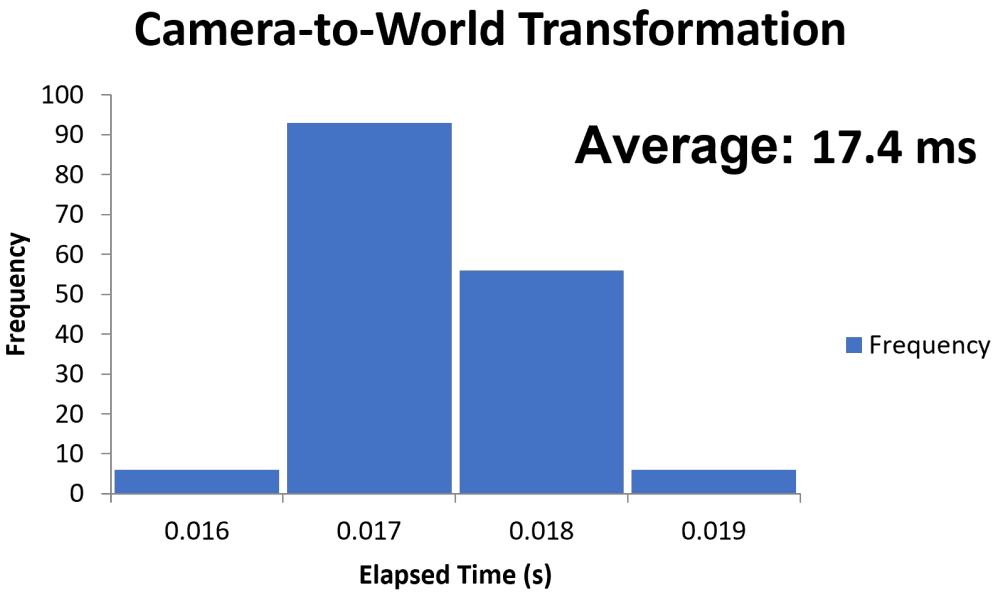

Figure 5.8: Histogram of the elapsed time for the camera-to-world transformation algorithm.

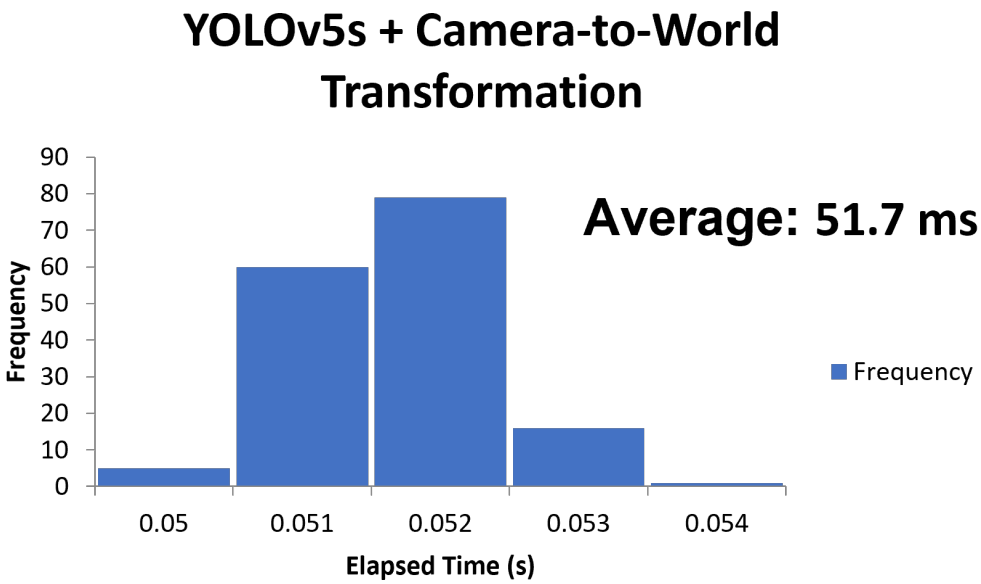

Figure 5.9: Histogram of the elapsed time for the complete system.

5.1 Summary

This chapter presented the procedure performed to test the proposed system and the positioning results obtained. The potential sources of error in the positioning estimates were mentioned and, finally, the time efficiency of the complete system was analysed.

Chapter 6

Conclusions and Future Work

This chapter provides the final observations on the work developed, together with the conclusions drawn from the results obtained. To conclude, future work is presented.

In this dissertation, a 3D VLP system based on ML and OCC, capable of providing real-time indoor position estimates on the order of centimeters, was presented. The system uses a set of LED luminaires modulated using OOK with unique frequencies as references, which allows it to be used also for general-purpose illumination. Its aim is to calculate the camera pose from a single input photograph, assuming that at least three LED luminaires are visible in the image. The detection and recognition of the LED-IDs is done using YOLOv5s, which makes the system fast and robust. Each block of the system and the tools used were explained and illustrated with experimental data. The system was validated using a real-world sized setup and the average 2D and 3D positioning error obtained was approximately 3.5 centimeters. Meanwhile, in terms of the real-time ability, the average computational time of the proposed system is 52 milliseconds (approx. 19 times per second) and the maximum is 54 milliseconds. In comparison with systems presented in the literature, these values were considered good, particularly given the dimensions of the practical setup and the typical requirements demanded of an indoor positioning system. The system, in addition to providing the 3D positioning of the receiver also provides the orientation, however, these results have not been compared with real world measurements and so nothing can be said about their accuracy. The system can be easily adapted to identify more LED luminaires with unique frequencies, ensuring positioning in larger indoor environments. Furthermore, we can state that an automatic framework for dataset acquisition was successfully developed. Increasing the accuracy of the system will be one of the next goals, as there is still room for improvement.

The work presented in this dissertation proves the feasibility of the proposed system. However, possible future improvements were identified. The first is to test the performance of other object detection algorithms. Although YOLOv5s has presented very satisfactory results, other models may present interesting features for certain applications. Regarding the receiver module, it would be interesting to redesign the structure to allow the variation of the three Tait-Bryan angles and to incorporate it into a robotic platform, such as TurtleBot. Another interesting future development would be to test other modulation techniques in the emitters, in order to further increase the number of LED-IDs. For that, it is important to ensure that these modulation techniques present a pattern that allows their recognition by the ML algorithm. It would also be interesting to re-perform the camera calibration, to

ensure a minimum reprojection error, and to implement a Linear Kalman Filter for bad pose rejection. The use of a wide-angle camera would ensure the positioning of the receiver closer to the luminaires, so it would also be interesting to test the performance of the system with a wide-angle lens. Finally, it would be important to validate the system for different tilts and heights of the receiver.

Bibliography

[1] Gordon Wells, Christophe Venaille, and Carme Torras. Vision-based robot positioning using neural networks. *Image and Vision Computing*, 14:715–732, 12 1996.

[2] Navneet Singh, Sangho Choe, and Rajiv Punmiya. Machine learning based indoor localization using wi-fi rssi fingerprints: An overview. *IEEE Access*, 9:127150–127174, 2021.

[3] Francesco Pascale, Ennio Andrea Adinolfi, Massimiliano Avagliano, Venanzio Giannella, and Andres Salas. A low energy iot application using beacon for indoor localization. *Applied Sciences*, 11:4902, 05 2021.

[4] Trong-Hop Do and Myungsik Yoo. An in-depth survey of visible light communication based positioning systems. *Sensors*, 16(5), 2016.

[5] Ali Yassin, Youssef Nasser, Mariette Awad, Ahmed Al-Dubai, Ran Liu, Chau Yuen, Ronald Raulefs, and Elias Aboutanios. Recent advances in indoor localization: A survey on theoretical approaches and applications. *IEEE Communications Surveys Tutorials*, 19(2):1327–1346, 2017.

[6] S. Gezici, Zhi Tian, G.B. Giannakis, H. Kobayashi, A.F. Molisch, H.V. Poor, and Z. Sahinoglu. Localization via ultra-wideband radios: a look at positioning aspects for future sensor networks. *IEEE Signal Processing Magazine*, 22(4):70–84, 2005.

[7] Zhaocheng Wang, Qi Wang, Wei Huang, and Zhengyuan Xu. Visible light communications : Modulation and signal processing. 2017.

[8] Jullia Cristiana Romina BIRSAN, Florica Moldoveanu, Alin Moldoveanu, Maria-Iuliana Dascalu, and Anca MORAR. Key technologies for indoor positioning systems. In *2019 18th RoEduNet Conference: Networking in Education and Research (RoEduNet)*, pages 1–7, 2019.

[9] Neha Chaudhary, Luis Nero Alves, and Zabih Ghassemblooy. Current trends on visible light positioning techniques. In *2019 2nd West Asian Colloquium on Optical Wireless Communications (WACOWC)*, pages 100–105, 2019.

[10] Jianli Jin, Lifang Feng, Jianping Wang, Danyang Chen, and Huimin Lu. Signature codes in visible light positioning. *IEEE Wireless Communications*, pages 1–7, 2021.

[11] Aleksandar Jovicic, Junyi Li, and Tom Richardson. Visible light communication: opportunities, challenges and the path to market. *IEEE Communications Magazine*, 51(12):26–32, 2013.

[12] Peixi Liu, Tianqi Mao, Ke Ma, Jiaxuan Chen, and Zhaocheng Wang. Three-dimensional visible light positioning using regression neural network. In *2019 15th International Wireless Communications Mobile Computing Conference (IWCMC)*, pages 156–160, 2019.

[13] Jie Lian, Zafer Vatansever, Mohammad Noshad, and Maïté Brandt-Pearce. Indoor visible light communications, networking, and applications. *Journal of Physics: Photonics*, 1(1):012001, jan 2019.

[14] David Plets, Sander Bastiaens, Luc Martens, and Wout Joseph. An analysis of the impact of led tilt on visible light positioning accuracy. *Electronics*, 8(4):389, 2019.

[15] Nicu Sebe, Ira Cohen, and Ashutosh Garg. *Machine Learning in Computer Vision*, volume 29. 01 2005.

[16] Md Habibur Rahman, Mohammad Abrar Shakil Sejan, Jong-Jin Kim, and Wan-Young Chung. Reduced tilting effect of smartphone cmos image sensor in visible light indoor positioning. *Electronics*, 9(10), 2020.

[17] Weipeng Guan, Xinjie Zhang, Yuxiang Wu, Zekun Xie, Jingyi Li, and Jieheng Zheng. High precision indoor visible light positioning algorithm based on double leds using cmos image sensor. *Applied Sciences*, 9(6), 2019.

[18] Xu Sun, Wenxiao Shi, Qing Cheng, Wei Liu, Zhuo Wang, and Jiadong Zhang. An led detection and recognition method based on deep learning in vehicle optical camera communication. *IEEE Access*, 9:80897–80905, 2021.

[19] Ramon Brena, Juan García-Vázquez, Carlos Galván Tejada, D. Munoz, Cesar Vargas-Rosales, James Fangmeyer Jr, and Alberto Palma. Evolution of indoor positioning technologies: A survey. *Journal of Sensors*, 2017, 03 2017.

[20] Miguel Rêgo and Pedro Fonseca. Occ based indoor positioning system using a smartphone camera. In *2021 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, pages 31–36, 2021.

[21] Md Habibur Rahman and Mohammad Abrar Shakil Sejan. Performance analysis of indoor positioning system using visible light based on two-leds and image sensor for different handhold situation of mobile phone. In *2020 IEEE Region 10 Symposium (TENSYMP)*, pages 1515–1518, 2020.

[22] Weipeng Guan, Shihuan Chen, Shangsheng Wen, Zequn Tan, Hongzhan Song, and Wenyuan Hou. High-accuracy robot indoor localization scheme based on robot operating system using visible light positioning. *IEEE Photonics Journal*, 12(2):1–16, 2020.

[23] Turtlebot. `https://www.turtlebot.com/`. (Accessed on 11/20/2021).

[24] Muhammad Saadi, Touqeer Ahmad, Yan Zhao, and Lunchakorn Wuttisttikulkij. An led based indoor localization system using k-means clustering. In *2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 246–252, 2016.

[25] Why deep learning over traditional machine learning? — by sambit mahapatra — towards data science. https://towardsdatascience.com/why-deep-learning-is-needed-over-traditional-machine-learning-1b6a99177063. (Accessed on 10/14/2021).

[26] Weipeng Guan, Jingyi Li, Shangsheng Wen, Xinjie Zhang, Yufeng Ye, Jieheng Zheng, and Jiajia Jiang. The detection and recognition of rgb-led-id based on visible light communication using convolutional neural network. *Applied Sciences*, 9(7), 2019.

[27] Github - ultralytics/yolov5: Yolov5 in pytorch > onnx > coreml > tflite. https://github.com/ultralytics/yolov5. (Accessed on 09/13/2021).

[28] Coco - common objects in context. https://cocodataset.org/#home. (Accessed on 11/12/2021).

[29] Linlin Zhu, Xun Geng, Zheng Li, and Chun Liu. Improving yolov5 with attention mechanism for detecting boulders from planetary images. *Remote Sensing*, 13(18), 2021.

[30] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection, 2017.

[31] Kaixin Wang, Jun Hao Liew, Yingtian Zou, Daquan Zhou, and Jiashi Feng. Panet: Few-shot image semantic segmentation with prototype alignment, 2020.

[32] Qisong Song, Shaobo Li, Qiang Bai, Jing Yang, Xingxing Zhang, Zhiang Li, and Zhongjing Duan. Object detection method for grasping robot based on improved yolov5. *Micromachines*, 12(11), 2021.

[33] Anchor boxes for object detection - matlab & simulink. https://www.mathworks.com/help/vision/ug/anchor-boxes-for-object-detection.html. (Accessed on 11/16/2021).

[34] Bin Yan, Pan Fan, Xiaoyan Lei, Zhijie Liu, and Fuzeng Yang. A real-time apple targets detection method for picking robot based on improved yolov5. *Remote Sensing*, 13(9), 2021.

[35] The beginner's guide to implementing yolov3 in tensorflow 2.0. https://machinelearningspace.com/yolov3-tensorflow-2-part-1/#nms-unique. (Accessed on 11/11/2021).

[36] Fangbo Zhou, Huailin Zhao, and Zhen Nie. Safety helmet detection based on yolov5. In *2021 IEEE International Conference on Power Electronics, Computer Applications (ICPECA)*, pages 6–11, 2021.

[37] Guanhao Yang, Wei Feng, Jintao Jin, Qujiang Lei, Xiuhao Li, Guangchao Gui, and Weijun Wang. Face mask recognition system with yolov5 based on image recognition. In *2020 IEEE 6th International Conference on Computer and Communications (ICCC)*, pages 1398–1404, 2020.

[38] First autonomous indoor drone by blue jay which navigates using vlc technology - philips lighting. https://www.signify.com/global/our-company/news/press-

release-archive/2017/20170615-children-in-dutch-hospital-play-game-with-worlds-first-autonomous-indoor-drone-developed-by-blue-jay. (Accessed on 11/09/2021).

[39] Philips lighting installs first supermarket with indoor positioning in germany - philips lighting. https://www.signify.com/global/our-company/news/press-release-archive/2017/20170306-philips-lighting-installs-first-supermarket-with-indoor-positioning-in-germany. (Accessed on 11/09/2021).

[40] Personalized shopping experience – aswaaq, dubai — interact retail. https://www.interact-lighting.com/global/customer-stories/aswaaq. (Accessed on 11/09/2021).

[41] Christoph Herrmann. Human eeg responses to 1-100 hz flicker: resonance phenomena in visual cortex and their potential correlation to cognitive phenomena. *Experimental Brain Research, v.137, 346-353 (2001)*, 137, 04 2001.

[42] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. In David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *Computer Vision – ECCV 2014*, pages 740–755, Cham, 2014. Springer International Publishing.

[43] Ross B. Girshick. Fast r-cnn. *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1440–1448, 2015.

[44] Trong-Hop Do and Myungsik Yoo. Analysis on visible light communication using rolling shutter cmos sensor. In *2015 International Conference on Information and Communication Technology Convergence (ICTC)*, pages 755–757, 2015.

[45] Trong-Hop Do and Myungsik Yoo. Performance analysis of visible light communication using cmos sensors. *Sensors*, 16(3), 2016.

[46] Ml practicum: Image classification — google developers. https://developers.google.com/machine-learning/practica/image-classification/convolutional-neural-networks. (Accessed on 11/20/2021).

[47] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

[48] Albumentations: fast and flexible image augmentations. https://albumentations.ai/. (Accessed on 10/17/2021).

[49] Object detection metrics with worked example — by kiprono elijah koech — towards data science. https://towardsdatascience.com/on-object-detection-metrics-with-worked-example-216f173ed31e. (Accessed on 09/13/2021).

[50] Test platform for visible light positioning. https://1library.org/document/zk3j4g4y-test-platform-for-visible-light-positioning.html. (Accessed on 10/18/2021).

[51] Repositório institucional da universidade de aveiro: Visible light positioning using a camera. https://ria.ua.pt/handle/10773/29720. (Accessed on 11/04/2021).

[52] Fusion 360 — 3d cad, cam, cae & pcb cloud-based software — autodesk. https://www.autodesk.com/products/fusion-360/overview. (Accessed on 12/15/2021).

[53] Herkulex rds-0101 brackets by dmitrydzz - thingiverse. https://www.thingiverse.com/thing:1934614. (Accessed on 10/22/2021).

[54] 3d printable raspberrypi 3/4 b+ case by stamos p. https://www.myminifactory.com/object/3d-print-raspberrypi-3-4-b-case-113761. (Accessed on 10/25/2021).

[55] Raspberry pi camera case - 55 degree mounting angle by iamaustin - thingiverse. https://www.thingiverse.com/thing:4560164. (Accessed on 10/29/2021).

[56] Intuitively understanding convolutions for deep learning — by irhum shafkat — towards data science. https://towardsdatascience.com/intuitively-understanding-convolutions-for-deep-learning-1f6f42faee1. (Accessed on 10/27/2021).

[57] Why you should use convolutions in your next neural net — using tensorflow — by tom allport — better programming. https://betterprogramming.pub/why-you-should-use-convolutions-in-your-next-neural-net-using-tensorflow-37d347544454. (Accessed on 10/27/2021).

[58] Activation functions in neural networks - geeksforgeeks. https://www.geeksforgeeks.org/activation-functions-neural-networks/. (Accessed on 10/27/2021).

[59] Understanding relu: The most popular activation function in 5 minutes! — by bharath k — towards data science. https://towardsdatascience.com/understanding-relu-the-most-popular-activation-function-in-5-minutes-459e3a2124f. (Accessed on 10/26/2021).

[60] Batch normalization in 3 levels of understanding — by johann huber — towards data science. https://towardsdatascience.com/batch-normalization-in-3-levels-of-understanding-14c2da90a338. (Accessed on 10/25/2021).

[61] Why is batch normalization useful in deep neural network? — towards data science. https://towardsdatascience.com/batch-normalisation-in-deep-neural-network-ce65dd9e8dbf. (Accessed on 11/24/2021).

[62] Cnn — introduction to pooling layer - geeksforgeeks. https://www.geeksforgeeks.org/cnn-introduction-to-pooling-layer/. (Accessed on 10/26/2021).

[63] Why dropout is so effective in deep neural network? — towards data science. https://towardsdatascience.com/introduction-to-dropout-to-regularize-deep-neural-network-8e9d6b1d4386. (Accessed on 11/24/2021).

[64] The most intuitive and easiest guide for convolutional neural network — by jiwon jeong — towards data science. https://towardsdatascience.com/the-most-intuitive-and-easiest-guide-for-convolutional-neural-network-3607be47480. (Accessed on 10/27/2021).

[65] Fully connected layer: The brute force layer of a machine learning model. https://iq.opengenus.org/fully-connected-layer/. (Accessed on 10/28/2021).

[66] Convolutional neural networks — a beginner's guide — by krut patel — towards data science. https://towardsdatascience.com/convolution-neural-networks-a-beginners-guide-implementing-a-mnist-hand-written-digit-8aa60330d022. (Accessed on 10/28/2021).

[67] Softmax activation function with python. https://machinelearningmastery.com/softmax-activation-function-with-python/. (Accessed on 10/26/2021).

[68] Home - opencv. https://opencv.org/. (Accessed on 11/16/2021).

[69] Opencv: Camera calibration and 3d reconstruction. https://docs.opencv.org/3.4/d9/d0c/group__calib3d.html#ga549c2075fac14829ff4a58bc931c033d. (Accessed on 09/13/2021).

[70] George Terzakis and Manolis I. A. Lourakis. A consistently fast and globally optimal solution to the perspective-n-point problem. Springer International Publishing, 2020.

[71] Opencv: Camera calibration and 3d reconstruction. https://docs.opencv.org/3.4/d9/d0c/group__calib3d.html#ga61585db663d9da06b68e70cfbf6a1eac. (Accessed on 11/16/2021).

[72] Opencv: Camera calibration and 3d reconstruction. https://docs.opencv.org/3.4/d9/d0c/group__calib3d.html#ga1aaacb6224ec7b99d34866f8f9baac83. (Accessed on 11/16/2021).