IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 15 January 2024, accepted 28 January 2024, date of publication 1 February 2024, date of current version 12 February 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3361326

RESEARCH ARTICLE

Precise Local Positioning of a Mobile Device Based on Pose Reconstruction From a Visible Light Beacon

JUAN D. GUTIÉRREZ[1], TEODORO AGUILERA[2], FERNANDO J. ÁLVAREZ[2], (Senior Member, IEEE), JORGE MORERA[2], AND FERNANDO J. ARANDA[2], (Member, IEEE)

[1]Department of Electronics and Computing, University of Santiago de Compostela, Lugo, 27002 Galicia, Spain
[2]Department of Electrical Engineering, Electronic and Automation, University of Extremadura, Badajoz, 06006 Extremadura, Spain

Corresponding author: Teodoro Aguilera (teoaguibe@unex.es)

This work was supported in part by MCIN/AEI/10.13039/501100011033 and European Union NextGenerationEU/PRTR under Project TED2021-129310B-I00, and in part by MCIN/AEI/10.13039/501100100011033 and FEDER—A way to make Europe under Project PID2021-122642OB-C42.

ABSTRACT This paper proposes an indoor positioning system for mobile devices using visible light beacons. The device's camera is used to acquire images in which the beacon appears. An algorithm processes these images to reconstruct the camera's pose at the photo acquisition time. This reconstruction allows for estimating the camera position accurately. The system operation is tested by a simulator based on Blender. This simulator allows for setting the beacon and camera characteristics, taking samples in rooms of different dimensions and analysing and studying the results obtained. In addition, an Android application for the real system has been developed to take samples and analyse them to estimate the mobile device's position. Finally, a comparison between the real and simulated systems is made. For this purpose, a test bench grid of 0.8 m × 1.1 m with 391 test points is designed. The simulator offers an average fidelity 97 % and can automate the sampling process. An average error of 7.49×10^{-3} m and coverage of 100 % are achieved with the camera and LED panel facing each other. The test bench real system achieves an average positioning error of less than 20.44×10^{-3} m, having a coverage close to 80 % and decreasing performance when the beacons are not fully captured. Finally, an experimental study of the system scalability has been carried out in a test room where four coded beacons have been deployed covering an area of $2.4 \times 3.6\,\text{m}^2$. Results over two different trajectories show reasonable losses of accuracy and coverage compared to the simulation and test bench, especially in the transition between beacons.

INDEX TERMS Local positioning, mobile device, pose reconstruction, visible light.

I. INTRODUCTION

In recent years, several lines of research have been developed to address the need for more accuracy and availability of GNSS in indoor environments. One of these research lines is based on visual light communication, where several approaches can be found [1]. For example, works such as those developed by [2] and [3] are based on the fingerprinting technique, i.e. they measure the Received Signal Strength (RSS) of light beacons and then apply different artificial intelligence techniques to the resulting radio map to determine with low accuracy the user's position. Others, such as the work presented by [4], propose developing a Visible Light Positioning System (VLPS) based on four LED lamps as transmitters and a Quadrant Photodiode Angular Diversity Aperture (QADA) as a receiver. This system was validated by simulation and experimental tests, obtaining 3-D positioning average errors below 13 cm in the worst case.

The associate editor coordinating the review of this manuscript and approving it for publication was Hadi Tabatabaee Malazi.

A different approach is developed by [5] where is presented a system based on Light-Emitting Diode (LED) beacons and CMOS cameras. With the help of additional sensors (an inclinometer and a magnetometer), they achieve accuracy in the low decimeter range. The centroid of each beacon is used for the location estimation. A 30 Frames per Second (FPS) video stream from a static digital camera, placed at a known height, is captured and processed in an auxiliary computer. The greater the number of beacons used, the better the accuracy provided in position estimation. The system has a coverage of $15 \times 20\,m^2$, and the positioning is evaluated at different heights offering a Mean Absolute Error (MAE) of 0.17 m when four or more beacons are used.

Another work of interest is that carried out by [6] in which a high-speed indoor VLPS is proposed by designing an elaborate flicker-free line coding scheme and a lightweight image processing algorithm. This system can provide cell phone positioning up to speeds of 5 m/s and achieve an accuracy of 7.5 cm with a processing time of 22.7 ms for a single beacon and 35.7 ms for two beacons. This work represents an evolution in this type of system since it allows image processing on the cell phone and offers processing times that make real-time positioning possible. Another work of similar characteristics can be found in [7], where the authors improve the system's accuracy up to 3.25 cm.

However, although the positioning of these systems is quite accurate, there is still room for improvement. The system proposed in this paper takes advantage of the whole beacon geometry by implementing the Pose Reconstruction Algorithm (PRA) [8] for the first time on an Android phone. This way, a minimum error of 0.28 mm is achieved with only one beacon, employing a processing time of 0.5 s, which is adequate to provide a real-time positioning rate. Also, the system proposed in this work is accompanied by a simulator that saves much time and effort in the system characterization. This tool allows the simulation of extreme conditions to test the system limits, analyze phenomena that could affect its performance, or study the implementation of possible improvements in the real system.

The usefulness of this type of simulator is evidenced by the large number of works in which a previous modelling of the real system is done. Some examples are the work developed by [9] to model a positioning system based on magnetic technology or the one developed in [10] for a UWB system. Some other works can be found when restricting the search to systems based on visible light. In [11], the authors propose modelling a VLPS to study its performance and accuracy depending on a series of configurable parameters, such as the location of the LED beacons, the room luminosity, and the camera's responsibility, among others. In [12], the authors propose a Indoor Location-Based Services (ILBS) based on cell identification (Cell-ID). In this work, four LED luminaires are simulated to position a receiver with the help of CandLES [13], a visible light communication simulator conducted by Boston University. This system has an average error of 20 cm on the selected cell and a maximum variance of $0.25\,m^2$ in the cells farthest from the beacons. In [14], LED beacons are proposed to provide accurate positioning within subway tunnels. When the beacons are 4 m high, the maximum error at ground level reaches 0.60 m for a person and 0.27 m for a vehicle. These results are based on the development of an analytical model. More recent work using the same approach can be found in [15], where visible light is combined with Machine Learning (ML) and theoretical models to evaluate the system's performance. The mean error when using the 2^{nd}-order regression model is within 4 cm and goes up to 5 cm when using the polynomial ML model. Finally, the work proposed by [16] uses a tool called *LedMapper* that allows the mapping of LED beacons in the environment using a pose reconstruction algorithm from the beacon image received through a smartphone camera. This tool is used to calibrate the visible light-based positioning system they proposed. To the authors' knowledge, none of these works mentioned above offer the possibility of interacting with the models or making extensions to adapt the simulator to the requirements of different experiments.

This work presents an evolution of the Blender simulator developed in [17], allowing it to switch from one simulated environment to another without further complication. Besides, the PRA [8] has been implemented for the first time in an Android phone, allowing to compare the phone's real positioning results with those obtained in the simulator. Additional improvements included in this work are summarized below:

- The simulator source code has been updated to work with the most recent Blender release.
- The positioning estimation is now based on the SQPnP pose reconstruction algorithm, which improves the overall system performance.
- A real-world test bench has been built to compare simulated and experimental results.
- A different round-corner detection algorithm has been included to provide a more accurate beacon detection.
- A system scalability study has been conducted in an office room where four coded beacons were deployed.

The rest of this article is organized as follows. In Section II, the simulator, its operation and the real system are described. Next, in Section III, some foundational ideas are discussed. A positioning algorithm is tested in the simulator and in the real system in Section IV. Finally, in Section V, the conclusions and future work derived from this work are extracted.

II. SYSTEM DESCRIPTION

A. BLENDER-BASED SIMULATOR

In this work, a Blender-based simulator is introduced. Blender [18] is a 3D modelling software distributed as open-source, which is also extensible via Python add-ons [19]. This simulator represents a significant long-term investment in developing improvements in the real system. It allows to systematically study the system performance under different conditions simply by modifying their corresponding

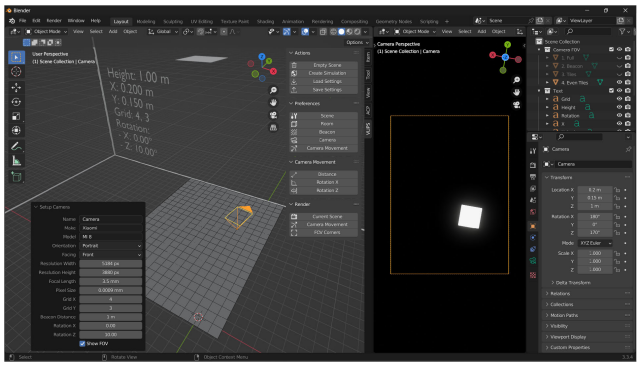


FIGURE 1. Blender add-on interface.

simulation parameters, saving researchers time and effort in taking measurements. The Blender project developed is shown in Fig. 1.

The proposed simulator should meet some specific requirements. First, it has to provide an interface to introduce the camera's characteristics, either a general-purpose camera or a smartphone one. In the latter case, its location (front or rear) should also be offered as an option. The said camera will be located at the origin of the coordinates, facing the ceiling and to a certain distance from it.

Also, the beacon side length (a squared panel) and its location should be configurable. Besides, a method to take either single samples or a series of them should be available to the user. The camera's location and rotation should be customizable when taking a series of samples. The resulting samples should include all the data needed to reproduce the same sampling session further. The proposed solution is provided as a Python add-on [20] that can be easily installed.

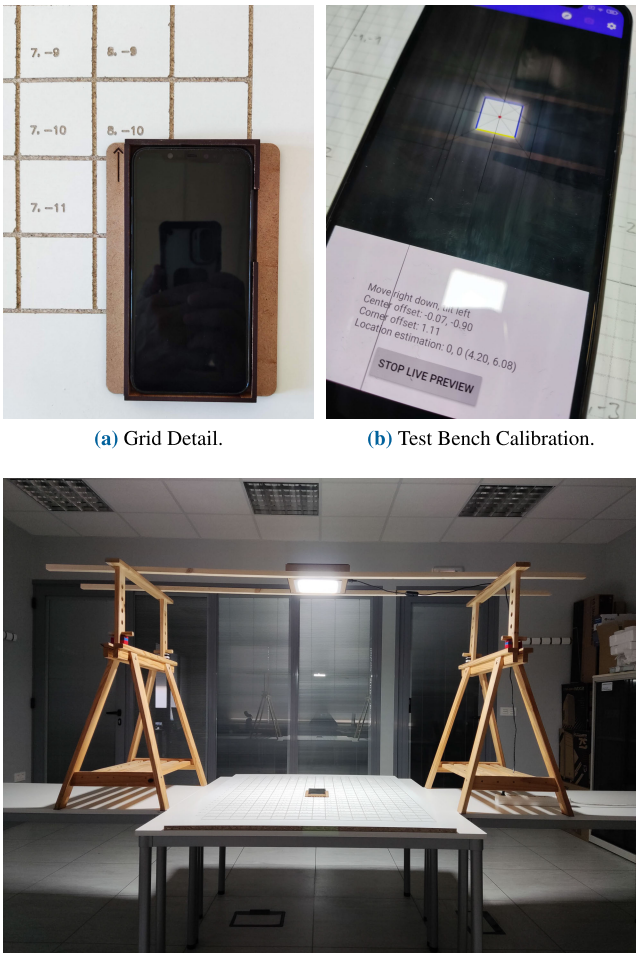
B. ACTUAL IMPLEMENTATION

An experimental test bench has been developed to validate the results obtained in the Blender-modeled system.

The first step is to create an accurate grid similar to the simulator's design. The grid has been carved with a Computer Numerical Control (CNC) machine in a wood plank whose dimensions are 0.8 m width by 1.1 m length, according to the simulation's Field of Vision (FOV) described in Section III-A.

Grid details are depicted in Fig. 2a, where each cell is marked with its coordinates. The coordinates' origin is in the centre of the grid. Each cell has a side of 0.05 m, resulting in 391 positions on the whole grid. The X-axis has been labelled, from left to right, with values from -8 to $+8$. The Y-axis takes values from 11 to -11, from top to bottom.

A squared beacon of 0.174 m side has been installed at 1 m height on this grid with the help of two adjustable stands. The smartphone's front camera faces the beacon-illuminated plane. The sampling application provides a calibration mode to determine when the camera axis is aligned just below the beacon centre as described in Fig. 2b. Once the grid centre is located, the grid plank is anchored to prevent its movement. The assembled test bench is shown in Fig. 2c.



(a) Grid Detail. (b) Test Bench Calibration.

(c) Full Setup.

FIGURE 2. Test bench.

The smartphone is placed at each point of the grid, starting at the upper left corner and ending at the lower right one, taking a sample and advancing to the next point until all 391 possible samples are taken. That means the first sample is taken at the box with coordinates $(-8, 11)$ and the last at $(8, -11)$. The smartphone is placed inside a framed box with rails in the back that perfectly fit the carved grid of the wooden plank. This system makes placing the camera in the desired coordinates easier, a detail that can be observed in Fig. 2.

III. FUNDAMENTALS

A. FIELD OF VISION

The camera's FOV could be defined as the portion of reality it can perceive. Calculating this area with accuracy is essential for two main reasons. First, if its dimensions are known, it can be divided into smaller cells, enabling a discretisation to evaluate the positioning system, comparing the actual position with the estimated one. Second, if the surface covered by each beacon is known, the number of beacons needed to cover a given surface can also be calculated. The cameras included in the smartphones follow the pinhole

model [21]. A 2D projection of the pinhole camera model is shown in Fig. 3.

The camera sensor dimensions (width and height) are represented by the 2×1 matrix $\boldsymbol{a}$. Similarly, the FOV dimensions (width and height) are represented by the 2×1 matrix $\boldsymbol{A}$. The distance between the camera and the beacon is D, and the camera's focal length is f. Note that the horizontal line that divides Fig. 3 through its centre defines four right-angled triangles. In the left lower triangle, $sin(\alpha_{ij}) = \frac{a_{i,j}}{2}$, where α_{ij} is the ij-th element of the 2×1 $\boldsymbol{\alpha}$ matrix and $f = cos(\alpha_{ij})$.

Similarly, in the upper right triangle $sin(\alpha_{ij}) = \frac{A_{ij}}{2}$, being A_{ij} the ij-th element of the 2×1 $\boldsymbol{A}$ matrix and $D = cos(\alpha_{ij})$. That way:

$$\left.\begin{aligned} \tan\boldsymbol{\alpha} &= \frac{\frac{a_{ij}}{2}}{f} \\ \tan\boldsymbol{\alpha} &= \frac{\frac{A_{ij}}{2}}{D} \end{aligned}\right\} \quad \Rightarrow \quad \frac{\frac{a_{ij}}{2}}{f} = \frac{\frac{A_{ij}}{2}}{D} \quad \Rightarrow \quad A_{ij} = a_{ij} \cdot \frac{D}{f} \tag{1}$$

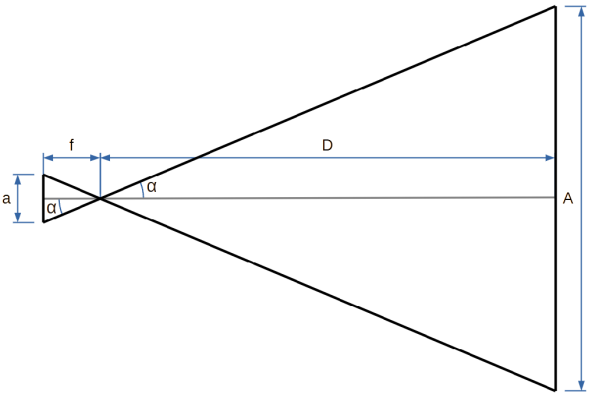


FIGURE 3. Pinhole camera model.

If 2×1 $\boldsymbol{R}$ matrix contains the camera samples width and height in pixels and P represents the pixel size (mm/px):

$$\boldsymbol{S} = \boldsymbol{R} \cdot P \tag{2}$$

where 2×1 $\boldsymbol{S}$ matrix elements are the width and height of the camera sensor in mm. Combining 1 and 2:

$$\boldsymbol{FOV} = \boldsymbol{S} \cdot \frac{D}{f} \tag{3}$$

where $\boldsymbol{FOV}$ contains the width and height of the FOV as a 2×1 matrix. The FOV dimensions can be obtained using 3, so it can be represented in the simulation. However, only half of the beacon would be visible when the camera location is on the FOV edges. Reducing the FOV's area by half of the beacon side length in each dimension this problem is solved:

$$\boldsymbol{FOV} = \boldsymbol{S} \cdot \frac{D}{f} - \boldsymbol{B} \tag{4}$$

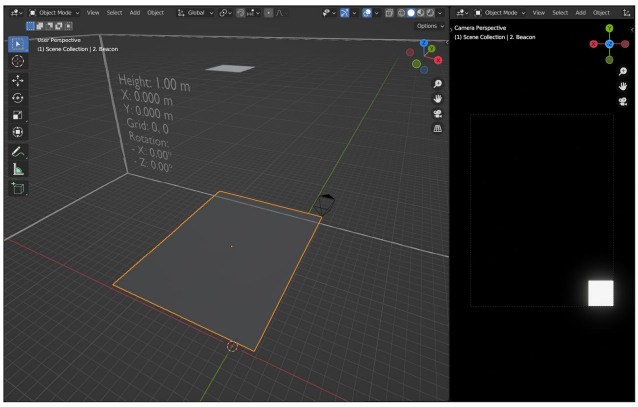


FIGURE 4. Camera in a corner of the beacon-adjusted FOV.

where $\boldsymbol{B}$ contains the beacon's width and height as a 2×1 matrix. Two variables define a digital camera's FOV: its sensor size and its focal length [22]. In this case, a third factor is present: the beacon size. Nevertheless, sampling every point in the FOV is not a practical way to evaluate the positioning performance in the simulated system. This evaluation requires discretising the available space, placing the camera in the corner of each cell and taking a sample in that position. The FOV centre is the origin of the coordinates. Dividing both sides of the FOV by the side of the cell and discarding the rest, the FOV calculated in 4 can be paved with a grid of square cells with known side:

$$\boldsymbol{\Delta} = \left\lfloor \frac{\boldsymbol{FOV}}{\delta} \right\rfloor \tag{5}$$

where $\boldsymbol{\Delta}$ components are the FOV's width and height in cells, expressed as a 2×1 matrix, and δ is the side length of each squared cell. Finally, the origin of coordinates could be in the centre of one cell if the division into cells results in an odd number of them. Obtaining $\boldsymbol{\Delta}$ mod 2 solves this case. If equal to zero, the result remains; else, the number of FOV cells are made even by decreasing them by one unit:

$$\boldsymbol{\Delta} = \begin{cases} \boldsymbol{\Delta} & \text{if } \boldsymbol{\Delta} \bmod 2 = 0 \\ \boldsymbol{\Delta} - 1 & \text{if } \boldsymbol{\Delta} \bmod 2 \neq 0 \end{cases} \tag{6}$$

With this in mind, the FOV can be obtained:

$$\boldsymbol{FOV} = \Delta \cdot \delta \tag{7}$$

Fig. 4 shows the FOV obtained using these equations to simulate a Xiaomi MI 8 [23] front camera, placed at 1 m height. It is equipped with a S5K3T1 sensor [24] developed by Samsung [25], with a pixel size of 0.9 µm. This camera is assembled with a focal length of 3.52 mm and its samples dimensions are 3880 px × 5184 px (width × height). The cell side length is 0.05 m. The resulting FOV is 0.8 m wide × 1.1 m long, containing 16 cells wide × 22 cells long.

B. POSITION ESTIMATION

This section describes the system used to estimate a person's position inside a room using a LED beacon and a photograph

taken with the camera of a smartphone held by the person himself. This system is developed both in simulation and in the real world.

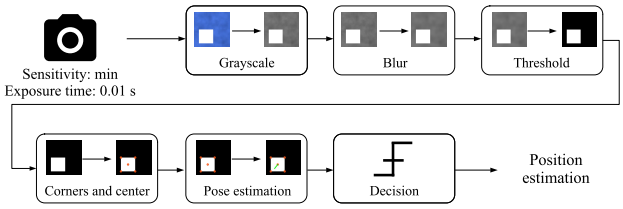


FIGURE 5. Position estimation system operation diagram.

Before going into detail about each position estimation system steps, described in Fig. 5, the process can be summarized as follows: first, the camera is prepared to take samples with a specific configuration that highlights the beacon outline. Next, these samples are further processed to accentuate the beacon outline even more. Following, the image is analyzed using computer vision techniques (such as contour localization or polygon approximation) to locate the beacon corners and find the beacon centre. Subsequently, the pose and the relative position between the camera and the beacon are estimated with the data obtained. This makes it possible to decide and estimate where the camera was inside the room when the sample was taken. Each step of the process is described in detail below.

1) SAMPLE COLLECTION

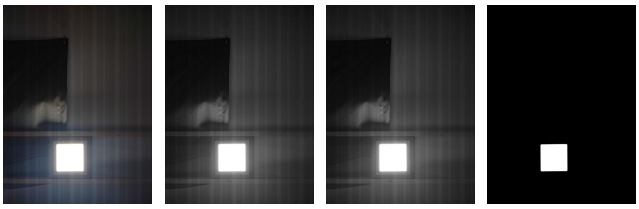
The sample must be taken to determine the beacon contour accurately. For this purpose, the camera sensitivity is set to take its minimum value. In addition, the exposure time is set to 0.01 s since it has been experimentally proven that this value offers the best results. A sample taken on the real system test bench at a distance of 1.2 m from the beacon is depicted in Fig. 7b.

2) SAMPLE PROCESSING

Before working with the samples obtained in the previous step, performing some operations is necessary to leave them in an optimal state. This processing consists of the following steps:

- **grayscale image conversion**: the acquired samples are in YUV format [26], which allows direct access to the luminous intensity data without the need to perform any operation on the original data. Since the image chromatic components contribute nothing to the decoding process, they can be discarded to obtain a grayscale image.
- **image blur**: a slight smoothing eliminates the image noise and optimizes the following steps results.
- **image thresholding**: to decode the image, it is necessary to work with zeros and ones instead of values between 0 and 255; the binary thresholding only keeps the most intense image values.

Everything that is not the beacon (black colour) shall correspond to the value zero, while the remaining area (white colour) shall be the beacon and take the value one. The steps taken in this process are described in Fig. 6 from the original sample until the final image is ready for analysis.



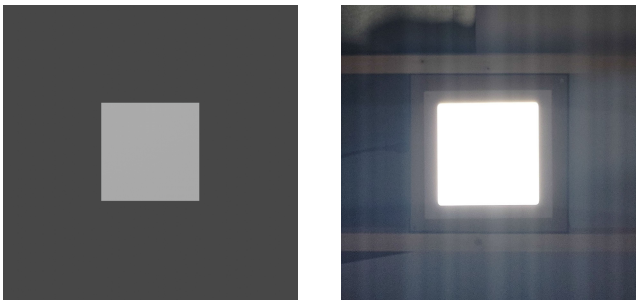
(a) Original. **(b)** Grayscale. **(c)** Blur. **(d)** Threshold.

FIGURE 6. Sample processing.

3) SAMPLE ELEMENTS SEARCH

OpenCV is a cross-platform, open-source package that provides multiple tools to perform computer vision tasks. As described in [8], a minimum of 4 points of the sample taken is needed to estimate the camera's pose. Using the geometry of the beacon, whose luminous surface is close to the shape of a perfect square, it is possible to obtain these four points by locating its corners. A fifth point, the centre of the beacon, is obtained for extra safety.

Locating the beacon corners of an ideal square beacon is a relatively simple task using the computer vision tools available in OpenCV. Fig. 7a shows a perfect square beacon. However, the beacon used in the simulator and real-world testing, shown in Fig. 7b, has rounded corners. This peculiarity must be considered when designing the algorithm to locate the beacon corners.



(a) Square Beacon. **(b)** Real Beacon..

FIGURE 7. Square corners vs rounded corners.

The beacon corner detection process begins with detecting the contours present in the black-and-white image resulting from the thresholding operation. Due to the data in these samples, the white areas correspond to the beacon, and the rest will be black. The contours detected in these samples will completely envelop the beacon, delimiting the area in which it is located. Even if the beacons were perfectly square, it does not mean that their outline is composed of four straight segments, corner to corner. Due to imperfections in the image acquired in the simulation and the real world, such a contour may consist of more than four line segments. These defects

can lead to differences in the corner location process for the ideal and the real beacon. The process, in both cases, starts with the beacon contour present in the sample but is addressed differently.

a: SQUARE CORNERS BEACON CASE

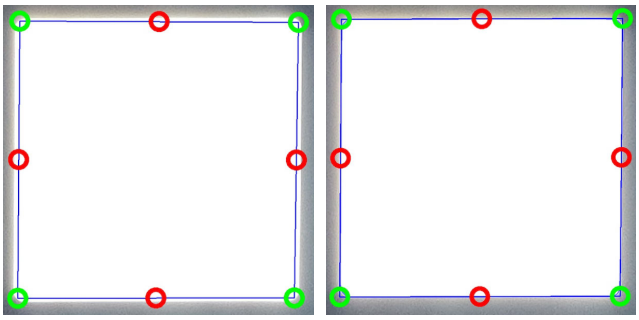
It is enough to approximate the contour to a polygon with fewer vertices to reduce the contour to its minimum expression. For this purpose, the Ramer-Douglas-Peucker algorithm [27] is used. This algorithm can reduce the number of segments defining a curve. As a criterion, a value determines the maximum distance between the original curve and its approximation. In this particular case, it is one-hundredth of the contour perimeter. The result is a contour with four corners in all cases tested. However, the detection process would be considered incorrect if the resulting contour did not have four corners.

b: ROUNDED CORNERS BEACON CASE

Due to the beacons used in the simulation and the real world having rounded corners, the starting contour comprises two distinct zones. On the one hand, there are straight zones on each beacon side and, on the other hand, curved zones at each corner. For this reason, approximating the contour using the Ramer-Douglas-Peucker algorithm is not optimal, as depicted in Fig. 8. The final contour is close to the expected contour but not containing the beacon totally since it leaves a gap without coverage at each rounded corner. To solve this problem, the first step is to convert the contour found into a convex hull using Sklansky's algorithm [28] and then use the Ramer-Douglas-Peucker algorithm, but this time keeping identical the perimeter size. As a result, the contour is maintained around the beacon, but the number of vertices is smaller, facilitating further work.

Subsequently, the list of points that trace the beacon contour is arranged. This list of points is transformed into a list of consecutively linked segments. The last segment is linked to the first segment to complete the contour. A list of segment lengths is stored together with the list of segments. The segments with the shortest length are those that define the corner curves. The segment's average length is calculated, discarding those below this value. The longest segments, which compose the beacon's sides, are concatenated to form a single segment. Finally, the four longest ones are taken, and their intersection, which corresponds to the corners of the beacon, is calculated. The result can be seen in Fig. 8b.

In both cases (simulation and real world), segments are labelled counterclockwise. The corners are marked in green dots, and the midpoint of each side is in red. The intersection of two diagonal segments determines the beacon's centre. With the four corners and the centre, the camera's pose can be estimated from a single sample. In Fig. 8a, the beacon contour is not optimally detected because the rounded corners of the beacon are not correctly identified. However, in Fig. 8b, this problem is solved, and the contour is identified accurately.



(a) Wrongly detected. **(b)** Correctly detected.

FIGURE 8. Contour detection comparison.

Note that the rounded corner detection improves the distance estimation to the beacon. If the estimated contour area is smaller (see Fig. 8a) than the actual one (see Fig. 8b), the PRA will calculate that the beacon is farther away than it really is.

4) CAMERA TRANSLATION AND ROTATION ESTIMATION

As discussed in [8], the SQPnP algorithm optimises the PnP problem by finding the minima that satisfy the problem conditions. OpenCV implements this algorithm through the `solvePnP` method. This method provides an implementation of the SQPnP algorithm if the value `SOLVEPNP` is given, and the following data are available:

- the object model points whose pose is to be estimated.
- the points of that object in an image.
- the camera intrinsic matrix used to capture the image.
- the camera distortion coefficients.

a: OBJECT MODEL POINTS

In this work, the object model describes its characteristics as if it were precisely in the camera projection plane. That is, it is the beacon's dimensions if there is no distance between its surface and the handheld camera. To create this model, it is necessary to know the side of the square beacon and the camera's pixel size. The first value is available since the beacon geometrical properties are known. The second value is provided by the optical sensor manufacturer.

The centre of the squared beacon model is at the origin of coordinates $(0, 0)$, which divides the beacon into four quadrants. The side of the object model is calculated by multiplying the side length by the camera pixel size. The beacon side is measured using the decimal metric system. The pixel size units are in those same units per pixel. The result, therefore, is in pixels. Hence, if the beacon side measured 174 mm and the camera pixel size were 0.9 µm/px, the Beacon Model Side (BMS) in pixels would be:

$$\begin{aligned} \text{BMS} &= \frac{\text{beacon side}}{\text{pixel size}} = \frac{174}{0.9}\frac{\text{mm}}{\mu\text{m/px}} \\ &= \frac{174}{0,9}\frac{10^{-3}}{10^{-6}}\frac{\text{m}}{\text{m/px}} = 193\,333.33\,\text{px} \end{aligned} \tag{8}$$

Consequently, if the beacon plane were attached to the camera and its sensor could capture the beacon completely,

its side would occupy 193 333.33 px. At this value, with the beacon centre at the coordinate origin, each corner of the beacon model, starting from the upper left and moving clockwise, would be at:

- (−96 666.66 , +96 666.66).
- (+96 666.66 , +96 666.66).
- (+96 666.66 , −96 666.66).
- (−96 666.66 , −96 666.66).

For different beacon contours, the object model would have to be created in an equivalent way. Since the Levenberg–Marquardt Algorithm (LMA) needs at least four points to estimate a flat object's pose, the beacon's geometry could be at least triangular, using the three vertices and the barycenter. At the opposite extreme would be beacons with circular or oval shapes, where the necessary points should be taken from their contour to have an error below the tolerance margin established when approximating their shape by a polygon.

b: OBJECT IMAGE POINTS

How the object points are located in the sample is described in Section III-B3. There, this task appears second in the list of requirements for pose estimation because it is the second parameter that the `solvePnP` method needs. In practice, it is the condition without which pose estimation cannot proceed. Therefore, pose estimation cannot be performed if it is impossible to find the four corners and the centre of the beacon in the sample.

c: INTRINSIC CAMERA MATRIX

The camera intrinsic matrix I contains camera-specific details, such as its focal length and the deviation of its optical centre. This matrix is part of the camera sensor's projection system of real-world objects. The focal length $f = (f_x, f_y)$ is a parameter that depends on the handheld device manufacturer and determines the space left between the lens and the camera sensor. It usually has the same value in both coordinates, so $f_x = f_y$. This value is in the specifications of the handheld device and usually appears among the Exif fields [29] included in the images captured with the sensor. Moreover, the camera's optical centre $c = (c_x, c_y)$ matches the centre of the captured image.

During the development of the positioning system, the front camera of a Xiaomi Mi 8 was used. Its pixel size is 0.9 µm/px, its focal length 3.52 mm, and the samples, in portrait mode, have a width of 3880 px and a height of 5184 px. Therefore, in this particular case, the intrinsic matrix of the camera is:

$$f_x = f_y = \frac{3.52}{0.9}\frac{\text{mm}}{\mu\text{m/px}} = \frac{3.52}{0.9}\frac{10^{-3}}{10^{-6}}\frac{\text{m}}{\text{m/px}} = 3911.11\,\text{px}$$

$$c_x = \frac{3880\,\text{px}}{2} = 1940\,\text{px} \quad ; \quad c_y = \frac{5184\,\text{px}}{2} = 2592\,\text{px}$$

$$\mathbf{I} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 3911.11 & 0 & 1940 \\ 0 & 3911.11 & 2592 \\ 0 & 0 & 1 \end{bmatrix} \tag{9}$$

d: CAMERA DISTORTION COEFFICIENTS

In the pinhole camera model (see Section III-A), a lens is placed between the object and the optical sensor to concentrate the light in the sensor. These lenses are imperfect and can exhibit different types of distortion that affect image formation. The most commonly used system for estimating these coefficients consists of a calibration process during which the camera takes a certain number of photographs of a known pattern, which are then analyzed for aberrations. After this process, the distortion coefficients are incorporated into the image processing process to obtain a more accurate result [30].

Although adding a calibration step to the system developed here could improve the results obtained, it has been discarded because the process would annoy the user. Future improvements could be incorporated into the system, including not only the estimation of distortion coefficients but the development of a calibration process that is as unobtrusive as possible. However, in its current state, the system assumes no distortion exists in the lens.

All the data needed to perform the pose estimation are available now. The main result of the process is a Boolean flag that determines whether or not it is possible to reconstruct the pose. If it is not, another sample is taken to repeat the entire process from the beginning. If it was, `solvePnP` provides a rotation vector and a translation vector to estimate the relative position between the camera and the beacon when the sample was taken.

IV. EXPERIMENTAL RESULTS

A. SIMULATION FIDELITY

Before comparing the results obtained in the simulated and real environments, measuring the simulation's fidelity is necessary. To do so, a comparison between the samples obtained in both environments has been performed. The camera characteristics correspond to the rear camera of a smartphone, Xiaomi Mi 8 [31]. It is equipped with an IMX363 Exmor RS sensor [32] developed by Sony [33], with a pixel size of 1.40 µm. This camera is assembled with a focal length of 4.216 mm and its samples dimensions are 3024 px × 4032 px (width × height).

Fig. 9 compares the samples obtained in the simulator and the real world when the camera is aligned parallel to the beacon surface. The simulated sample is the inner white square, while the real sample is the outer greyish square.

A Python script that accepts a simulated sample and a real one has been developed to compare them. The comparison process starts thresholding both samples to obtain their black-and-white representation, so the beacons' luminous parts are isolated. Then, the length in pixels of the top beacon side is measured for both images to establish the comparison. Finally, the results are stored in a comma-separated values (CSV) file and can be seen in Fig. 10.

The samples compared are taken at intervals ranging from 1 m to 3 m, in 0.1 m steps. The abscissa axis shows these distances. The ordinate axis shows the ratio between the top

FIGURE 9. Comparison of a sample in the simulation and real-world (inner and outer squares, respectively).

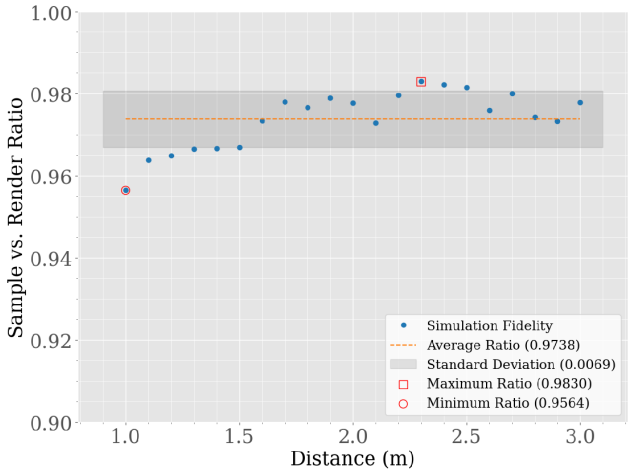


FIGURE 10. Simulation fidelity.

side length of the real and the simulated beacons. It has been obtained an average ratio of 0.9738 with a standard deviation of 0.0069. Detecting the beacon's rounded corners contributes a 1% improvement to this result.

In the real-world tests, the beacon will be located in the ceiling at a minimum distance of 1 m to the camera. In that case, the fidelity ratios are above 0.95. Under these circumstances, it can be concluded that the simulation reliability is acceptable.

B. SIMULATED SYSTEM RESULTS

A set of samples covering the whole FOV has been taken to validate the simulator experimentally. The side of each square cell is 0.05 m. With a distance of 1 m between the beacon and the camera, said FOV is composed of 391 positions. Its

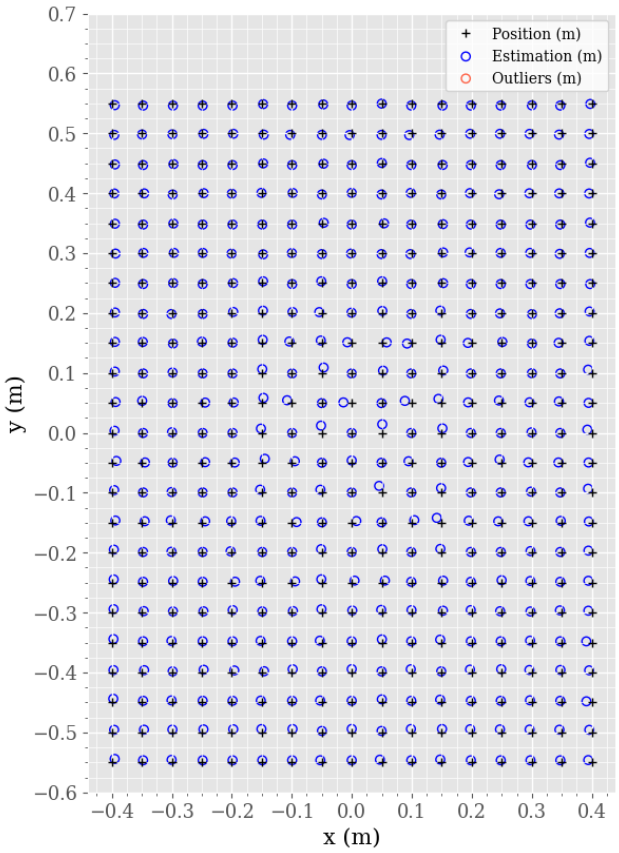


FIGURE 11. Simulated positioning results in the entire FOV.

wide is 0.8 m and its length is 1.1 m. Each sample is processed with the algorithm based on the camera pose reconstruction presented in Section III-B. In Fig. 11 can be observed the result obtained.

The black cross marks the position at which each sample is taken. The positions estimated by the positioning algorithm are marked with blue circles, and potential outliers with a 2D error above the length of the cell side are also evaluated and marked with red circles. This figure shows that the positioning accuracy is relatively high in all the evaluated points, with no outliers appearing.

Fig. 12 shows these positioning results as a heat map. Note that positioning results deviate from the ideal behaviour around the centre of the FOV, although their errors are not enough to be considered outliers. These higher errors are due to the closer distance between the camera and the beacon in these areas, obtaining larger and more detailed beacon images in which the realistic effects modelled in the simulator have a greater influence on the results.

The positioning results and the coverage percentage obtained are detailed in TABLE 1.

Reproducing the limiting cases is possible thanks to the simulator interactivity, making it easier to study different adverse situations and search for possible solutions that lead to improvements in the performance of the position estimation algorithm.

TABLE 1. Simulation positioning results (10^{-3} *m*).

Average Error	7.49
Standard Deviation	1.32
Minimum Error	5.07
Maximum Error	15.82
Coverage (%)	100

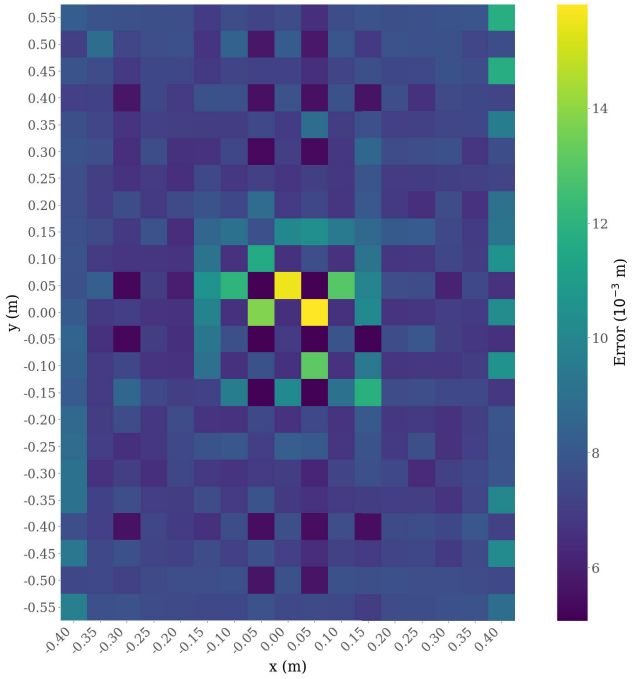


FIGURE 12. Heatmap of positioning results over the entire FOV in the simulation.

C. REAL SYSTEM RESULTS

The positioning results of the tests performed on this bench are shown in Fig. 13. Analogously to what was established in the simulator, outliers have been considered as those whose error in two dimensions exceeds the confidence margin, established as the side length of each cell, i.e. 0.05 m.

To better understand the results obtained in the real system, Fig. 14 shows the error made in the estimation at each position of the test bench as a heat map. As shown, the most significant errors are found at the left and right ends of the grid, where the beacon could not be fully captured. These results confirm the negative impact that partial beacon captures have on the performance of the positioning system. This effect is also noticeable in the bottom end of the grid, where a smaller portion of the beacon surface was lost when taking the samples.

Similarly, there is a circular area around the centre of the test bench where the error is slightly higher than the confidence margin (maximum 0.11 m vs confidence margin of 0.5 m), so that the estimated positions are marked in red in Fig. 13 and lighter blue in Fig. 14. This error is due to the distortion produced by the camera lens used to take the samples [34].

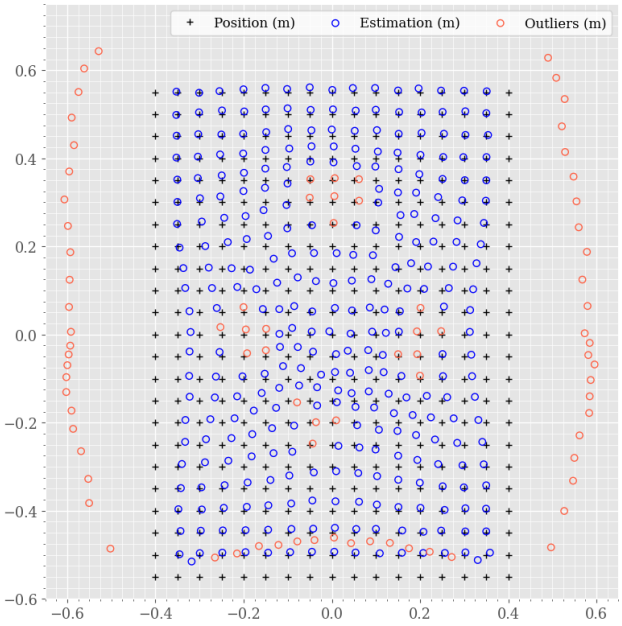


FIGURE 13. Real system positioning results in the entire FOV.

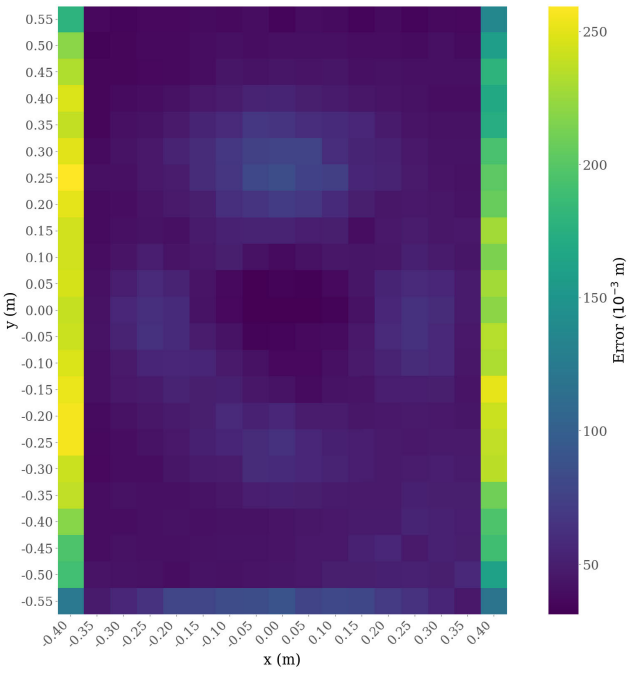


FIGURE 14. Heatmap of positioning results over the entire FOV in the real system.

The coverage obtained in the simulator is 100 % (see Fig. 11). In this test bench, the coverage obtained under the same circumstances is almost 80 %. The greatest loss of coverage is at the edges of the test bench, where the beacon cannot be completely captured. Besides, the lens distortion also influences the coverage in the central area. It should also be noted that the difference between samples taken in the periphery for the simulator and reality is due to smartphone lens aberration, which is not modelled in the

simulator. Incorporating this characteristic into the simulator will be explored in future work.

In TABLE 2, some parameters that quantitatively describe the accuracy in two-dimensional positioning and the coverage achieved in this test are provided. These values are calculated only for those samples with an estimation error below the confidence margin, as was done in the simulation tests. The mean error is 20.44 m, achieving a 79.03 % coverage across the entire grid of test points.

TABLE 2. Test bench positioning results (10^{-3} *m*).

Average Error	20.44
Standard Deviation	13.65
Minimum Error	0.28
Maximum Error	49.79
Coverage (%)	79.03

D. SYSTEM SCALABILITY

To study the system's behaviour in an everyday positioning environment, four luminaires are placed on the ceiling of a clear office room at 2.72 m above the floor. Each beacon has been coded by Visual Light Communication (VLC), emiting a 5-bit header followed by a 8-bit Biphase Mark Code (BMC) [35] at a sampling frequency of 2.5 kHz. Hence, the cell phone can identify the captured luminaire image, which are taken every 0.5 s continuously. A total positioning surface of $2.4 \times 3.6\,\mathrm{m}^2$ has been established where two trajectories have been followed, one in a zigzag and the other in a convergent square spiral. In both trajectories, the user has held the cell phone to a height of approximately 1.32 m. The test room set-up details are depicted in Fig. 15.

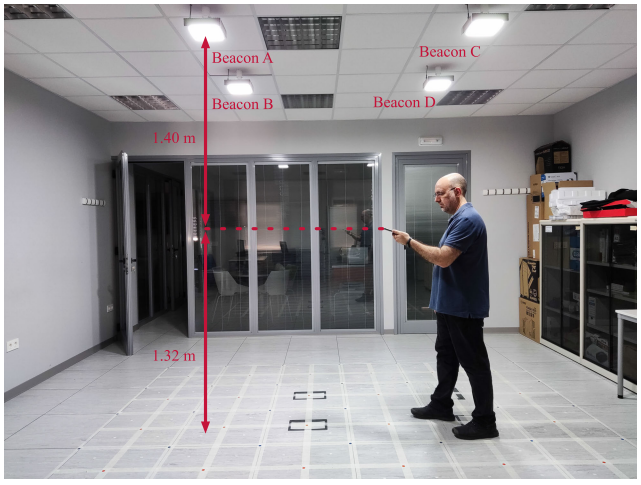


FIGURE 15. Test room with four beacons, showing the distance between them and the cell phone camera.

The chosen distance between the luminaire and the phone camera is based on a study to evaluate the success rate in detecting the codes emitted by VLC as a function of the reception distance. The results of this study are shown in Fig. 16, where for the chosen luminaire size, it can be noticed that a detection success rate of 100 % is achieved up to distances of 1.3 m between the luminaire and the camera. Beyond this distance, the success rate gradually decreases until the codes are undetectable at a distance of 3.0 m. Nevertheless, if greater detection distances are required, this can be solved by installing luminaires with a larger surface area.

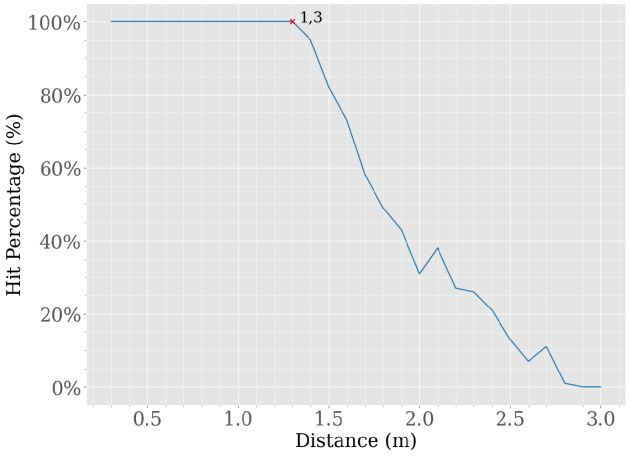


FIGURE 16. Success rate in identifying the codes associated with the beacons as a function of distance.

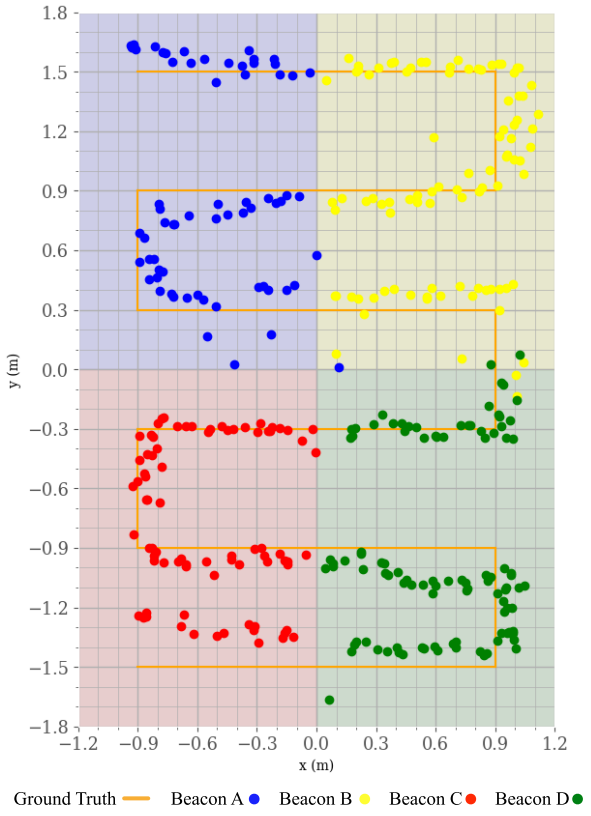


FIGURE 17. Positioning results in the zigzag trajectory.

An approximate minimum distance of 1.40 m from the camera to the beacons is chosen for the trajectories, meaning the maximum code detection rate is around 90 %, according

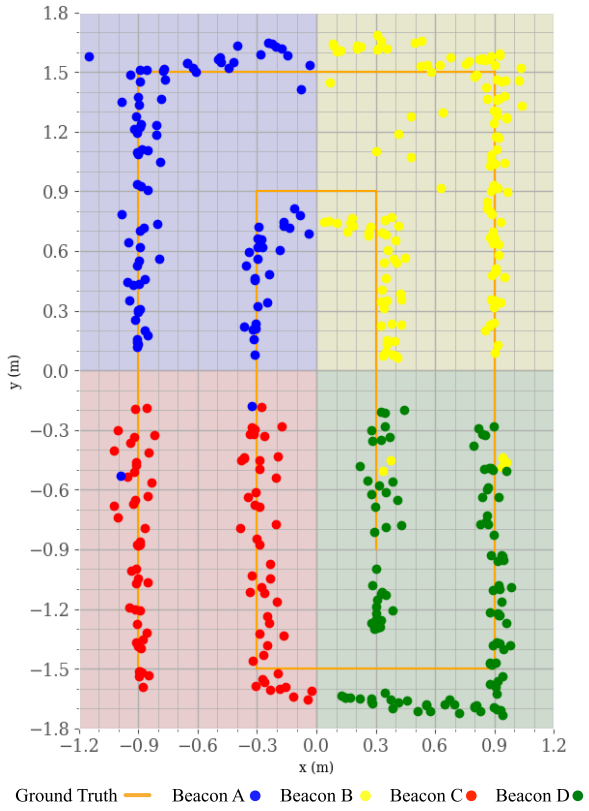


FIGURE 18. Positioning results in the spiral trajectory.

to Fig. 16. The positioning results obtained for the zigzag and the convergent square spiral trajectories can be seen in Fig. 17 and Fig. 18, respectively. In these figures, each beacon's coverage area has been coloured in a different soft tone. The positions estimated from the images captured from each one have been coloured with dots of their corresponding intense colour. In addition, a line in orange has also been drawn to define the path the person follows when moving along each trajectory.

Fig. 17 and Fig. 18 show reliable results close to the Ground Truth line along the complete trajectories. The density of points obtained during these trajectories is also adequate to provide a real-time user experience.

On the other hand, these results also show some losses of coverage and accuracy in the transition areas between different beacons. These accuracy losses are due to partially capturing one or more luminaire contours in these transition regions. This problem could be easily solved, for example, by including a fifth beacon in the center of the positioning area. Other accuracy losses are mainly produced because the cell phone trajectory is not precisely guided, but rather, the device is held in the person's hand leisurely, as a user would do in realistic everyday use. Despite these circumstances, the system still provides accurate positioning data as long as the tilt of the cell phone in the hand is not so extreme that beacons disappear from the captured image or make it unable to reconstruct its contour.

V. CONCLUSION AND FUTURE WORK

This work presents a visible-light indoor positioning system running on smartphones and a Blender simulator. The simulator enables the creation of rooms, cameras, and beacons. Also, it can take individual samples with a simulated camera with the same physical properties as a real camera to perform an analysis. Furthermore, multiple samples can be taken using the automation tools provided, moving the camera along different positions and even varying its orientation. The data necessary to reconstruct the camera and room characteristics are included in each sample. The simulator's reliability has been studied by comparing a set of samples obtained in the simulator with those obtained in the real world. The simulation is faithful to reality with a mean fidelity of 97 %.

The performance of the positioning system using the simulator was evaluated by placing the modelled smartphone and beacon face to face and 1 m away. A 391 test point grid, with dimensions 0.8 m $\times$ 1.1 m was used, obtaining a 100 % coverage and a mean error of 7.49×10^{-3} m. Subsequently, a real-world test bench replicating this setup was built, focusing on minimizing every possible human error that could affect the result. The grid was machined to a wood plank, and an enclosure to position the smartphone camera with accuracy was built. The positioning results offered a coverage close to 80 % and a mean error of 20.44×10^{-3} m.

In addition, an experimental study of the system scalability has been carried out in a test room where four coded beacons have been deployed. Two trajectories have been carried out, showing the system's feasibility in everyday positioning environments. However, there is some loss of accuracy and coverage compared to the simulation and one-beacon test bench, especially in the transition areas between beacons.

Work is in progress on various aspects to enhance the system scalability, such as reconstructing the contour of partially captured luminaire images and developing techniques to improve the Visual Light Communication (VLC) performance at longer distances. Besides, future system evolutions will consider more hostile environments for positioning where obstacles such as columns or furniture are present, relying on additional phone sensors such as the gyroscope, accelerometer and magnetometer. Regarding the simulator, improving its performance in light pollution conditions or modelling the lens distortion is necessary to bring its results closer to reality. Nevertheless, this simulator is a valuable analysis tool that saves time and effort when studying the behaviour of visible light indoor positioning systems.

REFERENCES

[1] J. Luo, L. Fan, and H. Li, "Indoor positioning systems based on visible light communication: State of the art," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 4, pp. 2871–2893, 4th Quart., 2017.

[2] N. Faulkner, F. Alam, M. Legg, and S. Demidenko, "Watchers on the wall: Passive visible light-based positioning and tracking with embedded light-sensors on the wall," *IEEE Trans. Instrum. Meas.*, vol. 69, no. 5, pp. 2522–2532, May 2020.

[3] A. H. A. Bakar, T. Glass, H. Y. Tee, F. Alam, and M. Legg, "Accurate visible light positioning using multiple-photodiode receiver and machine learning," *IEEE Trans. Instrum. Meas.*, vol. 70, pp. 1–12, 2021.
[4] E. Aparicio-Esteve, Á. Hernández, J. Ure na, and J. M. Villadangos, "Visible light positioning system based on a quadrant photodiode and encoding techniques," *IEEE Trans. Instrum. Meas.*, vol. 69, no. 8, pp. 5589–5603, Aug. 2020.
[5] G. Simon, G. Zachár, and G. Vakulya, "Lookup: Robust and accurate indoor localization using visible light communication," *IEEE Trans. Instrum. Meas.*, vol. 66, no. 9, pp. 2337–2348, Sep. 2017.
[6] J. Fang, Z. Yang, S. Long, Z. Wu, X. Zhao, F. Liang, Z. L. Jiang, and Z. Chen, "High-speed indoor navigation system based on visible light and mobile phone," *IEEE Photon. J.*, vol. 9, no. 2, pp. 1–11, Apr. 2017.
[7] M. H. Rahman and M. A. S. 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 *Proc. IEEE Region 10 Symp. (TENSYMP)*, Jun. 2020, pp. 1515–1518.
[8] G. Terzakis and M. Lourakis, "A consistently fast and globally optimal solution to the perspective-n-point problem," in *Computer Vision—ECCV* (Lecture Notes in Computer Science), A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, Eds. Cham, Switzerland: Springer, 2020, pp. 478–494.
[9] L. Ferrigno, M. Laracca, F. Milano, G. Cerro, P. Bellitti, M. Serpelloni, and O. C. Piedrafita, "Magnetic localization system for short-range positioning: A ready-to-use design tool," *IEEE Trans. Instrum. Meas.*, vol. 70, pp. 1–9, 2021.
[10] K. Paszek, D. Grzechca, and A. Becker, "Design of the UWB positioning system simulator for LOS/NLOS environments," *Sensors*, vol. 21, no. 14, p. 4757, Jul. 2021. [Online]. Available: https://www.mdpi.com/1424-8220/21/14/4757
[11] J. Q. J. Quan, B. B. Bo Bai, S. J. S. Jin, and Y. Z. Y. Zhang, "Indoor positioning modeling by visible light communication and imaging," *Chin. Opt. Lett.*, vol. 12, no. 5, pp. 52201–52204, 2014. [Online]. Available: https://opg.optica.org/col/abstract.cfm?URI=col-12-5-052201
[12] A. M. Vegni and M. Biagi, "An indoor localization algorithm in a small-cell LED-based lighting system," in *Proc. Int. Conf. Indoor Positioning Indoor Navigat. (IPIN)*, Nov. 2012, pp. 1–7.
[13] M. B. Rahaim, T. Borogovac, and J. B. Carruthers, "CandlES: Communication and lighting emulation software," in *Proc. 5th ACM Int. Workshop Wireless Netw. Testbeds, Exp. Eval. Characterization (WiNTECH)*. New York, NY, USA: ACM, Sep. 2010, pp. 9–14, doi: 10.1145/1860079.1860082.
[14] N. Krommenacker, Ó. C. Vásquez, M. D. Alfaro, and I. Soto, "A self-adaptive cell-ID positioning system based on visible light communications in underground mines," in *Proc. IEEE Int. Conf. Automatica (ICA-ACCA)*, Oct. 2016, pp. 1–7.
[15] Y.-C. Chuang, Z.-Q. Li, C.-W. Hsu, Y. Liu, and C.-W. Chow, "Visible light communication and positioning using positioning cells and machine learning algorithms," *Opt. Exp.*, vol. 27, no. 11, pp. 16377–16383, May 2019.
[16] Q. Liang, Y. Sun, C. Liu, M. Liu, and L. Wang, "LedMapper: Toward efficient and accurate LED mapping for visible light positioning at scale," *IEEE Trans. Instrum. Meas.*, vol. 71, pp. 1–12, 2022.
[17] J. D. Gutiérrez, T. Aguilera, F. J. Álvarez, J. Morera, and F. J. Aranda, "A blender-based simulation tool for visible light positioning with portable devices," in *Proc. IEEE Int. Instrum. Meas. Technol. Conf. (I2MTC)*, May 2022, pp. 1–6.
[18] Blender Foundation. *Home of the Blender Project—Free and Open 3D Creation Software*. Accessed: Feb. 5, 2024. [Online]. Available: https://www.blender.org/
[19] *Introduction—Blender Manual*. Accessed: Feb. 5, 2024. [Online]. Available: https://docs.blender.org/manual/en/2.93/advanced/scripting/
[20] *Add-Ons—Blender Manual*. Accessed: Feb. 5, 2024. [Online]. Available: https://docs.blender.org/manual/en/latest/editors/preferences/addons.html
[21] J. E. Solem, *Programming Computer Vision With Python: Tools and Algorithms for Analyzing Images*. Sebastopol, CA, USA: O'Reilly Media, 2012.
[22] R. Szeliski, *Computer Vision: Algorithms and Applications*. Cham, Switzerland: Springer, 2010. [Online]. Available: http://szeliski.org/Book/
[23] *Mi Global Home*. Accessed: Feb. 5, 2024. [Online]. Available: https://www.mi.com/global/mi8
[24] Samsung Official. *ISOCELL Slim 3T1 | Mobile Image Sensor*. Accessed: Feb. 5, 2024. [Online]. Available: https://www.samsung.com/semiconductor/image-sensor/mobile-image-sensor/S5K3T1/
[25] *Mobile | TV | Home Electronics | Home Appliances | Samsung U.S.* Accessed: Feb. 5, 2024. [Online]. Available: https://www.samsung.com/us/
[26] *FXScript Reference: RGB and YUV Color*. Accessed: Feb. 5, 2024. [Online]. Available: http://joemaller.com/fcp/fxscript_yuv_color.shtml
[27] D. H. Douglas and T. K. Peucker, "Algorithms for the reduction of the number of points required to represent a digitized line or its caricature," *Cartographica, Int. J. Geographic Inf. Geovisualization*, vol. 10, no. 2, pp. 112–122, Dec. 1973.
[28] J. Sklansky, "Finding the convex hull of a simple polygon," *Pattern Recognit. Lett.*, vol. 1, no. 2, pp. 79–83, Dec. 1982.
[29] (May 2019). *Exchangeable Image File Format for Digital Still Cameras: Exif Version 2.32*. [Online]. Available: http://cipa.jp/std/documents/download_e.html?DC-008-Translation-2019-E
[30] J. Weng, P. Cohen, and M. Herniou, "Camera calibration with distortion models and accuracy evaluation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 14, no. 10, pp. 965–980, Oct. 1992.
[31] (2018). *Xiaomi Mi 8—Specifications*. [Online]. Available: https://www.devicespecifications.com/en/model/c52c4895
[32] (May 2020). *Sony Exmor Camera Sensor*. [Online]. Available: https://web.archive.org/web/20100419131019/http://www.sony.net/SonyInfo/technology/technology/theme/exmor_r_01.html
[33] *Sony Semiconductor Solutions Group*. Accessed: Feb. 5, 2024. [Online]. Available: https://www.sony-semicon.co.jp/e/
[34] J. Park, S.-C. Byun, and B.-U. Lee, "Lens distortion correction using ideal image coordinates," *IEEE Trans. Consum. Electron.*, vol. 55, no. 3, pp. 987–991, Aug. 2009.
[35] P. Horowitz, "Biphase coding," in *The Art of Electronics*, 3rd ed. New York, NY, USA: Cambridge Univ. Press, 2015, p. 1041.

JUAN D. GUTIÉRREZ received the Ph.D. degree in visible LED light-based indoor positioning systems (IPS). He is currently an Assistant Professor with the University of Santiago de Compostela (USC). His training includes programming in different languages, system administration, application design, and databases and internet. What began as a fun experience in the mid-nineties has ended up being a real passion for him. With more than 20 years of experience in the computer world. He enjoys computing but, above all, learning new things. He has written more than 20 computer science books and translated another ten from English to Spanish. His current research interest includes the application of artificial intelligence to different fields of knowledge.

TEODORO AGUILERA received the bachelor's and M.Sc. degrees in physics and the Ph.D. degree in electronics from the University of Extremadura, Spain, in 2009 and 2011, respectively. Since 2009, he has been with the Department of Electrical Engineering, Electronics and Automation, University of Extremadura, where he was an Associate Lecturer and a Research Fellow with the Sensory Systems Group. His work lies in the design of acoustic local positioning systems (ALPS) and signal processing.

FERNANDO J. ÁLVAREZ (Senior Member, IEEE) received the M.Sc. degree in physics from the University of Sevilla, Spain, in 1998, the Ph.D. degree in electronics from the University of Alcalá, Alcalá de Henares, Spain, in 2006, the M.Sc. degree in electronic engineering from the University of Extremadura, Badajoz, Spain, in 2012, and the M.Sc. degree in signal theory and communications from the University of Vigo, Vigo, Spain, in 2014. Since 2001, he has been with the Department of Electrical Engineering, Electronics and Automation, University of Extremadura, where he is currently a Full Professor and the Head of the Sensory Systems Research Group. In 2008, he joined the Intelligent Sensors Laboratory, Yale University, New Haven, CT, USA, as a Postdoctoral ''Jose Castillejo'' Fellow. His current research interests include local positioning systems, acoustic signal processing, and embedded computing.

JORGE MORERA received the degree in physics from the University of Extremadura, Spain, in 1982. He is currently an Associate Professor of physics and electronics with the Department of Electrical Engineering, Electronics and Automation, University of Extremadura, where he is also a Sensory Systems Research Group Member. He has participated in five national research projects. His current research interests include local positioning systems and visible light technology.

FERNANDO J. ARANDA (Member, IEEE) received the B.Sc. degree in physics and the M.Sc. degree in the simulation of science and engineering problems, specializing in the simulation of physical phenomena, from the University of Extremadura, Badajoz, Spain, in 2018 and 2019, respectively, where he is currently pursuing the Ph.D. degree. Since 2019, he has been a part of the Sensory System Research Group, where he started his research activity. His current research interests include fingerprinting positioning, machine learning, and radio-frequency propagation.