Lighthouse Localization of Miniature Wireless Robots

Said Alvarado-Marin, Cristobal Huidobro-Marin, Martina Balbi, Trifun Savić, Thomas Watteyne, Filip Maksimovic

Lighthouse Localization of Miniature Wireless Robots

Said Alvarado-Marin[1], Cristobal Huidobro-Marin, Martina Balbi,
Trifun Savić, Thomas Watteyne, Filip Maksimovic

***Abstract*—In this paper, we apply lighthouse localization, originally designed for virtual reality motion tracking, to positioning and localization of indoor robots. We first present a lighthouse decoding and tracking algorithm on a low-power wireless microcontroller with hardware implemented in a cm-scale form factor. One-time scene solving is performed on a computer using a variety of standard computer vision techniques. Three different robotic localization scenarios are analyzed in this work. The first is a planar scene with a single lighthouse with a four-point pre-calibration. The second is a planar scene with two lighthouses that self calibrates with either multiple robots in the experiment or a single robot in motion. The third extends to a 3D scene with two lighthouses and a self-calibration algorithm. The absolute accuracy, measured against a camera-based tracking system, was found to be 7.25 mm RMS for the 2D case and 11.2 mm RMS for the 3D case, respectively. This demonstrates the viability of lighthouse tracking both for small-scale robotics and as an inexpensive and compact alternative to camera-based setups.**

***Index Terms*—Localization, Multi-Robot System, Wheeled Robots, Lighthouse Positioning.**

I. Introduction

CAMERA-based localization systems, such as that used in Preiss et al. [1], provide precise indoor positioning for robotic swarms. Motion capture solutions like Vicon offer sub-millimeter accuracy and high refresh rates, without necessitating extra hardware on the robots [2]. Despite their accuracy, such systems can be prohibitively costly, often exceeding tens of thousands of dollars, and their centralized server architecture may limit scalability in large multi-robot setups. An alternative approach is the Valve/HTC lighthouses, originally designed for virtual reality motion tracking. This system employs basestations that project lasers onto receiver diodes, coupled with a front-end chip on each robot. This setup has several benefits: it is significantly more affordable (a basestation costs $200, with receiver diodes and front-end chips priced at $2.30 and $1.20 respectively, in small quantities) and supports a fully decentralized architecture, enhancing scalability. It also naturally resolves the correspondence problem inherent to multi-camera tracking. There are two versions of the lighthouses: v1 [3] and the improved v2, which offers enhanced accuracy and scalability at the expense of a computationally intensive signal demodulation process on the robot.

[1] said-alexander.alvarado-marin@inria.fr
All authors are affiliated with INRIA-AIO, Paris, FR, 75012
Dataset and code are available at: https://github.com/DotBots/alvarado23lighthouse.

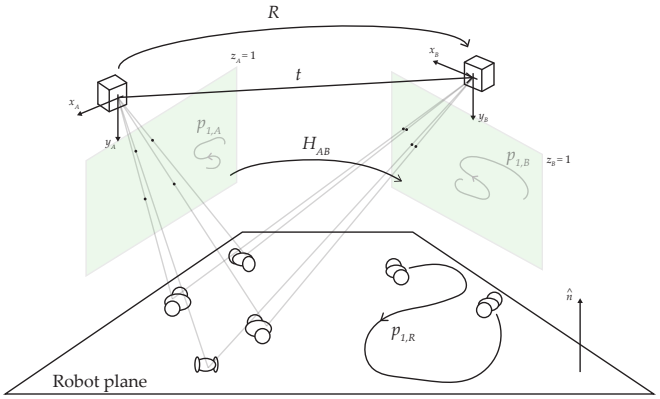

Fig. 1. Illustration of planar scene solving with either multiple robots or a single robot moving in a path consisting of non-colinear points.

The lighthouse system has been successfully implemented in robotic tracking applications, with Taffanel et al. [4] demonstrating centimeter-scale accuracy compared to commercial camera tracking systems.

Numerous studies have addressed localization using the lighthouse v1 system [3] [5] [6], but research on the lighthouse v2 system remains sparse. Most existing studies utilize the large official HTC receiver hardware [7] [8], which is unsuitable for small, cm-scale robots due to its size (70.9 × 79.0 × 44.1 mm). Prior studies on lighthouse localization in robotics, such as [4] and [9], have achieved centimeter-level accuracy in absolute positioning, and sub-centimeter accuracy with additional estimates. These systems, however, depend on multiple onboard receivers with known relative positions and employ the perspective-n-points (PnP) problem [10]. Specifically for lighthouse v2 compatible systems like [4], external FPGA-based hardware acceleration is necessary to manage the demodulation of multiple signals simultaneously. This requirement for multiple receivers and FPGA contributes to increased power usage, computational latency, cost, and size (3 cm x 1.5 cm in [4] and roughly 15 cm in [9], estimated from photographs), complicating miniaturization. Research on single-receiver systems does exist [5] [11], but their compatibility is limited to lighthouse v1. Demodulation on a cm-sized wireless microcontroller and localization with a single diode per robot would enable miniaturization and simple integration with existing robot platforms.

In this paper we propose a partially decentralized, lighthouse v2 compatible localization system for size-constrained multi-

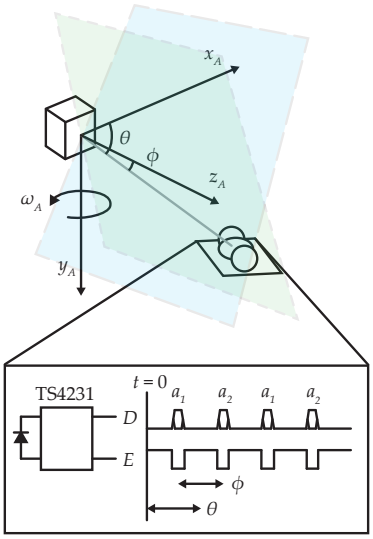

Fig. 2. Diagram demonstrating the general principle of Valve/HTC Lighthouse 2 operation.

robot system, that utilizes a single receiver diode per robot without external FPGA acceleration. Our work contributes two main innovations: First, a real-time demodulation algorithm for the lighthouse v2 signal, designed to be resistant to inter-symbol interference and low sample rates, functional on a single-core, low-power microcontroller. Second, three localization algorithms that work in 2D and 3D scenes using multiple observations from the lighthouse. The observations can be made with both a single moving robot or a group of disparate robots, as illustrated in Fig. 1. Notably, two of these algorithms do not require prior knowledge of basestation or robot configuration; they can self-calibrate from observations. This features makes the system portable and straightforward to deploy. The system implemented in this work is partially decentralized: After the initial scene-solving is performed, as long as the basestations remain stationary, each robot can independently estimate its global coordinates.

In our experiments, we decode the IR pulses from two lighthouses on a Cortex-M4 microcontroller, transmit them to a gateway using the IEEE 802.15.4 physical layer, then perform scene-solving on a computer in Python.

The eventual goal of this work is to apply lighthouse localization to swarms of robots similar to the Kilobot [12] or Zooid [13]. The Kilobot robots localize with a peer-to-peer distance estimate. The Zooid robots use a similar style to the lighthouse that relies on a grey-code projected pattern. IR falls within absorption range of diodes in standard CMOS processes which makes it possible to co-design a diode and front-end to receive lighthouse signals, as demonstrated in [11] on a 3 mm by 2 mm chip. This fits with the long-term goal of this work is to localize mm-scale robots, similar to the ones developed by Contreras et al. [14] or Wu et al. [15].

II. Lighthouse Operation

The lighthouse localization system operates by spinning planes of infrared (IR) light at a fixed and known angular

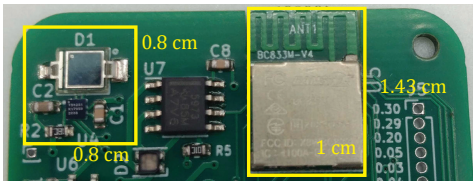

Fig. 3. Hardware integration of the receiver diode, front-end chip, and necessary passives, and the nRF in the compact Fanstel BC833 on a prototype PCB.

velocity ω. A given receiver can find the difference in time between the sweep start and the time at which the plane was received to calculate a spherical coordinate angle by multiplying by ω. The previous iteration of the Valve Lighthouse (LH1) used two separate sweeps, one horizontal and one vertical, to generate θ and ϕ spherical coordinates. The new iteration uses a single sweep consisting of two angled planes. The average of the times of arrival of the two planes is related to θ. The difference in times of arrival and the angle between the two planes is related to ϕ.

To indicate the time at which the sweep begins, the LH2 transmits a modulated signal that contains a coded count value that resets at the beginning of each sweep. This transmitted signal has two layers of modulation. The first contains a count value that, when received at a particular position, indicates the time at which the sweep signal was received. The modulation that contains this information can be interpreted as Differential Manchester (DM) encoded on-off key (OOK), and the coded bits are generated from a linear feedback shift register (LFSR) with known polynomial. Each lighthouse is configured to use a pair of polynomials that allow a receiver to distinguish which lighthouse transmitted a particular received bit sequence. The second layer of modulation, which contains factory calibration information, is encoded in which polynomial is used for a given sweep. The added complexity of the LH2 allows it to use a single rotating motor and localize without the use of a sync pulse, which extends the range as the IR can be more focused. In addition, the coded transmissions eliminate the need for multiple lighthouses to synchronize and deploy the system.

A. Hardware Implementation Details

The Triad TS4231 receiver chip acts as a diode front-end that rails the received IR signal to 3.3 V logic level. After configuration on startup, it has two outputs: an active low envelope line that indicates that valid data is incoming, and a data line that has the transmitted signal.

The hardware requires, at least, an optical front-end, and a wireless SoC to: (a) decode the modulated light and (b) wirelessly transmit the angle/location result to a central controller or processor. The optical receiver is built from commercially available parts (Osram BPW 32 S-Z diode and Triad TS4231 mixed-signal front-end). The wireless processor selected is the nRF52833 which is equipped with a Cortex-M4, 32-bit FPU, and a radio compatible with multiple wireless standards. The microcontroller SPI peripheral is used to sample the data line at a constant rate of 32 MS/s. Upon receiving four packets

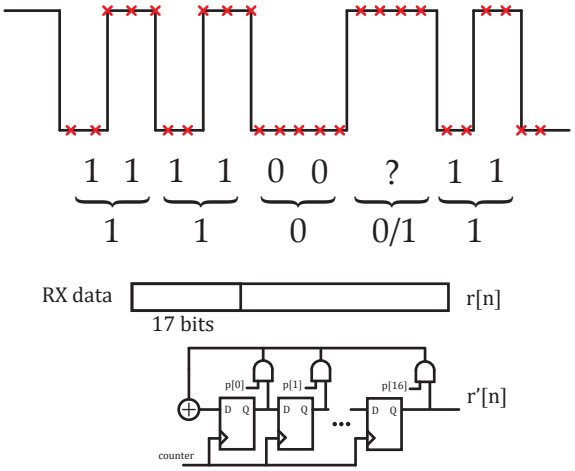

Fig. 4. Illustration of inter-symbol interference and the use of uncertain bits in the receiver and both Differential Manchester coding and known LFSR sequences to improve packet detection rate.

(two from each lighthouse) the packets are decoded to extract two sets of angles, as per (1).

$$\theta = \frac{\alpha_1 + \alpha_2}{2}$$
$$\phi = \tan^{-1}\left(\frac{\sin\left(\left(\alpha_2 - \alpha_1\right)/2 - \pi/3\right)}{\tan\left(\pi/6\right)\cos\left(\left(\alpha_2 + \alpha_1\right)/2\right)}\right) \tag{1}$$

A PCB with the front-end and processor is shown in Fig. 3 with dimensions of relevant components highlighted.

B. Decoding Algorithm

To convert the SPI samples to chips, we count the number of samples between level transitions on the data line, which effectively amounts to a zero-crossing detector. With a 32 MHz sample clock and short pulses of 83 ns, long pulses of 167 ns, there will be approximately three samples in a short pulse, and five in a long pulse. This technique is asynchronous with a sample rate that is not aligned with the data rate, so it is susceptible to bit misalignment and aliasing.

Furthermore, as shown in Fig. 4, certain bit transitions occasionally cause a dramatic increase in a "one" chip time or a decrease in a "zero" chip time, which appears as inter-symbol interference (ISI). The ISI is exacerbated at low signal strengths at long range, high angle of incidence, and at the beginning or end of a packet. It is observable even when the signal is over-sampled, indicating that it is caused by the front-end.

We mitigate ISI in two different ways. The first is by using the DM bit-wise encoding, as well as the LFSR polynomials if there are sufficient correct bits. The constraint from the DM encoding, that there must be an even number of "one" chips in a row, can be used to reduce the number of chip errors. Any remaining unknown bits after this procedure are treated later, during polynomial search, as unknowns in a soft-decision decoder.

$$r'_m[k+n+1] = \left(\sum_{n=0}^{16} p_m[n]r[k+n]\right)_2$$
$$\text{err}_m = d(r'_m, r) \tag{2}$$

The polynomials and LFSR sequences offer a further method to accommodate bit errors. First, it is necessary to find the correct sequence, which requires a minimum of 19 consecutive correct bits. If the polynomials were unknown (but had a known length of 17 bits), it would be necessary to decode 34 consecutive bits correctly (to apply the Berlekamp-Massey algorithm). However, the LFSR polynomials in this system are available. All that the receiver needs to do is determine the correct polynomial out of the candidate 32. In the implementation, we solve both problems by "sliding" each polynomial sequence through the chips and applying a Hamming error threshold proportional to the number of chips being checked. This simultaneously corrects chips and finds the polynomials, but requires a minimum of 24 consecutive bits with at most one error to uniquely select the LFSR from the 32 candidates.

Once the correct polynomial is found, we must locate the first known correct 17-bit sequence inside the full LFSR sequence. This can be accomplished by counting backwards or forwards through the LFSR from the received bits all the way to 0b1, which is the starting seed of the register. To speed up this process, we pre-computed 64 evenly spaced 17-bit sub-sequences, "checkpoints," at predetermined LFSR positions. These are stored in a hash table for rapid comparison during the search. The LFSR position of any successfully decoded packet is also stored as an additional checkpoint for the next search. This reduces computation time for slow-moving robots because there are small changes in the physical position and in the LSFR index. The checkpoint table allows the microprocessor to execute the search procedure with a worst-case 5 ms per received packet.

With a MATLAB implementation of the demodulation algorithm and oversampled TS4231 outputs from a logic analyzer, the proposed technique can still reliably decode packets with a minimum sample rate of 24 MHz at a measured range of 5 m. The main benefits of adding an FPGA to an existing robotic platform are speed and robustness of demodulation. The proposed technique is able decode in real time every lighthouse sweep from two simultaneous basestations, while only loading 19% of the microprocessor's CPU. A CPU-only solution also brings the following benefits: A high degree of integration, which makes the system substantially smaller and more applicable to cm-scale robots, lower sleep power consumption, and ease of use.

III. Localization

To perform receiver localization in 2- or 3-D coordinates relative to one of the lighthouses, it is necessary to estimate the translation and rotation of one lighthouse relative to the other. Then, the two rays, one from each lighthouse, can be used to determine Cartesian coordinates in the frame of one of the two lighthouses.

Alternatively, when only a single lighthouse is available, localization can be achieved by imposing a planar constraint on the receiver's motion. In order to calculate the rotation R and translation $\vec{t}$ from one lighthouse to the other, we borrow algorithms from computer vision and treat each lighthouse as a camera. Each receiver's angles intersect a plane 1 distance unit in front of the lighthouse to form a 2D image of the points. The coordinates of these points in (3) can be calculated directly from the angles in (1).

$$\begin{aligned} x &= -\tan(\theta) \\ y &= -\tan(\phi)/\cos(\theta) \\ z &= 1 \end{aligned} \tag{3}$$

This paper examines three distinct algorithms for scene-solving. These methods are categorized according to the number of lighthouses used, the dimensionality of the robots' movement (2D or 3D), and whether prior information about the scene is available.

A. 2D Scene - One Lighthouse

In planar scenarios, as long as there is a known set of correspondence points, the position of robots can be estimated using observations from one lighthouse. Such cases are relevant to this study because the robots mentioned in [12] and [13], as well as the wheeled robots employed in our experiments, are physically limited to motion on a floor or tabletop. Scene solving can be performed by using known correspondence between observations to establish a map linking a lighthouse image to a set of markers with predetermined positions in the global frame. This map is called a *Homography matrix*, and can be obtained using a direct linear transformation (DLT).

The minimum number of points to compute the function is four [16]. Additional points are used to reduce error with total least squares. The use of RANSAC and Levenberg–Marquardt (nonlinear least squares), as in OpenCV [17], would improve accuracy by removing outlying measurements or error from camera distortion. A comparison of bundling and least-squares techniques is beyond the scope of this work.

Once the homography matrix is estimated, the conversion of image points from the lighthouse frame to world coordinates involves computing a perspective transformation. This setup is illustrated in Figure. 5.

The transformed points will be accurate relative to the scale chosen for the markers in the global frame. In a robotic application, absolute scale could be estimated using an additional sensor, such as an IMU or a rotary encoder.

B. 2D Scene - Two Lighthouses

By introducing a second lighthouse, the scene can be solved without relying on markers with predetermined positions. This is achieved by comparing the image positions of corresponding points between two different lighthouse images. This process involves solving for the normal vector $\vec{n}$ of the plane in the frame of one lighthouse. As with the earlier algorithm, scene solving can be accomplished by determining the homography

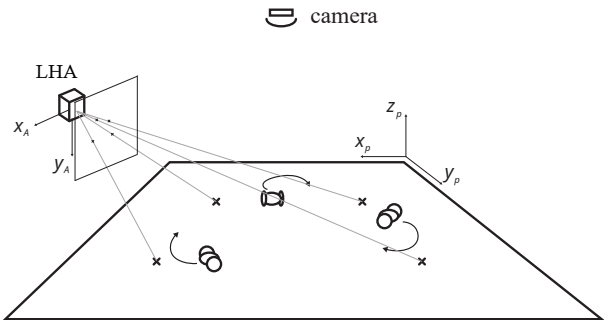

Fig. 5. Illustration of planar scene solving using four known planar markers via homography.

matrix that relates the observations from one lighthouse image to the other. This homography matrix can then be used to compute R and $\vec{t}$. It's worth noting that, in the absence of an external scale estimate, all coordinates will be correct to within a scaling factor.

2D algorithms require all points to be coplanar. Similar 3D algorithms (which use the fundamental matrix as opposed to the homography) always require at least one non-coplanar point [16]. Accurate hybrid approaches exist that rely on classification by outliers [18], but are not implemented in this work. Much like the "one lighthouse" scenario, the minimum number of points to compute the function is four, and additional points are used to reduce error with least squares.

After computing the homography matrix H, the scene can be solved by directly computing the SVD of H (Appendix 1 of [19]). The rationale behind the approach is that the cross product $\vec{t} \times \vec{n}$ is the second singular vector $\vec{v_2}$. And, in any planar approach, there will be two candidate solutions, only one of which is correct in the "real world." The SVD-based algorithm is summarized here.

First, compute the singular value decomposition of the 3×3 matrix H in the typical fashion: $H = U\Sigma V^T$ where U and V are orthonormal bases and Σ has the singular values of H along the diagonal sorted in descending order. Then, scale the singular values so that the second (middle) singular value is one. The factor ζ is used to scale a point from the first lighthouse's camera to the robot plane:

$$\zeta = \sigma_1/\sigma_3 \tag{4}$$

Then, the two candidates for the normal vector facing away from the lighthouse are:

$$\vec{n} = b\vec{v_1} \mp a\vec{v_3} \tag{5}$$

In (5), the two possible $\vec{n}$ from the sign of a are the two candidate vectors that describe the robot plane. The scalars a and b come from the singular values:

$$\begin{aligned} a &= \sqrt{\frac{1-\sigma_3^2}{\sigma_1^2-\sigma_3^2}} \\ b &= \sqrt{\frac{\sigma_1^2-1}{\sigma_1^2-\sigma_3^2}} \end{aligned} \tag{6}$$

To complete the scene solving, the two possible solutions for the R and $\vec{t}$ vectors must also be computed:

$$R = U \begin{bmatrix} c & 0 & \pm d \\ 0 & 1 & 0 \\ \mp d & 0 & c \end{bmatrix} V^T \tag{7}$$

Where c and d are given by:

$$\begin{aligned} c &= 1 + \sigma_1 \sigma_3 \\ d &= ab \end{aligned} \tag{8}$$

and are subsequently scaled so that $c^2 + d^2 = 1$.

Similarly, the two options for t can be computed:

$$\vec{t} = ev_1 \pm fv_3 \tag{9}$$

e and f, before scaling such that $e^2 + f^2 = 1$, are:

$$\begin{aligned} e &= -b/\sigma_1 \\ f &= -a/\sigma_3 \end{aligned} \tag{10}$$

Finally, once $\hat{n}$ is determined in the first lighthouse coordinate frame, each point in the first lighthouse frame can be calculated from its image point by multiplying by a scalar k_n such that:

$$k_n x_N \cdot \vec{n} = 1/\zeta \tag{11}$$

Determining which R, $\vec{t}$, and $\vec{n}$ are "correct" in the world view requires some assumptions as solutions from this algorithm will be internally consistent. One method is to use the sign of the $\vec{n}$ vector second element and the $\vec{t}$ vector third element with an assumption about how the camera is oriented with respect to the other camera and the robot plane. For instance, a typical setup will have an upright camera pointed at the plane, in which case the second element of $\vec{n}$ will be positive. If the two cameras point towards one another, the third element of the $\vec{t}$ will be positive.

C. 3D Scene - Two Lighthouses

Similar to the previously discussed method, known correspondence between the observations of two lighthouses can be leveraged to track an object in 3D space. The key distinction here lies in the type of matrix estimated from the correspondence points. Instead of the homography matrix, the fundamental matrix is used. This matrix relates corresponding non-coplanar points observed by both lighthouse cameras.

While the fundamental matrix can be estimated using DLT, similar to the homography matrix, it requires a minimum of seven corresponding points [20]. Employing more points can enhance the estimation when utilizing methods such as least squares or RANSAC. Due to the 3D nature of this algorithm, these points must be non-coplanar.

The fundamental matrix F can then be utilized to compute the essential matrix, E, which can then be decomposed into R and $\vec{t}$ that transform between the lighthouses' frames of reference. Then, the scene can be resolved by directly calculating the SVD of E. This computation yields four potential candidates for $[R|t]$. The correct $[R|t]$ from a world perspective can be determined by verifying the assumption that the reconstructed 3D points should lie in front of both lighthouses. For a more detailed discussion, refer to [21].

Once $[R|t]$ has been determined, each 3D point in the world frame can be derived from its corresponding image points as observed by both lighthouses using a linear triangulation method, such as the DLT. It is important to note that this algorithm is relative with respect to the scale of the reconstruction; all the estimated coordinates will be accurate only up to a scaling factor.

IV. Experimental Setup and Results

To evaluate the 2D localization algorithms, we fixed a receiver diode to a remote-controlled wheeled robot and tracked its trajectory on the floor. The robots capture and decode the lighthouse pulses, compute their angles (α_1 and α_2 from (1)), and transmit them to a computer. The computer performs the scene-solving, computes the robot locations, and records the data for further analysis. We simultaneously tracked the robots with a motion capture system from Qualisys—which has an accuracy of less than 1 mm—and used these measurements as ground-truth to determine the accuracy of the planar robot lighthouse localization. Figure 6 illustrates this setup.

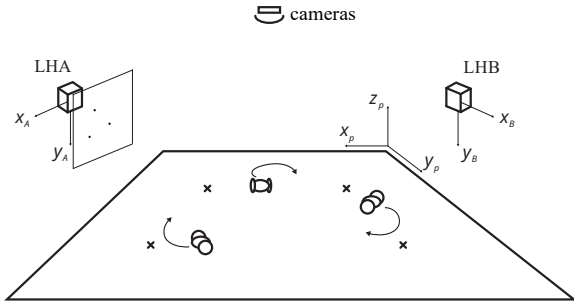

Fig. 6. (a) Three robots on a plane being tracked by two lighthouses, with overhead cameras in position. The known markers are denoted by Xs on the plane.

For the single lighthouse method, we delineated a 40×40 cm square on the floor, using its corners as the known markers required to calibrate the algorithm. In contrast, with the two lighthouse method, these markers were used exclusively to align the camera measurements with the lighthouse readings to determine accuracy; they were not used for scene solving.

For the 3D localization algorithm, we manually moved the receiver diode in the view of both a Qualisys motion capture system and two lighthouses. Accuracy of the lighthouse localization is compared against the motion capture ground-truth. In all experiments, the lighthouses were placed approximately 2 m away from the tracked objects.

Table. I provides a detailed evaluation of each algorithm's accuracy by showcasing the Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and Standard Deviation (SD) of the position of the robot as determined by the lighthouse.

A. Outliers and Noise Reduction

In Eq. 1 only four points are required for scene solving. However, this assumes that all four points are perfectly accurate. In reality, errors in the system are often caused by: manufacturing tolerances in the base station's motors, errors in

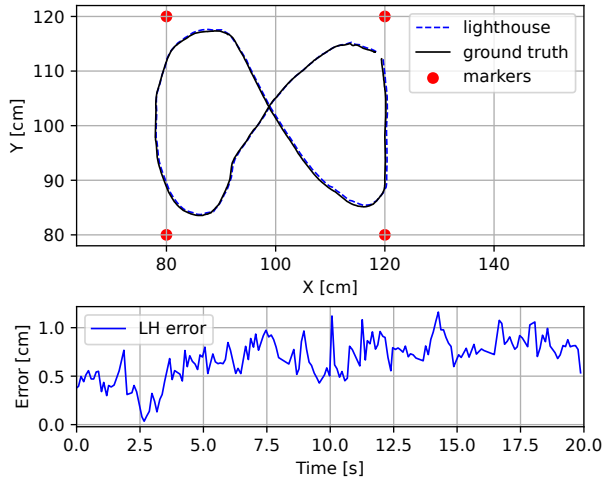

Fig. 7. X- and Y- coordinates of a robot's measured trajectory in the plane. The euclidean error of the trajectory is shown below.

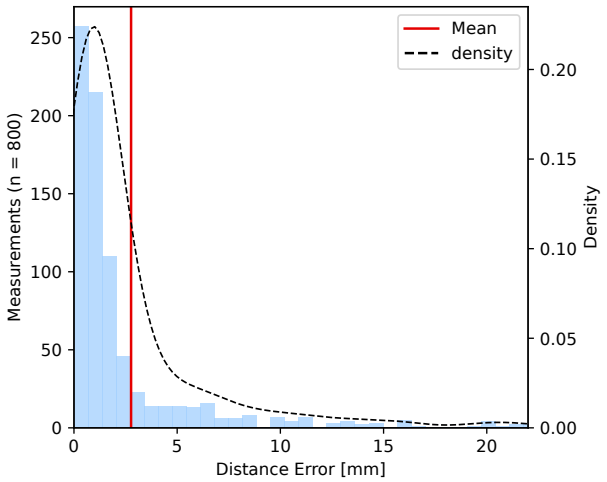

Fig. 8. Histogram of the euclidean MAE of the one-lighthouse localization.

demodulating the sweep signal, and numerical instabilities in scene solving. Manufacturing tolerances cause fluctuations in angle readings, which we measured to average around 300 microdegrees. These fluctuations are handled by the built-in total least-squares algorithm implemented in OpenCV functions. Demodulation errors cause the angle measurements to deviate by more than 2° for a single time-step, this significantly exceeds the system's average noise floor and can be filtered out with a difference threshold. Finally, a failed scene solution results in a distorted (unrealistically large) reconstruction, which can be detected and excluded with a distance threshold.

The Qualisys motion capture cameras emit infrared light pulses at the same wavelength as the lighthouse sweeps which interfere with the localization system. These camera pulses are modulated at 1 MHz, while the lighthouse data is modulated at 12 MHz. To address this interference, we automatically exclude any packet that contains 1 MHz because it indicates that the reading is dominated by the motion capture.

We found that these techniques are sufficient to remove the outliers observed in our experiments, and ensure accurate scene reconstructions.

B. 2D Scene - One Lighthouse

Figure. 7 displays a top-down view of the X-Y trajectory of a single robot. The positional error for this localization algorithm, as measured across several experiments, is illustrated in Figure. 8.

C. 2D Scene - Two Lighthouses

Figure. 9 presents a simultaneous reconstruction of the trajectories for three robots, illustrating a small-scale demonstration of multi-robot localization. Within this figure the recovered positions of the lighthouses are also included. Known markers in the floor were used to scale the plot to real-world dimensions. The histogram detailing the positional error of the localization algorithm is shown in Figure. 10.

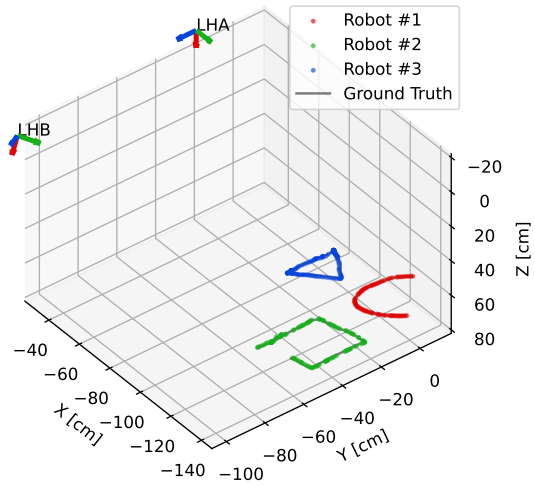

Fig. 9. Centralized scene solving and planar localization of three robots driving in regular geometric trajectories. This is a subset of the planar data used to determine accuracy. Note that the pose estimate was generated using the first 32 points of Robot 1's trajectory only and that solution was used for all subsequent points. The scene was not solved at each iteration.

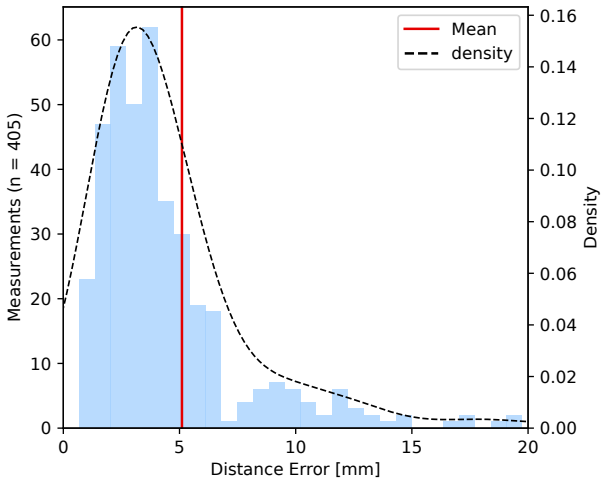

Fig. 10. Histogram of the Euclidean MAE of the two-lighthouse planar localization.

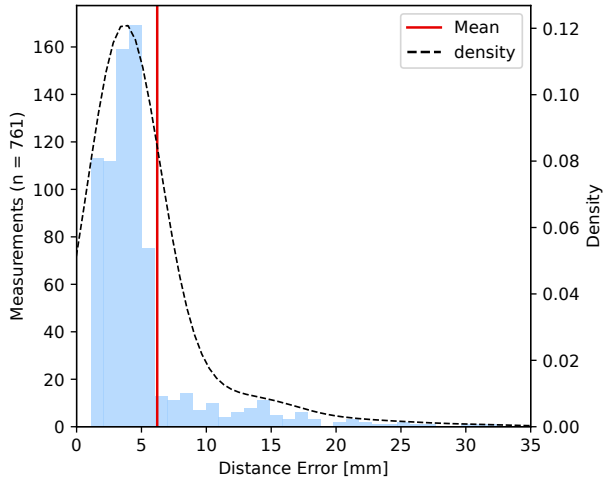

Fig. 11. Histogram of the Euclidean MAE of the two-lighthouse 3D localization.

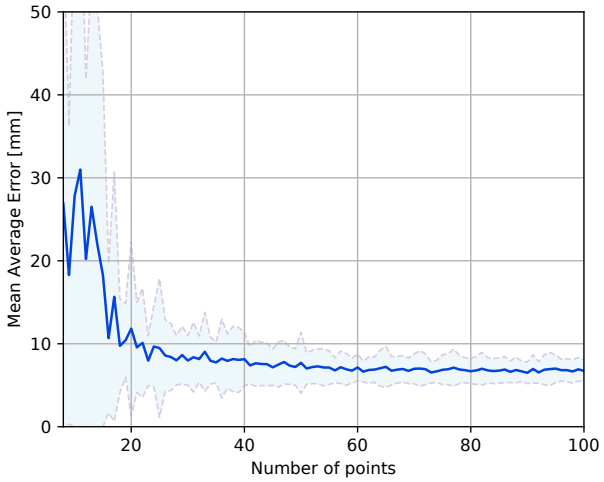

Fig. 12. Accuracy of the 3D reconstruction vs. the number of points used to solve the scene. Error bars, representing 1 standard deviation, are displayed in light blue around the MAE estimation.

TABLE I
LIGHTHOUSE POSITIONING ALGORITHMS' ERROR METRICS

Algorithm	MAE [mm]	RMSE [mm]	Std. Dev. [mm]
One LH - 2D	2.65	5.56	4.89
Two LH - 2D	5.1	7.25	5.15
Two LH - 3D	6.2	11.2	9.3

D. 3D Scene - Two Lighthouses

Unlike the previous 2D test, the 3D experiment's setup lacks pre-known markers to facilitate the alignment of the reconstructed data with the ground-truth. To address this, we utilized the algorithm detailed in [22]. The histogram in Figure. 11 displays the distribution of the positional error of the localization algorithm.

Additionally, we measured the system's precision when locating a stationary receiver. Table. II presents the standard deviation of these measurements, indicating that the lighthouse system can distinguish between two receiver diodes positioned as close as 1 mm apart.

It was observed that the accuracy of the 3D reconstruction is significantly influenced by the selection of data points used for scene solving. To further investigate this, we used 8 to 100 randomly selected non-coplanar measurements, each at least 4 cm apart, to solve a 3D scene. We then estimated the MAE of the reconstruction. The results are shown in Figure. 12.

From our findings, we deduce that a minimum of 20 non-coplanar measurements are necessary to ensure an accurate and stable scene reconstruction. Adding more measurements asymptotically enhances the accuracy of the reconstructed scene. Additionally, we also noted that the distance between the chosen measurements doesn't appear to significantly impact the overall localization accuracy.

TABLE II
3D SCENE - TWO LIGHTHOUSE ALGORITHM'S STATIC PRECISION

Axis	X	Y	Z
Std. Dev. [mm]	0.125	0.2435	0.137

E. Computation Time and Communication Latency

The localization procedure can be broken into four parts: signal acquisition and demodulation, calculation of polynomial from Eq. 2, calculation of location within the LFSR, and computation of position using equations 1, 3, and 11. Each step was implemented on an nRF52833 microcontoller and its computation time was measured with a logic analyzer. These computation times for the 2D scene algorithms are summarized in Table III. We were unable to implement the 3D localization algorithm on the microcontroller.

The total computation time (from pulse to LSFR index) requires approximately 1 ms of CPU time per lighthouse pulse. These results suggest that, once the scene has been solved, the pulse decoding algorithm and 2D localization can feasibly be computed on a low-power microcontroller.

TABLE III
COMPUTATION TIME ON THE ROBOT'S MICROCONTROLLER

Operation	Measured Computation Time (Mean $\pm$ Std. Dev.)
Demodulation	623 μs $\pm$ 3.6 μs
Determine Poly.	127 μs $\pm$ 27 μs
LFSR index	158 μs $\pm$ 155 μs
Position One LH - 2D	141 μs
Position Two LH - 2D	144 μs

F. Size and Power Consumption

Two benefits of CPU-only solutions over FPGAs are reduced size and lower power consumption. To quantify this, we used an OTII arc power analysis tool to measure the current consumption of an nRF52833 microcontroller and a TS4231 front-end receiver while they received IR signals from two lighthouse basestations. The circuit's physical dimensions

were estimated based on Figure 3. We then compared these metrics with those of the commercial FPGA-based solution utilized in [4], according to its datasheet [23]. It is important to note that the FPGA-based solution needs four TS4231 receivers to function. In contrast, our solution only requires one. The comparative analysis is summarized in Table IV.

TABLE IV
POWER CONSUMPTION AND CIRCUIT SIZE

Solution	Mean Current	Mean Power	Size
CPU-only	7.01 mA	23.1 mW	2.07 cm^2
FPGA-based [4]	30 mA	111 mW	10.53 cm^2

V. CONCLUSIONS

A limitation of the current scene-solving and localization methods is that they require some prior knowledge of whether the robots in question are planar or not. The 3D scene-solving technique has a constraint that it only works on non-coplanar points, whereas the 2D techniques have the opposite constraint.

Potential methods to improve accuracy in the future are outlier exclusion and bundling, sensor fusion, and use of the internal lighthouse IMUs to estimate pose as in [24]. Finally, further improvements could be achieved by developing a calibration method for the lighthouse basestations, which are currently modeled as ideal pinhole cameras. A more realistic model could help mitigate distortions introduced by the lens of the lighthouse and other manufacturing imperfections.

ACKNOWLEDGMENT

The authors would like to thank Dr. Raoul de Charette for his valuable discussions, as well as the contributors to the `libsurv` open-source VR project.

This document is issued within the frame and for the purpose of the OpenSwarm project. This project has received funding from the European Union's Horizon Europe Framework Programme under Grant Agreement No. 101093046.

REFERENCES

[1] J. A. Preiss, W. Hönig, G. S. Sukhatme, and N. Ayanian, "Crazyswarm: A Large Nano-quadcopter Swarm," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2017.

[2] P. Eichelberger *et al.*, "Analysis of accuracy in optical motion capture – a protocol for laboratory setup evaluation," *Journal of Biomechanics*, vol. 49, no. 10, pp. 2085–2088, 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0021929016305681

[3] J. Lwowski, A. Majumdar, P. Benavidez, J. J. Prevost, and M. Jamshidi, "HTC Vive Tracker: Accuracy for Indoor Localization," *IEEE Systems, Man, and Cybernetics Magazine*, vol. 6, no. 4, pp. 15–22, 2020.

[4] A. Taffanel *et al.*, "Lighthouse Positioning System: Dataset, Accuracy, and Precision for UAV Research," in *ICRA Workshop: Robot Swarms in the Real World*, Xian, China, June 2021.

[5] Maheepala, Malith and Joordens, Matthew A. and Kouzani, Abbas Z., "A Low-Power Connected 3-D Indoor Positioning Device," *IEEE Internet of Things Journal*, vol. 9, no. 11, pp. 9002–9011, 2022.

[6] Y. Yang, D. Weng, D. Li, and H. Xun, "An improved method of pose estimation for lighthouse base station extension," *Sensors*, vol. 17, no. 10, 2017. [Online]. Available: https://www.mdpi.com/1424-8220/17/10/2411

[7] Kuhlmann de Canaviri, Lara *et al.*, "Static and Dynamic Accuracy and Occlusion Robustness of SteamVR Tracking 2.0 in Multi-Base Station Setups," *Sensors*, vol. 23, no. 2, 2023. [Online]. Available: https://www.mdpi.com/1424-8220/23/2/725

[8] V. Holzwarth, J. Gisler, C. Hirt, and A. Kunz, "Comparing the accuracy and precision of steamvr tracking 2.0 and oculus quest 2 in a room scale setup," in *Proceedings of the 2021 5th International Conference on Virtual and Augmented Reality Simulations*, ser. ICVARS '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 42–46. [Online]. Available: https://doi.org/10.1145/3463914.3463921

[9] D. R. Quiñones, G. Lopes, D. Kim, C. Honnet, D. Moratal, and A. Kampff, "HIVE Tracker: A Tiny, Low-Cost, and Scalable Device for Sub-Millimetric 3D Positioning," in *Augmented Human International Conference (AH)*, Seoul, Republic of Korea, 2018.

[10] V. Lepetit, F. Moreno-Noguer, and P. Fua, "EPnP: An Accurate O(n) Solution to the PnP Problem," *International Journal of Computer Vision*, vol. 81, February 2009.

[11] B. G. Kilberg, F. M. R. Campos, F. Maksimovic, T. Watteyne, and K. S. J. Pister, "Accurate 3d lighthouse localization of a low-power crystal-free single-chip mote," *Journal of Microelectromechanical Systems*, vol. 29, no. 5, pp. 818–824, 2020.

[12] M. Rubenstein, C. Ahler, and R. Nagpal, "Kilobot: A Low Cost Scalable Robot System for Collective Behaviors," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2012.

[13] M. Le Goc, L. H. Kim, A. Parsaei, J.-D. Fekete, P. Dragicevic, and S. Follmer, "Zooids: Building Blocks for Swarm User Interfaces," in *Annual Symposium on User Interface Software and Technology (UIST)*, Tokyo, Japan, 2016.

[14] D. S. Contreras, D. S. Drew, and K. S. J. Pister, "First Steps of a Millimeter-Scale Walking Silicon Robot," in *International Conference on Solid-State Sensors, Actuators and Microsystems (TRANSDUCERS)*, 2017.

[15] Y. Wu *et al.*, "Insect-Scale Fast Moving and Ultrarobust Soft Robot," *Science Robotics*, vol. 4, no. 32, 2019.

[16] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge university press, 2003, pp. 88–293.

[17] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[18] D. Gallup, J.-M. Frahm, and M. Pollefeys, "Piecewise Planar and Non-Planar Stereo for Urban Scene Reconstruction," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010, pp. 1418–1425.

[19] B. Triggs, "Autocalibration from Planar Scenes," in *European Conference on Computer Vision (ECCV)*, 1998, pp. 89–105.

[20] R. I. Hartley, "Projective Reconstruction and Invariants from Multiple Images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, no. 10, pp. 1036–1041, 1994.

[21] D. Nistér, "An Efficient Solution to the Five-point Relative Pose Problem," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 6, pp. 756–770, 2004.

[22] K. S. Arun, T. S. Huang, and S. D. Blostein, "Least-Squares Fitting of Two 3-D Point Sets," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-9, no. 5, pp. 698–700, 1987.

[23] *Lighthouse Positioning Deck*, Bitcraze AB, 2022, rev. C. [Online]. Available: https://www.bitcraze.io/documentation/hardware/lighthouse_deck/lighthouse_deck-datasheet.pdf

[24] Y. Ding, J. Yang, J. Ponce, and H. Kong, "An Efficient Solution to the Homography-Based Relative Pose Problem With a Common Reference Direction," in *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 1655–1664.