Journal of Engineering Science and Technology
Vol. 16, No. 4 (2021) 3165 – 3184

LANDMARKS EXPLORATION ALGORITHM FOR MOBILE ROBOT INDOOR LOCALIZATION USING VISION SENSOR

MOHANAD N. NOAMAN*, MOHAMMED QASIM, OMAR Y. ISMAEL

Electronics engineering college, Ninevah University,
Al-Majmoaa Street, 41002, Mosul, Iraq
*Corresponding Author: mohanad.noaman@uoninevah.edu.iq

Abstract

In this paper, a novel Landmarks Exploration Algorithm (LEA) is presented for accurate, reliable, and efficient indoor localization. It operates in two stages. The first stage intends to search for artificial Color-Coded Landmarks (CCLs) and store their locations. Extended Kalman Filter (EKF) is exploited in this stage for continually updating the states of the robot, while the camera, equipped with image processing MATLAB code, is used for the detection of landmarks. The second stage attempts to make the robot find a location such that the distance to the detected CCLs is directly measured using a proximity sensor. At this stage, a trilateration method is applied to localize the robot. This paper also proposes an approach to estimate the heading angle of the robot. These two stages contribute to making the robot reach the target as a final step. Furthermore, the LEA performs localization even that one or two CCLs are detected at the same time and also specific order of CCLs is not required. The LEA is implemented, examined, and evaluated inside the CoppeliaSim environment. The simulation results indicate that the LEA provides the robot with the ability to explore CCLs, achieve an accurate localization, and reach the target.

Keywords: CoppeliaSim, Colour-coded landmarks/beacons, Image processing Localization, Trilateration.

1. Introduction

Robot technologies have rapidly developed in the last few decades. Hence, numerous concentrations have been paid to the autonomous navigation of mobile robot systems. However, there are yet many problems to be solved while making mobile robots more intelligent. One of these fundamental problems that a mobile robot frequently encounters is localization. The term "localization" refers to the ability of a mobile robot to localize itself within its environment [1]. In other terms, it refers to the position and orientation of a mobile robot within a certain coordinate. There are two notable kinds of localization: relative localization, which means that the robot has to find its position in a local reference frame, and global localization, which means that the robot tries to localize itself in a global reference frame [2].

Indoor localization has significantly gained importance in versatile applications such as transportation, logistics, surveillance, and warehouse which contains ambiguous areas and frequently changes [3]. Existing features can be insufficient for mobile robots to establish robust navigation behaviour in such an environment. Therefore, many practical applications rely on either passive or active artificial landmarks where they are placed throughout the robot trajectories to allow for precise localization [4, 5]. Although artificial landmarks are crucial for localization, they impose significant limits on how many landmarks can be placed. Therefore, the number and placement of landmarks can be optimized to be utilized by a localization algorithm of mobile robots.

The development of the LEA is motivated by the fact that most of the works using the trilateration method require at least three landmarks to be detected at the same time and the distance to the three landmarks to localize the robot under some constraints. The LEA presents the solution to that restriction and can localize the robot even though one or two landmarks can be seen at the same time by utilizing the combination of the Trilateration and Kalman filter techniques.

Over the past years, a broad range of approaches has been developed, examined, and applied to the problem of how to localize a mobile robot accurately. Many possible solutions have arisen with varying degrees of success. Han et al. [6] proposed a landmark-based particle localization algorithm (LPLA) using a vision sensor and it relies on at least two landmarks to localize the robot. In the same context, Betke and Gurvits [7] have used a number of landmarks to determine the pose of a robot using the position estimator.

Zhou [8] has derived an algorithm from a nonlinear least-square trilateration formula to provide the best approximation solution when there is no intersection at the actual position among circles. This algorithm utilizes a number of reference points to optimally estimate a position. Sabattini et al. [9] have used a monocular camera for positioning a low-cost mobile robot. The distances between the robot and three landmarks are measured by converting an image of the environment into a bird's eye view image. Then, the trilateration technique has applied using these distances to compute the robot's pose. However, the light conditions influence the accuracy and there was a 7cm mean error in the measurements of the position.

Moleski and Wilhelm [10] suggested a hybrid camera-LiDAR system for position estimation using the trilateration localization process. Four coloured landmarks have used to be identified by an RGB camera. This method was able to localize a vehicle with an 8cm error in 2D and a 261cm error in 3D. Unlike these

works which require three instantaneous landmarks to get an accurate estimation of the position, the proposed landmarks exploration algorithm (LEA) in this paper can localize a robot even if two or fewer landmarks are available. This can be done by exploiting the storing the locations of landmarks and odometry information.

On the other hand, many researchers have paid a concentration to optimize the localization methods depending on landmarks deployment. Rupp and Levi [11] considered geometrical observations to find the landmark locations in an indoor environment. Landmark positions on the walls have been selected close to a given set of localization points. They utilized the definition and optimization of a confidence level describing the expected localization estimation error to optimally place landmarks.

Marsland et al. [12] presented an automatic landmark selection algorithm that makes a mobile robot autonomously select notable landmarks without a pre-defined model of those landmarks. The approach utilizes a neural network to select landmarks by defining unexpected notable landmarks. Jourdan and Roy [13] minimized the average position error constrained in the sensor network by deploying a sensor network on the walls within the environment.

Unlike these methods, Beinhofer et al. [14] modelled the noise of the sensors and actuators of the robot to determine the minimal number of landmarks that guarantee a limit on the maximum deviation of the robot from its desired trajectory within its environment with high confidence. Magnago et al. [15] considered a greedy placement algorithm. Their approach is based on finding the optimal placement and a minimum number of deployed landmarks by collecting data, from environment observations to build a large number of trajectories to localize the robot accurately. Furthermore, Magnago et al. [16] extended the nearly-optimal greedy landmark placement approach by relying on environment contextual information for landmarks deployment. However, the LEA does not require landmarks ordering or deploying in some special ways to estimate the pose. It explores landmarks and stores their position. While some of the aforementioned approaches use active landmarks, the suggested LEA uses artificial CCLs as a passive artificial landmark. Unlike active landmarks, passive CCLs are simple to be made. Additionally, they do not require any power-supply sources to stay operable for an indefinite time.

In this paper, a novel landmarks exploration algorithm is proposed for mobile robot localization, which exploits EKF based dead reckoning, vision sensor, and the laser distance sensor to localize the robot. Many works in [17-19] have employed vision sensors for distance measurement. However, vision sensors are more sensitive to light, and colour interference could be a problem that affects the accuracy of distance measurement. Therefore, the laser distance sensor is used instead to determine a distance while the camera is only used for detecting the presence of landmarks.

The LEA works as follows. Firstly, it starts an EKF based dead reckoning for continually updating the mobile robot pose. This stage aims to seek CCLs and storing their locations. A camera equipped with image processing MATLAB code is used to detect these CCs landmarks. Secondly, the previous step is employed to determine the location at which the robot can measure the distance to at least three visible CCs landmarks directly by a laser sensor. On that occasion, the algorithm employs a trilateration localization technique to localize the mobile robot accurately inside the CoppeliaSim environment (formerly known as VREP, Virtual

Robot Experimentation Platform). This is crucial for the next step in which the robot pursues and reaches the target pose. In this respect, it is worth mentioning that the majority of localization based on trilateration methods needs to see at least three landmarks at the same time to correctly achieve localization as in [20-23]. However, the proposed LEA algorithm has the benefit of localizing the robot even without requiring seeing three landmarks at the same time.

To work under more realistic circumstances, it is crucial to spot some environmental assumptions. Firstly, the mobile robot moves around on the flat ground. Secondly, the environment should be illuminated enough for the camera to work properly.

The LEA has several characteristics that make it especially useful for mobile robot localization:

- It is considered as a new efficient method that can precisely localize the robot.
- It can also overcome the restrictions of the trilateration technique and it does not require landmarks ordering to achieve localization.
- It includes improvement in the method of determining orientation in which became more simple, efficient, and low computational than other methods.
- It can apply the trilateration technique and localize the robot without the need for three CCLs to be detected at the same time.

The rest of this paper is concerted as follows: section 2 presents the Extended Kalman Filter and sensors fusion. Section 3 discusses the trilateration approach. The CCLs detection algorithm is introduced in section 4. LEA is explained in detail in section 5. Section 6 shows and discusses the results. Section 7 concludes the paper and features the next step to be followed.

2. Modelling and pose estimation of a mobile robot

2.1. Mobile robot kinematic model

Consider the following unicycle model of the mobile robot moving in a horizontal plane which is shown in Fig. 1.

$$\begin{aligned} \dot{X} &= \|\mathbf{v}\| \cos\theta \\ \dot{Y} &= \|\mathbf{v}\| \sin\theta \\ \dot{\theta} &= \omega \end{aligned} \tag{1}$$

where $\|\mathbf{v}\|$ and ω denote the desired speed and angular velocity of the robot and θ is the angle between the x-axis of the global frame ($\mathbf{X}$) and the x-axis of the robot frame ($\mathbf{X}_{\mathrm{R}}$).

If the robot is only permitted to move along its $\mathbf{X}_{\mathrm{R}}$ axis, then $\dot{\mathbf{X}}_{\mathrm{R}} = \mathbf{v}$ and $\dot{\mathbf{Y}}_{\mathrm{R}} = 0$ (v_y in Fig. 1 is not necessarily zero, i.e., in case of slip). Moreover, the desired wheels' angular velocities ($\dot{\phi}_1, \dot{\phi}_2, \dot{\phi}_3$) can be obtained using the following equation [24].

$$\begin{bmatrix} \dot{\phi}_1 \\ \dot{\phi}_2 \\ \dot{\phi}_3 \end{bmatrix} = \frac{1}{r} \begin{bmatrix} \sqrt{3}/2 & -1/2 & l \\ 0 & 1 & l \\ -\sqrt{3}/2 & -1/2 & l \end{bmatrix} \begin{bmatrix} \dot{\mathbf{X}}_{\mathrm{R}} \\ \dot{\mathbf{Y}}_{\mathrm{R}} \\ \omega \end{bmatrix} \tag{2}$$

where r represents the radius of the wheel and l represents the distance from the centre of the robot to the wheel. A low-level PI controller is then used to make the wheels' motors track the desired angular velocities.

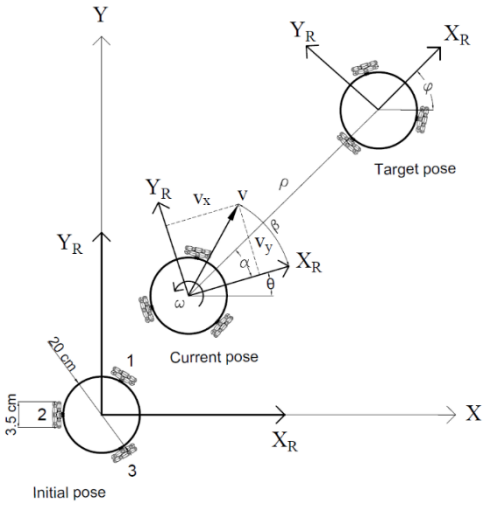

Fig. 1. Global frame and robot frame.

2.2. Extended Kalman filter

The EKF is utilized in this work to continuously estimate the pose of the robot when it moves within the environment. The direct use of sensor data could lead to an accumulative error over time. To improve the quality of updating robot pose, sensor fusion technique can be applied by using dedicated sensors such as an encoder and

Inertial Measurement Unit (IMU) contains a 3-axis gyroscope, accelerometer, and magnetometer. This can be done using the Kalman filter to correctly identifying the pose of a robot [25]. The Kalman filter in its basic form consists of two steps: prediction and update.

Suppose that measurements are received from the encoder and IMU at each sampling period T (T =50ms is used in this paper). The measured IMU signals contain noises, especially during the robot motion, and biases and are assumed in the following form

$$\begin{aligned} a_{xm} &= a_x + b_{ax} + \imath w_{ax} \\ a_{ym} &= a_y + b_{ay} + \imath w_{ay} \\ \omega_m &= \omega + b_\omega + \imath w_\omega \end{aligned} \tag{3}$$

where a_{xm} and a_{ym} denote the measured acceleration along the *x*-axis and *y*-axis of IMU respectively; ω_m is the measured angular velocity around the z-axis of the IMU; and b_{ax}, b_{ay} and b_ω are constant biases, i.e., $\dot{b}_{ax} = \dot{b}_{ay} = \dot{b}_\omega = 0$. The noises $\imath w_{ax}$, $\imath w_{ay}$ and $\imath w_\omega$ have zero mean and normal distribution.

The discrete model that describes the relationship between the robot position and acceleration and the robot orientation and angular velocity can be formulated based on Eqs. (1) and (3) as follows:

$$\mathbf{x}[k+1] = \mathbf{f}(\mathbf{x}[k], \mathbf{u}[k]) \tag{4}$$

$$\mathbf{z}[k] = H\mathbf{x}[k] \tag{5}$$

where $\mathbf{x}[k] = (x[k], y[k], \mathrm{v_x}[k], \mathrm{v_y}[k], b_{ax}[k], b_{ay}[k], \theta[k], b_\omega[k])$,

$$\mathbf{f}[k] = \begin{pmatrix} x[k] + T\|\mathbf{v}[k]\|\cos(\theta[k] + \beta[k]) \\ y[k] + T\|\mathbf{v}[k]\|\sin(\theta[k] + \beta[k]) \\ \mathrm{v_x}[k] + T(a_{xm}[k] - b_{ax}[k]) \\ \mathrm{v_y}[k] + T(a_{ym}[k] - b_{yx}[k]) \\ b_{ax}[k] \\ b_{ay}[k] \\ \theta[k] + T(\omega_m[k] - b_{\omega}[k]) \\ b_{\omega}[k] \end{pmatrix},$$

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix},$$

$\boldsymbol{z}[k] = (x[k], \quad y[k], \quad \theta[k])^T, \|\mathbf{v}[k]\| = \sqrt{\mathrm{v}_x[k]^2 + \mathrm{v}_y[k]^2}$, $\mathrm{v}_x[k]$ and $\mathrm{v}_y[k]$ are the robot velocities along $\mathbf{X}_\mathrm{R}$ and $\mathbf{Y}_\mathrm{R}$ axes respectively, $\beta[k] = \mathrm{atan2}(\mathrm{v_y}[k], \mathrm{v_y}[k])$, and $(x[k], y[k], \theta[k])$ are the robot pose relative to the global frame.

The EKF or unscented Kalman filter method can be used to estimate the states of the aforementioned model. Note that the measurement vector $\mathbf{z}[k]$ is obtained from encoder and magnetometer measurements for robot position and orientation respectively after proper sensors calibrations.

3. Trilateration Approach

3.1. The fundamental of trilateration

Throughout the years, in terms of location purposes, the trilateration approach has been applied in many different systems. It has also been employed in robotics technology, especially in a mobile robot. The trilateration approach involves the determination of the robot position based on distance measurements. It can be used for indoor or outdoor localization. Outdoor localization often uses the Trilateration approach with GPS to estimate the position [10]. While in indoor localization, the trilateration approach is applied with WiFi signal based [26], Bluetooth based [27], RFID technology [28], and Hybrid Camera-LiDAR System [10]. In this paper, the Trilateration approach is applied using a camera and a laser distance sensor mounted properly on the robot. The purpose of a camera is to detect CCLs. In this process, the distance between the mobile robot and the landmark can be measured using a laser distance sensor as shown in Fig. 2. To obtain the position of the mobile robot, the following three equations for the three circles are used:

$$\begin{aligned} (x - x_1)^2 + (y - y_1)^2 &= r_1^2 \\ (x - x_2)^2 + (y - y_2)^2 &= r_2^2 \\ (x - x_3)^2 + (y - y_3)^2 &= r_3^2 \end{aligned} \tag{6}$$

where $x_1, y_1, x_2, y_2, x_3, y_3$ denote the coordinates of the landmarks in a global reference frame, r_1, r_2, r_3 are the radii of circles representing the distances between the camera and landmarks and x, y represent the robot position.

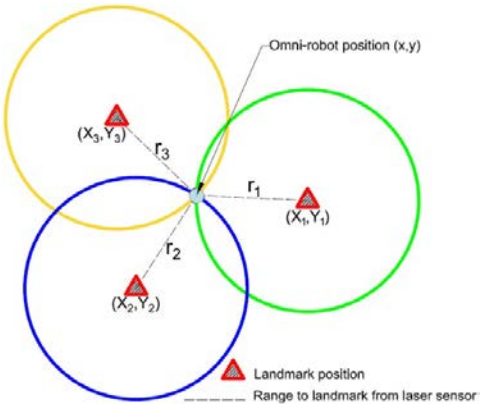

Fig. 2. Trilateration solution example at the intersection of three circles.

To solve for the position (x, y) of the robot, expand the squares of each of these three equations, then subtract the second equation from the first one and subtract the third equation from the second one to obtain

$$x = \frac{CE-FB}{EA-BD} \tag{7}$$

$$y = \frac{CD-AF}{BD-AE} \tag{8}$$

where $A = (-2x_1 + 2x_2)$, $B = (-2y_1 + 2y_2)$, $C = r_1^2 - r_2^2 - x_1^2 + x_2^2 - y_1^2 + y_2^2$, $D = (-2x_2 + 2x_3)$, $E = (-2y_2 + 2y_3)$ and $F = r_2^2 - r_3^2 - x_2^2 + x_3^2 - y_2^2 + y_3^2$.

The coordinates of landmarks relative to a global reference are deduced from the image processing technique. Additionally, the distance to the landmarks (r_1, r_2, r_3) are measured using a proximity sensor. Hereby, the x and y positions of the robot are computed depending on the abovementioned trilateration equations.

3.2. The restrictions of the trilateration approach:

As mentioned in the previous section, at least three distinguishable landmarks must be seen by the robot to localize itself in a certain environment. However, the trajectories or areas of an environment with less than three visible landmarks are inadequate to precisely localize the robot. Even though those three landmarks are usually suitable for the implementation of robot localization, this is not applicable in some cases. In other terms, a robot cannot localize itself in a case when a robot and landmarks lie in the same circumference (in the case of the triangulation technique). This is because the intersection of the two arcs is another arc, not a point [30, 31]. Even in an obstacle-free area, a problem emerges when a landmark becomes between a robot and another landmark. In this paper, these common restrictions will be overcome by applying the proposed elegant algorithm, which will be explained in detail in Section 5.

3.3. Calculation of the robot orientation:

For complete robot localization, the orientation should also be calculated. Although there are many ways to determine the orientation of a robot, the triangulation method with the help of landmarks is a proven technique. The majority of the many proposed triangulation approaches have several limitations such as requiring a

particular landmark ordering, having blind spots, or needing the triangle defined by the three landmarks to work.

Improvements to those approaches arose, but in contrast, they increased complexity, or they were required to cope with certain spatial arrangements separately [30]. In this context, minimal modification is performed, and the following new method is proposed in this work to compute the orientation. This becomes possible by the benefit of combining the trilateration and the triangulation method, which makes it easy-to-implement, low computational cost, and it can work in any arrangement without depending on a certain landmark ordering. After applying the trilateration method. The following formula is used to determine the heading of the robot:

$$\theta_R = atan2(y_i - y, x_i - x) - \emptyset_i \tag{9}$$

where (x_i, y_i) are the coordinates of a landmark i, (x, y) are the coordinates of the robot obtained from equations (7) and (8) and $\emptyset_i$ is the angle between the robot and the landmark as shown in Fig. 3 (a). For further simplification, the LEA ensures to make the robot face the detected landmark, which means no difference angle between them, and thus $\emptyset_i$ is zero as shown in Fig. 3 (b). Consequently, the final version of the equation is quite simple:

$$\theta_R = atan2(y_i - y, x_i - x) \tag{10}$$

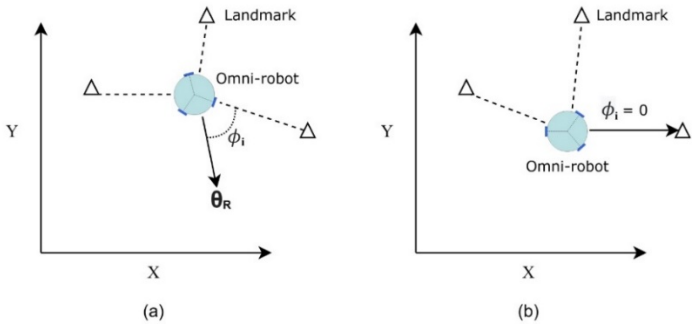

Fig. 3. The Omni-robot locations among landmarks.

4. Color-Coded Landmark Detection

This section focuses on the detection of the artificial CCLs which are based on the combination of three-square areas with two or three different colors assigned in-line near each other. These colors are red, green, and blue. The selected number of square areas is perfect in the context of detecting the CCLs with high reliability and fast response time. Moreover, it generates twenty-four CCLs.

Each video frame is passed from the robot camera in CoppeliaSim to MATLAB which is employed to detect the CCLs. RGB colour space is used in this work because the segmentation of the needed colors is more profound than other colour spaces. Furthermore, it can provide a high-performance color detection without any conversion stage despite some undesired properties under imbalance illumination.

RGB image can be separated into Red (R), Green (G), and Blue (B) components images with an intensity range of [0, 255] for each. The proposed framework of CCLs

detection is divided into three stages. In the first stage, each RGB channel is converted to a binary image based on a normalized intensity threshold in the range of [0, 1]. These binary channels are called BR, BG, and BB for the R, G, B components respectively. However, this thresholding method is not effective when the RGB image contains objects with colors that are produced from combining the high-intensity values of RGB components. To solve this issue, a grayscale intensity image in the range of [0, 255] is subtracted from each of the RGB channels before converting these channels to binary. This grayscale intensity image is earned from the weighted method with a contribution of 29.89% red, 58.70% green, and 11.4% blue.

The second stage is the image segmentation process. In this stage, a collection of pixel regions is produced for each binary image which leads to process only the important segments of the image instead of processing the entire image. The segmentation process is based on the region-growing method which has clear edges with good segmentation results. In this segmentation method, the neighboring pixels of the initial seed points are inspected according to an 8-connected neighborhood and calculated iteratively if the neighbors of the pixel need to be added to the region. Each region contains a collection of binary data called a Binary Large OBject (BLOB). BLOBs with a different number of pixels are held in each BR, BG, and BB channel depending on what the robot camera is seeing in the environment. Moreover, the small BLOBs which have fewer than P pixels are removed.

In the third stage, a matrix of [*x y*] centroid coordinates for the BLOBs in the segmented BR, BG, and BB channels are computed. Then, the CCLs are detected based on the relationship among the centroids of three BLOBs. In other words, the CCL detection is dependent on the vertical adjacent among these BLOBs with acceptable tolerance (T pixels) and the vertical distances among the three BLOBs centroids. At the end of the third stage, the numeric values of the CCLs with their counts and centroids are returned to the CoppeliaSim with the assumption of Red =1, Green = 2, Blue = 3. The proposed method of CCLs detection is illustrated in Fig. 4.

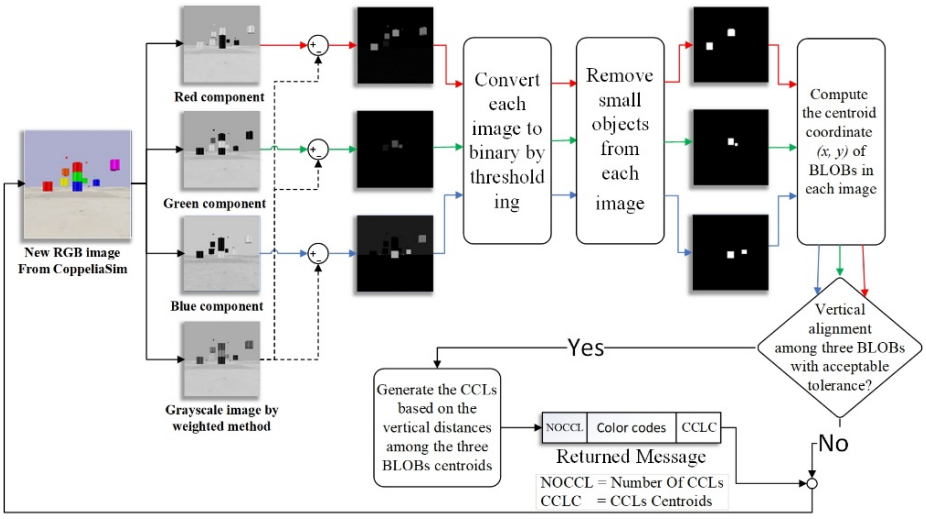

Fig. 4. The proposed CCL detection algorithm.

5. Landmarks Exploration Algorithm

The following scenario is considered: the robot is required to move to a target pose or multiple targets relative to the global (room) frame and tackle some tasks such as delivering materials, sensing, welding, etc. Additionally, the robot does not know its current pose relative to the global frame. The robot must recognize its location relative to the global frame, not necessarily for the entire time, so that it can move to the target pose using its onboard sensors and EKF.

The proposed landmark exploration algorithm (LEA) assumes the following: 1) the robot is equipped with a camera, or multiple cameras, mounted on the front side that detects the CCLs in 180° range in front of the robot; 2) the robot has a distance sensor mounted in front of the robot (at the same position of the front-facing camera but shifted upward in the z-axis direction) which can measure the distance between the landmark and the robot; it can be a single beam laser, IR sensor or sonar sensor; 3) the robot is equipped with an IMU and encoder to continually estimate its position and orientation using EKF; and iv) the environment is obstacles free; however, obstacle avoidance algorithm such as the one in [31] can be easily integrated with the LEA.

The LEA starts after finding the first landmark. Finding the first landmark can be performed by making the robot follow a wall, moving in a zigzag pattern, or any other suitable method. Finding the first landmark is not the focus of this paper.

Once the robot finds the first landmark, it stops and starts estimation using EKF with zero initial states. The EKF estimates the robot pose relative to the robot's frame (F_r) at the time it found the first landmark. The position of the first landmark relative to the frame F_r can be readily calculated since the robot pose relative to F_r is known as follows: 1) the robot rotates until the landmark becomes in the middle of the image coming from the front-facing camera. This can be achieved as follows: if the image width, in pixels, is normalized and the landmark position (P_{Li}) in the image width is known from the image processing algorithm. Then, the following PID control law centers the landmark on the image of the front-facing camera

$$\omega = K_p e(t) + K_d \dot{e}(t) + K_i \int_0^t e(\tau)\, d_\tau \tag{11}$$

where the error $e(t) = 0.5 - P_{Li}$; 2) the distance (d_{L1}) between the landmark and the robot is measured by the distance sensor mounted above the front-facing camera. Thus, the position of the landmark P_L relative to the frame F_r can be calculated as

$$\begin{aligned} P_{Lx} &= P_{rx} + d_L \cos\theta \\ P_{Ly} &= P_{ry} + d_L \sin\theta \end{aligned} \tag{12}$$

where P_r and θ is the robot position and orientation relative to the frame F_r respectively. The position P_L and the extracted global position of the landmark from the CC are recorded and the robot will ignore the landmark if it sees it again.

Once the position P_{L1} and the global position of the first landmark are recorded and given that landmarks are positioned in a way such that for every landmark there is at least one landmark such that the distance between them is $\leq 2d_{max}$ where d_{max} is the maximum range of the distance sensor, the robot rotates 90° CW or CCW using (11) with $e(t) = \theta_d - \theta$, $\theta_d = \hat{\theta} + \pi/2$, and $\hat{\theta}$ is the robot orientation at the time when the first landmark becomes at the centre of the image

of the camera. The robot then moves in a circular motion with a radius of d_{max} around the first landmark until it finds the second landmark.

If the desired velocity vector to make the robot moves in a circular path is generated as follows

$$\begin{bmatrix} \mathrm{v}_{dx} \\ \mathrm{v}_{dy} \end{bmatrix} = \begin{bmatrix} \cos\tilde{\theta} & -\sin\tilde{\theta} \\ \sin\tilde{\theta} & \cos\tilde{\theta} \end{bmatrix} \begin{bmatrix} k\cos wt \\ k\sin wt \end{bmatrix} \tag{13}$$

where $\tilde{\theta}$ is the robot orientation at the time it finishes the 90° rotation, $k > 0$ and w denotes the angular frequency (k and w are design parameters which can be tuned to make the robot move at the desired speed and the radius of the circle equal to d_{max} respectively), then the desired angle θ_d and the desired speed $\|\mathbf{v}\| = \dot{\mathbf{X}}_{\mathrm{R}}$ (assuming $\dot{\mathbf{Y}}_{\mathrm{R}} = 0$) of the robot can be determined using:

$$\theta_d = \mathrm{atan2}(\mathrm{v}_{dy}, \mathrm{v}_{dx}) \tag{14}$$

$$\dot{\mathbf{X}}_{\mathrm{R}} = \sqrt{{\mathrm{v}_{dx}}^2 + {\mathrm{v}_{dy}}^2} = k \tag{15}$$

ω is calculated using (11) with $e(t) = \theta_d - \theta$. Then, $\dot{\mathbf{X}}_{\mathrm{R}}$ and ω are plugged into (2) to obtain the desired wheels' angular velocities.

Next, the same steps after finding the first landmark are repeated. If the robot detects the first landmark again while moving around the second landmark, it repeats the same steps as if it found a new landmark but without recording the position since it has already been recorded for the first time. The robot is then guaranteed to find the third landmark as per the arrangement of the landmarks as shown in Fig. 5 (the case where the distance between the landmark 1 and 2 is exactly equal to $2d_{max}$).

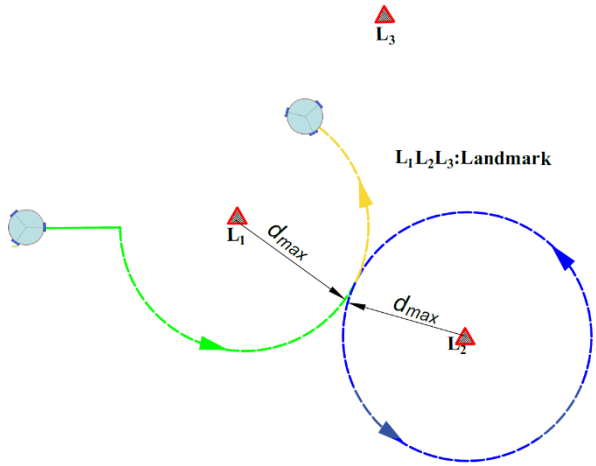

Fig. 5. The case when the robot detects the first landmark again before the third landmark.

At this point, the robot knows its position and the positions of three landmarks to the frame F_r. Now, it is straightforward to calculate the distance between the robot and each one of the three landmarks utilizing the distance formula between two points as follows

$$d_i = \sqrt{(P_{rx} - P_{Lix})^2 + (P_{ry} - P_{Liy})^2}, \quad \text{i} = 1,2,3 \tag{16}$$

To determine the robot's global position and orientation, eq (7), (8), and (10) are applied. However, the distances d_i, calculated based on the estimated robot's position and orientation from EKF, are not accurate due to the error resulting from the EKF. Consequently, this leads to an error when calculating the robot's global pose.

To overcome this issue, the robot may find a position such that it can detect all three landmarks and directly measure the distance between the robot and the landmarks using the distance sensor. In this case, the EKF error is eliminated and the only purpose of the EKF is to enable the robot to find the position where it can detect the landmarks and measure the distance using the distance sensor. The existing of such a position is not always ensured because it depends on the landmarks' arrangements. Provided that the positions P_r, P_{L1}, P_{L2} and P_{L3} are known, the *necessary* condition to find the position is that the distance between every two landmarks is $\leq 2d_{max}$. The *sufficient* condition is determined as follows: 1) two circles with radii d_{max} centred at any two landmarks are drawn and the two points of intersections are calculated. These intersection points represent the furthest point from the two landmarks by which the robot can measure the distance to them using the distance sensor, and 2) the distance between each intersection point and third landmark is calculated using the distance formula. If one of the distances is $\leq d_{max}$, then the position is found (see Fig. 6). The robot then moves to that position and rotates until the camera detects a landmark and then use the PID control law (11) to centre the landmark in the image and measure the distance to it using the distance sensor. The latter step is repeated until the distances to the remaining landmarks are measured.

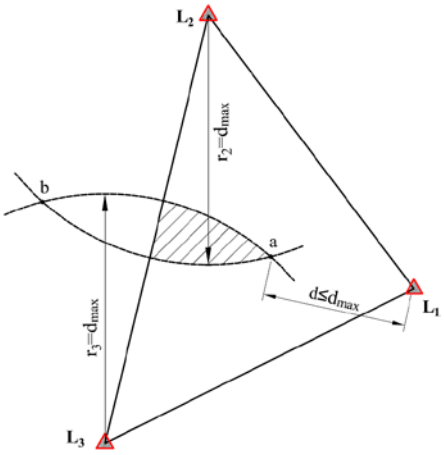

Fig. 6. The sufficient condition for finding the position to measure the distance to the landmarks using a laser sensor. Points a and b represent the points of intersection of the two circles centred at landmarks 2 and 3.

If the necessary condition is not met, i.e., $d > d_{max}$ in Fig. 6, it is still possible to reduce the estimation error of the EKF. This can be achieved if the robot moves to the nearest middle point between any two landmarks and repeats the abovementioned steps to measure the distance to the two landmarks and determine the distance to the third landmark using (16). If there is a distance between any two

landmarks that is $\geq d_{max}$, the same steps are repeated when the necessary condition is not satisfied. After finding the robot's global position, the robot ends the current EKF, starts a new one, and moves to the target position. The robot may start a new LEA and update its pose again if it detects a new landmark on its way to the target. This can occur if the target is too far from the position in which the robot knows its global position to reduce the error from the EKF. The pseudo-codes of the LEA is shown in Algorithm 1.

Algorithm 1 Landmarks Exploration Algorithm (LEA)

1: **if** the first landmark is detected by one of the cameras, then
2: stop the robot
3: start EKF with zero initial states
4: rotate the robot until the front-facing camera face the detected landmark
5: measure and record the distance to the landmark using the distance sensor
6: record the landmark position relative to current EKF coordinates and extract the global coordinates from the CC.
7: rotate the robot 90^o CW or CCW randomly
8: move the robot in a circular motion around the detected landmark until the next (second) landmark is detected
9: execute steps 2,4,5,6,7, and 8
10: **if** the first landmark is detected instead of the third one then
11: execute steps 2,4,7, and 8
12: **end if**
13: execute steps 2,4,5, and 6
14: **if** $d_{12} \leq 2d_{max}$ and $d_{23} \leq 2d_{max}$ and $d_{13} \leq 2d_{max}$then
15: find the points of intersection of the two circles centered at any two landmarks with the radius of d_{max}
16: calculate the distance from the points of intersection to the third landmark using the distance formula
17: **if** any of the distances $\leq d_{max}$ then
18: move the robot to the point of intersection and execute steps 4 and 5 for each of the three landmarks
19: **else**
20: move the robot to the nearest middle point between any two landmarks and execute steps 4 and 5 for each of the two landmarks
21: calculate the distance to the third landmark using the distance formula
22: **end if**
23: **else if** there is a distance between any two landmarks that is $\leq d_{max}$
24: execute steps 20 and 21
25: **else**
26: calculate the distances to landmarks one and two using the distance formula
27: **end if**
28: calculate the global pose of the robot using (7), (8), and (10)
29: **return** the global pose of the robot
30: **end if**

Note that after finding the third landmark, the robot moves from its current position to the position in which it can measure the distances to the landmarks and from that position to the target pose as follows: 1) the robot first rotates by the angle $\alpha = \varphi - \theta$ (see Fig. 1) using the PID control law (11) with $\theta_d = atan2(y_{tp} -$

$y_{cp}, x_{tp} - x_{cp}$) where (y_{cp}, x_{cp}) and (y_{tp}, x_{tp}) denote the cartesian coordinates for current and target robot positions respectively, and 2) the robot then moves to the target position using the following control law [32].

$$\dot{\mathbf{X}}_{\mathrm{R}} = K_{\rho}\rho\cos\alpha \tag{17}$$

where $K_{\rho} > 0$, $\rho = \sqrt{(x_{tp} - x_{cp})^2 + (y_{tp} - y_{cp})^2}$, and $\alpha = 0$ as per the first aforementioned robot rotation. Once the robot reaches the target position, it rotates to the target orientation (θ_t) using the PID control law (11) with $\theta_d = \theta_t$.

6. Results and Discussion

In this section, simulation studies by using the CoppeliaSim and MATLAB software are accomplished to validate the effectiveness of the proposed LEA. The mobile robot used in these studies is a three-wheeled omnidirectional robot whose dimensions and configuration are shown in Fig. 1. Two simulation scenarios are considered. In the two scenarios, the global frame is at the upper right corner of the room as shown in Fig. 7 where the red, green, and blue arrows represent the positive X, Y, and Z axes respectively. The robot is equipped with a camera that detects landmarks in 180^{o} range in front of the robot. The landmark detection algorithm is done inside MATLAB. The data transfer between CoppeliaSim and MATLAB is performed through a regular API. The camera's maximum range in which the landmark can be detected is set to equal to d_{max}. The distance sensor used in the simulation is a single ray laser sensor with a maximum range of 3.5m. In both scenarios, the robot is initially at point A and is required to go to the target pose (1.5 m, 18 m, 180^{o}). The robot does not know its position nor orientation relative to the global frame at point A. The global position of landmarks 1 and 2 are (10m,7m) and (5.5 m, 8 m) respectively in the first and second scenario and the global position of landmark 3 is (8m,12.5m) in the first scenario and (12.5m,11.5m) in the second one. The PID parameters used in (11) are 0.25, 0.1 and 0.15 for K_p, K_d and K_i respectively. The value of K_{ρ} used in (17) is 0.2.

In the first scenario which is shown in Fig. 7, the robot with an unknown pose starts moving forward (point A) as indicated by the blue arrow in front of the robot which always points to the current robot's direction. The camera then detects landmark 1. The robot stops (point B), starts EKF with zero initial states, and rotates until landmark 1 becomes in the center of the image received from the camera using the control law (11). After that, the robot measures the distance to the landmark, records its position relative to the current EKF coordinates using (12), and extracts and records the global coordinates of the landmark from the CC. Next, the robot rotates 90o CW using (11) and begins rotating around landmark 1 using (13) until the camera spots landmark 2 (point C). The same aforementioned steps are repeated except starting an EKF one until the camera sees landmark 3 (point D).

After finding three landmarks, the robot checks the necessary and sufficient conditions which are met in the first scenario case. Thereafter, the robot moves to the point of the two circles intersection (point E) using (11) and (17) and rotates until it positions landmark 1 in the center of the image using the control law (11) and measures the distance to landmark 1 using the distance sensor. The same steps are repeated for landmarks 2 and 3. Afterwards, the robot calculates its global position and orientation using (7), (8), and (10), ends the current EKF, starts a new

EKF with the computed global pose as an initial pose, and moves to the target using (11) and (17). It is worth noting that the global robot pose computed at point E is very accurate as shown in Table 1. This is due to that 1) the solution of the trilateration problem is unique and the error in the robot pose is due to the error in the distance sensor measurements (distance calculation in CoppeliaSim is very accurate) and 2) The estimation error of the EKF is eliminated since it is solely used to find the position in which the robot can measure the distance to the three landmarks using the distance sensor directly. The error in the robot poses when it reaches the target is shown in Table 1 as well.

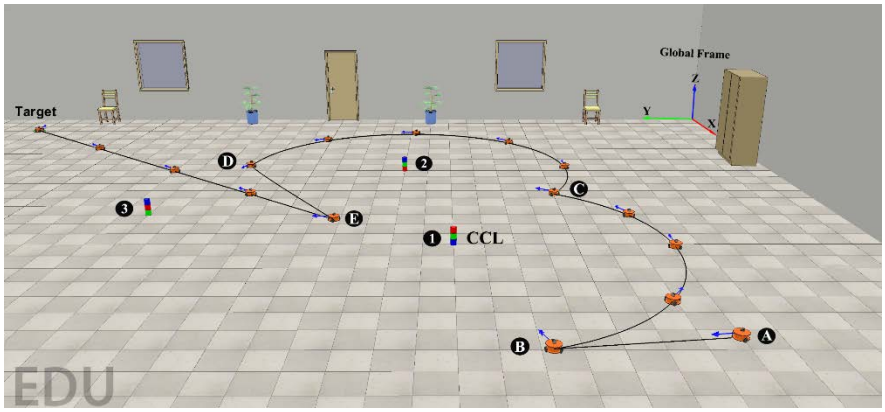

Fig. 7. The robot deploys LEA to reach the target pose in the first scenario.

Table 1. The first scenario's pose error.

Scenario 1	**Actual Pose - Determined Pose = Error**		
	X (m)	Y (m)	θ (°)
Tri. pose	9.1050-9.1051=0.0001	8.8276-8.8258=0.0018	71.638-71.619=0.019
Target pose	1.5000-1.5540=0.0540	18.000-18.076=0.0760	180.00-182.83=-0.283

The second scenario shown in Fig. 8 is different from the first one as the distance between landmark 1 and 3 is greater than d_{max}. The robot behaves in a similar way to the first scenario at points A, B, C. However, at point D the robot spots landmark 1 again before landmark 3. In this case, the robot repeats the same steps when spotting landmark 2 except recording the landmark position and extracting its global position from the CC since it has already been performed when the robot was at point B.

The robot then detects landmark 3 at point E and the necessary condition is not satisfied in this scenario since d_23≥d_max. Thus, the robot moves to the nearest middle point between any two landmarks provided that the distance between them is ≤d_max. Next, the robot measures the distance to landmarks 1 and 3 using the distance sensor in a similar way to the first scenario, whereas the distance to landmark 2 is measured using the distance formula since it is out of the distance sensor range. The robot then finds its pose and moves to the target as in the first scenario.

It can be observed that there is an error in the computed global pose at point F as shown in Table 2. This is because the distance to landmark 2 is computed based on the estimated position of the robot and landmark 2 using EKF. The error in the

robot poses when it reaches the target is shown in Table 2 as well. The error in estimating the robot pose in the second scenario could be eliminated if more restrictions of the landmark's arrangement are set. Furthermore, the error when the robot reaches the target could be reduced if the landmarks are placed near the target.

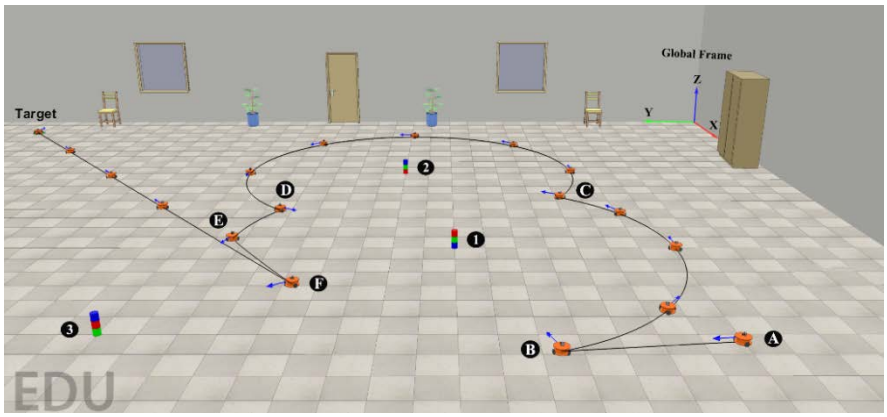

Fig. 8. The robot employs LEA to go to the target in the second scenario.

Table 2. The second scenario's pose error.

Scenario 2	**Actual Pose - Determined Pose = Error**		
	X (m)	Y (m)	θ (°)
Tri. pose	9.3748-9.3274= 0.0474	11.510-11.587=-0.0770	214.70-216.119=-1.4190
Target pose	1.5000-1.6010=-0.1010	18.000-18.132=-0.1320	180.00-184.880=-4.8800

As a practical consideration, a value of less than d_{max} is used in the landmark arrangement and the two circle intersection conditions. This is due to the estimation error ensuing from the EKF; consequently, the robot is not able to measure the distance, i.e., out of range, to the landmark if there is a small error in the computed intersection point.

The accuracy of the EKF estimation can be calculated as follows. The exact robot pose is known at each simulation step because the LEA is implemented inside the CoppeliaSim and it is seamless to get the object pose when using it. Then, the exact and estimated robot paths can be plotted to visualize the estimation error. To reduce the estimation error, the process noise covariance matrix (Q) and the measurement noise covariance matrix (R) can be manually tuned until the exact and estimated robot paths become close to each other. Alternatively, a performance index can be constructed as

$$J = \sum_{t_i=t_0}^{tf} \|e_i\|^2 = \sum_{t_i=t_0}^{tf} e_i^T e_i$$

where $e_i = \left(x_i[k] - \hat{x}_i[k], \quad y_i[k] - \hat{y}_i[k], \quad \theta_i[k] - \hat{\theta}_i[k]\right)^T$ is the error vector of the estimated states ($e = exact\ states - estimated\ states$), t_0 and t_f are the initial and final discrete-time of the tuning interval respectively. Then, the tuning

of Q and R matrices can be done using one of the optimization algorithms, e.g., genetic algorithm, to minimize that performance index.

7. Conclusion and Future Work

The main contribution of this research was to provide a novel LEA. The LEA enabled a mobile robot equipped with a vision sensor to execute a systematic exploration of CCLs, obtain its pose, and then go to a certain target. EKF and trilateration methods were employed in the LEA. This algorithm is based upon detecting CCLs, store their location, find the best position among them to apply the trilateration method, and reach the target in the final step. Certain assumptions were pointed out to work in a practical and reasonable environment. The impressive CoppeliaSim Simulator in conjunction with MATLAB, for landmarks detection only, was used as the environment for testing and evaluating the LEA. Two scenarios were considered to show the capability of the LEA for robot pose estimation. Simulation results demonstrated the satisfactory performance of the LEA and successfully estimated the mobile robot pose. Future developments will focus on the implementation of the LEA in actual experiments. Moreover, the algorithm may be extended to avoid obstacles while performing localization.

Nomenclatures

a_{xm}, a_{ym}	The measured acceleration along the *x*-axis and *y*-axis of IMU respectively
$b_{ax}, b_{ay}\ b_{\omega}$	Basis in terms of x, y, and angular.
d_{max}	The maximum range of the distance sensor
d_i	The distance between a landmark i and the robot.
F_r	Robot Frame
P_{Li}	Position of a landmark i.
P_r	Robot position
$w_{as}, w_{ay}\ w_{\omega}$	Noises in terms of x, y, and angular.
X_R	*X*-axis of the robot frame
x_i, y_i	Coordination of a Landmark i.
x, y	Coordination of the robot.
Y_R	*Y*-axis of the robot frame
y_{cp}, x_{cp}	The cartesian coordinates for the current robot position
y_{tp}, x_{tp}	The cartesian coordinates for the target robot position

Greek Symbols

$\dot{\phi}_1, \dot{\phi}_2, \dot{\phi}_3$	Angular velocities belong to Motors 1, 2, 3
$\emptyset_i$	Angle between a Landmark I and the robot
θ_R	Angle of the robot
θ_d	Desired Angle

Abbreviations

BLOB	Binary Large Object
CCLs	Colour code landmarks
CCW	Counter clockwise
CW	Clockwise

EKF	Extended Kalman Filter
IMU	Inertial Measurement Unit
LEA	Colour code landmarks

References

1. Zafari, F.; Gkelias, A.; and Leung, K.K. (2019). A survey of indoor localization systems and technologies. *IEEE Communications Surveys & Tutorials*, 21(3), 2568-2599.
2. Alatise, M.B.; and Hancke, G.P. (2017). Pose estimation of a mobile robot based on fusion of IMU data and vision data using an extended Kalman filter. *Sensors*, 17(10), 2164.
3. Brena, R.F.; García-Vázquez, J.P.; Galván-Tejada, C.E.; Muñoz-Rodriguez, D.; Vargas-Rosales, C.; and Fangmeyer, J. (2017). Evolution of indoor positioning technologies: A survey. *Journal of Sensors*, 2017, 21.
4. Gutmann, J.-S.; Eade, E.; Fong, P.; and Munich, M. (2010). A constant-time algorithm for vector field slam using an exactly sparse extended information filter. *Conference: Robotics: Science and Systems VI*, Universidad de Zaragoza, Zaragoza, Spain.
5. Nazemzadeh P.; Fontanelli D.; and Macii D. (2016). Optimal placement of landmarks for indoor localization using sensors with a limited range. *International Conference on Indoor Positioning and Indoor Navigation* (*IPIN*). Alcala de Henares, Spain, 1-8.
6. Han, S.-B.; Kim, J.-H.; and Myung, H. (2012). Landmark-based particle localization algorithm for mobile robots with a fish-eye vision system. *IEEE/ASME Transactions on Mechatronics*, 18(6), 1745-1756.
7. Betke M; and Gurvits, L. (1997). Mobile robot localization using landmarks. *IEEE transactions on robotics and automation*, 13(2), 251-263.
8. Zhou, Y. (2009). An efficient least-squares trilateration algorithm for mobile robot localization. *In* 2009 *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Missouri, USA 3474-3479.
9. Sabattini, L.; Secchi, C.; Fantuzzi, C. and Stefani, A. (2010). Bird's–eye view image for the localization of a mobile robot by means of trilateration. *IFAC Proceedings Volumes*, 43(16), 223-228.
10. Moleski, T.W.; and Wilhelm, J. (2020). Trilateration positioning using hybrid camera-LiDAR system. *In AIAA Scitech* 2020 *Forum*, Orlando, FL, 0393.
11. Rupp, T.; and Levi, P. (2000). Optimized landmark arrangement for absolute localization-a practical approach. *Proceedings* 2000 *IEEE/RSJ International Conference on Intelligent Robots and Systems* (*IROS* 2000)(*Cat No* 00*CH*37113). Takamatsu, Japan, 448-453.
12. Marsland, S.; Nehmzow, U.; and Duckett, T. (2001). Learning to select distinctive landmarks for mobile robot navigation. *Robotics and Autonomous Systems*, 37(4), 241-260.
13. Jourdan, D.B.; and Roy, N. (2008). Optimal sensor placement for agent localization. *ACM Transactions on Sensor Networks* (*TOSN*), 4(3), 1-40.

14. Beinhofer, M.; Müller, J.; and Burgard, W. (2013). Effective landmark placement for accurate and reliable mobile robot navigation. *Robotics and Autonomous Systems*, 61(10), 1060-1069.
15. Magnago, V.; Palopoli, L.; Passerone, R.; Fontanelli, D.; and Macii, D. (2017). A nearly optimal landmark deployment for indoor localisation with limited sensing. *International Conference on Indoor Positioning and Indoor Navigation* (*IPIN*). Sapporo, Japan, 1-8.
16. Magnago V.; Palopoli L.; Passerone R.; Fontanelli D. and Macii D. (2018). Optimal landmark placement for indoor positioning using context information and multi-sensor data. *IEEE International Instrumentation and Measurement Technology Conference* (*I2MTC*). Houston, TX, USA, 1-6.
17. Zhang, L.; Wei, K.; Qixiang, Y.; and Jianbin, J. (2014). A novel laser vision sensor for weld line detection on wall-climbing robot. *Optics & Laser Technology*, 60, 69-79.
18. Eric, N.; and Jang, J.W. (2017). Kinect depth sensor for computer vision applications in autonomous vehicles. 2017 *Ninth International Conference on Ubiquitous and Future Networks* (*ICUFN*). Milan, Italy, 531-535.
19. Jeon, Y.; Park, J.; Kang, T.; and Lee, J.O. (2013). A distance measurement system using a laser pointer and a monocular vision sensor. *Journal of the Korean Society for Aeronautical & Space Sciences*, 41(5), 422-428.
20. Rusli M.E.; Ali M.; Jamil, N.; and Din, M.M. (2016). An improved indoor positioning algorithm based on rssi-trilateration technique for internet of things (iot). 2016 *International Conference on Computer and Communication Engineering* (*ICCCE*). Kuala Lumpur, Malaysia, 72-77.
21. Yi, G.H; bin Djaswadi, G.W; bin Md Khir, M.H; and Ramli, N. (2018). An Adaptive Wi-Fi Trilateration-Based Indoor Localization. 2018 *International Conference on Intelligent and Advanced System* (*ICIAS*). Kuala Lumpur convention Centre, Malaysia, 1-6.
22. Monta, S.; Promwong, S.; and Kingsakda, V. (2018). Evaluation of ultra wideband indoor localization with trilateration and min-max techniques. 2016 13*th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology* (*ECTI-CON*). Chiang Mai, Thailand, 1-4.
23. Cotera, P.; Velazquez, M.; Cruz, D.; Medina, L.; and Bandala, M. (2016). Indoor robot positioning using an enhanced trilateration algorithm. *International Journal of Advanced Robotic Systems*, 13(3), 110.
24. Siegwart, R.; Nourbakhsh, I.R.; and Scaramuzza, D. (2011). *Introduction to autonomous mobile robots*. (2nd ed.) The MIT Press.
25. Pomárico-Franquiz, J.; Khan, S.H.; and Shmaliy, Y.S. (2014). Combined extended FIR/Kalman filtering for indoor robot localization via triangulation. *Measurement*, 50, 236-243.
26. El-Naggar, A.; Wassal, A.; and Sharaf, K. (2019). Indoor positioning using WiFi RSSI trilateration and INS sensor fusion system simulation. *Proceedings of the* 2019 2*nd International Conference on Sensors, Signal and Image Processing*. Prague, Czech, 21-26.
27. Kao, C.H.; Hsiao, R.S.; Chen, T.X.; Chen, P.S.; Pan, M.J. (2017). A hybrid indoor positioning for asset tracking using Bluetooth low energy and Wi-Fi.

2017 *IEEE International Conference on Consumer Electronics-Taiwan* (*ICCE-TW*). Taoyuan, Taiwan, 63-64.

28. Shen, J.; Wang, A.; Wang, C.; and Xiong, N.N. (2017). A RFID based localization algorithm applying trilateration for wireless sensor networks. *Journal of Internet Technology*, 18(5), 1167-1175.
29. Esteves, J.S.; Carvalho, A.; and Couto, C. (2003). Generalized geometric triangulation algorithm for mobile robot absolute self-localization. 2003 *IEEE International Symposium on Industrial Electronics* (*Cat No* 03*TH*8692). Rio de Janeiro, Brazil, 346-351.
30. Pierlot, V.; and van Droogenbroeck, M. (2014). A new three object triangulation algorithm for mobile robot positioning. *IEEE Transactions on Robotics*, 30(3), 566-577.
31. Qasim, M.S. (2016). *Autonomous collision avoidance for a teleoperated UAV based on a super-ellipsoidal potential function*. M.Sc. Dissertation. University of Denver, Colorado, USA.
32. Aicardi, M.; Casalino, G.; Bicchi, A.; and Balestrino, A. (1995). Closed loop steering of unicycle like vehicles via Lyapunov techniques. *IEEE Robotics & Automation Magazine*, 2(1), 27-35.