A Robust Approach for Improving the Accuracy of IMU based Indoor Mobile Robot Localization

Sundarrajan G, Suriya D. Murthy, Srivenkata Krishnan S,
Kiran Kassyap S, Ragul Bhagwanth and Vidhya Balasubramanian
Department of Computer Science and Engineering,
Amrita School of Engineering, Coimbatore
Amrita Vishwa Vidyapeetham,
Amrita University, India
b_vidhya@cb.amrita.edu

Keywords: Robot Localization, Indoor Localization, Sensor Fusion, Inertial Sensors, Dead Reckoning

Abstract: Indoor localization is a vital part of autonomous robots. Obtaining accurate indoor localization is difficult in challenging indoor environments where external infrastructures are unreliable and maps keep changing. In such cases the robot should be able to localize using their on board sensors. IMU sensors are most suitable due to their cost effectiveness. We propose a novel approach that aims to improve the accuracy of IMU based robotic localization by analyzing the performance of gyroscope and encoders under different scenarios, and integrating them by exploiting their advantages. In addition the angle computed by robots to avoid obstacles is used as an additional parameter and appropriately integrated using a complementary filter. Our experiments that evaluated the robot over different trajectories demonstrated that our approach improves the accuracy of localization over applicable existing techniques for above scenarios.

1 Introduction

Robotic indoor localization is a method in which the position and orientation of the mobile robot is determined with respect to the indoor environment and is an important part of any autonomous mobile robot. Autonomous robot systems are commonly being used during disaster response, in industries, as assistive robots etc. In order to support the effective functioning of robots in such scenarios there is a need for accurate and efficient indoor localization, navigation and mapping methods. One of the fundamental challenges in indoor environments is localization and this supports the other two functionalities. In this paper our primary goal is accurate self localization of mobile robots in indoor environments.

Several approaches have been designed for mobile robot localization, which include infrastructure and non-infrastructure based methods. Infrastructure based approaches use existing WiFi or RFID installations to help localize the robot (Choi et al., 2011; Li, 2012; Zhang et al., 2014). However the accuracy of these approaches are environment dependent, therefore we need solutions that do not rely on external infrastructure. Approaches that do not rely on infrastructure use laser range finders, on board cameras and inertial sensors for localization of the robot (DeSouza and Kak, 2002; Guran et al., 2014; Surmann et al., 2003). Laser range finders have high accuracy, but are expensive. Camera based approaches are effective in many situations, but are computationally complex and perform poorly in low light conditions and in presence of occlusions.

Cost effectiveness, practicality, and ease of use has popularized the use of inertial sensors and digital encoders for localization purposes (Guran et al., 2014), and are widely employed by dead reckoning based approaches. While accelerometers and encoders have been used for distance estimates, gyroscopes and magnetometer sensors have been used to estimate orientation. However the accuracy of these sensors are affected due to accumulation of noise and drift errors from accelerometers and gyroscopes respectively. Digital encoders are affected by slippage and other errors. To overcome these errors, map matching techniques and sensor fusion techniques have been proposed (Elmenreich, 2002; Xiao et al., 2014).

In map matching, accuracy is improved by considering salient features in the map and repositioning the robot based on it. Techniques like those described in (Xiao et al., 2014) use probabilistic approaches for map matching and thereby localization. Sensor fusion techniques can be used to combine the data from different sensors and improve their accuracy. Sensor fusion can be achieved by simple aggregation based approaches or by applying filters like Kalman, extended Kalman and Complementary filters (Kam et al., 1997). The current state of art in Kalman filter based approaches use two or more different position and orientation estimates. The inputs for the Kalman filter are from sensors embedded in the robot and external features(landmarks,corridors,walls etc.) present in the indoor environment where the robot navigates. However accurate both sensor fusion based and map matching based techniques may be, they are ineffective when maps are incomplete, and the environment consists of dynamic obstacles. Additionally they are computationally expensive. Therefore a cost effective approach that does not completely rely on a map is essential.

In this paper we propose an approach that effectively combines the advantages of the encoders and gyroscopes to improve the accuracy of localization of the robot. While existing techniques like gyrodometry (Ibrahim Zunaidi and Matsui, 2006) have been developed, which use Kalman Filters for sensor fusion, their performance is affected when accelerometers are ineffective (as in the case of wheeled robots). Our approach exploits the curvature of the robot's trajectory to determine when the gyroscopes and encoders are used. This helps limit the continuous use of the gyroscope thereby reducing drift errors. In addition we exploit the obstacle avoidance capabilities of autonomous robots to provide another source of orientation estimate. This is combined with the gyroscope's orientation estimate using complementary filter to improve the localization accuracy.

The rest of the paper is organized as follows: Section 2 outlines the state of art in robotic localization in detail and positions our work with respect to it. Next we discuss how the individual sensors are used for distance and orientation estimate and explain our proposed approach in Section 4. Finally we evaluate our approach for different scenarios (Section 5) and conclude in Section 6.

2 Related Work

Many localization techniques have been developed and implemented in robots over a period of time. Common indoor localization methods for robots rely on external infrastructures and sensors which are part of the robot. Localization techniques that use RF technologies such as RFID, WiFi and bluetooth(Choi et al., 2011; Li, 2012; Zhang et al., 2014) rely on additional external infrastructure such as RF antennas and ultrasonic transceivers placed in the environment. On the other hand, techniques that use cameras, laser range finders or inertial sensors(DeSouza and Kak, 2002; Guran et al., 2014; Surmann et al., 2003) do not rely on external infrastructure.

In the infrastructure based approaches, RF based localization is one of the most prevalent techniques and is easy to implement. In RF based methods, the robot requires external infrastructure like antennas/reference tags placed in environment for communication with receiver which is carried by the robot. RF based localization techniques can either use fingerprinting or non-fingerprinting approaches. The non-finger printing approaches include trilateration method and angulation techniques(AOA) (Liu et al., 2007). The trilateration method estimates the position of the target with respect to the reference points using the Received Signal Strength Indication (RSSI) values (Easton and Cameron, 2006; Granados-Cruz et al., 2014). This method fails to locate mobile node in multipath dense environment. The WiFi fingerprinting technique compares the RSSI observations made by the mobile node with a trained database to determine the location of the moving object (Li, 2012). Deterministic approaches such as K Nearest Neighbors(K-NN) (Kelley, 2015), decision tree methods (Erinc, 2013) and probabilistic methods that include Bayesian, Hidden Markov Model (HMM) have been used in fingerprinting approach. In RFID based localization, a large number of RFID tags placed in the environment act as reference points. Hence, when a new RFID tag enters the space, the signal strength is compared with the reference points' signal strength and location of the robot is determined(Choi et al., 2011; Zhang et al., 2014). In both RFID and WiFi finger printing approaches, the collection of data set (offline phase) is tedious and time consuming. It also requires frequent updation of fingerprint maps and special approaches are needed to reduce the cost of updation (Krishnan et al., 2014).

The techniques that do not rely on infrastructure involve the use of LIDAR (Light Detection and Ranging), cameras and inertial sensors. Camera based approaches use the entire visual information or interest points or combination of all these as input in determining the location of the robot(DeSouza and Kak, 2002). These methods are usually prone to errors due to occlusions, changes in scale, rotation and illumination. In LIDAR based approaches, lasers have been used to emit pulses that are reflected off a rotating mirror from which time of flight is determined and used to calculate distances(Surmann et al., 2003). The main drawback here is that the laser range finders are expensive.

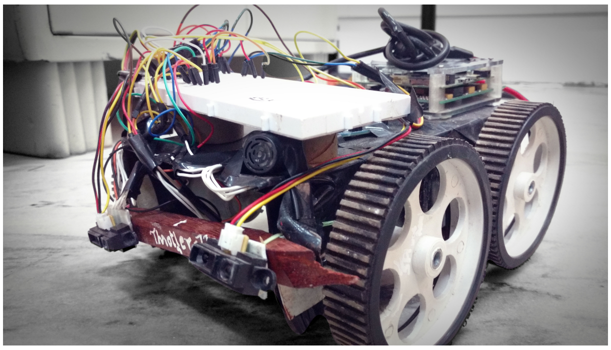
Inexpensive sensors such as accelerometers, gyroscopes and encoders have also been used in localization. These IMU sensors are commonly used in dead reckoning, where new state estimates are calculated with the help of prior states(Ahn and Yu, 2007). Accelerometer and encoders have been used to find the distance while gyroscope and magnetometer sensors have been used to determine the orientation (Guran et al., 2014). These methods do not give accurate results because accelerometer suffers from noise error, gyroscope suffers from static drift and magnetometers are error-prone in places where magnetic interferences are high, especially in indoor environments. The encoders suffers from either systematic(unequal wheel dimensions and kinematic imperfections) or non-systematic errors(wheel slippage and irregularities of the floor) and can be removed to a certain extent using the UMBmark test(Borenstein and Feng, 1994). To overcome the inefficiencies of these sensors we need to effectively combine the data from multiple sensors. Sensor fusion based approaches have been proposed to achieve this. The sensor fusion in this context is the translation of different sensory inputs into reliable estimates(Elmenreich, 2002). Filters like Kalman and complementary filters have been widely used for sensor fusion thereby improving the accuracy to some extent (Kam et al., 1997). Map matching based techniques have also been applied (Xiao et al., 2014) that improve the accuracy of localization accuracy using probabilistic approaches. As mentioned in previous section, the above methods fail when the map is not reliable and environment is dynamic.

Therefore we need a better method that is able to effectively localize the robot with minimum overhead and no external infrastructure. In addition it must be able to exploit the advantage of different sensors to improve accuracy and without relying on maps. In this paper we aim to achieve this using a novel decision based approach that uses the scenarios where different sensors work accurately. The next sections will describe our approach for improving the localization accuracy of indoor robots using inertial measurement units.

3 System Overview

Before we describe our approach we first introduce our robot, its system and its sensors. An image of our robot is shown below in Fig 1. Our robot is a four wheeled autonomous light weight vehicle and has a dimension of 105mm*55mm*57mm. Each of the 4 wheels has a diameter of 110mm. The robot is equipped with bStem (bst, 2015) single chip computer, an Arduino micro controller, sensors and motor drivers. The environment in which the robot navigates is a partially known environment, in which walls and corridors are known but any furniture or moving obstacles are unknown. However for the purpose of our localization we do not use any prior map information including walls.

Figure 1: Our 4-wheeled mobile robot



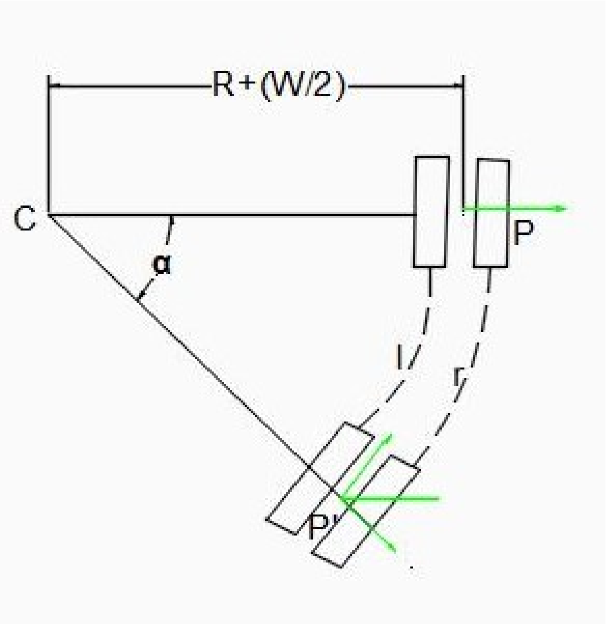
The robot is defined by its pose p which is given at time t as,

$$p = [x_t, y_t, \theta_t]$$

, where x_t, y_t represents the position estimate and θ_t represents the orientation estimate.

To measure the distance travelled by the robot, two quadrature encoders associated with the gear motors are added to the rear wheels of the robot. These encoders output the left and right ticks(l, r), using which the distance travelled by the robot is calculated. It also provides us with an estimate of the robot's orientation θ_e. The tri-axial gyroscope present in bStem provides the angular rates with respect to each axis. Using the z-axis angular rate we calculate the orientation angle θ_g. Two ultrasonic and two infrared sensors are embedded on either sides in-front of the robot. Both the sensors provide distance between obstacle and robot. This data serves to provide us with an additional orientation estimate. We use the encoders to determine the distance estimate and a combination of gyroscope, encoders and obstacle avoidance sensors for orientation estimate to effectively determine pose. The mathematical determination of position and orientation estimate from encoders, gyroscope and obstacle avoidance sensors are explained below.

Figure 2: When the robot takes left turn



3.1 Position and Orientation Estimation using Encoders

The encoders provide ticks l and r from left and right motors respectively. The width of the robot is defined as W. Using these, the position and orientation is calculated by applying geometric techniques (Jensfelt, 2001). In this technique the pose is estimated for the following two cases:

- Robot is taking turns.
- Robot is moving straight.

3.1.1 Case 1: Determination of pose when the robot is taking turns

When the robot is taking turns, there is a difference in the ticks between the right and left encoders. We use this information to estimate the curvature of the robots path (based on the previous position $p^{'}$) and hence determine the pose p. Figure 2 represents the situation where the robot is taking a left turn.

The angle α represents the change in heading angle of the robot during traversal from position $p^{'}$ to p. As shown in figure when the arc is extended, it results in a circle with center at C and radius R. Based on the arc we can characterize the left tick l and right tick r values, and therefore α as follows:

$$l = \alpha R \tag{1}$$

$$r = \alpha(R+W) \tag{2}$$

$$\alpha = (r-l)/W \tag{3}$$

The radius of the robot is proportional to l and inversely proportional to α,

$$R = l/\alpha \tag{4}$$

The heading direction changes when the robot turns. The current orientation is the heading angle which is based on the previous orientation, and α. The new orientation estimate $\theta_e(t)$ is calculated as follows,

$$\theta_e(t) = \theta_e(t-1) + \alpha \tag{5}$$

In order to obtain the position estimates $x(t), y(t)$, the center C(C_x,C_y) must be determined. The center is calculated with respect to point $p^{'}$ and its direction vector at point $p^{'}$.

$$C_x = x(t-1) - (R+(W/2))\sin(\theta_e(t-1)) \tag{6}$$

$$C_y = y(t-1) - (R+(W/2))(-\cos(\theta_e(t-1))) \tag{7}$$

The new position estimate ($x(t)$, $y(t)$) is determined using the center calculated previously,

$$x(t) = C_x + (R+(W/2))\sin(\theta_e(t)) \tag{8}$$

$$y(t) = C_y + (R+(W/2))(-\cos(\theta_e(t))) \tag{9}$$

The pose as determined above is calculated considering the case when the robot turns left. Similarly, this can be extended for the calculation of new pose when robot turns right.

3.1.2 Case 2: Determination of pose when the bot is moving straight

When the robot is moving straight, there is no change in the orientation of robot, and the left ticks are equal to the value of right ticks. So, the new orientation estimate is same as the orientation at time $t-1$,

$$\theta_e(t) = \theta_e(t-1) \tag{10}$$

Now, the corresponding pose $x(t), y(t)$ is calculated as given below,

$$x(t) = x(t-1) + l(\cos(\theta_e(t))) \tag{11}$$

$$y(t) = y(t-1) + l(\sin(\theta_e(t))) \tag{12}$$

3.2 Orientation estimate using gyroscope

Gyroscope is a sensor used to measure angular velocity. In general, angular velocity is the change in rotational angle per unit time, which is expressed in radians per second(rad/sec). In our robot, the L3GD20 MEMS motion sensor 3-axis digital gyroscope which is part of the bStem, provides angular velocities along three axes x,y,z respectively. The sensitivity of the gyroscope is set at 250dps, ideal for a robotic system that undergoes heavy vibrations due to motor and chassis movement. The goal is to calculate the orientation angle of the robot over a period of time, which is done by integration of angular velocity about the z-axis over a certain time interval.

In order to get reliable angle estimates we first calculate and eliminate the offset and noise of our gyroscope. The offset is calculated as an average of angular velocities when the robot is at rest.

$$offset = \frac{1}{N}\sum_{i=0}^{N} \quad \omega \qquad (13)$$

where ω is the angular velocity at that instant of time and N represents the number of angular velocity values taken. The offset is calculated separately for the three axes.

The noise is characterized by the maximum difference between angular velocity(ω) and $offset$ for values in which robot is at rest.

$$noise = \max(\omega_i - offset) \qquad 0 \leq i \leq N \qquad (14)$$

where ω_i is the angular velocity at that instant. Noise is calculated for all three axes.

The angular velocities which fall in the range of $[-noise, noise]$ are ignored and the remaining values are taken for the calculation of orientation. The orientation is the integration of angular rates over a interval of time. The orientation angle $\theta_g(t)$ at time t is given by,

$$\theta_g(t) = \theta_g(t-1) + \omega_i dt \qquad (15)$$

where dt represents the time period over which ω_i can be periodically integrated to determine the angle, in our case this value is equal to 20ms. Using the above $\theta_g(t)$ value we estimate the orientation and position.

The position estimate is calculated using previous position values, the encoder values and the newly calculated orientation value and is determined as follows.

$$dist = (l+r)/2 \qquad (16)$$

$$x(t) = x(t-1) + dist(\cos(\theta_g(t))) \qquad (17)$$

$$y(t) = y(t-1) + dist(\sin(\theta_g(t))) \qquad (18)$$

3.3 Orientation estimate using obstacle avoidance sensors

The distance between obstacle and robot is estimated by the ultrasonic and infrared sensors. The sensor specifications are given below,

1. Ultrasonic Range Finder XL-EZ3 (MB1230) which has a range of 20cm to 760cm and a resolution of 1cm is used to detect any obstacles present in long range.
2. Sharp GP2Y0A41SK0F IR Distance Sensor which has a range of 4cm to 30cm is used to determine obstacles in close range. These infrared sensors have faster response time than ultrasonic sensors and are used in addition.

A robot that is moving autonomously takes navigational decisions based on the output from the above sensors. An obstacle avoidance method (Widodo Budiharto and Jazidie, 2011), is adapted to calculate the orientation of the robot as is explained below.

When an obstacle is detected, the two ultrasonic sensors return the distance between the obstacle and robot, which is denoted by d_{lu}, d_{ru}. d_{lu} is the distance from obstacle and left ultrasonic sensor. Similarly, d_{ru} is the distance from obstacle and right ultrasonic sensor. d_{safe} is the minimum distance at which robot can start to maneuver to avoid obstacle. d_{safe} is called the flank safety distance. This value is calculated experimentally based on the range of the ultrasonic sensors and width of the robot.

To determine the angle from the distances given by ultrasonic and infrared sensors, the following cases must be considered.

3.3.1 case 1: $d_{lu} > d_{safe}$ and $d_{ru} > d_{safe}$

This situation occurs when there is no obstacle in front of the robot and the robot can continue to move straight, the orientation angle does not change. So the angle at time t, is given by,

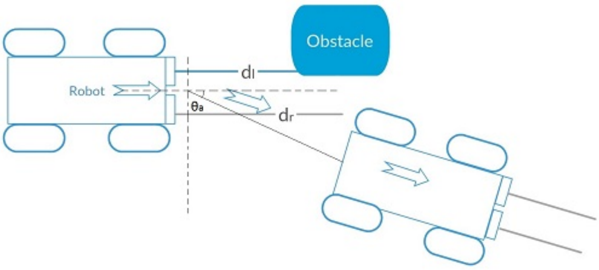
$$\theta_{oa}(t) = \theta_{oa}(t-1) \qquad (19)$$

3.3.2 case 2: $d_{lu} < d_{safe}$ and $d_{ru} > d_{safe}$

The situation is shown in Figure 3. In order to avoid the obstacle, the robot has to turn right. The ideal angle for robot to avoid the obstacle is calculated using distance from left ultrasonic sensor. The angle $\theta_{oa}(t)$ is inversely proportional to distance d_{lu} i.e. smaller the distance higher the angle and vice versa.

$$\theta_{oa}(t) = \theta_{oa}(t-1) - \frac{\pi}{2}(\frac{k_{\theta u}}{d_{lu}}) \qquad (20)$$

Figure 3: Right turn after detecting obstacle



where $k_{\theta u}$ represents the collision angle constant for the ultrasonic sensor, which is calculated based on d_{safe} and d_{lu}.

3.3.3 case 3: $d_{lu} > d_{safe}$ and $d_{ru} < d_{safe}$

When the bot has to turn left in order to avoid obstacle, the ideal angle is calculated using the distance from the right ultrasonic sensor and the angle is calculated similarly,

$$\theta_{oa}(t) = \theta_{oa}(t-1) + \frac{\pi}{2}(\frac{k_{\theta u}}{d_{ru}}) \tag{21}$$

3.3.4 case 4: $d_{lu} < d_{safe}$ and $d_{ru} < d_{safe}$

When there is a nearby obstacle, this case comes into picture. Here, we use the output from infrared sensors i.e., distance between the obstacle and robot to determine the angle.

Similar to ultrasonic sensors, we have cases in which the robot must turn left, right and in cases when the robot is too close to the obstacle, move backwards. The angle calculation for right turn is given by,

$$\theta_{oa}(t) = \theta_{oa}(t-1) - n \times (\frac{\pi}{2})(\frac{k_{\theta i}}{d_{ri}}) \tag{22}$$

Similarly, the angle calculation for left turn is given by,

$$\theta_{oa}(t) = \theta_{oa}(t-1) + n \times (\frac{\pi}{2})(\frac{k_{\theta i}}{d_{li}}) \tag{23}$$

where n is experimentally calculated to increase or decrease the speed appropriately to avoid close obstacles. $k_{\theta i}$ is the collision angle constant for infrared sensors and d_{li} is the distance between left infrared sensor and obstacle. Similarly d_{ri} is the distance between right infrared sensor and obstacle.

Finally, if the robot is too close to obstacle and is impossible for it to maneuver an obstacle smoothly, i.e. if $d_{lu} < d_n$, then the robot moves backwards by a certain distance and checks for all the above possibilities and estimates the orientation angle. d_n is the maximum distance before which the robot should start maneuvering to avoid the obstacle. This value is calculated experimentally based on the range of the infrared sensors and width of the robot. The region $\{d_{safe}$ - $d_n\}$, is the buffer region for smooth maneuvering.

Using the above equations, we estimate the orientation angle based on the data from the obstacle avoidance sensors. We assume that the robot follows this orientation angle to avoid the obstacle during movement. Based on this, the pose of the robot is estimated as follows.

$$dist = (l+r)/2 \tag{24}$$

$$x(t) = x(t-1) + dist(\cos(\theta_{oa}(t))) \tag{25}$$

$$y(t) = y(t-1) + dist(\sin(\theta_{oa}(t))) \tag{26}$$

Now that we have estimated the pose using the individual sensors, the goal of our system is to improve the accuracy by appropriately combining the data from all three sensors.

4 Curvature based Decision System

This section describes our novel approach to improving the location accuracy by fusing information from the different sensors appropriately.

In robotic localization inaccuracies in orientation measure affects the position more when compared to inaccuracies in distance measure(Borenstein and Feng, 1994). For example, a robot with 8 inch wheel base and slip of 1/2 inch in one of the wheels results in an error of approximately 3.5 degrees in orientation. So it is important to have a highly accurate value of the orientation measure when compared to that of a distance measure. Our method assumes that the distance(l,r) measured using the encoders is reliable, since the systematic errors can be easily removed using UMBenchmark and non-systematic errors do not affect the distance estimate much, as explained in the above example. The orientation estimate from encoders however are not reliable since they undergo huge non-systematic errors due to sudden turns and heavy slippage. Slight deviations in distance due to non-systematic errors results in huge deviations in orientations. The gyroscopes are generally accurate in estimating orientation, and is preferred over encoders. However over longer durations the performance degrades due to the accumulation of drift errors. Our approach also enhances the accuracy of orientation estimation in gyroscope by

1. reducing the period of time the gyroscope is used, so that drift errors are minimized

2. improve orientation estimation accuracy by fusing the estimates from gyroscope with those provided by obstacle avoidance system as discussed in Section 3.3.

To achieve above approach, the trajectory in which the robot travels is analyzed. We identify the part of the trajectory that have low non-systematic errors, so that orientation measure from encoders can be used reliably. In the other parts of the trajectory the gyroscope is employed. Since environments contain obstacles, orientation estimations can be obtained from the robot's obstacle avoidance system, and this is fused with the gyroscope data to get a better accuracy of localization. The main steps in our approach are as follows

1. Estimation of curvature of the robot's trajectory
2. Using the trajectory to determine the appropriate sensor to use
3. Enhance orientation estimation accuracy using a complementary filter to fuse orientation from above step and obstacle avoidance system

Figure 4 outlines the flow of the above steps in the form of an architecture diagram. The next paragraphs will discuss these steps in more detail.

4.1 Curvature Estimation

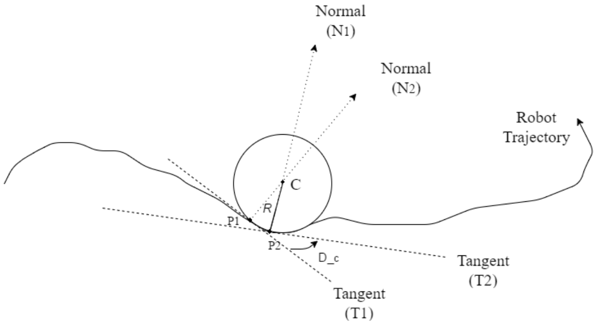
The robot's trajectory is a 2D smooth plane curve. The curvature of the past trajectory of the robot determines the sensor(s) to be used for the orientation estimation at the current point. In order to find the curvature, we find the angle between the slopes of tangents drawn to the previous points $P1(x,y)$ and $P2(x,y)$ on the trajectory. To draw the tangents we find the center of curvature C(C_x,C_y) (as explained in subsection 3.1.1 of the previous section) of the curve. We then calculate the angle between the slopes of the tangents, which gives the curvature at the point. This method is efficient in curvature estimation since we only store the previous two points of the trajectory and not the complete trajectory to decide on the appropriate sensors.

To find the slope $m1$ of the tangent vector T at point $P1$, we first find the slope $n1$ of the normal vector N at $P1$ passing through center of curvature C. The slope of $n1$ is calculated using

$$n1 = (C_y - P2(y))/(C_x - P1(x)) \qquad (27)$$

The slopes $m1$ of the tangent vector T, which is perpendicular to N is then calculated as the negative of the inverse of $n1$; $m1 = -1/n1$. Similarly we calculate $m2$, which is the slope of the tangent vector at point $P2$.

Figure 5: Determination of slope



Once these two slopes are calculated the degree of curvature D_c can be determined as follows

$$D_c = \tan^{-1}((m1 - m2)/(1 + m1 * m2)) \qquad (28)$$

Using this D_c value we calculate the orientation angle as explained next.

4.2 Gyroscope Controller

The key to determining an accurate orientation angle is to calculate the precise start and stop threshold values. These threshold values are calculated experimentally, which depend on the width, speed and turning radius of the robot. The start threshold value $\mu1$ is calculated by computing the average of degrees of curvature when the robot begins to make a turn. In scenarios where the robot completes a maneuver by avoiding an obstacle and starts to follow a linear trajectory, the encoders estimate cannot be instantaneously considered for computation. This is due to the fact that there occurs abrupt difference in the left and right encoder ticks immediately after a sharp turn and hence the average of degree of curvatures of multiple linear paths is computed and used as the stop threshold value $\mu2$.

When the D_c keeps increasing and goes over start threshold value $\mu1$, we initiate the gyroscope and use it for orientation measure and allow it to summate as the robot traverses the curved path. The use of gyroscope is ceased when the degree of curvature becomes closer to the stop threshold value $\mu2$. By doing this we ensure that the gyroscope is not made to run continuously for a long time, thereby ensuring that the drift errors from the gyroscopes are not accumulated. When D_c keeps decreasing and becomes equal to or less than $\mu2$, the encoders take over the orientation estimation of the robot. Using the encoders when the D_c is less than or equal to the $\mu2$, reduces the effect of the non systematic errors resulting in an accurate position calculation.

Figure 4: System Architecture

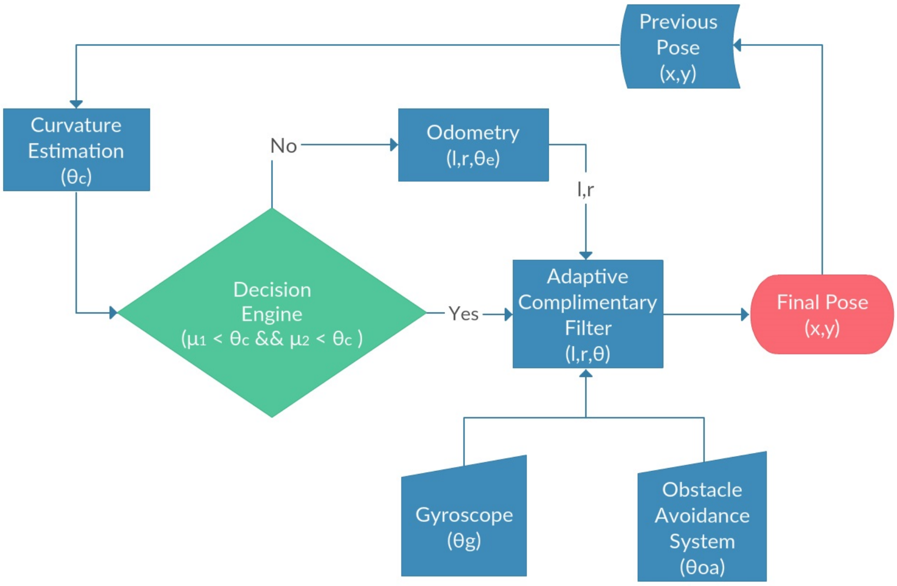
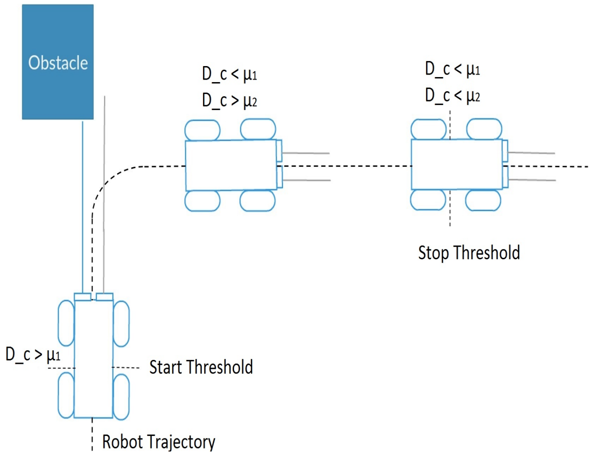


Figure 6: Determination of start and stop threshold



In order to achieve the above the following conditions have to be met,

1. $\mu 2 << \mu 1$, ensuring the left and right ticks are equal in straight paths.

2. $\mu 1$ is given higher precedence than $\mu 2$, ensuring gyroscope is started as soon as the robot begins to make a turn.

In order to further increase the orientation estimate obtained from gyroscope, it is fused with the angle obtained from the obstacle avoidance sensors explained below.

4.3 Gyro-Obstacle Fusion

To make the orientation estimate even more accurate, we get an angle from the obstacle avoidance sensors as explained in Section 3.3. This angle is obtained only when the autonomous robot encounters an obstacle. This angle is then used along with the gyroscope as a parameter to the complementary filter to obtain an accurate orientation. We use a complementary filter since it is very easy and light to implement making it perfect for embedded systems application as in our case. The complementary filter is given by the equation,

$$\theta = \alpha\theta_g + (1-\alpha)\theta_{oa} \tag{29}$$

where θ_g is the angle obtained from gyroscope, θ_{oa} is angle obtained from obstacle avoidance sensors and α is the factor which is determined experimentally.

Therefore, we get an accurate orientation estimate from encoders, gyroscope and obstacle avoidance system. This orientation is then combined with the distance measure(l,r) from encoders to get the pose estimate and are given by equations,

$$dist = (l+r)/2 \tag{30}$$

$$x_t = x_{t-1} + dist(\cos(\theta)) \tag{31}$$

$$y_t = y_{t-1} + dist(\sin(\theta)) \tag{32}$$

On fusing the data by considering the advantages of individual sensors, the position is obtained.

5 Experimental Results

The previous section discussed our approach for improving the accuracy of localization by combining data from different sensors used in our robot. In this section we analyze the performance of this approach. First we discuss the arena and experimental setup followed by the metrics used for analyzing our algorithm. We also discuss the methodology for obtaining the ground truth.

5.1 Environment Setup:

The experiments are carried out in an arena of dimensions $1710 * 480 cm^2$ using a four wheeled autonomous robot as shown in figure 1. The arena is then transformed into a grid of squares, each of area $900 cm^2$. Obstacles are placed in different parts of the arena. Different trajectories are considered for evaluating our algorithm. In order to generate the ground truth the grid is used to identify the points the robot passed through. These points are plotted on a raster map. The following algorithms are evaluated

1. Localization using encoders as described in Section 3.1
2. Localization using gyroscope as described in Section 3.2
3. Localization using Curvature based Decision Approach without integrating obstacle avoidance information (CDA-nO) as described in Section 4.1 and 4.2
4. Localization using Curvature based Decision Approach that uses all three sensors (CDA-O).

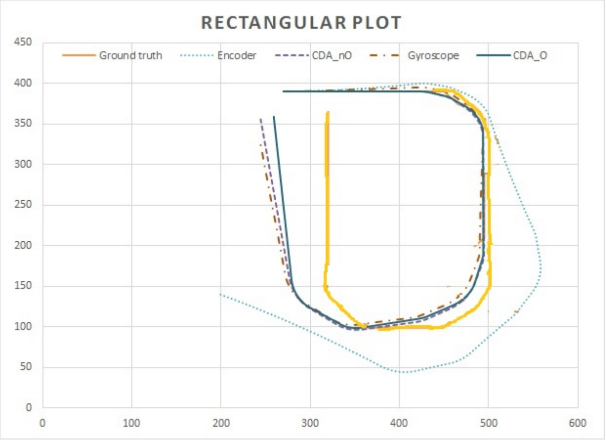
To evaluate the performance of these approaches in comparison to the ground truth we use the metrics as discussed next.

5.2 Metrics of Evaluation

The most important indicator in evaluating the performance of a localization algorithm is accuracy. Accuracy refers to how close the estimated trajectory as obtained from the localization method is to the ground truth. In order to determine the closeness, mean and standard deviation of the Euclidean distance between the ground truth and estimated point is calculated. The mean Euclidean distance is the sum of Euclidean distance at each corresponding point divided by the total number of such corresponding points which is given by N.

$$\mu = \frac{\sum_{i=0}^{N}(\sqrt{((x_{gt} - x_{est})^2 + (y_{gt} - y_{est})^2)})}{N} \tag{33}$$

Figure 7: Performance for a Rectangular Trajectory



(x_{gt}, y_{gt}) refers to the ground truth and (x_{est}, y_{est}) is the estimated position from the localization method used. While the mean provides the error estimates, the standard deviation gives an idea of how consistently the algorithm performs.

5.3 Performance Analysis

We evaluate the above mentioned approaches over the arena specified. The evaluation is done by making the robot navigate along different trajectories in the arena. We have considered three trajectories and their corresponding ground truth is plotted and used for comparison. Tables 1 and show the mean and standard deviation of the euclidean distance error in the corresponding trajectories and methods used for comparison respectively.

5.3.1 Case 1: Rectangular Trajectory

The first case is when the robot traversed along the periphery of the arena resulting in a rectangular trajectory. Figure 7 shows the plot of the estimated trajectories for different techniques along with the ground truth trajectory. From the figure and the table we can see that the average error of encoder based localization is comparatively high. This is because, in the trajectory, the robot has taken sharp turns, which leads to slips and it affects the performance of encoder based method. The performance of gyroscope based localization is better than encoder because its orientation estimates are more accurate during turns, and that dictates the accuracy of the trajectory. The curvature based approach improves the accuracy of the localization over the gyroscope on average. However using the orientation estimates from the obstacle avoidance system further improves the accuracy, which leads us

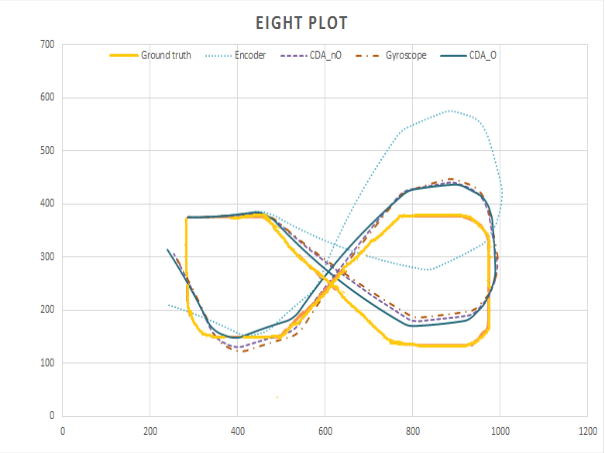
Table 1: Average Euclidean Distance Error

Methods implemented	Average Euclidean Distance error (cm)		
	Trajectory 1 (Square)	Trajectory 2 (Eight)	Trajectory 3 (Random path)
Encoders	121.96	89.27	333.65
Gyroscope	40.50	44.22	75.33
CDA-nO	28.64	39.50	65.38
CDA-O	20.25	34.68	20.49

Table 2: Standard deviation

Methods implemented	Standard deviation(cm)		
	Trajectory 1 (Rectangular)	Trajectory 2 (Eight)	Trajectory 3 (Random path)
Encoders	79.64	51.74	191.74
Gyroscope	22.49	26.86	44.26
CDA-nO	15.96	24.91	38.96
CDA-O	12.59	24.66	12.67

Figure 8: Performance for an Eight Shaped Trajectory



to the conclusion that orientation estimates during obstacle avoidance has high accuracies and can be reliably used.

5.3.2 Case 2: Eight shaped trajectory

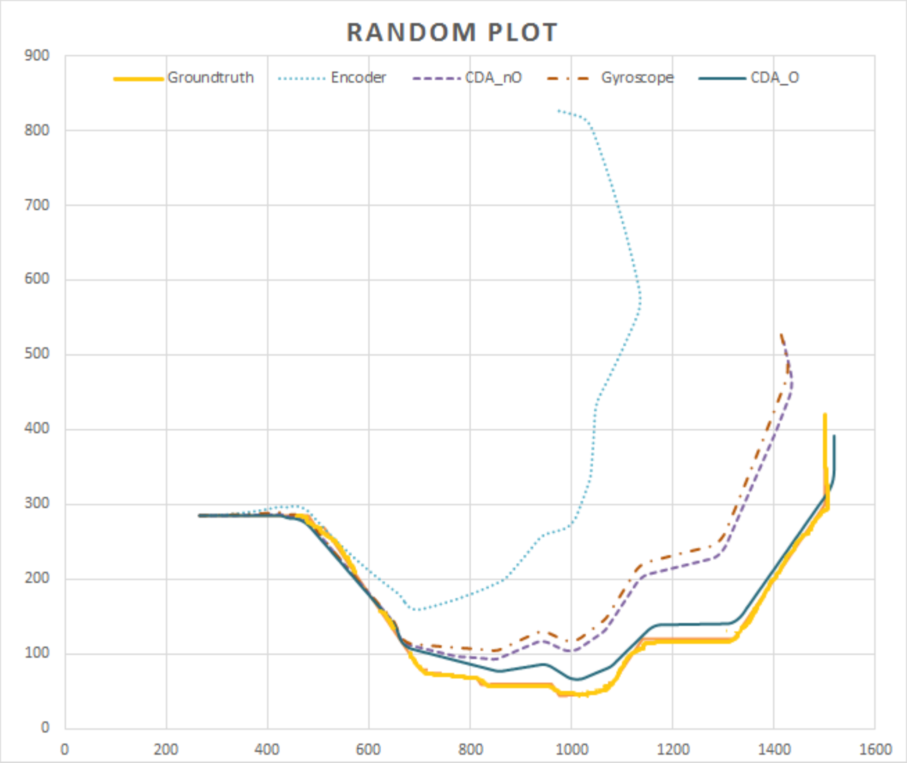
Figure 8 shows the output of the different localization methods when the robot is allowed to navigate in an eight shaped trajectory. Here we can notice that since there are not many sharp turns, the encoder performance improves. Here too our curvature based approaches (both with and without input from obstacle avoidance sensors) perform the best. While the gyroscope plays a major role in the improvement of accuracy, the nature of the trajectory limits the curvature based decision approach. However we can see that employing the input from obstacle avoidance improves the accuracy to a certain extent.

5.3.3 Case 3: Random trajectory

To obtain a better idea of how well our approach for localization works, we allowed the robot to take a random path filled with obstacles, and this path was much longer than the previous ones. When the path is longer the drift errors in the gyroscope play a role. Figure 9 shows the plots of the estimated trajectories for this path. We can see that here that our curvature based approach that utilizes the obstacle avoidance system performs exceedingly well in comparison to all other strategies. This is because the path has lot of obstacles and that is well exploited here. The gyroscope performs poorest here, since it is more prone to drift errors here. Therefore using the curvature based approach helps reduce the drift error, and improve accuracy to a small extent.

Overall we can see that our approach performs consistently well in all scenarios and gives an accuracy of around 20 to 30 cm. We also observe that considering the curvature and fusing the estimate of the orientation from the obstacle avoidance systems with gyroscopes significantly improves accuracy. The trajectories generated by our approach are very close to the trajectory both in terms of distance and the overall pattern, as can be seen in the plots. The standard deviation values, specially for the rectangular and random plots show that our approach not only provides good accuracies, but does it consistently along the path. Additionally the complexity of this approach is low and hence is easily implementable in any autonomous

Figure 9: Random plot



robot.

6 Conclusions and Future Research

In this paper we have described out approach for improving the localization accuracy of indoor mobile robots, which use inertial sensors for localization. The curvature of the robot's trajectory is analyzed to determine when the gyroscope and encoder data are used. In addition to improve the accuracy, orientation estimates from obstacle avoidance systems are fused with the gyroscope's orientation estimates. Our experiments have shown that by using only gyroscopes as and when needed, the drift errors are reduces considerably. By employing the additional data from obstacle avoidance sensors, we see that the accuracy is improved multifold. It is seen that since environments usually have obstacles, and robots have to navigate around them, using that as a parameter is an effective approach for localization. Our approach provides high accuracies at a low overhead, and this approach is computationally efficient compared to many probabilistic approaches. However when the turns are very sharp, and acute, our approach suffers a bit. Experiments are ongoing to improve the accuracy of complex trajectories that include acute angles.

REFERENCES

(2015). bstem developer kit integrated robotics platform: http://www.braincorporation.com.

Ahn, H.-S. and Yu, W. (2007). Indoor mobile robot and pedestrian localization techniques. In *Control, Automation and Systems, 2007. ICCAS'07. International Conference on*, pages 2350–2354. IEEE.

Borenstein, J. and Feng, L. (1994). Umbmark: a method for measuring, comparing, and correcting dead-reckoning errors in mobile robots.

Choi, B.-S., Lee, J.-W., Lee, J.-J., and Park, K.-T. (2011). A hierarchical algorithm for indoor mobile robot localization using rfid sensor fusion. *Industrial Electronics, IEEE Transactions on*, 58(6):2226–2235.

DeSouza, G. N. and Kak, A. C. (2002). Vision for mobile robot navigation: A survey. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(2):237–267.

Easton, A. and Cameron, S. (2006). A gaussian error model for triangulation-based pose estimation using noisy landmarks. In *Robotics, Automation and Mechatronics, 2006 IEEE Conference on*, pages 1–6. IEEE.

Elmenreich, W. (2002). Sensor fusion in time-triggered systems.

Erinc, G. (2013). Appearance-based navigation, localization, mapping, and map merging for heterogeneous teams of robots.

Granados-Cruz, M., Pomarico-Franquiz, J., Shmaliy, Y. S., and Morales-Mendoza, L. J. (2014). Triangulation-based indoor robot localization using extended fir/kalman filtering. In *Electrical Engineering, Computing Science and Automatic Control (CCE), 2014 11th International Conference on*, pages 1–5. IEEE.

Guran, M., Fico, T., Chovancova, A., Duchon, F., Hubinsky, P., and Dubravsky, J. (2014). Localization of irobot create using inertial measuring unit. In *Robotics in Alpe-Adria-Danube Region (RAAD), 2014 23rd International Conference on*, pages 1–7. IEEE.

Ibrahim Zunaidi, Norihiko Kato, Y. N. and Matsui, H. (2006). Positioning system for 4wheel mobile robot: Encoder, gyro and accelerometer data fusion with error model method. In *CMU Journal*, volume 5.

Jensfelt, P. (2001). *Approaches to mobile robot localization in indoor environments*. Citeseer.

Kam, M., Zhu, X., and Kalata, P. (1997). Sensor fusion for mobile robot navigation. *Proceedings of the IEEE*, 85(1):108–119.

Kelley, K. J. (2015). Wi-fi location determination for semantic locations. *The Hilltop Review*, 7(1):9.

Krishnan, P., Krishnakumar, S., Seshadri, R., and Balasubramanian, V. (2014). A robust environment adaptive fingerprint based indoor localization system. In *In Proceedings of the 13th International Conference on Ad Hoc Networks and Wireless (ADHOC-NOW-2014)*, volume 8487, pages 360–373.

Li, J. (2012). Characterization of wlan location fingerprinting systems. Master's thesis, School of Informatics, University of Edinburgh.

Liu, H., Darabi, H., Banerjee, P., and Liu, J. (2007). Survey of wireless indoor positioning techniques and systems. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 37(6):1067–1080.

Surmann, H., Nüchter, A., and Hertzberg, J. (2003). An autonomous mobile robot with a 3d laser range finder for 3d exploration and digitalization of indoor environments. *Robotics and Autonomous Systems*, 45(3):181–198.

Widodo Budiharto, D. P. and Jazidie, A. (2011). A robust obstacle avoidance for service robot using bayesian approach. *International Journal of Advanced Robotic Systems*, 8:52–60.

Xiao, Z., Wen, H., Markham, A., and Trigoni, N. (2014). Lightweight map matching for indoor localisation using conditional random fields. In *Information Processing in Sensor Networks, IPSN-14 Proceedings of the 13th International Symposium on*, pages 131–142.

Zhang, H., Chen, J. C., and Zhang, K. (2014). Rfid-based localization system for mobile robot with markov chain monte carlo. In *American Society for Engineering Education (ASEE Zone 1), 2014 Zone 1 Conference of the*, pages 1–6. IEEE.