•••

***Abstract*—**

***Keywords*—**

I. 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 in disaster response and rescue, in industries, as assistive robots etc. In order to support the effective functioning of the robots in such scenarios there is a need for accurate and efficient indoor localization, navigation and mapping methods. One of the primitive challenges in indoor environments is localization and this supports the other two functionalities. In this paper we primarily focus on indoor localization of mobile robots.

Over the past decade, there have been significant developments in mobile robot localization. Several approaches have been used, which include infrastructure and non-infrastructure based methods. Infrastructure based approaches use existing WiFi or RFID installations to help localize the robot. These approaches becomes infeasible in challenging environments such as disasters, chemical spills etc. where infrastructure will not be available. Approaches that do not rely on infrastructure use laser range finders,camera and inertial sensors for localization of the robot. Laser range finder based approaches provide accuracy, but are very expensive. On the other hand, camera based approaches are computationally complex and performs poorly in low light conditions and presence of occlusions.

The use of inertial sensors and digital encoders for localization have gained more popularity because of its low cost, small dimensions and low power consumption. These sensors are commonly used in dead reckoning based localization methods. The dead reckoning method computes new state estimates with the help of previous state estimates. Accelerometers and encoders have been used for distance estimates while gyroscopes and magnetometer sensors have been used for orientation estimates. The accuracy of these methods are affected due to accumulation of noise and drift errors from accelerometers and gyroscopes respectively. Digital encoders tend to produce errors because of slippage, mechanism symmetry and sensor accuracy limitations. To overcome these errors, sensor fusion techniques and map matching techniques have been proposed.

The current map matching techniques that have been used for location estimation use recursive Bayesian filters, such as HMMs, kalman filter and particle filter. These techniques are computationally complex and require ample sensor data to estimate accurate trajectories. Over the years there have been several sensor fusion approaches that varies from simple aggregation to complex systems which use approaches such as kalman filter,extended kalman filter etc. 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. In the absence of map or in an unknown environment, the external features cannot be used as an input for kalman filter, which decreases the efficiency of the kalman filter. In such situations, a cost effective and an efficient approach is essential.

In our paper, we propose an efficient localization approach. In this approach, we assume encoders to be a reliable source of input for distance calculation as inaccuracies in encoder readings caused due to systematic and non-systematic errors do not reflect much in distance estimate compared to that of orientation estimate. The impact of these errors in orientation estimate is considerably high. So, we propose a system which uses the inputs from encoders, gyroscope, ultrasonic and infrared sensors for accurate orientation estimate. The data from the ultrasonic and infrared sensors of the autonomous navigation system are used for taking advantage of encoders over the disadvantage of gyroscope and vice versa. The orientation obtained in the above methods are subjected to complimentary filter for further increasing the accuracy. The proposed algorithm results in a computationally efficient solution for the localization problem.

The main technical novelties of the paper of this paper are

1) We use complimentary filter for fusing orientation estimates, which is more feasible when parameters used are minimum. It is also computationally less intensive.
2) In obstacle avoidance based orientation calculation,
    a) When the turns are determined, we use gyroscope's orientation estimate. The reason is that non-systematic errors(slips or irregularities in the floor) predominantly occur during turns which affects the angle estimated with encoders.
    b) We use encoder's orientation estimate when the bot moves in linear paths. The reason is that, in a typical indoor environment, the number of turns are comparatively less to number of linear paths. When we use gyroscope for linear paths, drift errors gradually accumulates, making the data obtained from gyroscope redundant.

II. Related Work

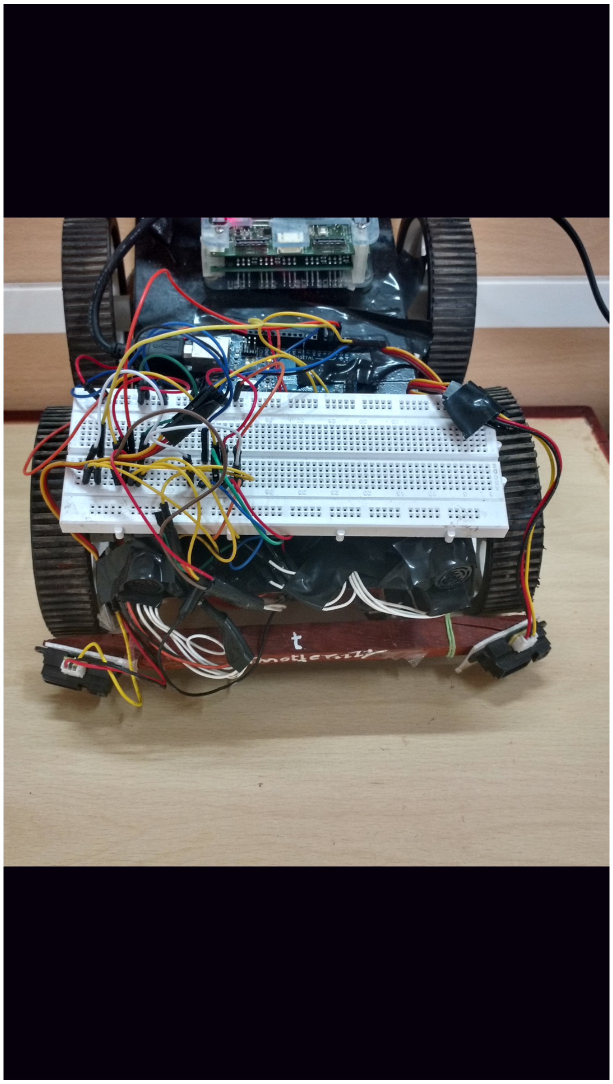
Many localization techniques have been developed and implemented in robots over a period of time. Common indoor localization methods for robots rely on external infrastructures and sensors which are part of the robot. Localization techniques that use RF technologies such as RFID, WiFi and bluetooth[3, 12, 15] 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[4, 9, 14] 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 tags or receiver which is carried by the robot. RF based localization techniques can either be finger printing or non-finger printing. The non-finger printing approaches include trilateration method and angulation techniques(AOA) [13]. The trilateration method estimates position of the target with respect to the reference points using the Received Signal Strength Indication (RSSI) values[5, 8].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[12]. Deterministic approaches such as K Nearest Neighbors(K-NN) [11], decision tree methods [7] 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[3, 15]. In both RFID and WiFi finger printing approaches, the collection of data set (offline phase) is tedious and time consuming.

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[4]. 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[14]. The main drawback here is that the laser range finders are expensive.

Inexpensive sensors such as accelerometers, gyroscopes and encoders have been used in localization. These IMU sensors are commonly used in dead reckoning method. In dead reckoning approach new state estimates are calculated with the help of old state estimates[1]. The accelerometer and encoders have been used to find the distance while gyroscope and magnetometer sensors have been used to determine the orientation[9]. 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. The encoders suffers from either systematic or non-systematic. Systematic errors are caused by the kinematic imperfections of the mobile robot. The most notorious systematic errors are unequal wheel diameters, uncertainty about effective wheelbase etc. Non-systematic errors are caused by the wheel slippage or irregularities in the floor. The systematic errors are removed to a certain extent with the UMBmark test[2]. Using only one of these sensors is insufficient to provide desired results, since each of them have high error rates. To overcome these errors, sensor fusion based approaches have been proposed. The sensor fusion in this context is the translation of different sensory inputs into reliable estimates[6]. The fusion of sensors can be from either complimentary or redundant sensors. [10] presents the survey of existing sensor fusion based approaches. The complexity of the environment and the errors from the sensors make the robot localization less feasible.

Figure 1. Robot



III. System Overview

This section describes about 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[] 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. This information from the map is not used as an input in determination of pose. This map is converted into an obstacle grid where each grid cell is marked by either 1 or 0, where 1 indicates that the cell contains an obstacle and 0 indicates that it is free space.

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

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

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

The robot of width W is equipped with an array of sensors, which assist in estimating its current pose p. 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 a rough estimate of angle given as θ_e. The tri-axial gyroscope present in bStem provides the angular rates with respect to three axes. Using the z-axis angular rate we calculate the heading direction given as θ_g. Two ultrasonic and two infrared sensors are embedded on either sides of the front of the robot. Both the sensors determine the distance between obstacle and robot from which further accurate orientation angle is calculated during obstacle avoidance. We use the distance estimate from encoders and the orientation estimate from gyroscope and encoders coupled with estimate from obstacle avoidance sensors to effectively determine pose estimate.

The mathematical determination of position and orientation estimate from encoders, gyroscope ,and obstacle avoidance sensors are explained below.

A. Position and Orientation estimate using Encoders

The encoders provide ticks from left and right motor which is represented as l and r respectively. The width of robot is defined as W. Using this, the position and orientation is calculated [], the details of which are explained below.

The robot pose has to be determined separately for two cases:

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

1) Case 1: Determination of pose when the bot is taking turns: Figure 1 represents the situation where the robot is taking a left turn. Let $p^{'}$ be the previous position of robot and p be the current position of the robot, we need to calculate the orientation and position. The angle α represents the degree of turn taken by robot to traverse from position $p^{'}$ to p. α is determined with the help of figure 2.

In figure 2 , when the arc is extended, it results in a circle with center at C and radius R units. Practically, l is ticks given by left encoder to traverse from A to $A^{'}$ and r is given by right encoder to traverse from B to $B^{'}$.

Mathematically, l is determined by considering $A^{'}AC$,

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

Similarly, r is determined by considering $B^{'}BC$,

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

Figure 2. When the bot takes left turn

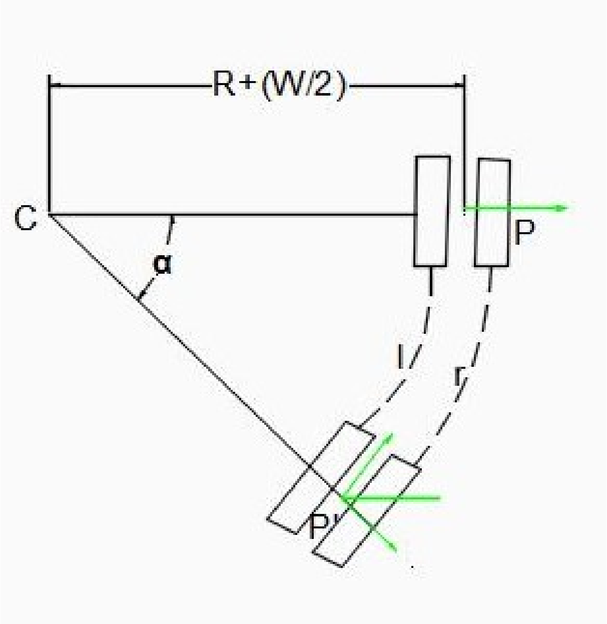
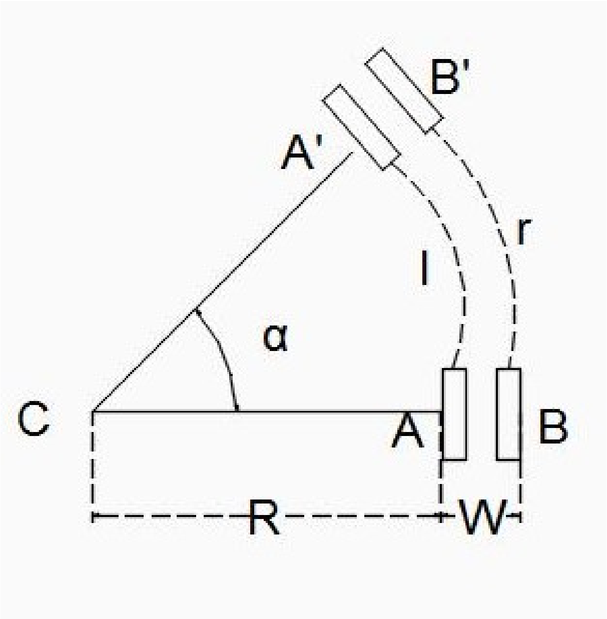


Figure 3. Determination of α



Using the mathematically obtained l and r, α is given as,

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

The radius of bot is proportional to l and inversely proportional to α,

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

Now in figure 1, at point p, the direction vector representation of θ_t is determined. The direction changes when the robot turns right or moves straight.

The new orientation estimate is summation of θ at time $t-1$ and calculated α value and is given as,

$$\theta_t = \theta_{t-1} + \alpha \tag{5}$$

In order to obtain position estimates x_t, y_t, center C(C_x, C_y) must be determined. The center is calculated with respect to point $p^{'}$ and its direction vector at point $p^{'}$.

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

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

Now x_t and y_t the new position estimate is determined with the center calculated previously and is given as,

$$x_t = C_x + (R + (W/2))\sin(\theta_t) \tag{8}$$

$$y_t = C_y + (R + (W/2))(-\cos(\theta_t)) \tag{9}$$

The above calculated orientation and position estimates are used to determine pose at time t,

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

. This determination of pose is when the bot takes left turn. Similarly, this can be extended to calculating the new pose when robot turns right.

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. So, the new orientation estimate is same as the orientation at time $t - 1$,

$$\theta_t = \theta_{t-1} \tag{10}$$

Now, the corresponding x_{t+1}, y_{t+1} is calculated with previous position estimates,l, r and is given as,

$$x_t = x_{t-1} + l(\cos(\theta_t)) \tag{11}$$

$$y_t = y_{t-1} + l(\sin(\theta_t)) \tag{12}$$

Here, both left tick and right tick values are equal that is $l == r$

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

Sensors always require to be calibrated before it can be put to proper use. So we calculate the offset and noise of our gyroscope to get reliable angle estimates.

The offset is calculated as an average of angular velocities when the robot is at rest(initial) and is given by,

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

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

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

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

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

Using the calculated noise and offset values, we determine the orientation of robot θ_g. The ω values which fall in the range of $[-noise, noise]$ are ignored as noise and the remaining values are taken for the calculation of orientation angle. The orientation is the integration of angular rates over a interval of time. The angle is given by,

$$\theta_g = \theta_g^{'} + \omega_i dt \tag{15}$$

Where $\theta_g^{'}$ represents the previous angle estimate and dt, the time period of the rate gyro which is calculated experimentally to determine the time period over which it can be periodically integrated to determine the angle from angular velocity, in our case this value is equal to 20ms. Using the above θ_g value we estimate the orientation and position.

The position estimate is calculated with previous position values,l, r, newly calculated orientation value and is given as,

$$dist = (l + r)/2$$

$$x_t = x_{t-1} + dist(\cos(\theta_g))$$

$$y_t = y_{t-1} + dist(\sin(\theta_g))$$

C. Orientation estimate using obstacle avoidance sensors

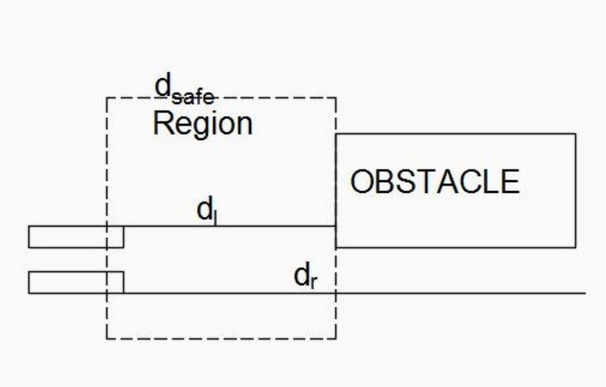
Infrared and ultrasonic sensors are used for measuring distances of obstacles with respect to the robot. In unknown environments/semi-known environments localization becomes tough. An autonomous robot tends to take navigational decisions based on obstacles detected. We exploit this to estimate orientation when obstacle is encountered.

In this system, 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. Sharp GP2Y0A41SK0F IR Distance Sensor which has a range of 4cm to 30cm is used to determine obstacles in close range as they have faster response time than ultrasonic sensors. A obstacle avoidance method[], from which the orientation of the robot is calculated is explained below.

When an obstacle is detected, the two ultrasonic sensors provide us with the distance between the obstacle and the 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}, the Flanks Safety Distance, which is the minimum distance from which robot can start maneuvering smoothly to avoid the obstacle. This value is calculated experimentally based on the range of the ultrasonic sensors and width of the robot.

To determine the angle, the following cases must be considered.

Figure 4. Right turn after detecting obstacle



1) case 1: $d_{lu} > d_{safe}$ and $d_{ru} > d_{safe}$: This situation is the case when there is no obstacle in front of the robot and the robot can continue to move straight, in this case the orientation angle doesn't change. So the angle is given by,

$$\theta_{oa} = previous(\theta_{oa}) \tag{16}$$

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 angle θ_{oa} is inversely proportional to distance obtained from the sensor that is smaller the distance higher the angle and vice versa.

$$\theta_{oa} = previous(\theta_{oa}) - \frac{\pi}{2}(\frac{k_{\theta u}}{d_{lu}}) \tag{17}$$

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

3) case 3: $d_{lu} > d_{safe}$ and $d_{ru} < d_{safe}$: The situation is similar to the one shown in figure 3, but its the other way round. In order to avoid the obstacle, the robot has to turn left and the angle is given as,

$$\theta_{oa} = previous(\theta_{oa}) + \frac{\pi}{2}(\frac{k_{\theta u}}{d_{ru}}) \tag{18}$$

where $k_{\theta u}$ represents the constant for collision angle for ultrasonic sensor and d_{ru} is the distance from obstacle and the right ultrasonic sensor.

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 infrared sensors to obtain distance from obstacles which is denoted by d_{li}, d_{ri}, where d_{li} is the distance from obstacle and left infrared sensor and similarly, d_{ri} is the distance from obstacle and right infrared sensor.

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

$$\theta_{oa} = previous(\theta_{oa}) - n \times (\frac{\pi}{2})(\frac{k_{\theta i}}{d_{ri}}) \tag{19}$$

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

$$\theta_{oa} = previous(\theta_{oa}) + n \times (\frac{\pi}{2})(\frac{k_{\theta i}}{d_{li}}) \tag{20}$$

where n is experimentally calculated value to increase/decrease the speed to avoid close obstacles, $k_{\theta i}$ is the constant of collision angle for infrared sensor 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.

The goal of our system is to accurately determine the pose at time t,

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

given the pose estimate at time $t - 1$,

$$p^{'} = [x_{t-1}, y_{t-1}, \theta_{t-1}]$$

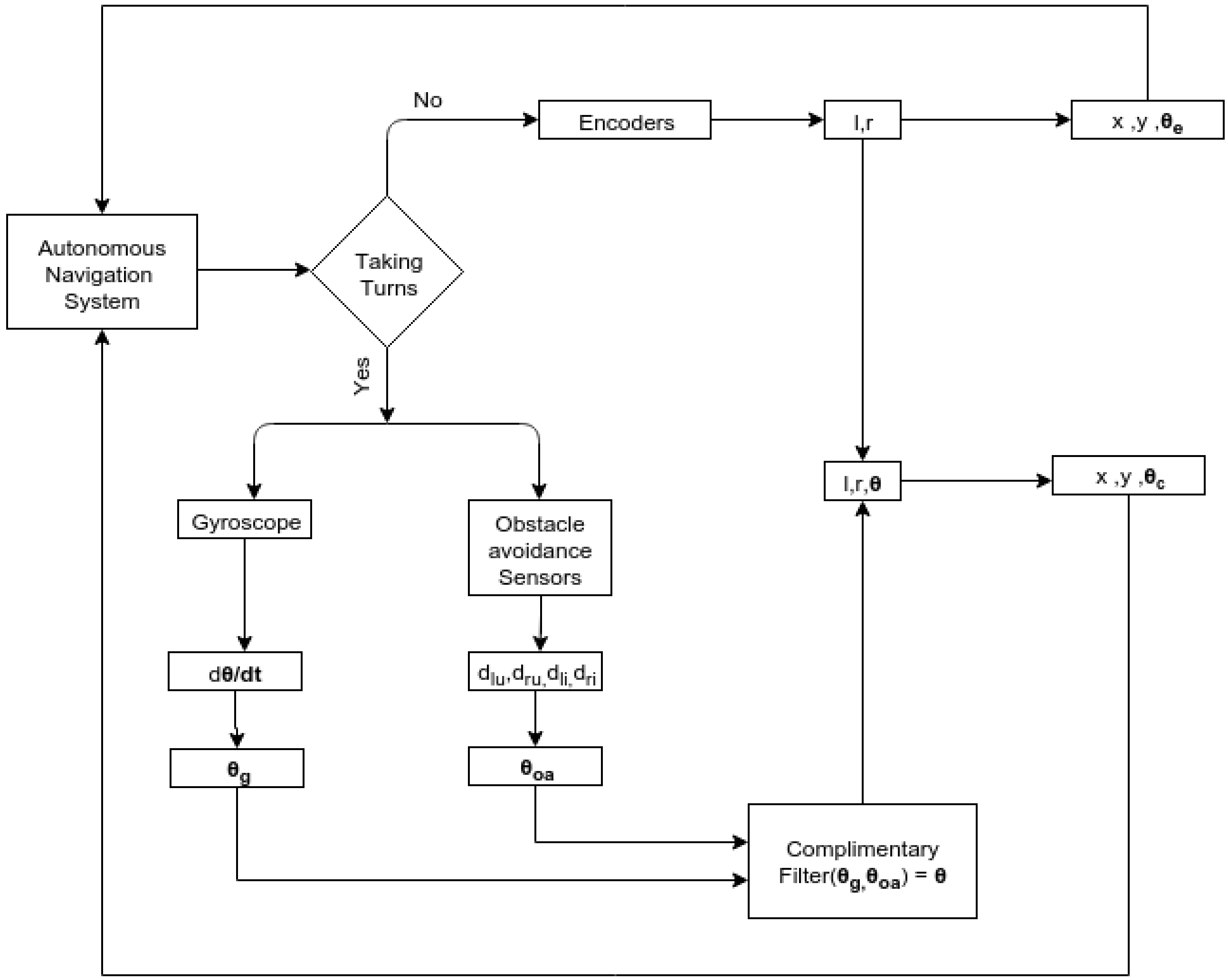
Initial pose is given to the system. Our objective is to minimize the difference between estimated pose p and ideal pose p_{ideal} during the above dead reckoning process.

IV. Proposed Method

In the previous section, we discussed how the individual sensors estimate the position of the bot. This section describes our novel approach to combine these sensors in a way that takes the advantage of one sensor to compliment over the disadvantage of other sensor. This hybrid method of sensor fusion is explained below.

The goal of this approach is to estimate pose accurately. The pose is calculated using two components namely distance and orientation(θ). Inaccuracies in orientation measure affects the position estimate when compared to inaccuracies in distance measured based on UMBmark test[2]. 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. Therefore, a accurate orientation measure is necessary for accurate pose estimation when compared with the distance measure. We consider that distance(l, r) estimate provided by encoders to be reliable. In order to get an accurate orientation, we consider the orientation measures obtained from encoders, gyroscope and obstacle avoidance sensors and combine these sensors by considering their individual properties.

The trail in which the robot navigates plays a major role in determining which orientation estimate to be used. This can be explained in two scenarios: First,when the bot is moving straight and Second, when the bot is taking turns.


No
Encoders
l,r
x ,y ,θe
Autonomous Navigation System
Taking Turns
Yes
Gyroscope
Obstacle avoidance Sensors
dθ/dt
dlu,dru,dli,dri
θg
θoa
Complimentary Filter(θg,θoa) = θ
l,r,θ
x ,y ,θc

A. *Scenario 1: When the robot moves straight*

Gyroscope sensors accumulate drift error over time. In an ideal indoor environment, linear paths have more frequency of occurrence than turning paths. So, use of gyroscope to determine orientation when the robot moves in a linear path will accumulate drift errors and the result obtained will not be as desired. While in encoders, the impact of non-systematic errors in determination of orientation estimate is comparatively less in linear paths. Therefore, we use orientation measure from encoders when the robot is taking linear path and is given by equation.

$$\theta = previous(\theta) + \alpha \tag{21}$$

α is determined by equation 3.

B. *Scenario 2: When the robot is taking turns*

As mentioned earlier, the frequency of occurrence of turns are less. Hence, the accumulation of drift errors in gyroscope would be less. But, while taking turns slip occurs. So, encoders cannot be used whereas the gyroscope is resilient to this. Turns are detected using obstacle avoidance sensors and orientation measure of gyroscope is calculated using equation 15. Further, we use the orientation measure from obstacle avoidance sensors. These two orientation measures are combined using the complimentary filter to increase accuracy.

1) Complimentary Filter: The primary purpose of complimentary filter is to combine the orientation measure from both gyroscope and obstacle avoidance sensors to obtain an accurate angle. The complimentary filter is given by the equation,

$$\theta = \alpha\theta_g + (1 - \alpha)\theta_{oa} \tag{22}$$

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

For the above mentioned scenarios, we get the orientation measure respectively. This orientation and distance measure(l, r) are combined to get the pose estimate and are given by equations,

$$dist = (l + r)/2$$

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

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

References

[1] Hyo-Sung Ahn and Wonpil Yu. "Indoor mobile robot and pedestrian localization techniques". In: *Control, Automation and Systems, 2007. ICCAS'07. International Conference on*. IEEE. 2007, pp. 2350–2354.

[2] Johann Borenstein and Liqiang Feng. "UMBmark: a method for measuring, comparing, and correcting dead-reckoning errors in mobile robots". In: (1994).

[3] Byoung-Suk Choi et al. "A hierarchical algorithm for indoor mobile robot localization using RFID sensor fusion". In: *Industrial Electronics, IEEE Transactions on* 58.6 (2011), pp. 2226–2235.

[4] Guilherme N DeSouza and Avinash C Kak. "Vision for mobile robot navigation: A survey". In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 24.2 (2002), pp. 237–267.

[5] Adam Easton and Stephen Cameron. "A gaussian error model for triangulation-based pose estimation using noisy landmarks". In: *Robotics, Automation and Mechatronics, 2006 IEEE Conference on*. IEEE. 2006, pp. 1–6.

[6] Wilfried Elmenreich. "Sensor fusion in time-triggered systems". In: (2002).

[7] Gorkem Erinc. "Appearance-Based Navigation, Localization, Mapping, and Map Merging for Heterogeneous Teams of Robots". In: (2013).

[8] Moises Granados-Cruz et al. "Triangulation-based indoor robot localization using extended FIR/Kalman filtering". In: *Electrical Engineering, Computing Science and Automatic Control (CCE), 2014 11th International Conference on*. IEEE. 2014, pp. 1–5.

[9] Matej Guran et al. "Localization of iRobot create using inertial measuring unit". In: *Robotics in Alpe-Adria-Danube Region (RAAD), 2014 23rd International Conference on*. IEEE. 2014, pp. 1–7.

[10] Moshe Kam, Xiaoxun Zhu, and Paul Kalata. "Sensor fusion for mobile robot navigation". In: *Proceedings of the IEEE* 85.1 (1997), pp. 108–119.

[11] Keith J Kelley. "Wi-Fi Location Determination for Semantic Locations". In: *The Hilltop Review* 7.1 (2015), p. 9.

[12] Jiwei Li. "Characterization of WLAN Location Fingerprinting Systems". In: ().

[13] Hui Liu et al. "Survey of wireless indoor positioning techniques and systems". In: *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on* 37.6 (2007), pp. 1067–1080.

[14] Hartmut Surmann, Andreas Nüchter, and Joachim Hertzberg. "An autonomous mobile robot with a 3D laser range finder for 3D exploration and digitalization of indoor environments". In: *Robotics and Autonomous Systems* 45.3 (2003), pp. 181–198.

[15] Hui Zhang, Joseph C Chen, and Kai Zhang. "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*. IEEE. 2014, pp. 1–6.