Laser Based EKF Localization on TurtleBot

James Wells
Cooperative Distributed Systems Lab
Department of Mechanical Engineering
University of Cincinnati
Cincinnati OH, United States of America
wells2jz@mail.uc.edu

Oyindamola Omotuyi
Cooperative Distributed Systems Lab
Department of Mechanical Engineering
University of Cincinnati
Cincinnati OH, United States of America
omotuyoa@mail.uc.edu

***Abstract*—This paper presents a technique on the localization of a ground robot using the Extended Kalman Filter (EKF) with the use of a Light Detection and Ranging (Lidar) sensor. The use of EKF in localizing robots has been highly successful when working with small maps for many years. The goal of this project was to design and implement an algorithm which would replicate what has already been done using Python, ROS and Gazebo. This software environment was chosen because of its widespread use in industry and its ability to enable a quick transition between hardware and simulation. The simulation was based on the TurtleBot3 Burger platform. This platform was equipped with a 360 Laser Distance Sensor LDS-01. This platform was chosen as it is inexpensive, off the shelf and readily available.**

Keywords— Extended Kalman Filter, Gazebo, Laser, Localization, ROS, TurtleBot

I. Introduction

A. Background

Localization has become an important topic in the field of robotics. It involves estimating the state or pose of the robot given a known map of the environment. Robots have to depend on noisy sensors to determine where they are located and their attitude in the world. This is a fundamental problem because a few meters' error in the location of the robot in the world can be very costly [2]. For instance, self-driving cars in the real world need to have the best estimate of their location at real-time in order to avoid collisions with other cars. Unfortunately, the position of the robot is not directly sensed and most of the sensors used are not noise-free [1]. Therefore, robot position must be inferred from data. Herein lies the problem of mobile robot localization and the need for some filtering algorithm of the sensor data in order get the best estimate of the state of the robot in the environment.

B. Problem Statement

As can be seen through the information provided above in the background, robot localization is a crucial area in the field of mobile robotics and self-driving cars. Being able to accurately estimate position in an environment is key for mobile robots to be able to do any meaningful work within a known environment. Large amounts of work have already been completed on localization, so as an introduction to this area of study, so this paper will focus on the verification and implementation of the laser based Extended Kalman Filter (EKF), an algorithm which has already been studied at great lengths. This implementation will focus on using the algorithm and hardware on a simulated TurtleBot in Gazebo. For this algorithm, several sensors could have been chosen for sensor input including acoustic sensors, stereo vision sensors or RGB-D sensors, however Lidar was the most advantageous. Three major reasons behind the selection of a Lidar for the localization algorithm were: the versatility, the accuracy and the availability. According to [4], laser-based systems are able to obtain robust results in both indoor and outdoor environments, making this algorithm versatile when it is time to implement on actual hardware. Also, according to [4], laser range finders offer high speed and accuracy. This will be beneficial in providing a high rate of accurate data to the algorithm. The final reason for selecting a Lidar was the availability. Based on findings in [5], a Lidar sensor was readily available in Gazebo.

II. Related Works

A. Extended Kalman Filter SLAM

Similar work has already been done further improving upon what is being presented in this paper. An example of this is seen in [2]. The work done there was to implement a full Simultaneous Localization and Mapping (SLAM) problem, while the focus of this paper is simply to focus on the localization aspect. With that being said, many aspects overlap. The localization algorithm presented in this paper and the SLAM algorithm in [2] both have the Extended Kalman Filter at its core as well. In both cases the EKF algorithm is used as the state estimator. The major difference between what was done for this paper and the paper referenced was in this paper, the map was provided to the robot while in [2] the robot had to both create and localize itself within the map.

B. Localization

According to Thrun et al in [1], localization problems can be classified based on the difficulty. This difficulty is created by the nature of robot's environments and knowledge available to the robot initially. The following paragraphs explain the metrics which Thrun classifies the difficulty of the localization problem.

1. Classification based on the nature of the environment

There are two major types of environments a robot can be located in – static and dynamic environments. In static environments, it is only the pose of the robot that is changing.

However, in dynamic environments, both the locations of the features in the environment and the pose of the robot changes. This makes localization in dynamic environment a more challenging task

2. *Classification based on the initial knowledge of the robot*

The difficulty level of localization problem also depends on the type of information available to the robot at the initial stage and at run-time. The following are three class of problems according to their level of difficulty according to [1].

1. Position Tracking: In this problem, the initial pose of the robot is known. The pose error during motion is also assumed to be minimal. This is also called a local problem [3].

2. Global localization: In this problem, the initial pose of the robot is unknown. The robot is assumed to be placed anywhere in the environment without the knowledge of where it is. The robot's current pose has to be determined by sensors [3].

3. Kidnapped Robot Problem: While in global localization problem, the robot knows that it does not know where it is, in this problem, the robot does not know that it does not know where it is, therefore having a false information of where it is. This concept is important in testing different localization algorithms to see if they have the ability to recover from global localization failures. Autonomous robots require the ability to recover failures. [1]

C. Robot Operating System and Gazebo

Robot Operating System (ROS) is an open source platform for developing robot applications. There are four major concepts explored while using ROS – Nodes, Topic, Messages and Services. Nodes use a publish-subscribe method to send and receive messages. Messages is a ROS data type that is used by nodes when subscribing or publishing to a topic. Therefore, a node can either subscribe to a topic to receive messages or publish so that other nodes can subscribe to its messages.[1] Gazebo is a 3D, physics based, simulator for visualizing and simulating robot's movement in a simple or complex environment.

III. METHODOLOGY

When creating the laser-based localization algorithm, several key parts had to come together. The two major aspects of the laser localization algorithm consisted of the Extended Kalman Filter and the geometry the problem is rooted in. The geometry is how the robot is able to relate the information about the world coming from its sensors to its location in the map. The Extended Kalman filter is what filters the incoming stream of data and then fuse it together to increase its reliability.

A. Application of the Extended Kalman Filter Localization Algorithm

The standard Kalman Filter is only applicable to linear systems. The ground robot simulated in this project is a nonlinear system hence the need for Extended Kalman Filter (EKF). EKF is a form of Kalman Filter which is used estimating the state and covariance of nonlinear systems by linearizing the estimate of the current mean and covariance of the system. This algorithm involves two steps: prediction and measurement update. In order to localize the robot, we needed to determine the current estimate of the state of the system based on the state estimated using the kinematics of the system and state estimated by measurements of the environments. The state of the robot is defined as the position of the robot along x-axis and y-axis and the angular rotation, theta about the z-axis as seen in equation (1).

$$State_{robot} = \begin{bmatrix} x_k \\ y_k \\ \theta_k \end{bmatrix} \quad (1)$$

The EKF algorithm implemented has the robot's location defined using the map of the environment which is based on X and Y coordinates of the objects within the environment. In this case, equally spaced cylinders were placed at four locations as shown in Figure (1).

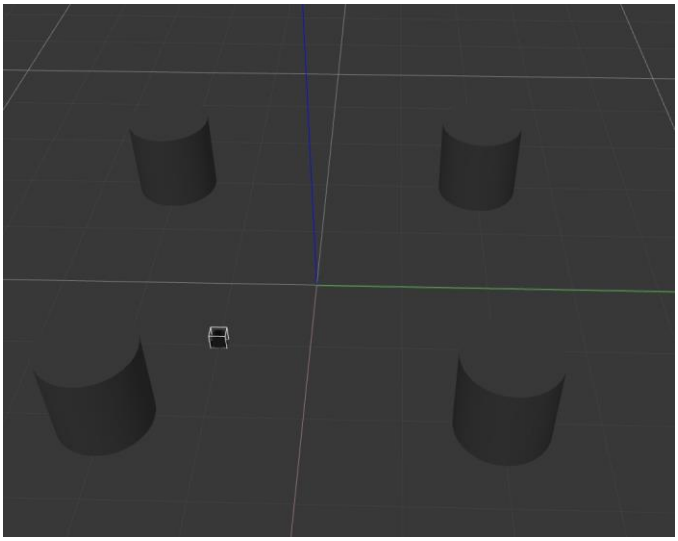

Figure 1: TurtleBot initialized in custom made world with cylinders of 1 meter radius position at (2,2), (-2,2), (2,-2), (-2,-2).

In ROS, a node was created, and it subscribed to three topics –nav_msgs/Odometry, geometry_msgs/cmd_vel and /scan. The **f**ollowing shows how the algorithm was divided into the prediction and update steps and explains the process in each step.

I. Prediction

The odometry data was used in estimating the current state of the TurtleBot $\hat{x}_k^-$ defined by the equations (2). The covariance matrix Q was defined by a diagonal matrix in which each element of the diagonal is from the covariance data in the

[1]http://wiki.ros.org/ROS/Tutorials/UnderstandingNodes

odometry topic. Also, the rotation data was also converted from quaternion to Euler by importing the euler_from_quaternion module from tf.transformations to obtain theta.

At the prediction stage, the relative distance between the object and the robot along the ΔX_{odom} and ΔY_{odom} direction was calculated at each time step. This is seen in equation (6) and equation (7). This is used for classification of the point clouds obtained by the laser scan to detect the object that the robot is going to localize itself based off. The equations necessary for the prediction step of the algorithm are shown below in (2) through (7).

Prediction:

$$\hat{x}_k^- = A\hat{x}_{k-1} + Bu_k \quad (2)$$

$$P_k^- = AP_{k-1}A^T + Q \quad (3)$$

$$A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (4)$$

$$B = \begin{bmatrix} \Delta T * \cos(\theta_{k-1}) & 0 \\ \Delta T * \sin(\theta_{k-1}) & 0 \\ 0 & \Delta T \end{bmatrix} \quad (5)$$

$$\Delta X_{odom} = X_{obj\_Loc} - X_{odom\_robot} \quad (6)$$

$$\Delta Y_{odom} = Y_{obj\_Loc} - Y_{odom\_robot} \quad (7)$$

II. Measurement Update

After the prediction step, the algorithm proceeds into a measurement update step. The measurement update step confirms the prediction and corrects any errors based on the data the robot has now collected. The measurement data was derived from the range data from the laser on the TurtleBot. The distance, written as D, is from the edge of the object to the center of the robot. The radius of the cylinder was added to the minimum distance measurement to find the estimate the distance to the cylinder center. This was estimated using the geometry seen in Figure (2) and the relative distance between the object and the robot along the x and y direction was calculated as ΔX_{laser} *and* ΔY_{laser} .

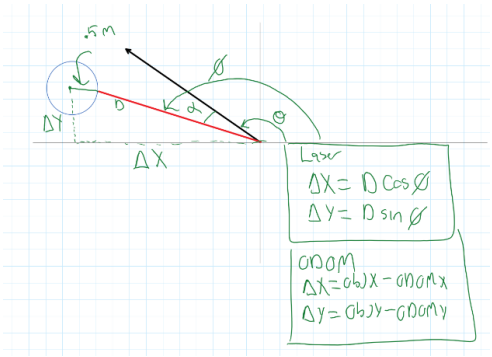

Figure 2: Geometry Diagram showing the geometry behind the localization

Using a minimum distance classifier with inputs of delta x, delta y, odometry and laser data, the objects location was predicted. The estimated robot location was calculated using the equations (10) and (11). This is passed into the algorithm as the y_k variable as seen in equation (13). The covariance matrix R was defined by a diagonal matrix in which each element of the diagonal was set with a mean of 0 and standard deviation of 0.01.

Update

Measurement: /scan

$$\Delta X_{laser} = D * \cos(\emptyset) \quad (8)$$

$$\Delta Y_{laser} = D * \sin(\emptyset) \quad (9)$$

$$X_{robot_{laser}} = X_{predicted_{object_{location}}} - \Delta X_{laser} \quad (10)$$

$$Y_{robot_{laser}} = X_{predicted_{object_{location}}} - \Delta X_{laser} \quad (11)$$

$$K_k = \frac{P_k^- C^T}{C P_k^- C^T + R} \quad (12)$$

$$\hat{x}_k = \hat{x}_k^- + K_k(y_k - C\,\hat{x}_k^-) \quad (13)$$

$$P_k = (I - K_k C)P_k^- \quad (14)$$

IV. Results

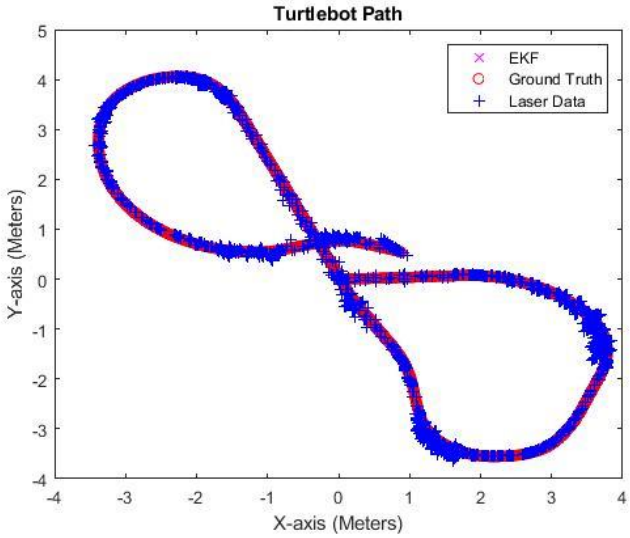

Figure 3: Trajectory of robot localization based on EKF Localization, Laser Data and Ground Truth.

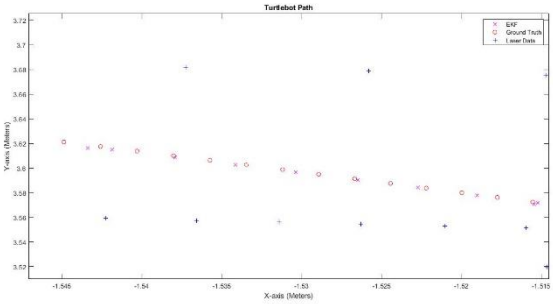

Figure 4: Zoomed in View of Data Comparing EKF Results with Pure Laser Data and the Ground Truth of the Robot

Figures 3 and 4 above show the results from a teleoperation test of the TurtleBot. In these plots, the location of the TurtleBot has been recorded based on just the laser (Blue pluses), the EKF Localization algorithm (Magenta circles) and the ground truth data (red x's). From the plots, it should be noted that the EKF based localization has estimated the position much more accurately when compared to the laser alone.

Figures 5 and 6 below show the X and Y position and estimate with respect to time. Figure 5 is over the entire time of the test while figure 6 is zoomed in on a time period to show how close the data actually matches. Again it can be seen that the data aquired strictly from the laser is not nearly as acurate as the data which has been filtered and fused with the EKF. The EKF data matches the ground truth data with a higher degree of acuracy.

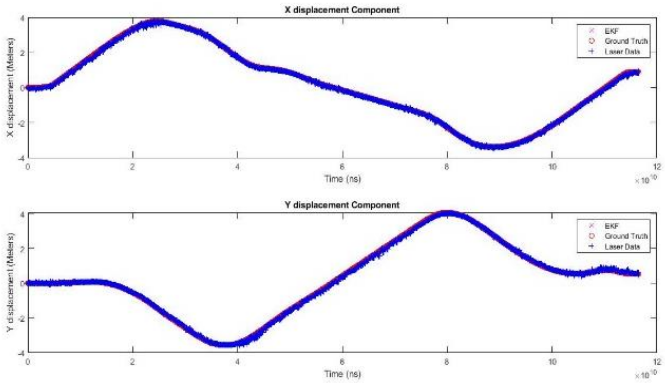

Figure 5: EKF, Laser and Ground Truth X and Y Position of the Robot vs. Time

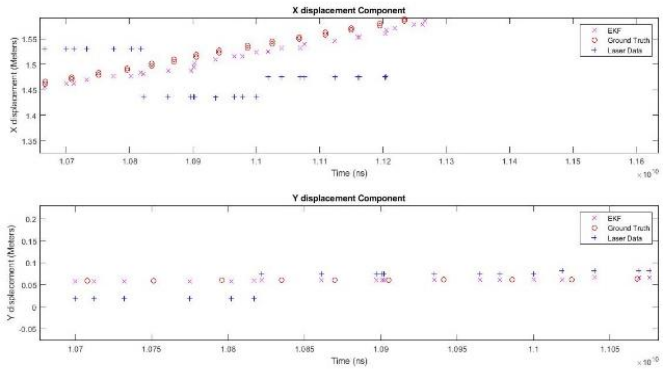

Figure 6: Zoomed in view of the data collected in both X and Y components of the filter. This shows how the data varies over time.

V. Discussion and Analysis

The implementation of a Laser Based EKF localization algorithm was a success on a limited scale. Right now, the algorithm created isn't of practical use in the real world. As a proof of concept, it works extremely well. The plots shown in figure 4 and figure 6 clearly displays that the EKF slam algorithm enables a near perfect match of the ground truth while relying solely on the data from the laser produces relatively large inaccuracies. The EKF algorithm takes in both an estimated position based on the odometer data and updates it based off the position it receives from the laser. In this instance, the odometry data has limited noise and matches the ground truth much more reliably. When the laser is introduced, more errors occur. With this being said, the odometry data is very clean and not representative of the actual world. In the actual world, the odometer data wouldn't be as good due to factors such as wheel slippage, encoders miss counting and other external forces which will cause the accuracy to drop off. When this happens, the laser will then become the more reliable measurement. Finally, according to [5], the data coming from the odometry can become very inaccurate as the TurtleBot moves long distances. The tests run were over limited distances, so error did not have much time to accumulate to create noticeable results. In the future, errors could be introduced artificially to the odometer data in order to more effectively evaluate the performance of the laser.

Another aspect seen in figure 4 and 6 and briefly stated in the previous paragraph is that the laser data is far off. Two major reasons for this error are the low resolution of the laser scan and the improper classification of the center distance. The simulated laser used only took measurements in one-degree increments. This limited resolution could cause the minimum distance between the cylinder and the robot to be missed. This short coming could be overcome using a laser with a higher resolution or changing the clustering algorithm. Relying only on the center point also induces error. Improving the clustering algorithm to look for both the minimum distance and several points around to estimate the arc which is seen in the laser could potentially improve its accuracy.

VI. Conclusion and Future Work

Now that the algorithm has been implemented on a simple case consisting of just cylinders with a constant radius, the algorithm needs to be generalized and improved. This entails developing a better clustering and tracking algorithm. This will enable the environment to be more realistic to what it would see in the real world. Before this algorithm can be used in a real-world application, the fundamental algorithm would need to be changed. Research on EKF based localization has greatly slowed down over the years and the focus has been shifted to particle filter-based localization algorithms. Particle filter-based algorithms are more robust and can handle larger environments better then EKF based algorithms.

Another improvement which needs to be made is estimated the yaw angle. Currently, the algorithm only estimates the X and Y location of the robot and trusting the theta angle which is provided by the IMU. Estimating orientation based on laser information would require the use of several objects, each of which being known with high certainty corresponding to an object in the map, the distances between these objects and the angles which they were detected. With all of this in mind, it was not feasible given the equipment used. Before this would be deployed in an environment in which large disturbances to the orientation of the robot would occur, adding this would be a necessity because the current algorithm is dependent on this angle being known from odometry data.

After this algorithm is completed, the real goal is to create a SLAM (Simultaneous Localization and Mapping) algorithm. These algorithms are much more robust and useful in practice because they enable the robot to localize itself in an environment it hasn't previously seen and create a map of this environment. This type of algorithm has a wide range of applications such as disaster scenarios, military applications and finally exploration of the solar system. The SLAM algorithm which would become the focus would first an EKF based SLAM algorithm.

According to Thrun, in [6], the EKF SLAM is earliest and perhaps the most influential. This type of SLAM would be an excellent starting point in the future because of its strong presence and it follows the work presented in this paper and what was learned in Decision Engineering class.

REFERENCES

[1] Thrun, S., Burgard, W., & Fox, D. (2005). Probabilistic robotics. Cambridge, Mass: MIT Press.

[2] Hasan, Hamid & H Mohammed, T. (2017). Implementation of Mobile Robot's Navigation using SLAM based on Cloud Computing. Engineering and Technology Journal. 35. 634-639.

[3] Kasparaviciute, G. (2016). Systematic Evaluation of Selected Algorithms for Sensor Based Localization for a Mini Autonomous Vehicle.

[4] Chong, T. J., Tang, X. J., Leng, C. H., Yogeswaran, M., Ng, O. E., & Chong, Y. Z. (2015). Sensor technologies and simultaneous localization and mapping (SLAM). Procedia Computer Science, 76, 174-179.

[5] Fairchild, C., & Harman, T. L. (2016). ROS robotics by example: Bring life to your robot using ROS robotic applications. Birmingham, UK: Packt Publishing Limited.

[6] Thrun, S. (2007). Simultaneous localization and mapping. In Robotics and cognitive approaches to spatial mapping (pp. 13-41). Springer, Berlin, Heidelbe