

Simulation of EKF-Localization in a 2D Environment

I. PROBLEM AND ITS IMPORTANCE

IN order for an autonomous system to make informed decisions, it must have knowledge of its environment and its state within the environment. For example, if a robot is instructed to carry an item to a specific room in a building, the robot must have knowledge of its current location in order to plan an efficient route. Likewise, a robot cannot inform safety crews of the location of hazardous materials if the robot itself does not know its current location. However, in order for the robot to localize itself, it also needs previously constructed maps to correlate observed landmarks to landmarks described in the maps. It was determined that the problems of localization and mapping within the robotic domain were actually coupled and thus, the problem of Simultaneous Localization and Mapping (SLAM) was properly defined [1]. The development of SLAM also made it possible for a robotic system, without any prior knowledge of its environment, to explore the state-space and construct a map using landmarks and its estimated position. This result has obvious implications for the use of robots to explore dangerous areas such as enemy encampments, other planets in the solar system, and underwater environments. The purpose of this research is to simulate the first aspect of SLAM, autonomous localization in an environment with known correspondences.

II. RELATED WORK

The SLAM problem was first conceived at the 1986 IEEE Robotics and Automation Conference held in San Francisco, California [1]. A number of researchers were using state estimation processes to try to separately solve the localization and mapping problems until the robotic community realized that the two problems were coupled [2]. The Extended Kalman Filter (EKF) variant of SLAM was implemented and discussed in several papers [3], [4], [5]. Of paramount importance in the previously mentioned papers was discussion on the convergence of EKF-SLAM. SLAM with the use of sonar has recently been of interest to the autonomous underwater vehicle community due to the lack of GPS data underwater [6], [7]. One of the major issues with the mentioned papers and in the field of EKF-SLAM in general is that errors in estimation are introduced during EKF linearization.

III. APPROACH

In this research, the Extended Kalman Filter (EKF) was implemented in order to localize an autonomous robot within an environment. The algorithm made use of known correspondences in the environment, thus, saving the problem of extracting features from unknown landmarks for future research. The EKF algorithm was implemented and simulated within

the Robot Operating System (ROS) and Stage architectures. ROS is a publish-and-subscribe communication architecture that allows for software nodes to transfer data. A typical robotic platform might have a node to control actuators, a node to process video data, and a node for higher level planning functions. A great aspect of ROS is that the code developed can be directly applied to the target robotic platform without code modification. ROS also provides a node that wraps the Stage robotic simulation environment so that code can be vetted in simulation before being executed on the target platform. In this research, an EKF node was written that correlates color blob data to landmarks in the environment and then uses the EKF algorithm to estimate its location. The simulation was initialized with five landmarks, each of a different color and each landmark location was known to the robot. The robot also knew its own state at the start of the simulation. The robot's motion was modeled after a differential drive, such as a Pioneer or Seekur. Thus, its motion equations relied upon translational velocity, v_t , rotational velocity, ω_t , and pose, θ , with the equations of motion defined by the following equation.

$$\begin{bmatrix} x' \\ y' \\ \theta' \end{bmatrix} = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} + \begin{bmatrix} -\frac{v_t}{\omega_t} \sin(\theta) + \frac{v_t}{\omega_t} \sin(\theta + \omega_t \Delta t) \\ \frac{v_t}{\omega_t} \cos(\theta) - \frac{v_t}{\omega_t} \cos(\theta + \omega_t \Delta t) \\ \omega_t \Delta t \end{bmatrix}$$

The state variables, x' , y' , and θ' denote the predicted state after time step Δt . At each time step, $\Delta t = 0.01 \text{seconds}$, the EKF algorithm predicts its current state variables based on its internal motion model and estimated system noise. It then predicts the state of the covariance matrix using the following equation

$$\bar{\Sigma}_t = G_t \Sigma_{t-1} G_t^T + V_t M_t V_t^T$$

In the previous equation, G_t is a linearization of the motion model, V_t captures the system velocity, M_t represents system uncertainty, $\bar{\Sigma}_t$ is the predicted covariance matrix, and Σ_t is the previous iteration's covariance matrix. Error accumulates in the prediction of the state variables and the covariance matrix until a known landmark is detected, at which point the EKF algorithm can perform an update and resolve state inconsistencies. In this simulation, a color blob finder was utilized to detect landmarks based on their color. Each landmark was assigned a specific color and the robot associated landmark colors with landmark positions. Upon detection of a landmark, the EKF algorithm creates a predicted measurement based on the robot's predicted position and the map of landmarks. The predicted measurement is compared against the actual measurement to update the robot's estimated state. The difference between the predicted and actual measurements

is multiplied by the Kalman gain, K_t , to update the estimated state in the following equations.

$$z_t = \begin{bmatrix} \text{atan2}(m_y - \bar{\mu}_t, m_x - \bar{\mu}_t) - \bar{\mu}_{t,\theta} \\ m_s \end{bmatrix}$$

$$S_t = H_t \bar{\Sigma}_t H_t^T + Q_t$$

$$K_t = \bar{\Sigma}_t H_t^T S_t^{-1}$$

$$\bar{\mu}_t = \hat{\mu}_t + K_t(z_t - \hat{z}_t)$$

In the definition of z_t , the first element is the distance between the robot and the detected landmark, the second element is the angle to the landmark from the robot, and the third element is the landmark's signature, which is irrelevant in EKF. Furthermore, Q_t models measurement error and H_t is the Jacobian of the measurement model. More information about the calculation of K_t , can be found in Sebastian Thrun's textbook [8]. The process of predicting and propagating the state variables and covariance matrix is recursive and allows for the robot to localize itself.

The robot in the simulation used a simple motor schema to navigate through the environment while avoiding obstacles. The robot merely moved in a straight line until its laser sensors detected an obstacle within 3 meters, at which point, the robot turned away from the vector pointing from the robot to the obstacle. Also, the robot decreased its velocity proportionally to the distance to the obstacle. This simple motion allowed the robot to navigate the environment and detect landmarks as shown in Figure 1.

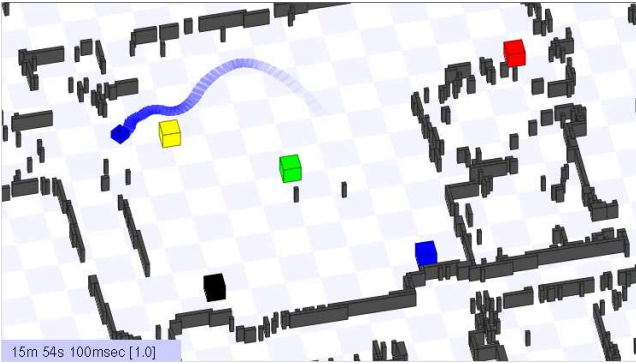


Figure 1. Differential robot navigating the Stage environment.

In Figure 1, the landmarks are denoted by the yellow, green, red, blue, and black boxes. The robot is denoted by the smaller blue box with a tail showing the recent locations of the robot.

IV. EVALUATION OF RESULTS

The performance of the EKF localization algorithm was compared against localization only using dead-reckoning. The metric that was used to compare the two localization techniques was the percent error between the estimated position

of the robot and the actual position of the robot. The percent error formula is given by the following formula.

$$\%Error = 100 * \frac{\text{prediction} - \text{actual}}{\text{actual}}$$

Both the dead-reckoning odometry and the EKF localization estimated coordinates were compared against the actual robot coordinates generated by the simulation in figures 2, 3, 4, and 5. In each figure, the x-axis is the simulation time and the y-axis is the percent error.

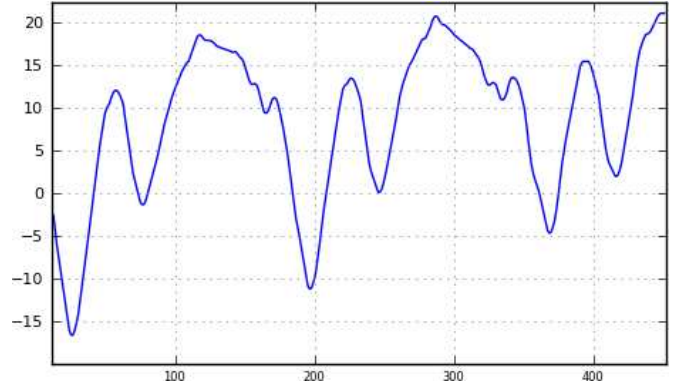


Figure 2. Dead-reckoning odometry error in x-direction.

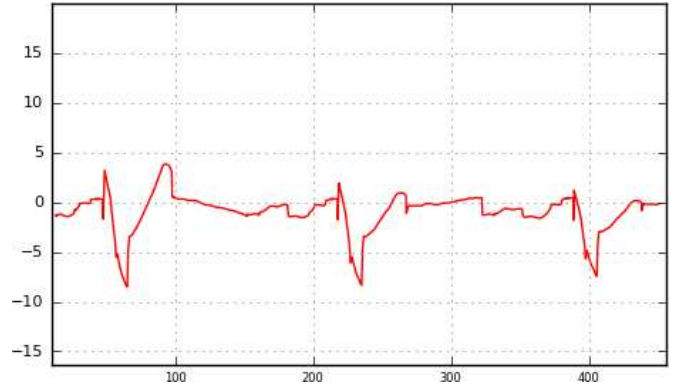


Figure 3. EKF error in x-direction.

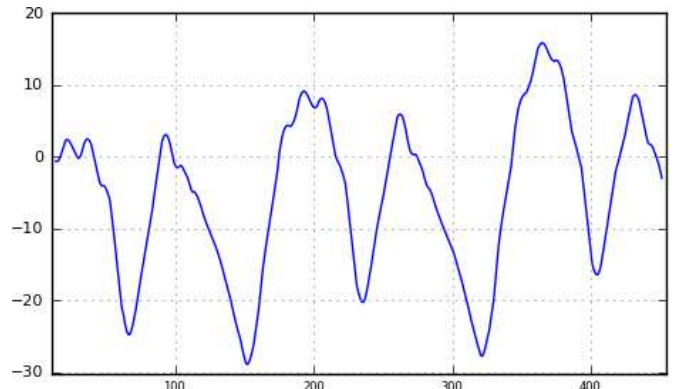


Figure 4. Dead-reckoning odometry error in y-direction.

When the percent errors displayed in the figures are compared it is clear that the EKF localization technique predicted

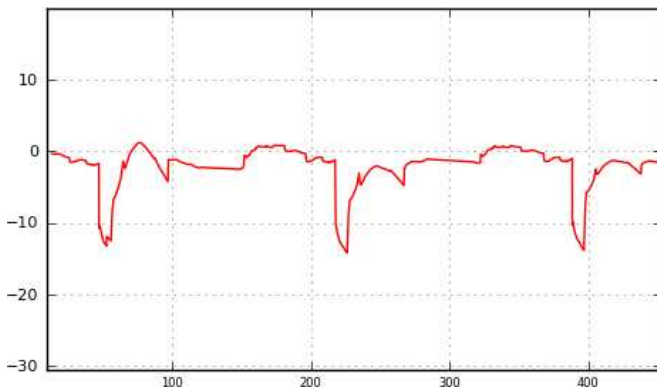


Figure 5. EKF error in y-direction.

state variables that deviated less from the actual state variables than the state variables predicted by dead-reckoning. When comparing the figures it is important to note that the dead-reckoning percent error increased to over -29% while the maximum percent error in EKF was only around -12%. Also, the EKF algorithm usually maintained a percent error near the 0% mark. However, the dead-reckoning technique predicted state variables that oscillated and drifted wildly. A final note about the plots is that the EKF plots show evidence of a periodic waveform. In the EKF plots, both the x and y predicted states begin to deviate from the actual robot position rapidly at the same time. The error in the x and y predicted values then decreased at the same time. This occurred in a periodic fashion. Even though the plots of the dead-reckoning states oscillated, the oscillations in the x and y error plots were not time-correlated.

V. DISCUSSION

Given that the dead-reckoning technique produced a larger state variable error compared to the EKF technique, it is clear that the EKF technique produced a better estimation of the actual robot coordinates. However, there are other interesting insights into the EKF technique that are embedded in the periodic nature of figures 3 and 5. The periodic increases in error in the two figures are due to the placement of the landmarks. As the robot uses its simple motor schema to navigate the environment, it eventually positions itself in a hallway that does not have any landmarks. Thus, the robot has to rely solely upon its internal motion model and covariance matrix to predict its state. However, due to noise in the system, this results in error accumulation because it cannot correct its position by observing a landmark. The EKF percent error figures show a periodic nature because the small environment and simple motor schemas result in the robot moving through similar trajectories in the environment. Thus, the robot moves into the hallway without any landmarks several times on a regular interval.

While working on this project, the researcher became extremely acquainted with the relationships between the EKF matrices and how they effected each other. Also, while the learning curve was steep, the ROS architecture proved to be a capable framework for developing and simulating the

EKF algorithm. However, it should be noted that the Stage simulator communication node provided by the standard ROS distribution was not sufficient for simulating a robot with a color blob finder. A ROS node that was developed by one of the ROS community members had to be installed in order to access all of the Stage simulator messages and commands. Also, the Eigen C++ library proved to be a great help in the implementation of the EKF algorithm. Eigen is a C++ library that facilitates the use of linear algebra in C++. Eigen provides a simple syntax for initializing matrices and it even provides functions for multiplying, transposing, and taking the inverses of matrices.

There are definitely improvements that could be made to the EKF algorithm as well as the overall architecture of the system. An inherent problem in EKF is the fact that linearization results in rounding errors in the internal motion model. Other Kalman filters should be explored such as the Unscented Kalman filter. Also, the motion planner for the robot could be improved in order to maximize landmark observations. Currently, the robot's motion planner is only concerned with moving in a straight line and avoiding obstacles when they are within a certain range. The robot's motion planner is not affected by the presence or absence of landmarks. However, the robot's state prediction could be improved if the robot actively sought out landmarks as it navigated the environment. When goals are eventually added to the robot's planner, the robot could allocate time to seeking out landmarks to improve state estimation while on its way to its objective. Finally, the use of multiple agents in the simulator is an interesting research problem. The multiple agents would communicate with each other and observe each other in order to improve individual localization. In multi-agent systems, it would be interesting to determine if the improved localization is worth the increased use of communication bandwidth.

REFERENCES

- [1] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: part i," *Robotics & Automation Magazine, IEEE*, vol. 13, no. 2, pp. 99–110, 2006.
- [2] S. Williams, P. Newman, G. Dissanayake, and H. Durrant-Whyte, "Autonomous underwater simultaneous localisation and map building," in *Robotics and Automation, 2000. Proceedings. ICRA '00. IEEE International Conference on*, vol. 2, 2000, pp. 1793–1798 vol.2.
- [3] S. Huang and G. Dissanayake, "Convergence and consistency analysis for extended kalman filter based SLAM," *Robotics, IEEE Transactions on*, vol. 23, no. 5, pp. 1036–1049, 2007.
- [4] J. A. Castellanos, R. Martinez-Cantin, J. D. Tardos, and J. Neira, "Robocentric map joining: Improving the consistency of EKF-SLAM," *Robotics and Autonomous Systems*, vol. 55, no. 1, pp. 21–29, 2007.
- [5] R. Vazquez-Martin, P. Nunez, J. del Toro, A. Bandera, and F. Sandoval, "Adaptive observation covariance for EKF-SLAM in indoor environments using laser data," in *Electrotechnical Conference, 2006. MELECON 2006. IEEE Mediterranean*, 2006, pp. 445–448.
- [6] S. Barkby, S. Williams, O. Pizarro, and M. Jakuba, "An efficient approach to bathymetric SLAM," in *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, 2009, pp. 219–224.
- [7] C. Roman and H. Singh, "Improved vehicle based multibeam bathymetry using sub-maps and SLAM," in *Intelligent Robots and Systems, 2005. (IROS 2005). 2005 IEEE/RSJ International Conference on*, 2005, pp. 3662–3669.
- [8] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. The MIT Press, Sep. 2005.