Precision in Motion: Enhancing 2D Robot Pose through Integrated Encoders, IMU, and Kalman Filter for Dynamic Line Following

Dexter Fernandes
2375876

Pruthvi Geedh
2239674

Sadashiv Vaidya
2467558

***Abstract*—This paper introduces a tailored strategy designed specifically for the 3pi+ Pololu robot, addressing the critical task of precise localization and orientation in a 2D plane. Accurate determination of the robot's position (x, y) and orientation (θ) is essential for a range of applications, from autonomous navigation to intricate manipulation tasks.**

The method utilises encoders and an inertial measurement unit (IMU) with sensors such as accelerometers, gyroscopes, and a magnetometer, combined with a line-following algorithm. Real-world experiments validate the system's performance, demonstrating a significant improvement in pose estimation and orientation accuracy compared to individual sensors. The approach's adaptability shines in tests across diverse line shapes, showcasing robustness in various environments.

This tailored solution extends beyond the 3pi+ Pololu robot, offering reliability for diverse robotic systems and contributing to the ongoing discourse on sensor integration. The clarity, precision, and tangible results presented in this work make a lasting contribution to the field, pushing the boundaries of possibilities in robotics

***Index Terms*—3pi+ Pololu robot, Inertial Measurement Unit (IMU), Encoders, Precise localization, Odometry, 2D plane navigation, Line-following algorithm, Kalman Filter, Pose estimation accuracy, Sensor fusion, Autonomous navigation.**

I. Introduction

In the realm of robotics, achieving precision in 2D plane localization and orientation stands as a pivotal challenge with broad applications, spanning autonomous navigation to intricate manipulation tasks. This research presents a distinct strategy tailored explicitly for the 3pi+ Pololu robot, recognizing the significance of accurately determining the robot's position (x, y) and orientation (θ). The implementation of encoders and an inertial measurement unit (IMU), coupled with a line-following algorithm, forms the cornerstone of this approach, leveraging inherent functionalities to capture essential data on wheel rotations, accelerations, and angular rates.

A notable feature of this strategy lies in enhancing the robot's navigation capabilities across diverse line shapes, achieved through the integration of a line-following algorithm and post-processing using the extended Kalman filter. This integration addresses nuanced challenges presented by varied terrains and scenarios, contributing to a more precise, versatile, and adaptive robotic system.

Meticulous real-world experiments validate the system's performance, revealing a substantial enhancement in pose and orientation estimation accuracy compared to reliance on individual sensors. The approach's adaptability is underscored through tests across different line shapes, emphasizing its robustness in diverse environments.

Beyond 3pi+ Pololu robot, this tailored approach extends its utility, providing a reliable solution for diverse robotic systems. The integrated line-following algorithm enhances versatility, making it practical for various applications.

II. Literature Review

The Dead Reckoning problem of getting the position of the moving object is discussed a lot because of its significance in robotics applications.

In the case of a 2-wheeled differential drive 3Pi+ robot, encoders are used with left and right wheels to get the distance travelled by each of them. The position and yaw angle of the mobile robot are calculated by the rotary angle, the diameter of the wheel, and the body width of the mobile robot. This method is called odometry[1]. Position estimation using odometry has unbounded errors due to slippage, mismatches in system parameters, and noise from the encoder signals. Hence, an inertial measurement unit (IMU) can be used to solve the dead reckoning problem.

Use of IMU prevents errors in measurements due to wheel slippage. An IMU consists of an Accelerometer, Gyroscope, and magnetometer to get accelerations, angular velocities, and pole direction respectively. The main problem in implementing IMU is the drift caused by to double integration of readings and it is discussed thoroughly in [2, 3]. Solutions include calibration of the sensors at the start of the experiments [4] and the use of filters mentioned in [2] to combine data from accelerometer and gyroscope readings.

One of the effective filters used for reducing measurement and process noise is the Kalman Filter. Kalman filter predicts the state using previous states and updates the current state with the help of Kalman gain functions [5, 6, 7].

In this paper, we have implemented encoders and IMU separately to understand errors caused by position and orientation estimation. Then to reduce this error we have effectively used the Kalman filter on each sensor reading.

A. Hypothesis Statement

Hypothesis 1 (H1): Pose estimation using wheel encoders alone is anticipated to exhibit noticeable errors, especially during turns, stemming from limitations in the kinematics model of wheel encoder measurements.

Hypothesis 2 (H2): We anticipate that estimating position and orientation using the IMU (Accelerometer +

Gyroscope) will have less error when compared with the encoders.

Hypothesis 3 (H3): Application of the Kalman filter aims to reduce measurement noise in position measurements from both encoders and the IMU.

In summary, the hypotheses collectively address the expected challenges and solutions in the 2D pose estimation of the 3pi+ Pololu robot. They explore the limitations of individual sensors, propose strategies for noise reduction and sensor fusion, and anticipate improvements in precision and reliability through the application of the Kalman filter.

III. Implementation

A simple line-follow algorithm with the bang-bang controller was implemented on Polulu 3pi+ robot to understand noise and error in the position and orientation estimates of a two-wheeled differential drive robot. Encoders and IMU were used to conduct experiments, and x, y, and theta were calculated in a global coordinate frame.

A. Line Following

To enable left-edge line following, a 3pi Pololu robot with five circularly arranged line sensors initializes motors, line sensors, and a bang-bang controller. Continuous sensor readings detect the line's position relative to the left edge, with errors calculated for deviation assessment. The bang-bang controller prompts left turns when the robot is too far and right turns when too close, adjusting thresholds for optimal response. Dynamic motor speed adjustments align the robot with the left edge, while adaptive time steps ensure real-time responsiveness.

B. Wheel Encoders

Encoders were initialized at the start to implement encoders for a 3Pi Polulu robot. 3Pi robot uses interrupt service routine (ISR) to count ticks of encoders. Readings from the left and right wheel counts were then converted into the distance traveled by each wheel. Then using ICC equations, position (x, y) and orientation (θ) were calculated in global frame coordinates.

In following equations D_r and D_l are distances travelled by the right and left wheel, D_c is the distance travelled by centre of the robot, and L is the wheelbase distance.

$$\omega = \frac{v_r - v_l}{2} \tag{1}$$

$$\Delta\theta = \frac{D_r - D_l}{L} \tag{2}$$

$$x = x + D_c cos(\theta) \tag{3}$$

$$y = y + D_c sin(\theta) \tag{4}$$

$$\theta = \theta + \Delta\theta \tag{5}$$

C. Inertial Measurement Unit

3pi+ robot uses LSM6 IMU which consists of an accelerometer, gyroscope, and magnetometer. The accelerometer gives overall acceleration in x, y, and z directions, hence we need to subtract the effect of gravity from each component. Then, we can calculate position by double integrating linear accelerations that we get from the previous step. However, due to significant noise in the accelerometer values and double integration, we see a lot of drift in position estimation as shown in Fig. 1[2]. It makes it difficult to use IMU for position estimation.

The gyroscope is calibrated at the start of each run by using a sampling method to reduce excess noise in IMU readings. The output of the gyroscope is in mg, where 'g' refers to the acceleration due to gravity. To get the reading in required units of degrees per second we need to multiply it by 0.00875 as mentioned in algorithm. By integrating the angular velocities obtained, we can get the orientation of the robot in the x, y, and z directions.

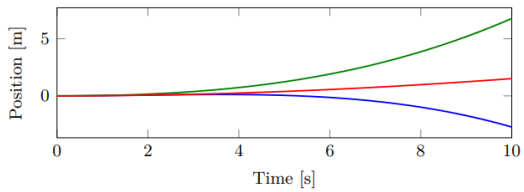


Fig. 1: Position estimation graph in all three axes when the robot is stationary

Algorithm 1 Robot Pose and Line Following with Encoders and LSM6 IMU

Input: LeftEncoderCount, RightEncoderCount, WheelRadius, WheelBase, GyroZ, AccelX, AccelY, LineSensors, dt, runTime
Output: x, y, encoderTheta, imuTheta
while $time < runTime$ **do**
    Calculate Pose (x,y) and orientation (θ) from Encoders using ICC equation:
    Get IMU Angle from:
    Calibrate Gyroscope using a sampling method ▷ Assuming Gyrocalvalues is the calibration value
    $GyroZ \leftarrow (GyroZ - \text{Gyrocalvalues}) \times 0.00875$ ▷ Apply calibration to GyroZ
    $imuTheta \leftarrow imuTheta + GyroZ \times dt$ ▷ Update imuTheta using the calibrated GyroZ
    Line Following with Bang-Bang Controller:
end while
Report Data:
return x, y, $encoderTheta$, $imuTheta$

D. Kalman Filter

The Kalman filter gracefully handles linear system dynamics and measurements, displaying best results where predictability is key. However, when the terrain becomes more intricate, featuring non-linear componets due to cos(θ) and sin(θ) involved, it's the Extended Kalman Filter (EKF) that presents as the optimal solution.

Extended Kalman Filter Equations

Prediction Step:: The EKF's prediction step estimates the current state by incorporating the non-linear system dynamics function f and the control input u_k:

$$\hat{x}_{k|k-1} = f(\hat{x}_{k-1|k-1}, u_k) \tag{6}$$

The prediction error covariance is then updated using the Jacobian matrix H_k, capturing the linearisation of the system dynamics:

$$P_{k|k-1} = H_k \cdot P_{k-1|k-1} \cdot H_k^T + Q_k \tag{7}$$

Update Step:: The update step corrects the prediction based on measurements. The Kalman gain K_k is determined by the Jacobian matrix H_k of the measurement function and the measurement noise covariance R_k:

$$K_k = P_{k|k-1} \cdot H_k^T \cdot (H_k \cdot P_{k|k-1} \cdot H_k^T + R_k)^{-1} \tag{8}$$

The predicted state is then corrected using the Kalman gain and the difference between the actual measurement z_k and the predicted measurement:

$$\hat{x}_k = \hat{x}_{k|k-1} + K_k \cdot (z_k - h(\hat{x}_{k|k-1})) \tag{9}$$

The error covariance matrix is updated to refine the uncertainty estimation:

$$P_k = (I - K_k \cdot H_k) \cdot P_{k|k-1} \tag{10}$$

1st Degree Taylor Series Expansion:

Linearization, a key concept in EKF, employs the 1st degree Taylor series expansion. It approximates non-linear functions with linear counterparts.

Jacobian Matrix

At each time step, the Jacobian is evaluated with current predicted states. These matrices can be used in the Kalman filter equations. This process essentially linearises the non-linear function around the current estimate.

$$H = \left.\frac{\partial f}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{a}} \tag{11}$$

Here, $\mathbf{x}$ represents the state variables, and f is the non-linear system dynamics function.

In the EKF context, $f(a)$ corresponds to the previous state, $\frac{\partial f}{\partial x}$ is the Jacobian, and x is the current state.

The computation of the Jacobian matrix takes place within the extendedKalmanFilter() function of the Control Systems Toolbox of MATLAB.

State transition functions for Extended Kalman Filter

The following equations describe the computation of the new state of position (x, y) and orientaion (θ):

$$x_{t+1} = x_t + v_t \cos\theta_t \Delta t \tag{12}$$

$$y_{t+1} = y_t + v_t \sin\theta_t \Delta t \tag{13}$$

$$\theta_{t+1} = \theta_t + \omega_t \Delta t \tag{14}$$

where,

t = timestep

x_k = x-coordinate at timestep k

y_k = y-coordinate at timestep k

v_k = linear velocity of the robot

θ_k = robot orientation at timestep k

Δt = time interval between measurements

ω_k = $\left(\frac{v_R - v_L}{L}\right)$ = angular velocity at timestep k

To improve the estimation of the position (x, y) and orientation (θ) of the robot, we have empirically obtained the following matrices:

for estimation using wheel encoders,

$$ProcessNoise = \begin{bmatrix} 5e-1 & 0 & 0 \\ 0 & 5e-1 & 0 \\ 0 & 0 & 9e-1 \end{bmatrix}$$

$$MeasurementNoise = \begin{bmatrix} 1e-3 & 0 & 0 \\ 0 & 1e-3 & 0 \\ 0 & 0 & 1e-1 \end{bmatrix}$$

for estimation using IMU,

$$ProcessNoise = \begin{bmatrix} 5e-1 & 0 & 0 \\ 0 & 5e-1 & 0 \\ 0 & 0 & 9e-6 \end{bmatrix}$$

$$MeasurementNoise = \begin{bmatrix} 1e-3 & 0 & 0 \\ 0 & 1e-3 & 0 \\ 0 & 0 & 1e-6 \end{bmatrix}$$

A relatively higher value of Process noise and Measurement noise was chosen for the estimation of θ using wheel encoders during to the unreliability in measurement observed.

IV. Metrics

Mean Squared Error (MSE): For a higher penalty for large errors

$$\frac{1}{M}\sum_{i=1}^{M}\sqrt{((x_2 - x_1)^2 + (y_2 - y_1))^2} \tag{15}$$

where,

M = number of measurements

x_i = measurement of the x-coordinate

x_i = measurement of the x-coordinate

The Mean Squared Error (MSE) is a valuable metric for comparing the performance of encoder and IMU systems in robot path-following. MSE determines the difference between the robot's position and the ideal path of the robot i.e. the center of the line at any given point. It emphasizes minimizing differences between actual and measured positions by penalizing large errors more. We calculate MSE for both systems and compare the values. A lower MSE indicates better accuracy, meaning the system with lower MSE approximates the robot's path more accurately. This metric is useful for highlighting the importance of reducing larger errors and providing a comprehensive assessment of system performance to make informed decisions about capturing precise robot trajectories.

V. Experiment Methodology

The robot used in the experiment was a 3Pi+ Pololu robot equipped with wheel encoders and an LSM6 IMU sensor. The bang-bang controller implemented a left-edge line following. To obtain linear velocity of each robot wheel, a coefficient of **0.247** was multiplied with each wheel's PWM signal. This coeffiecient was obtained empirically. At the beginning of each trial, calibration of the wheel encoders and IMU sensor was performed to ensure accurate measurements.

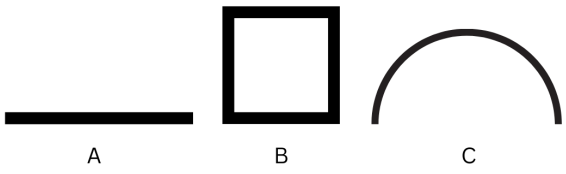


Fig. 2: Different line types (Not to scale)

A. Straight Line Experiment

1) Configuration: The robot was positioned at the start of a straight line path measuring 57 cm.

2) Procedure: The robot was initiated, and encoder and IMU data were recorded at intervals of 170 ms. This process was repeated five times for consistency with a run time of 6800 ms.

B. Square Box Experiment

1) Configuration: The robot was placed at the starting point of a square box shape with sides measuring 11.5 cm.

2) Procedure: Similar to the straight line experiment, the robot executed the box trajectory five times in each orientation, recording data at the defined intervals. The experiment was done in clockwise and anticlockwise direction to check for any bias with a run time of 8500 ms.

C. Semi Circle Experiment

1) Configuration: The robot was positioned at the starting point of a semicircle path with a diameter of 38.4 cm.

2) Procedure: Similar to the straight line and square box experiments, the robot executed the semicircle trajectory five times in each orientation. Data were recorded at defined intervals to capture the robot's movements accurately.The experiment was done in clockwise and anticlockwise direction to check for any bias with a run time of 8500 ms.

D. Data Collection and Analysis

1) Process: In the data collection phase, Measurements were acquired at intervals of 170 ms, and a minimum of 5 trials were conducted for each line type. To increase precision, the actual coordinates and lengths of each line were meticulously determined by measuring the endpoints and interpolating the intermediate coordinates, assuming uniform motion of the robot.

Notably, the data was stored using the EEPROM library, ensuring the retention of crucial information even when the robot was unplugged. Upon re-connection via a serial monitor cable, the data was seamlessly retrieved for post-processing

The integration of MATLAB for in-depth analysis and Python post-processing added a layer of sophistication to the methodology. This comprehensive approach not only guarantees a profound understanding of the robot's performance across various trajectories and contributes to the reliability of the results.

2) Data Visualisation: Python, Matplotlib, Seaborn and other visualization tools were employed to plot graphs for a comprehensive understanding of the robot's performance.

E. Overview of Method

The experiment aimed to evaluate the performance of the 3Pi+ Pololu robot equipped with wheel encoders and an LSM6 IMU sensor using a bang-bang controller for left-edge line following. The robot traversed predefined paths, including straight lines, square boxes, and semicircles, to assess its ability to follow distinct trajectories. The key components of the experiment included the implementation details, variable configurations, experimental procedures, data collection, and analysis methodologies.

F. Discussion of Variables

1) Independent Variables: In the experiments, independent variables were manipulated to observe their impact on the robot. In the straight line, distance influenced linear motion response. In the semicircle, clockwise and anticlockwise orientations were independent variables, affecting curved trajectories. The square box experiment explored the impact of box orientation (clockwise and anticlockwise) on the robot's performance in box trajectories.

2) dependent variables: Dependent variables measured outcomes tied to manipulated factors. In the straight line, it gauged the robot's linear motion response to distance changes. In the semicircle, it assessed the robot's behavior in curved paths, influenced by orientation variations. The square box experiment focused on the robot's performance during box trajectories, impacted by changes in box orientation. These metrics were crucial for evaluating overall performance.

3) controlled variables: To ensure the validity and reliability of the experiments, certain variables were carefully controlled to minimize their potential impact on the robot's behavior. Two crucial controlled variables in these experiments were the flat surface and ambient lighting conditions.

a) Flat Surface: The experiments were conducted on a flat and uniform surface to ensure a consistent foundation, eliminating potential variations in terrain that could influence the robot's movement.

b) Ambient Lighting Conditions: Ambient lighting was consistently maintained to stabilize the visual environment for the robot's sensors. This minimized potential inaccuracies in perception that could arise from fluctuations in lighting, avoiding confounding variables.

to the last page, then move this whole earlier in your document to get it to display on an earlier

VI. Results

In the evaluation of the robot path-following system, This involved comparing the robot's path as determined by each sensor—encoder, IMU, and EKF—with the actual real-world coordinates.

A. Straight line type

The simple nature of the straight line path causes the wheel encoders to give a better orientation measurement when compared to the IMU. Employing the EKF for optimal position estimation has yielded much more precise measurements while the same for orientation estimation could use improvement.

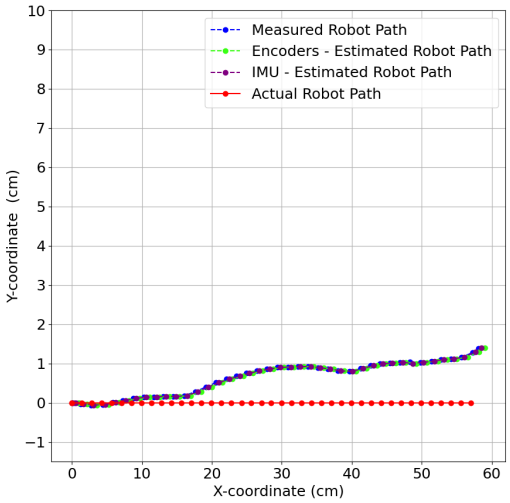
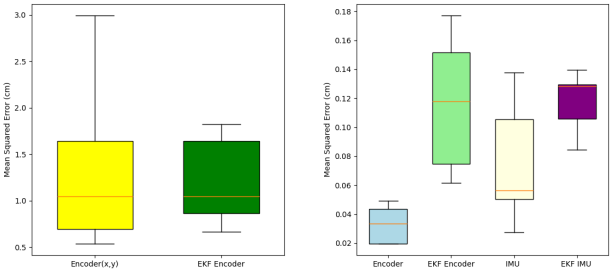


Fig. 3: Straight line Robot path comparison



(a) Encoder vs EKF for (X,Y) (b) Encoder & IMU vs EKF theta

Fig. 4: MSE for straight line comparing (X,Y) and Theta

B. Square Box line type

The use of IMU significantly reduces MSE in orientation(θ) estimation when compared with the encoder's orientation estimation. However, implementing EKF on encoders' data to estimate better position(x, y) does not show improvement in results. Also, implementing EKF on orientation estimation shows a slight improvement in precision.

C. Semi Circle line type

As expected, when traversing a curving path, the MSE between the IMU measurements and the actual measurements is relatively lower when compared to that of the wheel encoders. Encoders perform poorly due to wheel slippage in the turn. The use of Extended Kalman filter yields little-to-no improvement in the measurements of the position (x, y) and the orientation (θ) of the robot.

D. Hypothesis Validation

Hypothesis 1 (H1): Encoders alone exhibited noticeable errors, especially during turns, confirming the need for additional sensor integration.

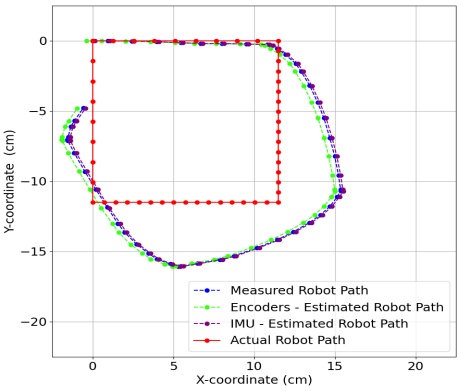
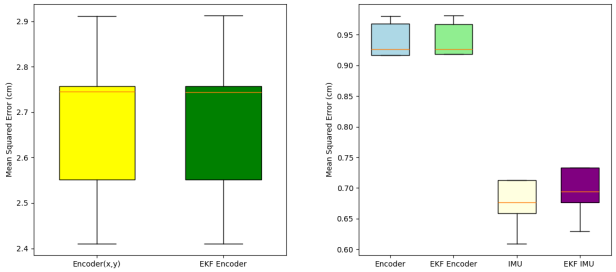


Fig. 5: Square Box Robot path comparison



(a) Encoder vs EKF for (X,Y) (b) Encoder & IMU vs EKF theta

Fig. 6: MSE for Square box comparing (X,Y) and Theta

Hypothesis 2 (H2): IMU-based estimation showed less error compared to encoders alone, validating the effectiveness of IMU in navigation.

Hypothesis 3 (H3): Application of the Kalman filter slightly reduced measurement noise in position measurements from both encoders and the IMU.

VII. Discussion and Conclusion

To understand errors in measurements from wheel encoders and IMU, the MSE metric was plotted against real-world coordinates for position (x, y) and orientation(theta). To avoid biased results we have performed experiments on different line shapes as shown in Fig.2 .

A. Regarding Position(x,y) Estimation

Using wheel encoders, position (x,y) estimation in the global frame is reliable when the traversed path does not have any turns, as shown in Fig. 6 . However, pose estimations give significant errors when the traversed path has turned as seen in the small box and semi-circle experiments Fig.5,7. Even though ICC equations were used to calculate the heading or the robot, errors persisted because of measurement errors in the kinematic model. To improve position (x,y) estimation we attempted to use IMU (accelerometer), however, we discovered constant drift in the values of x and y. This is potentially due to the presence of noise in accelerometer reading which constantly increases due to the double integration function required to get position from acceleration. To further improve the quality of our measurements from wheel encoders, we performed

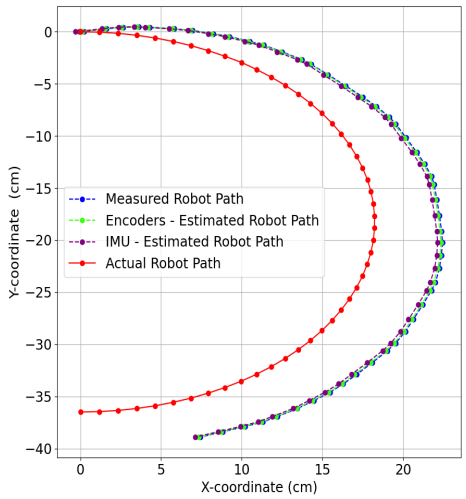
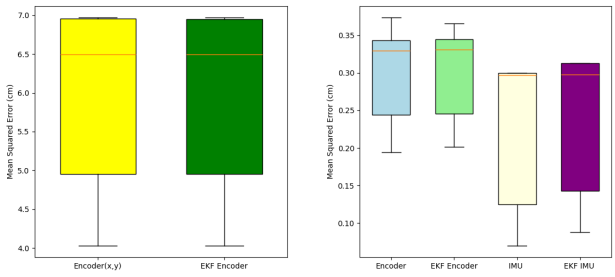


Fig. 7: Semi Circle Robot path comparison



(a) Encoder vs EKF for (X,Y) (b) Encoder & IMU vs EKF theta

Fig. 8: MSE for Semi-circle comparing (X,Y) and Theta

post-processing in MATLAB using Extended Kalman Filter (EKF). As seen from the Fig.3, we obtained more precise results in a straight-line experiment. However, when EKF is carried over to small box and semi-circle traversal little to no improvement is observed which suggests further work is required in tuning noise parameters in EKF.

B. Regarding Orientation(θ)

As shown in Fig.4 orientation estimations for straight line using encoders give better results than IMU and EKF-implemented encoder and IMU. When similar experiments were conducted on small box and semicircles, encoders performed relatively poorly as seen in Fig.6,8. when compared with IMU because of kinematic model error in calculation of orientation. Thus proving reliability of gyroscope data for orientation estimation. To further improve measurements, we plotted MSE of post-processed data using EKF. However, EKF implementation does not improve the measurements by large margin.

VIII. FUTURE WORK

Firstly, there's room for enhancing the kinematic model, delving deeper into factors like wheel slippage and dynamic changes in wheel properties. Concurrently, dynamic sensor behavior analysis could benefit from advanced algorithms to tackle drift, noise, and temporal variations. Improving parameters used in the Extended Kalman Filter, robust to different line types, can also provide another frontier to estimate optimal pose. Implementing advanced sensor fusion techniques, such as the Madgwick filter, holds promise in resolving drift issues and enhancing the accuracy of x and y position estimates derived from the Inertial Measurement Unit (IMU). This presents an avenue for future research to explore and refine filtering processes that optimize the fusion of encoder and IMU data, pushing the boundaries of the 3pi+ Pololu robot into realms of increased autonomy and adaptability.

REFERENCES

[1] B.-S. Cho, W.-s. Moon, W.-J. Seo, and K.-R. Baek, "A dead reckoning localization system for mobile robots using inertial sensors and wheel revolution encoding," *Journal of Mechanical Science and Technology*, vol. 25, pp. 2907–2917, Nov. 2011.

[2] M. Kok, J. D. Hol, and T. B. Schön, "Using inertial sensors for position and orientation estimation," *Foundations and Trends® in Signal Processing*, vol. 11, no. 1–2, pp. 1–153, 2017.

[3] J. Zhao, "A review of wearable imu (inertial-measurement-unit)-based pose estimation and drift reduction technologies," *Journal of Physics: Conference Series*, vol. 1087, p. 042003, Sept. 2018.

[4] D. Tedaldi, A. Pretto, and E. Menegatti, "A robust and easy to implement method for imu calibration without external equipments," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, May 2014.

[5] E. I. Al Khatib, M. A. Jaradat, M. Abdel-Hafez, and M. Roigari, "Multiple sensor fusion for mobile robot localization and navigation using the extended kalman filter," in *2015 10th International Symposium on Mechatronics and its Applications (ISMA)*, IEEE, Dec. 2015.

[6] H. Ferdinando, H. Khoswanto, and D. Purwanto, "Embedded kalman filter for inertial measurement unit (imu) on the atmega8535," in *2012 International Symposium on Innovations in Intelligent Systems and Applications*, IEEE, July 2012.

[7] R. I. Alfian, A. Ma'arif, and S. Sunardi, "Noise reduction in the accelerometer and gyroscope sensor with the kalman filter algorithm," *Journal of Robotics and Control (JRC)*, vol. 2, no. 3, 2021.