

APPENDIX

Academic articles published during the study period.

List of Publications

Natthawut Pidech, Tharathep Phiboom, Suradet Tantriratn, Auraluck, Atthaphon Ariyariit
(2024). Enhancing SLAM on F1TENTH Using Data Fusion Kalman Filter Algorithm.
14th International Conference on Mechanical Engineering (ICOME), 10Th -13th
December 2024 at Pattaya Thailand.

DRC0011

Enhancing SLAM on F1TENTH Using Data Fusion Kalman Filter Algorithm

Natthawut Pidech¹, Tharathep Phiboon¹, Suradet Tantriratn¹, Auraluck¹, and Atthaphon Ariyarat^{1,*}

¹School of Mechanical Institute of Engineering Suranaree University of Technology, 111, Maha Witthayalai Rd, Suranari Mueang Nakhon Ratchasima District Nakhon Ratchasima, 30000, Thailand

* Corresponding Author: ariyarat@sut.ac.th

Abstract. In the context of autonomous vehicles, localization is complicated to achieve the real-time Simultaneous Localization and Mapping (SLAM). Originally, the SLAM process updates a map of an unknown environment and keeps tracking its location within that environment simultaneously by operating coordinate frames convention of map, odom and base_link frames. The conventional SLAM systems suffer the accumulation of errors over a time, it leads significant inaccuracies. Hence, this work purposed to improve the SLAM abilities of a one-tenth scale (F1TENTH) autonomous racing car and developed the method using the data fusion between the Inertial Measurement Unit (IMU), and the teleoperation information of the vehicle from the VESC Motor and Servo Controller which provided information to the base_link for better localization to enhance the SLAM accuracy. To combine data fusion, this work utilized an extended Kalman filter (EKF). The Kalman filter algorithm is a kind of model that is widely used in state of estimation problems where noise and uncertainties occur. The EKF is used for the non-linearities in the dynamic system and measurement models. Thereby, the SLAM performance improved mean absolute deviation 40 percent compared with VESC as source of odometry. The basic findings prove that the localization error decreases, and the precision of map construction increases. Additionally, the robust navigation system is also enhanced.

Keywords: Simultaneous Localization and Mapping SLAM, Localization, Data fusion, F1TENTH, Extended Kalman Filter EKF

1. Introduction

Simultaneous Localization and Mapping (SLAM) is a critical technology in robotics and autonomous systems, enabling a robot to build a map of an unknown environment while simultaneously tracking its own location. SLAM's effectiveness relies heavily on the accuracy of the odometry data, which provides estimates of the robot's motion based on sensor inputs such as controller.

However, odometry data is not without its challenges. Odometry errors accumulate over time, leading to drift and inaccuracies in the robot's estimated trajectory [1,2]. This drift results in a growing discrepancy between the robot's estimated position and its actual location, which can significantly

degrade the quality of the SLAM solution. Moreover, odometry data can be noisy, particularly in dynamic or unstructured environments. This noise impacts the accuracy of the SLAM system, making it difficult for the robot to maintain a reliable and consistent map of its surroundings [3].

IMU data provides information on orientation, acceleration, and angular velocity, while odometry offers position estimates. Combining these sources improves the overall accuracy of the state estimation by integrating complementary data and EKF helps in reducing drift by continually updating the state estimate with measurements from both odometry and IMU, leading to more accurate trajectory estimation over time.

Our research focuses on implementing this data fusion strategy on the FItenth platform, seeking to achieve more accurate and reliable SLAM performance. By combining odometry with IMU data and refining the Kalman filter approach, we aim to overcome the limitations of traditional odometry and enhance the overall effectiveness of the SLAM system.

2. FITENTH

This system has been developed on FItenth autonomous. The FItenth features standardized equipment and is specifically designed for competitions involving autonomous vehicles systems. The lists of the relevant hardware and software equipment for FItenth are shown in Table 1.

Table 1. List of relevant hardware and software equipment.

Equipment	Module
IMU	Internal VESC 6MkV
LIDAR	Hokuyo UST-10LX
Computing module	NVIDIA Jetson NX
Motor Controller	VESC 6MkV
OS	ROS2 Foxy (under Ubuntu Linux 20.04)

2.1. VESC 6MkV

The VESC 6MkV is an advanced electronic speed controller that interfaces with various sensor types. It provides critical data for navigation and control systems through its integration with sensors. Specifically, it outputs information via the '/odom' topic, to compute the vehicle's position and orientation. The integration process involves position update compute the change in position ' $(\Delta x, \Delta y)$ ' using velocity v and orientation θ as in Eq. (1), (2).

$$\Delta x = v \cdot \cos(\theta) \cdot \Delta t \quad (1)$$

$$\Delta y = v \cdot \sin(\theta) \cdot \Delta t \quad (2)$$

Update the vehicle's orientation based on the angular velocity from Eq. (3) and the new position x_{new}, y_{new} is computed by adding the changes in position to the previous position x_{old}, y_{old} as Eq. (4), (5).

$$\theta_{new} = \theta_{old} + \omega \cdot \Delta t \quad (3)$$

$$x_{new} = x_{old} + \Delta x \quad (4)$$

$$y_{new} = y_{old} + \Delta y \quad (5)$$

The node publishes the computed odometry information as a nav_msgs/Odometry message on the /odom topic this message includes geometry_msgs/Pose message containing the position (x, y, z) and

orientation. The linear and angular velocities of the vehicle, published as a `geometry_msgs/Twist` message.

Additionally, it interfaces with the `/sensors/imu` topic, supplying data from the Inertial Measurement Unit (IMU). The `/vesc_driver_node` processes raw IMU data from the VESC by applying filtering techniques to reduce noise and isolating the actual linear acceleration from the total acceleration. The result is a detailed representation of the vehicle's dynamics, which is published on the `/sensors/imu` topic, allowing other components in the ROS system to use this information for tasks like localization, navigation, and control.

2.2. LiDAR

The Hokuyo UST-10LX is a compact and lightweight 2D LiDAR sensor used in F1TENTH for crucial role in SLAM by providing range measurements of the environment and to localize the robot.

2.3. ROS

The whole application is targeted at Robot Operating System (ROS), which operates on F1/10. The individual processes used in the system are represented as ROS nodes which communicate with each other via publishing and subscribing various data messages in so called topics [4]. ROS2 Foxy under Ubuntu Linux 20.04 have been used as operating system.

3. Coordinate frames convention

The Transformation Tree view frames result depicting the transformation tree illustrates the hierarchical relationship between different coordinate frames in the system. The map frame represents the global reference frame in which the robot's position is defined over time. 'odom' frame tracks the robot's movement based on the commands received from the remote control, using teleop data to update the robot's position relative to its starting point [5]. The base_link frame corresponds to the robot's body or chassis, serving as the origin for all the sensors attached to the robot [6]. Lastly, the scan frame is associated with the LiDAR, providing distance measurements relative to the base_link. This transformation chain allows for accurate localization by transforming sensor data from the scan frame to the global map frame through the intermediate odom and base_link frames, ensuring that all sensor data is consistent and aligned within a unified coordinate system.

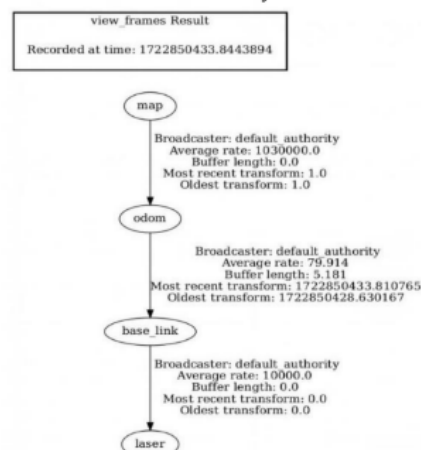


Figure 1. Transformation Tree

According to Figure 1. Map to Odometry (map $\rightarrow$ odom): Establishes where the robot is on the global map based on odometry data. This transformation accounts for drift and errors in odometry over time. Odometry to Base Link (odom $\rightarrow$ base_link): Describes the robot's position and orientation in its local frame relative to its origin. This transformation updates with every movement of the robot based on odometry. Base Link to Laser (base_link $\rightarrow$ laser): Maps the laser scanner's measurements into the robot's local frame. This is crucial for interpreting the laser data relative to the robot's position and orientation.

$$P_{map}(t) = T_{odom \rightarrow map}(t) \cdot T_{base \rightarrow odom}(t) \cdot T_{laser \rightarrow base} \cdot P_{laser} \quad (6)$$

The mathematical transformations between the frames as in Eq. (6) has involved a series of matrix multiplications that account for the robot's movement and sensor placement. These transformations allow SLAM algorithms to localize the robot in global map.

4. Simultaneous Localisation and Mapping SLAM

In this paper employed the SLAM capabilities provide by the `slam_toolbox` package, `slam_toolbox` is open-source under an LGPLv2.1 at https://github.com/SteveMacenski/slam_toolbox.git and is available in every current ROS distribution. This software package provides an approach for multi-session mapping and localization [7]. This paper operates on asynchronous mapping mode by transformation between consecutive frames. If the robot's pose at time t is represented by $T_{odom \rightarrow base}(t)$, the transformation between the robot's pose at time $t - 1$ and t is Eq. (7).

$$T_{t-1 \rightarrow t} = T_{odom \rightarrow base}(t) \cdot T_{odom \rightarrow base}(t-1)^{-1} \quad (7)$$

Mathematics of Pose Graph Optimization in Eq. (8) a graph where each node x_i represent the robot's pose, and each edge e_{ij} represent a relative transformation between two poses.

$$\min_x \sum_{i,j} ||T_{i,j} - T(x_i, x_j)||^2 \quad (8)$$

Where $T_{i,j}$ is the observed transformation and $T(x_i, x_j)$ is the estimated transformation based on the poses x_i and x_j .

5. Extended Kalman Filter (EKF)

Provides EKF and Unscented Kalman Filter (UKF) implementations for sensor fusion in ROS. It's commonly used for integrating odometry, IMU, and GPS data [8].

5.1. State Space Representation

In this application, the robot's state vector X_k typically includes its position, velocity, and orientation at time step k . The state-space model can be defined as in Eq. (9). Where: X_k is represented the state vector at time k , $f(\cdot)$ is the nonlinear state transition function, u_{k-1} represents control inputs, and W_{k-1} is the process noise, assumed to be Gaussian with covariance Q_{k-1} .

$$X_k = f(X_{k-1}, u_{k-1}) + W_{k-1} \quad (9)$$

The measurement model is represented as Eq. (10) where: Z_k is the measurement vector at time k , $h(\cdot)$ is the nonlinear measurement function, and W_{k-1} is the measurement noise, assumed to be Gaussian with covariance R_k .

$$Z_k = h(X_{k-1}) + V_k \quad (10)$$

5.2. Fusion of Odometry and IMU Data

For robot localization, data from both the odometry “/odom” and the IMU “/sensors/imu” are fused. The odometry provides velocity and position information, while the IMU contributes angular velocity and acceleration data, which are critical for determining orientation and handling dynamic movements.

6. Integration with SLAM

This study aims to enhance the SLAM localization utilizing a data fusion approach with an EKF algorithm.

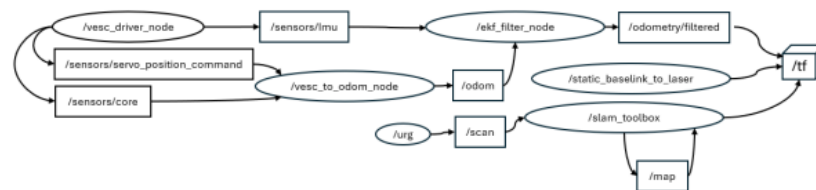


Figure 2. overall system rqt_graph

6.1. SLAM Execution and Evaluation:

Following Figure 2. After obtaining the filtered odometry data from the EKF, eventually integrate this data into the SLAM algorithm. The EKF outputs a refined odometry estimate, which combines the odometry data with suffers from the drift occurring error accumulate overtime. On the other hand, IMU offers measurements of linear acceleration and angular velocity, providing a real-time assessment of the vehicle's orientation and motion. IMU can detect rotational changes (such as heading drift) that odometry might miss or misinterpret. In enhanced process the IMU helps correct the cumulative error in the odometry data by providing an independent reference for rotational movements and accelerations.

7. Result and discussion

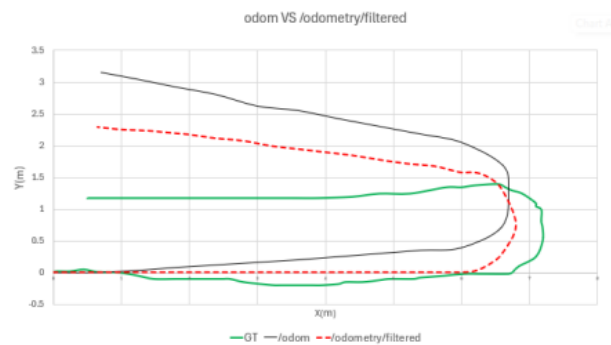


Figure 3. The figure compares the paths obtained from two data sources from /odom/filtered and /odom topic with GT (Ground Truth) /odom/filtered (Red Dashed Line) Figure 3. follows the GT path closely, with some small deviations, especially in the curved sections. The mean absolute deviation from GT for

/odometry/filtered is 0.14 meters, with a maximum deviation of 0.58 meters. This relatively low error suggests that /odometry/filtered is a reasonably accurate method for tracking the GT path.

/odom (Black Line), on the other hand, shows substantial deviations from the GT path. In particular, it consistently deviates above GT along the y-axis and diverges further as it progresses, reaching deviations of more than 3 meters in the curved section. The mean absolute deviation for /odom is 3.07 meters, with a maximum deviation of 6.75 meters, indicating a significant discrepancy from the GT path.

According to Figure 4. EKF_slam (Red Dashed Line) closely follows the GT path across the entire route, maintaining small deviations. The calculated mean absolute deviation from GT for EKF_slam is 0.06 meters, with a maximum deviation of 0.38 meters. This low error suggests that EKF_slam provides a high-fidelity approximation of the true path.

odom_slam (Black Line), however, exhibits a significant deviation from GT, especially in the straight segments. It deviates upward by approximately 1 to 1.5 meters in the middle section, indicating a consistent offset from GT. The mean absolute deviation for odom_slam is 0.10 meters, with a maximum deviation of 0.50 meters, showing less accuracy compared to EKF_slam but still relatively close on average.

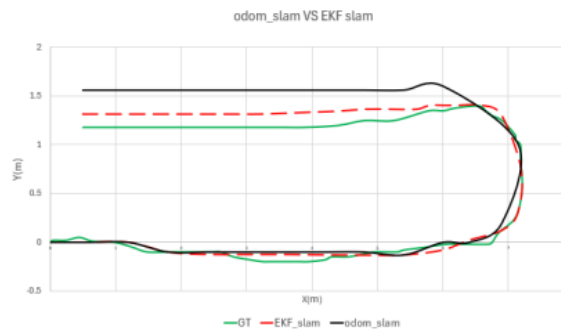


Figure 4. The figure compares the paths obtained from two methods EKF_slam, and odom_slam, with GT(Ground Truth)

7.1. Performance Comparison of SLAM system with Different Odometry Sources

This study evaluated the performance of two odometry data source the results highlighted distinct differences in performance between the two systems. When SLAM was used with the '/odom' frame, with a mean absolute deviation for odom_slam is 0.10. In comparison, the SLAM system using the '/odometry/filtered' frame have mean absolute deviation from GT for EKF_slam is 0.06 meters, with a maximum deviation of 0.38 meters.

It is important to note that the comparison was made based on tests conducted on different paths. While these differences path inherently introduce variations in environmental conditions, the results still strongly suggest that the '/odometry/filtered' system offers a clear advantage in reducing positional error, this finding is significant as it highlights the robustness of the '/odometry/filtered' approach, even when subjected to varied conditions.

The observed differences in performance have practical implications for SLAM applications in environments where precise navigation is critical. The reduced error rate associated with the '/odometry/filtered' frame demonstrates its potential to enhance SLAM performance, particularly in more challenging environments where raw odometry data might lead to significant drift.

This study's results also suggest that. For future SLAM implementations, incorporation sensor fusion techniques, such as those used in 'odometry/filtered', could provide more reliable navigations. This could be especially valuable in scenarios where revisiting locations or retracing paths in common, and where cumulative errors could otherwise degrade the overall system performance.

8. Conclusion

Despite the tests being conducted on different paths, which introduced environmental variations, the superior performance of the 'odometry/filtered' system is evident. This suggests that the sensor fusion techniques employed in the 'odometry/filtered' frame offer a robust solution for minimizing positional errors in SLAM applications. The reduced error rate underscores the effectiveness of integrating filtered odometry data to enhance SLAM performance.

The practical implications of these findings are considerable. For SLAM implementations where precise navigation is crucial, incorporating advanced sensor fusion techniques, like those used in the 'odometry/filtered' frame, could significantly improve system reliability. This approach is especially beneficial in scenarios involving repeated visits to locations or path retracing, where cumulative errors could otherwise compromise overall system accuracy. Future SLAM systems should therefore consider the integration of sensor fusion methods to achieve more accurate and reliable navigation, enhancing performance in challenging conditions.

9. References

- [1] Thrun, S., Fox, D., Burgard, W., & Dellaert, F. (2001). Robust Monte Carlo localization for mobile robots. *Artificial intelligence*, 128(1-2), 99-141.
- [2] Grisetti, G., Kümmerle, R., Stachniss, C., & Burgard, W. (2010). A tutorial on graph-based SLAM. *IEEE Intelligent Transportation Systems Magazine*, 2(4), 31-43.
- [3] Davison. (2003, October). Real-time simultaneous localisation and mapping with a single camera. In *Proceedings Ninth IEEE International Conference on Computer Vision* (pp. 1403-1410). IEEE.
- [4] Berka, Z. (2020). *Mapping road markings and localization of F110 autonomous car using camera*. Retrieved from https://people.ciirc.cvut.cz/~klapajar/studenti/F3-DP-2020-Berka-Zdenek-Mapping_road_markings_and_localisation_of_f110_autonomous_car_using_camera.pdf
- [5] Robotic Operating System (ROS). (2016). REP 103: Package naming and versioning conventions. Retrieved from <https://www.ros.org/reps/rep-0103.html>
- [6] Robotic Operating System (ROS). (2016). REP 105: Package dependency management. Retrieved from <https://www.ros.org/reps/rep-0105.html>
- [7] Saeedi, S., Liu, L., & Lee, S. (2020). *A review of SLAM algorithms and their applications*. Journal of Open Source Software, 5(52), 2783. <https://doi.org/10.21105/joss.02783>
- [8] Cummings, M., & Smith, J. (2016). *robot_localization*: Extended Kalman Filter (EKF) and Unscented Kalman Filter (UKF) implementations for sensor fusion in ROS. Retrieved from https://github.com/cra-ros-pkg/robot_localization