COMP0217 Final Project Report
Sub-Terranean Navigation Project

Artemis Androutselli, Afthab Shiraz, Eesha Mashruwala, Sofiya Flenova
Department of Computer Science, University College London, London, UK
{artemis.theotoki.24, afthab.shiraz.24, eesha.mashruwala.24, sofiya.flenova.24}@ucl.ac.uk

***Abstract*—This report presents our state estimation approach for the COMP0217 Sub-Terranean Navigation Project. The objective is to estimate the robot's 2D position and yaw in real time using onboard inertial and Time-of-Flight (ToF) sensors in a motion capture arena without GPS. We describe the estimator design, calibration strategy, sensor fusion pipeline and evaluation on the provided datasets. Results are compared against ground truth to assess accuracy, robustness and failure modes.**

***Index Terms*—Sensor fusion, state estimation, Extended Kalman Filter, inertial sensing, Time-of-Flight, robotics**

I. INTRODUCTION

This project considers real-time localisation of a mobile robot in a bounded indoor arena without GPS. The robot is equipped with inertial sensors and three ToF range sensors, while motion capture provides ground truth for development and evaluation. The goal is to estimate planar position (x, y) and yaw ψ accurately during both simple and more complex trajectories.

In this report, we describe the estimation problem, motivate our modelling choices, present the sensor fusion algorithm and evaluate performance on the provided training datasets.

II. PROJECT SETUP AND PROBLEM FORMULATION

A. Available Sensor Streams

Table I summarises the main logged data streams available in the provided datasets.

TABLE I: Available Logged Data Streams Used by the Estimator.

Signal	Nominal Rate
GT position, GT rotation, GT time	200 Hz
Sensor ACCEL, Sensor GYRO	104 Hz
Sensor LP_ACCEL, Sensor Temp	100 Hz
Sensor MAG	50 Hz
Sensor ToF1, Sensor ToF2, Sensor ToF3	10 Hz

Ground truth rotation is provided as quaternions. In this project, we use it only for initialisation and evaluation, while the estimator itself relies on onboard sensing.

B. Estimation Objective

The objective of the estimator is to provide a real-time estimate of the robot's planar pose, consisting of position (x, y) and heading ψ, using onboard accelerometer, gyroscope and Time-of-Flight (ToF) measurements. In addition, planar velocity states are included to support smooth propagation between sparse range updates.

The state vector is defined as

$$\mathbf{x}_k = \begin{bmatrix} x_k & y_k & \psi_k & v_{x,k} & v_{y,k} \end{bmatrix}^\top, \tag{1}$$

where (x_k, y_k) denote the robot position in the global frame, ψ_k is the yaw angle, and $v_{x,k}, v_{y,k}$ are the planar velocity components.

This state representation is intentionally compact. Rather than estimating inertial sensor biases as additional states, gyroscope and accelerometer biases are estimated offline.

C. Tasks

The estimator is evaluated on two core tasks:

1) **Task 1:** Straight-line motion with forward and backward segments.
2) **Task 2:** A more complex circuit trajectory with varying motion.

The same estimator should be used for both tasks. Calibration data has been used to estimate biases, scale factors, thresholds, or noise parameters before hand and hardcoded into the EKF simulink block.

III. METHODOLOGY

A. System Model

We model the estimator dynamics in discrete time as

$$\mathbf{x}_{k+1} = f(\mathbf{x}_k, \mathbf{u}_k, \Delta t) + \mathbf{w}_k, \tag{2}$$

where $\mathbf{u}_k$ denotes the sensor-derived motion input used in the prediction step, Δt is the sampling interval and $\mathbf{w}_k \sim \mathcal{N}(0, \mathbf{Q}_k)$ is zero-mean process noise.

In our implementation, the prediction input is formed from IMU measurements, with accelerometer inputs contributing weakly due to strong velocity damping in the motion model.

The measurement model is written as

$$\mathbf{z}_k = h(\mathbf{x}_k) + \mathbf{v}_k, \tag{3}$$

where $\mathbf{z}_k$ contains the available sensor observations and $\mathbf{v}_k \sim \mathcal{N}(0, \mathbf{R}_k)$ is zero-mean measurement noise.

In practice, the update step is multi-sensor and may include Time-of-Flight (ToF) range observations, depending on measurement validity and gating conditions.

For the ToF sensors, each range observation is modeled as

$$d_k^{(i)} = r^{(i)}(\mathbf{x}_k) + n_k^{(i)}, \quad i \in \{1, 2, 3\}, \tag{4}$$

where $r^{(i)}(\mathbf{x}_k)$ is the predicted distance from sensor i to the nearest arena wall along the corresponding ray and $n_k^{(i)}$ is measurement noise.

The geometric model assumes a square arena with known boundaries. For each ToF sensor, a ray is cast from the robot position in the sensor heading direction, determined by the robot yaw and the sensor mounting angle and the predicted range is taken as the distance to the first wall intersection. Missing returns and anomalous readings are handled by validity checks based on range limits, sensor status fields and innovation gating. This is important because some arena walls contain holes, which can produce erroneous ToF measurements.

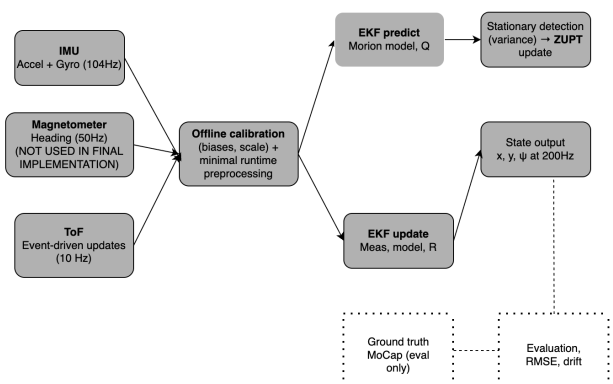

Fig. 1: Block diagram of the EKF sensor fusion pipeline.

B. Estimator Design

The Extended Kalman Filter (EKF) is used for state estimation. The prediction step is given by

$$\hat{\mathbf{x}}_{k|k-1} = f\left(\hat{\mathbf{x}}_{k-1|k-1}, \mathbf{u}_k, \Delta t\right), \tag{5}$$

$$\mathbf{P}_{k|k-1} = \mathbf{F}_k \mathbf{P}_{k-1|k-1} \mathbf{F}_k^\top + \mathbf{Q}_k, \tag{6}$$

where $\mathbf{F}_k$ is the Jacobian of the process model with respect to the state.

The update step is applied sequentially for each available sensor modality. For a generic measurement $\mathbf{z}_k$, the update equations are

$$\mathbf{S}_k = \mathbf{H}_k \mathbf{P}_{k|k-1} \mathbf{H}_k^\top + \mathbf{R}_k, \tag{7}$$

$$\mathbf{K}_k = \mathbf{P}_{k|k-1} \mathbf{H}_k^\top \mathbf{S}_k^{-1}, \tag{8}$$

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k\left(\mathbf{z}_k - h\left(\hat{\mathbf{x}}_{k|k-1}\right)\right), \tag{9}$$

$$\mathbf{P}_{k|k} = \left(\mathbf{I} - \mathbf{K}_k \mathbf{H}_k\right) \mathbf{P}_{k|k-1}, \tag{10}$$

where $\mathbf{H}_k$ is the Jacobian of the measurement model.

In the final implementation, only ToF range observations are used for correction, while magnetometer updates were omitted due to noise and inconsistency.

C. Calibration and Pre-processing

Calibration is performed offline prior to runtime execution. In the final implementation, sensor bias and scale parameters are extracted from dedicated calibration datasets and then hard-coded into the online EKF used inside Simulink.

a) Offline IMU Calibration: Two provided calibration datasets, `calib2_straight.mat` and `calib1_rotate.mat`, are used to estimate inertial sensor correction parameters. A separate calibration script, `extract_simulink_calibration_from_calib.m`, is used to compute:

- gyroscope bias,
- accelerometer bias, and
- a yaw-rate scale factor for the gyroscope.

These values are then embedded directly in the EKF implementation as fixed constants. This approach was adopted because the Simulink model interface does not provide a separate runtime calibration stage.

b) Bias-Corrected Prediction Inputs: At runtime, the gyroscope and accelerometer channels used by the motion model are corrected using the precomputed calibration constants. In particular, the yaw-rate input is bias-corrected and scaled, while the forward and lateral acceleration channels are bias-corrected before being transformed into the world frame.

c) Stationary Detection and ZUPT: A zero-velocity update (ZUPT) mechanism is used to reduce velocity drift. Stationarity is detected online using a sliding-window variance test on the gyroscope yaw-rate channel and the forward acceleration channel. When available, the low-pass filtered accelerometer signal is used for the stationarity test to improve robustness. The robot is treated as stationary when

$$\operatorname{var}(\omega) < 0.0005, \qquad \operatorname{var}(a) < 0.005. \tag{11}$$

When this condition is met, the velocity states are corrected towards zero.

d) ToF Measurement Handling: The ToF sensors operate at a lower effective rate than the EKF update loop. Rather than resampling, the implementation detects whether a new ToF reading has arrived by comparing the current value with the previous one. A ToF update is only applied when a new measurement is detected.

ToF measurements are also validated before use. A reading is rejected if it is non-finite, smaller than $0.02\,\mathrm{m}$, or larger than $2.5\,\mathrm{m}$. In addition, innovation gating is applied so that updates with residual magnitude above $0.4\,\mathrm{m}$ are discarded.

e) Motion-Adaptive ToF Weighting: To improve robustness during turning, the ToF measurement model is made less influential at high angular velocity. The heading component of the ToF Jacobian is attenuated as a function of yaw rate, and the effective ToF measurement noise is inflated during rapid rotation. This reduces unstable updates caused by linearisation errors when the robot is turning quickly.

f) Initialisation: At runtime, the EKF state is initialised with a fixed nominal starting pose. In the final implementation, the internal state is initialised to $x = 0.82\,\mathrm{m}$, $y = 0$, $\psi = 0$, with zero initial velocity. Ground-truth data are not used during runtime execution. This fixed initialisation was introduced to better match the deployment setup and improve early filter convergence.

Overall, the final design combines offline calibration for systematic IMU correction with lightweight online pre-processing for stationarity detection, ToF validation, and robust measurement gating.

D. Implementation Details

The estimator is implemented as an online Extended Kalman Filter (EKF) inside a Simulink MATLAB Function block. The filter processes one sample at a time and stores its internal state using persistent variables.

a) Runtime Structure: The deployed function has the fixed interface `myEKF(acc, gyro, mag, ToF1, ToF2, ToF3, Temp, LP_acc)`. At each call, the current sensor values are used to execute one EKF update step. A fixed sampling interval of $\Delta t = 0.005\,\mathrm{s}$ is assumed, corresponding to the $200\,\mathrm{Hz}$ Simulink time base.

b) Prediction and Update Order: Each iteration of the EKF follows the sequence:

1) prediction using gyroscope yaw rate and accelerometer-derived motion,
2) optional zero-velocity update (ZUPT) if stationarity is detected,
3) sequential ToF measurement updates for right, forward, and left sensors, when new valid readings are available.

This sequential update structure allows individual measurements to be gated or rejected independently.

c) Measurement Timing: The IMU-driven prediction runs at every function call, while ToF updates are event-driven. Since ToF readings are effectively held between updates, a new-sample detection mechanism is used to determine whether each ToF sensor has produced a fresh measurement. This avoids repeatedly applying the same held value.

d) Persistent State and Outputs: The EKF maintains its current state estimate, covariance, previous ToF readings, and stationary-detection buffers using persistent variables. The filter outputs the current state estimate and covariance at each time step.

To comply with the fixed Simulink model interface, the estimated state vector is padded with zeros before being returned. In addition, the reported position components are remapped to match the coordinate convention expected by the Simulink model, with the output position formed from a rotated version of the internal EKF state. These interface adjustments do not change the EKF computation itself, but ensure compatibility with the deployment model.

e) Offline Calibration Integration: Bias and scale calibration are not performed inside the runtime block. Instead, they are computed offline from calibration datasets and inserted into the MATLAB Function block as fixed constants. This allows the deployed EKF to remain lightweight while still using calibrated inertial inputs.

f) Development and Evaluation: A separate MATLAB development script is used outside Simulink for debugging, plotting, and RMSE evaluation against ground truth. This script loads logged `.mat` files, runs the EKF, and compares estimated position and yaw against the reference trajectory.

IV. Experimental Setup

A. Datasets and Evaluation Protocol

The estimator was developed using the datasets provided as part of the project, with separate files used for calibration and evaluation.

a) Parameter Calibration & Tuning: Estimator parameters, including process noise, measurement noise, innovation gating thresholds, stationary-detection thresholds, and velocity decay, were tuned empirically during development using the provided task datasets. Once satisfactory behaviour was achieved, the same parameter set was retained for all subsequent evaluation runs.

b) Task 1 Evaluation: Task 1 performance was evaluated using the four provided Task 1 files: `task1_1`, `task1_2`, `task1_3`, and `task1_4`. These datasets contain straight-line forward and backward motion and were used to assess estimator stability, drift behaviour, and consistency under relatively simple motion.

c) Task 2 Evaluation: Task 2 performance was evaluated using the four provided Task 2 files: `task2_1`, `task2_2`, `task2_3`, and `task2_4`. These datasets contain more complex trajectories with turning and varying motion, and were used to assess robustness under more challenging dynamic conditions.

d) Initialisation Strategy: At runtime, the EKF state is initialised to a fixed nominal pose, with position approximately set to $(x, y) = (0.82, 0)$ and zero initial velocity and heading. Ground-truth measurements are not used by the deployed estimator for initialisation. Instead, convergence is achieved through the combination of calibrated IMU prediction, zero-velocity updates during stationary periods, and sequential ToF corrections.

e) Evaluation Method: For development and analysis, the estimator outputs were compared against the provided ground-truth position and rotation signals. Performance was assessed using trajectory plots and root-mean-square error (RMSE) in position and yaw. Ground truth was used only for evaluation and not as an input to the final online estimator.

B. Performance Metrics

The position RMSE over a trajectory of length N is defined as

$$\mathrm{RMSE}_{\mathrm{pos}} = \sqrt{\frac{1}{N}\sum_{k=1}^{N}\left[(\hat{x}_k - x_k^{\mathrm{gt}})^2 + (\hat{y}_k - y_k^{\mathrm{gt}})^2\right]}, \quad (12)$$

which captures the overall spatial deviation of the estimated trajectory.

In addition, axis-wise RMSE values are computed for further insight:

$$\mathrm{RMSE}_x = \sqrt{\frac{1}{N}\sum_{k=1}^{N}(\hat{x}_k - x_k^{\mathrm{gt}})^2} \tag{13}$$

$$\mathrm{RMSE}_y = \sqrt{\frac{1}{N}\sum_{k=1}^{N}(\hat{y}_k - y_k^{\mathrm{gt}})^2}. \tag{14}$$

Yaw estimation accuracy is evaluated using a wrapped angular error to account for periodicity:

$$\mathrm{RMSE}_\psi = \sqrt{\frac{1}{N}\sum_{k=1}^{N}\left(\hat{\psi}_k - \psi_k^{\mathrm{gt}}\right)^2}, \tag{15}$$

In addition to RMSE, qualitative evaluation is performed using trajectory plots comparing estimated and ground-truth paths in the (x, y) plane.

V. Results

A. Quantitative Results

TABLE II: Estimator Performance

Dataset	Pos RMSE (m)	Yaw RMSE (deg)	Final Error (m)
Task 1_1 1	0.027	3.06	0.020
Task 1_2 1	0.022	5.63	0.022
Task 1_3	0.025	6.41	0.025
Task 1_4	0.022	6.86	0.022
Task 2_1 1	0.044	4.94	0.013
Task 2_2 1	0.100	7.77	0.214
Task 2_3 1	0.036	5.50	0.011
Task 2_4	0.287	8.95	0.007

Table II summarises the results for position RMSE(m), yaw RMSE(deg), and final error(m).

B. Trajectory and Time-Series Plots

Figures 2 and 3 show planar trajectories of the EKF and the ground truth for tasks 1.1 and 2.3 respectively. Figures 4 and 5 show the corresponding time-series data for x-position, y-position, and heading (yaw) for tasks 1.1 and 2.3.

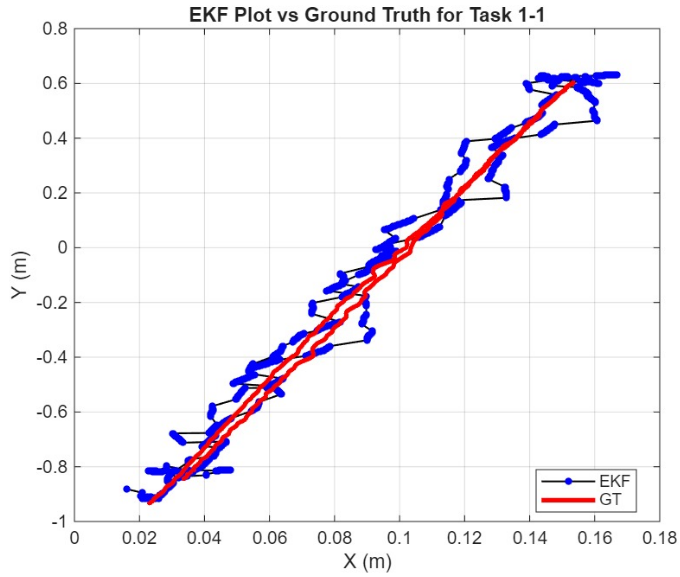

Fig. 2: Task 1.1 trajectory

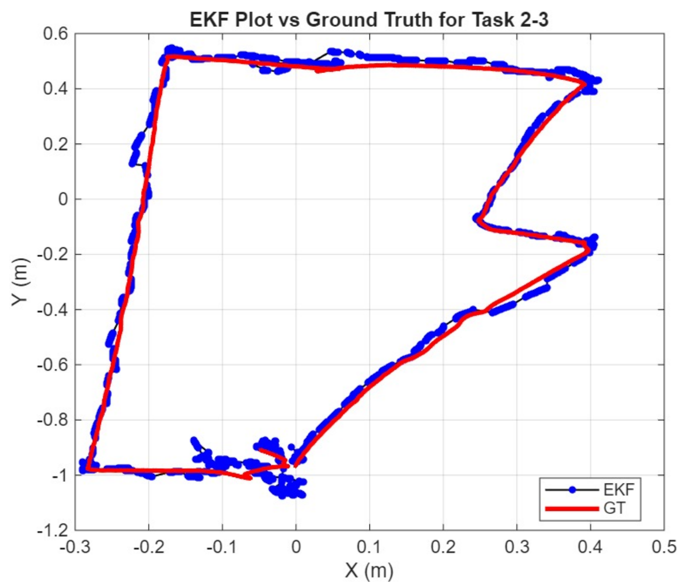

Fig. 3: Task 2.3 trajectory

C. Task-by-Task Analysis

a) Task 1: Straight-Line Motion: The estimator performs consistently well across all Task 1 datasets, achieving low position and yaw RMSE values. This behaviour is expected given the relatively simple motion profile, which consists primarily of straight-line forward and backward movement with minimal rotational dynamics.

In this regime, the prediction model is highly effective. The gyroscope provides stable yaw estimates, and the velocity states remain well-behaved due to the exponential velocity decay term in the motion model. As a result, drift is naturally limited even without frequent corrective updates.

Time-of-Flight (ToF) measurements further improve accuracy by correcting small position errors when valid measurements are available. Since the robot orientation remains approximately constant, the ToF measurement model is well-conditioned and the linearisation remains accurate.

Overall, the dominant factors contributing to strong performance in Task 1 are:

- stable yaw estimation from the gyroscope,
- low model mismatch due to simple motion,
- reliable ToF corrections with minimal geometric ambiguity.

b) Task 2: Complex Trajectories: Task 2 introduces a more challenging motion, including frequent turns and varying velocities. This leads to increased estimation error compared to Task 1, as reflected in the higher RMSE values for some datasets (notably Task 2_2 and Task 2_4).

A key difficulty arises from the increased sensitivity of the

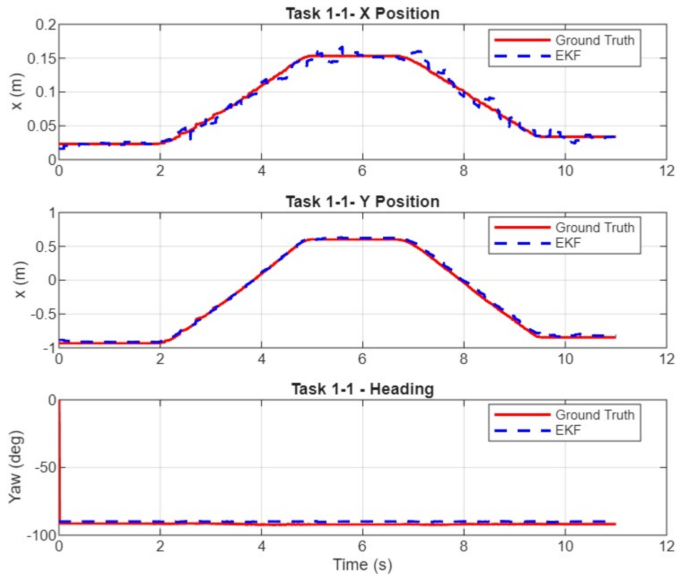

Fig. 4: Time-series x, y, yaw for task 1.1

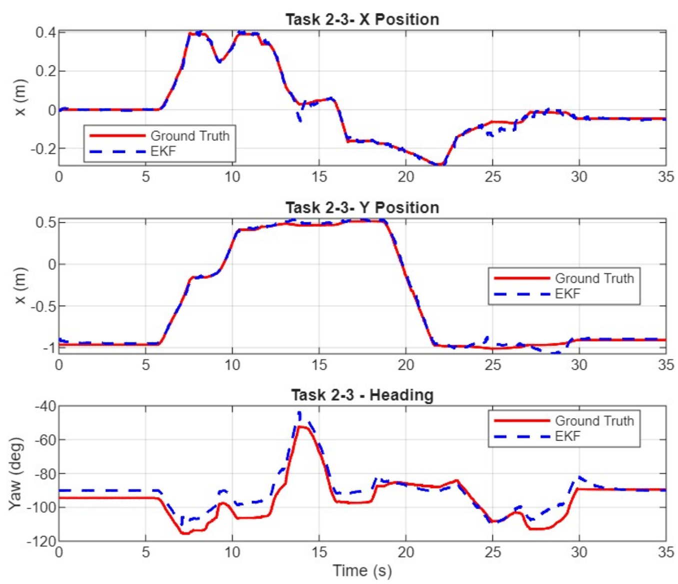

Fig. 5: Time-series x, y, yaw for task 2.3

ToF measurement model during rotation. When the robot turns rapidly, the linearisation of the wall-intersection model with respect to heading becomes less accurate. This can lead to unstable updates if not properly managed. To mitigate this, the implementation decreases the heading component of the ToF Jacobian and increases the measurement noise during high angular velocity. This improves robustness but reduces the corrective strength of ToF updates during turning.

The most important improvement for Task 2 performance is the inclusion of the zero-velocity update (ZUPT). During stationary or near-stationary periods, detected via variance thresholds on gyroscope and accelerometer signals, the velocity states are corrected towards zero. This significantly reduces drift accumulated during motion and stabilises the estimator, particularly after turns or during pauses.

Without ZUPT, velocity drift propagates into position error, leading to large trajectory deviations. With ZUPT enabled, the estimator is able to repeatedly re-anchor the velocity estimate.

c) Sensor Contributions and Failure Modes: Across both tasks, different sensors dominate in different regimes:

- **Gyroscope:** critical for short-term heading propagation; errors accumulate slowly due to bias and scale imperfections.
- **Accelerometer:** contributes weakly to motion prediction due to strong damping, primarily aiding smooth propagation rather than absolute accuracy.
- **ToF sensors:** provide absolute position correction relative to arena walls; most effective during straight motion and less reliable during rapid rotation.

The largest errors occur in Task 2 during sharp turns and in segments with reduced ToF reliability (e.g. near wall discontinuities or when measurements are rejected by gating). In these cases, the estimator relies more heavily on IMU prediction, which leads to temporary drift until corrective measurements or ZUPT updates are applied.

VI. Discussion

The results demonstrate that the proposed EKF achieves reliable localisation performance across both simple and complex trajectories, but also reveals several important trade-offs inherent to the design.

a) Sensitivity to Bias and Drift: The estimator remains sensitive to residual IMU bias and modelling errors. Although gyroscope and accelerometer biases are estimated offline and hard-coded, any mismatch between calibration conditions and runtime behaviour introduces drift, particularly in heading. Since the motion model relies heavily on gyroscope integration, small yaw errors propagate into position error through the coordinate transformation. The use of velocity damping and ZUPT reduces this effect, but does not fully eliminate long-term drift during extended motion without correction.

b) Robustness to ToF Failures: The ToF sensors provide essential absolute position corrections, but are inherently unreliable due to wall holes, missing returns, and measurement noise. Robustness is achieved through multiple mechanisms, including range validation, innovation gating, and motion-adaptive measurement weighting. In particular, rejecting measurements outside $[0.02, 2.5]$ m and applying a gating threshold prevents large outliers from destabilising the filter. However, this also reduces the number of accepted updates, meaning that during periods of poor ToF visibility the estimator relies more heavily on IMU prediction.

c) Effect of Filter Tuning and Model Mismatch: Filter performance is strongly dependent on tuning parameters such as process noise, measurement noise, gating thresholds and velocity decay. The chosen tuning reflects a compromise between responsiveness and stability. Model mismatch is particularly evident during rapid rotations, where the linearised ToF measurement model becomes less accurate. The implemented attenuation of the heading Jacobian and inflation of measurement noise solves this issue, but the correction strength is reduced.

d) State Representation: The chosen state vector, consisting of position, heading, and velocity, provides a good balance between model expressiveness and computational simplicity. Including velocity states enables smooth propagation between sparse ToF updates and supports the ZUPT mechanism. However, the decision not to estimate sensor biases online limits the filter's ability to adapt to changing conditions. This increases reliance on accurate offline calibration.

e) Computational Efficiency and Real-Time Suitability: The estimator is computationally lightweight and well-suited for real-time deployment. The EKF operates at a fixed 200 Hz update rate with a small state dimension, and uses simple matrix operations and sequential measurement updates. The use of persistent variables avoids unnecessary memory allocation, and the event-driven handling of ToF measurements reduces unnecessary computation.

f) Failure Cases: The largest estimation errors occur during rapid turns combined with unreliable ToF measurements. In these situations, the filter relies primarily on IMU prediction, and heading errors lead to noticeable position drift. Additional failures can occur when ToF measurements are repeatedly rejected due to gating or invalid readings, resulting in temporary loss of absolute correction. Task 2.4 illustrates this behaviour, where increased trajectory complexity and reduced measurement reliability lead to higher position RMSE.

g) Future Improvements: Several improvements could further enhance performance. Incorporating online bias estimation would allow the filter to adapt to changing sensor characteristics and reduce drift. A more accurate motion model, for example including angular velocity dynamics or slip effects, could improve prediction during aggressive manoeuvres. Additionally, incorporating more robust measurement models, such as probabilistic ray-casting or multi-hypothesis updates, could improve resilience to ToF failures. Also, replacing the EKF with a nonlinear filtering approach such as an Unscented Kalman Filter (UKF) may improve performance under strong nonlinearities, particularly during rapid rotations.

VII. Conclusion

This project developed an Extended Kalman Filter (EKF) for real-time localisation of a mobile robot in a bounded indoor arena using inertial and Time-of-Flight (ToF) sensors. The estimator uses a five-dimensional state comprising position, heading and velocity, with prediction driven primarily by gyroscope integration and weakly by accelerometer inputs. Robustness is achieved through measurement gating, motion-adaptive ToF updates and a zero-velocity update (ZUPT) mechanism based on online stationary detection. Sensor biases are estimated offline from calibration datasets and hard-coded into the model.

The estimator performs well in simple motion scenarios, achieving low position and yaw errors in Task 1. In more complex trajectories (Task 2), performance degrades due to increased sensitivity to heading errors and reduced reliability of ToF measurements during rapid rotations. However, the inclusion of ZUPT and adaptive measurement handling significantly improves stability and limits drift, enabling the estimator to maintain reasonable accuracy across all datasets.

Future improvements would focus on reducing reliance on offline calibration by incorporating online bias estimation, improving the motion model to better capture dynamic behaviour, and enhancing robustness to ToF failures. More advanced nonlinear filtering techniques, such as the Unscented Kalman Filter, could also be explored to better handle strong nonlinearities in the measurement model. Overall, the final system demonstrates an effective balance between accuracy, robustness, and computational efficiency for real-time deployment.