ECE 276A Project 3: Visual-Inertial SLAM

Unay Shah **PID:** A59015777

Keywords: SLAM, visual inertial, landmarks, extended kalman filter, localization, feature detection, feature matching, robotic estimation

Abstract:

Simultaneous localization and mapping (SLAM) is a way to map an area while simultaneously tracking an agent that is traversing the area. Visual-inertial SLAM refers to a way to correctly localize a robot in the world based on visual cues and features extracted from its surroundings, while mapping the area. We are provided with features extracted from images taken by a camera mounted on a car. We also have the associated IMU data from the car. We combine these data using Extended Kalman Filter (EKF) to iteratively improve the mapping and localize the car.

Introduction

Simultaneous localization and mapping (SLAM) is used to map unknown environments using a robot carrying various sensors, continuously collecting data of its surroundings. These data points need to be processed in order to create a clean and accurate map. The robot might face some disturbances or the sensor may collect noisy data, which needs to be cleaned. This is where visual inertial SLAM along with various other techniques used in this project are useful. We have IMU and stereo image data from a car. Features have been extracted from the images and have been provided with corresponding angular and linear velocity data.

We start by creating a dead reckoning trajectory of the car using just the IMU data. We then start by improving on the location of the features using EFK. We use the observation model to get the deviation between predicted feature locations from the IMU and actual data captured by the camera to correct the location of features. We proceed with using these corrected feature mappings to get deviations in the car's trajectory and try to correct the trajectory.

Our robot in this scenario is a car that is being driven on public roads and has a GPS/IMU, a Velodyne LiDAR sensor, perspective cameras and a fisheye stereo camera setup. The data provided is synchronized beforehand.

Problem Formulation

IMU Localization

The time varying pose can be modelled as $\dot{T}(t) = T(t)\hat{\xi}(t)$, where T is the pose of the robot at time t, ζ is the twist denoted as $\zeta = \begin{bmatrix} v(t) \\ \omega(t) \end{bmatrix}$ and $\hat{\zeta} = \begin{bmatrix} \hat{v} & \omega \\ \mathbf{0} & 0 \end{bmatrix}$. For discrete cases, the pose kinematics can be modelled as:

$$T_{t+1} = T_t \exp(\tau_t \hat{\zeta}_t)$$

Where τ_t is the time difference between the next and current timestamp measurement. We assume that the pose does not change during this duration.

The camera calibration matrix for a stereo camera is denoted as

$$K_s = \begin{bmatrix} fs_u & 0 & c_u & 0 \\ 0 & fs_v & c_v & 0 \\ fs_u & 0 & c_u & -fs_u b \\ 0 & fs_v & c_v & 0 \end{bmatrix} = \begin{bmatrix} u_L \\ v_L \\ u_R \\ v_R \end{bmatrix}$$

Which can be simplified as

$$\begin{bmatrix} u_L \\ v_L \\ d \end{bmatrix} = \begin{bmatrix} fs_u & 0 & c_u & 0 \\ 0 & fs_v & c_v & 0 \\ 0 & 0 & 0 & fs_u b \end{bmatrix}$$

Where K_s is the camera calibration matrix, f is the focal length, (s_u, s_v) denote the pixel scaling, (c_u, c_v) denote the principal point used to translate the image frame origin, b is the baseline distance between the two cameras (it is assumed that they are aligned horizontally with the same z-coordinates in world frame) and $(u_L, v_L), (u_R, v_R)$ denote the pixel coordinates in the left and right camera.

Landmark Mapping

Data association has already been carried out, which can be denoted by $\Delta_t: \{1, \dots, M\} \to \{1, \dots, N\}$. The observation model with measurement noise $N \sim (0, V)$ is created as

$$z_{t,i} = h(T_t, m_j) + v_{t,i} \coloneqq K_s \pi\left({}_O T_I T_t^{-1} \underline{m_j}\right) + v_{t,i}$$

Where π is the projection function defined as

$$\pi(\boldsymbol{q}) = \frac{1}{q_3}\boldsymbol{q} \quad \frac{d\pi}{d\boldsymbol{q}}(\boldsymbol{q}) = \frac{1}{q_3}\begin{bmatrix} 1 & 0 & -\frac{q_1}{q_3} & 0 \\ 0 & 1 & -\frac{q_2}{q_3} & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & -\frac{q_4}{q_3} & 1 \end{bmatrix}$$

And the noise term is defined as $I \otimes V = \begin{bmatrix} V & & \\ & \ddots & \\ & & V \end{bmatrix}$

The EKF update step is defined using μ mean, Σ variance, K Kalman gain and H observation model Jacobian, where all these terms are calculated using the previous observations of these terms.

$$\tilde{z}_{t+1,i} = K_s \pi\left({}_O T_I T_t^{-1} \underline{\mu_{t,j}}\right)$$

$$H_{t+1,i,j} = \begin{cases} K_s \frac{d\pi}{dq}\left({}_O T_I T_t^{-1} \underline{\mu_{t,j}}\right) {}_O T_I T_t^{-1} P^T, & if\ \Delta_t(j) = i \\ 0, & else \end{cases}$$

$$P = [\boldsymbol{I} \quad 0\,] \epsilon \mathcal{R}^{3\times 4}$$

$$K_{t+1} = \Sigma_t H_{t+1}^T (H_{t+1} \Sigma_t H_{t+1}^T + I \otimes V)^{-1}$$

$$\mu_{t+1} = \mu_t + K_{t+1}(z_{t+1} - \tilde{z}_{t+1})$$

$$\Sigma_{t+1} = (I - K_{t+1} H_{t+1}) \Sigma_t$$

We try to estimate and correct the location of the feature points using the predicted positions and observed positions, trying to obtain μ that is close to the actual position of the landmarks in the world frame.

Visual Inertial SLAM

Having obtained an algorithm to create a trajectory from IMU data and to locate landmarks using the trajectory and image data, we try to correct the robot trajectory combining both these algorithms. We have the pose in $SE(3)$ state which needs to be converted to a Gaussian distribution to use EKF. Hence, to obtain a motion model for EKF, we perturb pose kinematics equations to obtain $T \approx \mu(I + \widehat{\delta\mu})$which consists of 2 terms:

Nominal: $\dot{\mu} = \mu\hat{u}$ Perturbation: $\dot{\delta\mu} = -\widehat{u'}\delta\mu + w$

Where $\hat{u}' = \begin{bmatrix} \hat{\omega} & \hat{v} \\ 0 & \hat{\omega} \end{bmatrix} \epsilon \Re^{6\times 6}$

The prediction step then requires computing $\mu_{t+1|t}$ and $\Sigma_{t+1|t}$.

$$\mu_{t+1|t} = \mu_{t|t} \exp(\tau_t \hat{u}_t)$$

$$\Sigma_{t+1|t} = \exp(-\tau_t \hat{u}_t') \Sigma_{t|t} \mu_{t|t} \exp(-\tau_t \hat{u}_t')^T + W$$

The update step with pose perturbation $\delta\mu$ is:

$$z_{t+1,i} \approx K_s \pi\left({}_O T_I T_t^{-1} \underline{m_j}\right) - K_s \frac{d\pi}{dq}\left({}_O T_I T_t^{-1} \underline{m_j}\right) {}_O T_I \left(\mu_{t+1|t}^{-1} \underline{m_j}\right) \odot \delta\mu + v_{t+1,i}$$

This new perturbed motion model results in

$$H_{t+1,i} = -K_s \frac{d\pi}{dq}\left({}_O T_I T_t^{-1} \underline{m_j}\right) {}_O T_I \left(\mu_{t+1|t}^{-1} \underline{m_j}\right) \odot$$

While rest of the calculations remain the same as the landmark mapping step.

Technical Approach

EKF is a type of Kalman Filter algorithm that approximates a function to a linear equation in the state and noise variables and models it to improve the state and observed values. The noise is modelled to be Gaussian noise. It starts by using the motion model to get the current state of the robot. This refers to finding the position of the car using its velocities in this case. Some noise is added to the state to add some variation in the motion model. We calculate the coefficients of the state and noise in the modelled Kalman filter, denoted by $F_t = \frac{df}{dx}(\mu_{t|t}, u_t, 0)$ and $Q_t = \frac{df}{dw}(\mu_{t|t}, u_t, 0)$. We can thus find the mean ($\mu_{t+1|t}$) and variance ($\Sigma_{t+1|t}$) of the state at current timestamp. Being the prediction step, there are no changes in mean and variance, but there is a new state that is calculated for this iteration.

This is followed by the update step, which makes use of the observation model. In this case, the observation model is obtained by the camera images. These images are processed by feature detection algorithms, which identify important and similar features in the left and right camera, returning their pixel coordinates. Pixel coordinate data along with location and trajectory data is used to find the deviation between values from observation model and motion model. This difference when multiplied by

the Kalman gain gives the corrected mean at the current timestamp. The observation model is also modelled as a linear equation with $H_t = \frac{dz}{dx}(\mu_{t|t-1}, 0)$ and $R_t = \frac{dz}{dv}(\mu_{t|t-1}, 0)$.

To obtain features in an image, an algorithm like SIFT (Scale Invariant Feature Transform) can be used. This also has an associated mapping algorithm, where it can identify closely related objects given multiple images.

IMU Localization

First, we find the dead reckoning trajectory of the car, We start by rotating the IMU frame to correct the orientation by rotating around the x axis, followed by creating the twist matrix using the linear and angular velocities. After stacking the data, the adjoint is calculated to get the $\hat{\zeta}$ matrix. Using the difference in consecutive timestamps and twist matrix, we can get the dead reckoning trajectory of the car. Along with this, we can project the feature coordinates transformed into the world frame from the optical frame to plot landmark locations with the trajectory.

Landmark Mapping

After obtaining the trajectory, we can use these values to predict location of landmarks using EKF. We are assuming that the trajectory obtained is accurate for the time being. First we identify and separate the landmarks that are seen at the current timestamp. We track the indexes of the ones that have already been seen before and those that are new. We map the new features into the world frame from the pixel values using the trajectory at the current timestamp. Using the previously stored world frame coordinates, we use the indexes that are visible at both previous and current timestamps. From the world frame coordinates, we project them into the optical frame and obtain pixel coordinates. These pixel coordinates are compared with the actual pixel coordinates from the feature data. This gives us the error in the data. This is then multiplied with the Kalman gain and we obtain the required change in pixel values that needs to be done to correct the feature position. We calculate the covariance at the current timestamp.

Visual Inertial SLAM

The visual inertial SLAM step follows a similar approach to landmark mapping, but now we update the cars pose using corrected values from previous landmark data. We start by calculating the pose (nominal) at the current timestamp, using the twist and previous timestamps, followed by the perturbation using the noise and adjoint of twist. We use the motion model to find the equations for these, using the mean from the previous iteration to predict these values.

This is followed by the observation model. This is calculated only for the features that have been seen in both, previous and current iterations to find the error and correct it. The predicted trajectory is used to predict position of features in the world frame, and then these features are projected into pixel coordinates. They are compared with the feature pixel coordinates at current timestamp. This comparison is between features observed and extrapolated into the current step and the actual pixel obtained. This difference helps correct the mean and covariance at each iteration.

In this step, we also correct the pose of the car along with landmark position. The mean and covariance of the car is stored along with those of the landmarks, and updated in each iteration. For EKF, some noise is added to perturb the variance and Kalman gain at the start and end of each iteration respectively. In an ideal situation, both the car's IMU sensor and camera features work in unison to provide a corrected path and more accurate mapping of landmarks in the world.

Results

From IMU Localization, we can see the approximate path that the car takes and the position of landmarks around its path.

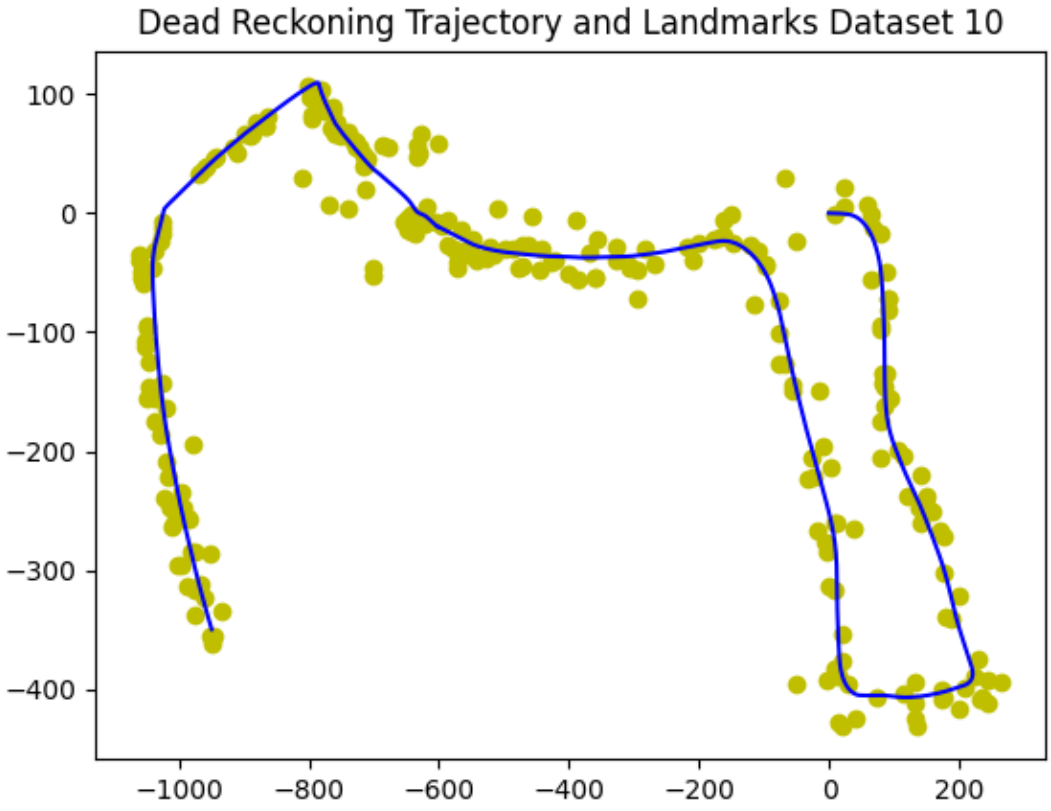
We assume that the trajectory is correct and try to reduce error between the landmarks positions

obtained from previous pose and that from the current pose.

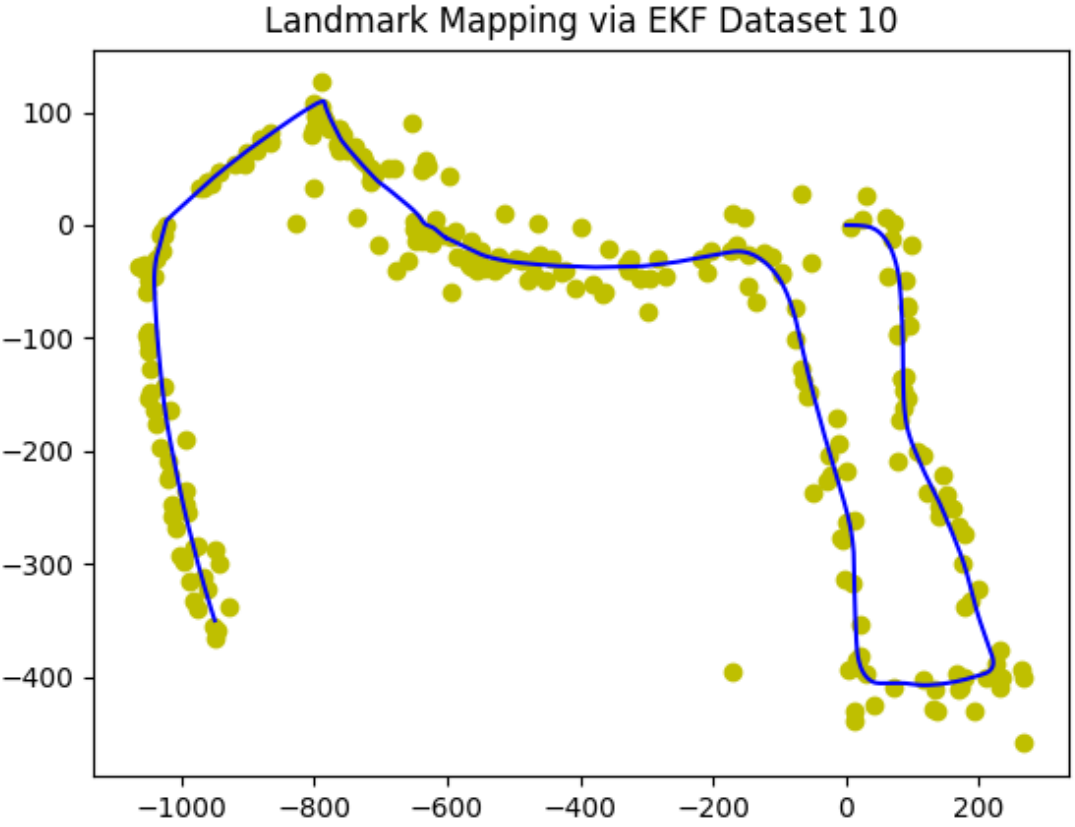
In the last step of SLAM, we try to correct both the car's trajectory and landmarks positions using the values obtained from the other

Dataset 10

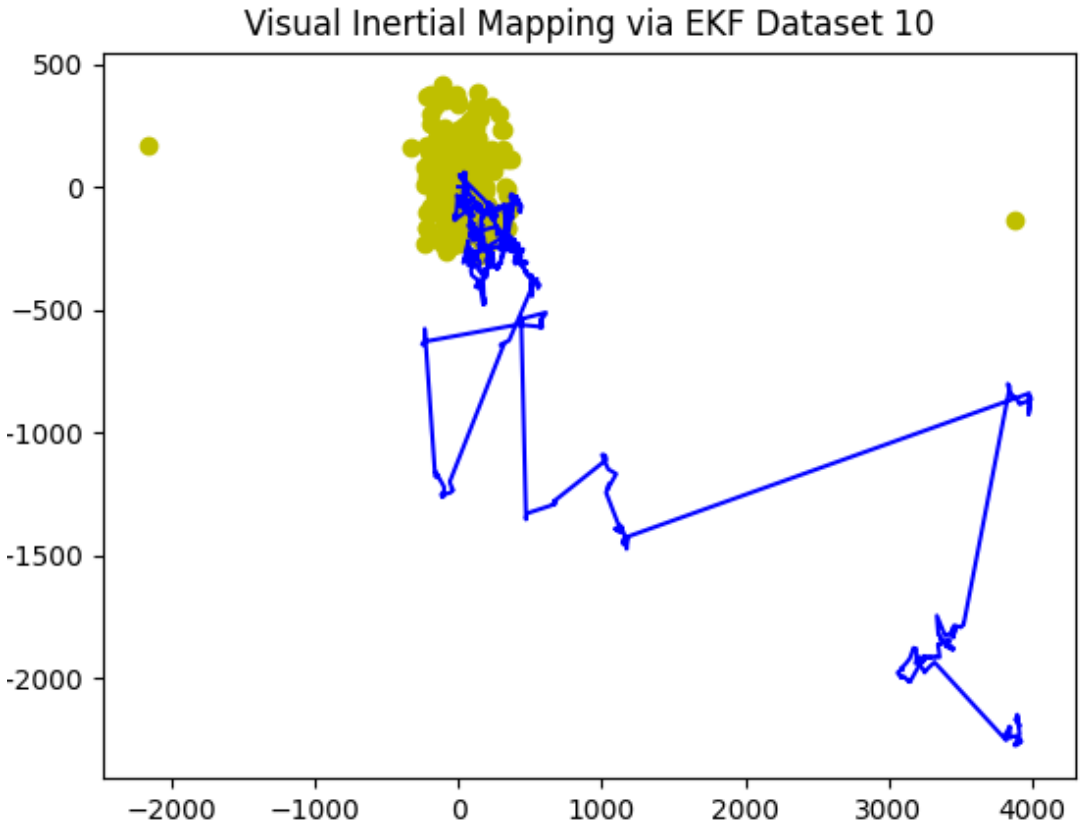
Dead Reckoning



Landmark Mapping with EKF

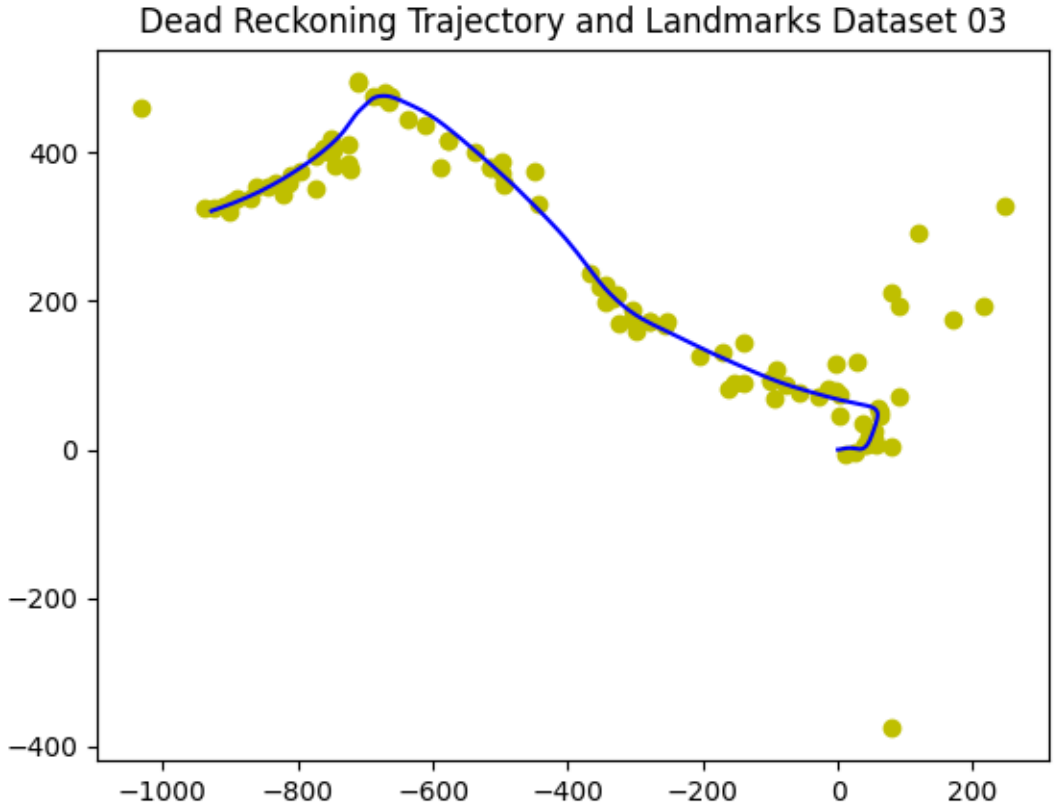


Visual Inertial SLAM:

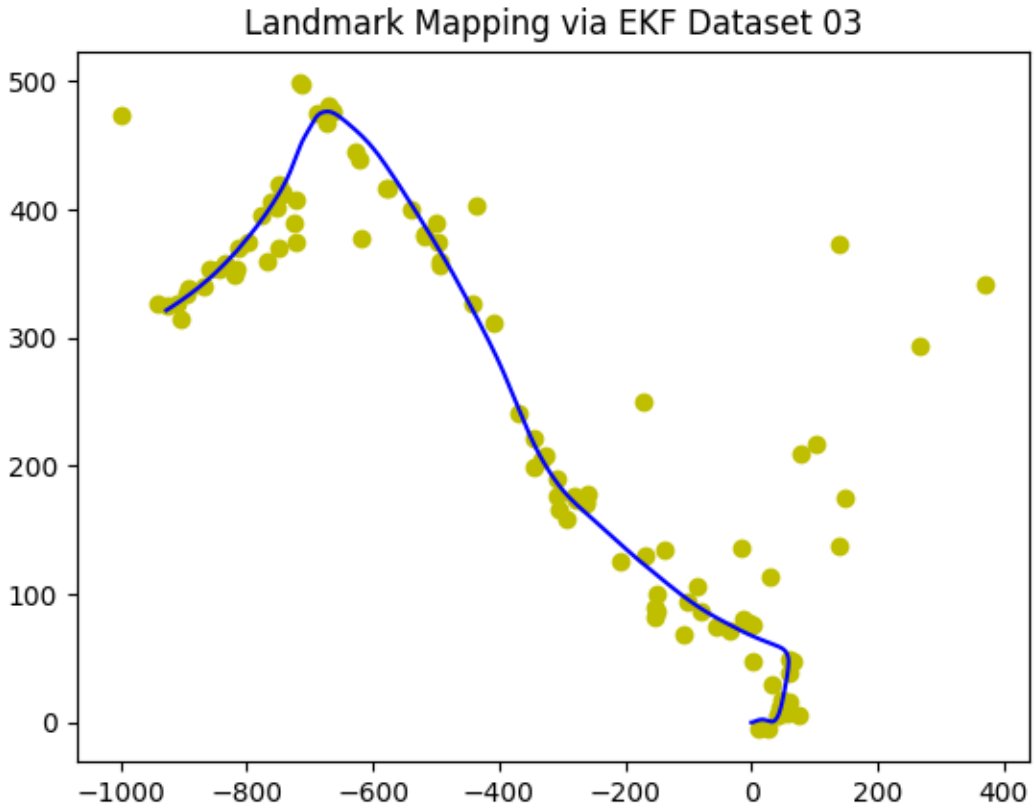


Dataset 11

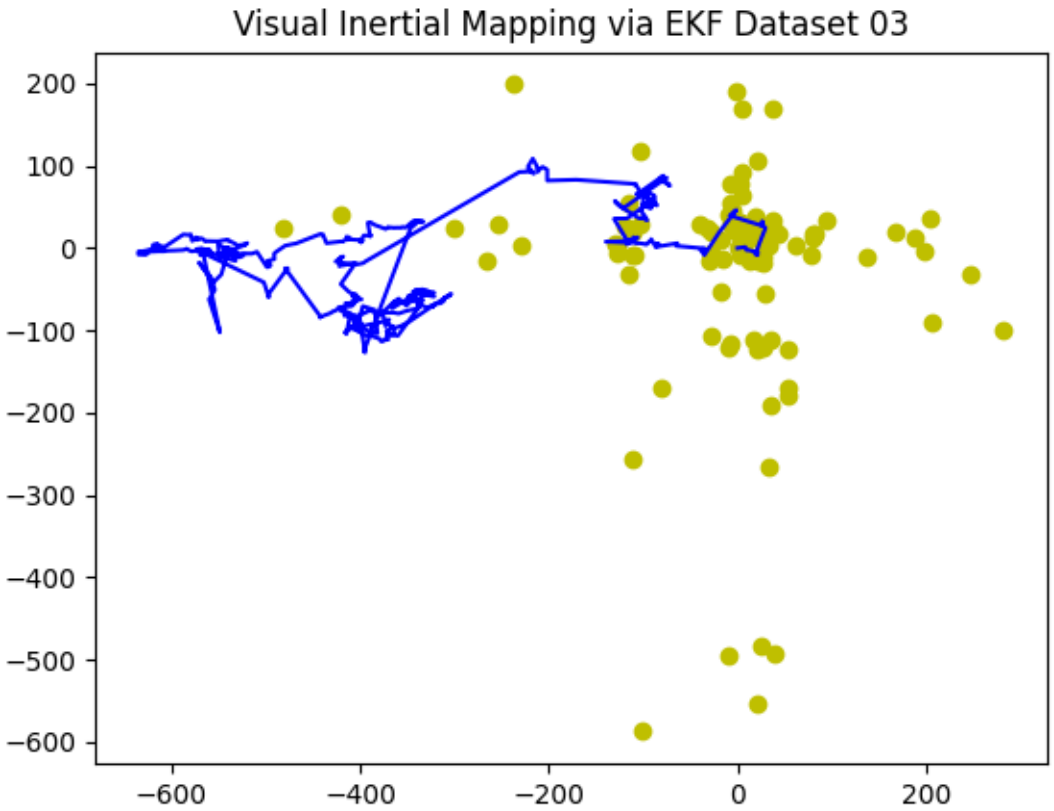
Dead Reckoning



Landmark Mapping with EKF



Visual Inertial SLAM:



I was able to obtain results for IMU Localization and Landmark mapping, but could not correctly program SLAM. There is an issue in the update step where I was unable to correctly update Σ and μ values for correcting the trajectory.