Visual-Inertial SLAM

Varun Sankar Moparthi
University of California San Diego
vmoparthi@ucsd.edu

Nikolay Antasnov
University of California San Diego
natanasov@ucsd.edu

***Abstract*—This paper presents a Visual-Inertial Simultaneous Localization and Mapping (VI-SLAM) system that integrates an Extended Kalman Filter (EKF) for state estimation and landmark mapping. The approach fuses inertial measurement unit (IMU) data with stereo camera observations to achieve robust localization in dynamic environments. The EKF prediction step employs SE(3) kinematics to propagate the robot's pose using IMU-derived velocity measurements, while the update step refines the pose based on visual feature observations. Feature detection, stereo matching, and temporal tracking techniques are applied to extract reliable correspondences. To improve computational efficiency and mapping accuracy, unreliable feature points are filtered, and noise parameters are optimized. The proposed method is validated using real-world data from Clearpath Jackal robots operating in an outdoor environment. Experimental results demonstrate accurate trajectory estimation and precise landmark localization, underscoring the effectiveness of IMU-visual fusion in enhancing SLAM performance.**

I. Introduction

Simultaneous Localization and Mapping (SLAM) is a fundamental problem in robotics, particularly for autonomous systems navigating unknown and dynamic environments. The challenge lies in a robot's ability to simultaneously localize itself within its environment while building a map of the surroundings, all without relying on external infrastructure. Traditional visual SLAM systems rely heavily on camera-based observations, which can struggle in environments with poor lighting, textureless regions, or dynamic objects. In contrast, inertial measurement units (IMUs) provide high-frequency motion data, but by themselves, they often suffer from cumulative drift over time.

To address these challenges, the integration of visual and inertial data has become a key approach in enhancing the robustness and accuracy of SLAM systems. Visual-Inertial SLAM (VI-SLAM) combines the strengths of both sensors: the visual system offers rich environmental features for map construction, while the IMU ensures continuous state estimation even in the absence of visual landmarks. However, the fusion of these data streams requires careful handling of sensor noise, temporal inconsistencies, and the non-linear nature of motion.

This paper proposes a VI-SLAM approach based on an Extended Kalman Filter (EKF), designed to merge IMU data and stereo camera observations for accurate localization and mapping. The proposed system utilizes SE(3) kinematics to propagate the robot's pose using IMU measurements, while visual features tracked across stereo camera frames refine the estimation of the robot's trajectory and landmarks. A key contribution of this work is the development of an efficient pipeline for feature detection, matching, and temporal tracking, which ensures the reliable extraction of visual data despite environmental challenges.

The effectiveness of the proposed VI-SLAM system is demonstrated through a set of experiments conducted using real-world data collected from Clearpath Jackal robots navigating dynamic environments. These experiments reveal the potential of sensor fusion in improving localization accuracy and mapping precision, addressing the inherent limitations of both visual and inertial sensing in isolation. The results indicate that the integration of IMU and stereo camera data offers a significant advantage in building more reliable and accurate SLAM systems for autonomous navigation.

Identify applicable funding agency here. If none, delete this.

II. Problem Formulation

A. IMU localization via EKF prediction

The goal of IMU localization is to estimate the pose of an Inertial Measurement Unit (IMU) over time. The IMU provides measurements of linear velocity $\mathbf{v}_t \in \mathbb{R}^3$ and angular velocity $\boldsymbol{\omega}_t \in \mathbb{R}^3$ at discrete time steps. The pose of the IMU, denoted as $\mathbf{T}_t \in \mathrm{SE}(3)$, consists of both the position and orientation of the IMU in a three-dimensional space. It is represented by a 4×4 matrix as:

$$\mathbf{T}_t = \begin{bmatrix} \mathbf{R}_t & \mathbf{p}_t \\ \mathbf{0}^\top & 1 \end{bmatrix}, \tag{1}$$

where $\mathbf{R}_t \in \mathrm{SO}(3)$ represents the rotation matrix and $\mathbf{p}_t \in \mathbb{R}^3$ represents the position vector of the IMU at time t.

The system dynamics are governed by the kinematic equations of rigid body motion:

$$\dot{\mathbf{p}}_t = \mathbf{R}_t \mathbf{v}_t, \tag{2}$$

$$\dot{\mathbf{R}}_t = \mathbf{R}_t \left[\boldsymbol{\omega}_t\right]_\times, \tag{3}$$

where $\left[\boldsymbol{\omega}_t\right]_\times$ is the skew-symmetric matrix of the angular velocity vector $\boldsymbol{\omega}_t$.

At each time step, the task is to estimate the state $\mathbf{x}_t = [\mathbf{p}_t, \mathbf{R}_t]^\top$, which is the pose of the IMU. The goal is to propagate this state over time using the linear and angular velocity measurements. The prediction step is typically carried out using a recursive estimation technique, such as the Extended Kalman Filter (EKF), to account for the uncertainty in the system dynamics and the sensor measurements.

Given:

- IMU measurements: $\mathbf{v}_t$ (linear velocity), $\boldsymbol{\omega}_t$ (angular velocity) at time t,
- State vector: $\mathbf{x}_t = [\mathbf{p}_t, \mathbf{R}_t]^\top \in \mathbb{R}^6$, representing the position and orientation of the IMU,
- Process noise covariance: $\mathbf{Q}_t$,

the task is to predict the state $\mathbf{x}_{t+1}$ of the IMU at the next time step using the system's dynamics. The state update is based on the kinematic model of the IMU and the provided velocity measurements.

B. Feature detection and matching

Given a sequence of stereo image pairs, the goal is to detect and track visual features across both the left and right camera frames over time. Let t denote the timestep, and M denote the total number of features detected across all images. For each timestep t, we construct a matrix $\mathbf{z}_t \in \mathbb{R}^{4\times M}$, where each column corresponds to a specific feature. The matrix $\mathbf{z}_t$ is defined as:

$$\mathbf{z}_t = \begin{bmatrix} l_{x,1} & l_{x,2} & l_{x,3} & \dots & l_{x,M} \\ l_{y,1} & l_{y,2} & l_{y,3} & \dots & l_{y,M} \\ r_{x,1} & r_{x,2} & r_{x,3} & \dots & r_{x,M} \\ r_{y,1} & r_{y,2} & r_{y,3} & \dots & r_{y,M} \end{bmatrix},$$

where:

- $l_{x,j}$ and $l_{y,j}$ are the x and y pixel coordinates, respectively, of the j-th feature in the left camera frame at timestep t.
- $r_{x,j}$ and $r_{y,j}$ are the x and y pixel coordinates, respectively, of the j-th feature in the right camera frame at timestep t.

If the j-th feature is not present in the t-th frame, the corresponding entries are set to -1:

$$l_{x,j} = l_{y,j} = r_{x,j} = r_{y,j} = -1.$$

C. Landmark Mapping via EKF Update

The objective of this problem is to estimate the positions of static landmarks $\mathbf{m} \in \mathbb{R}^{3\times M}$ observed in visual measurements using an Extended Kalman Filter (EKF). The predicted trajectory of the Inertial Measurement Unit (IMU) is assumed to be correct, and the focus is on estimating the landmark positions efficiently while maintaining computational feasibility.

Let the landmark positions be represented as:

$$\mathbf{m} = \begin{bmatrix} m_1 & m_2 & \dots & m_M \end{bmatrix} \in \mathbb{R}^{3\times M}, \tag{4}$$

where each landmark $m_i = (x_i, y_i, z_i)^T$ is assumed to be static. The estimation process does not involve a prediction step for the landmarks.

At each time step t, the robot captures visual observations of the landmarks:

$$\mathbf{z}_t = h(\mathbf{m}, \mathbf{x}_t) + \mathbf{v}_t, \tag{5}$$

where $\mathbf{x}_t$ represents the robot's pose, $h(\mathbf{m}, \mathbf{x}_t)$ is the measurement function that projects the landmark positions into the camera frame, and $\mathbf{v}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_t)$ is the observation noise.

Given the sparsity of the covariance matrix $\mathbf{P}$ associated with the landmark estimates, efficient matrix representations such as Compressed Sparse Row (CSR) or List of Lists (LIL) formats should be considered to reduce computational complexity.

Since the robot's motion does not provide sufficient movement along the z-axis, the estimation of z coordinates for the landmarks is expected to be inaccurate. Thus, the focus of estimation should be on the x and y coordinates of the landmarks.

D. Visual-Inertial SLAM

The goal of this problem is to integrate IMU-based motion prediction with visual landmark updates to achieve a complete SLAM solution. The IMU pose $\mathbf{T}_t \in SE(3)$ is estimated using an EKF update step based on stereo-camera observations.

Let the IMU pose be represented as:

$$\mathbf{T}_t = \begin{bmatrix} \mathbf{R}_t & \mathbf{t}_t \\ 0 & 1 \end{bmatrix} \in SE(3), \tag{6}$$

where $\mathbf{R}_t \in SO(3)$ is the rotation matrix and $\mathbf{t}_t \in \mathbb{R}^3$ is the translation vector.

The IMU prediction step propagates the state using the motion model:

$$\mathbf{T}_{t+1} = f(\mathbf{T}_t, \mathbf{u}_t) + \mathbf{w}_t, \tag{7}$$

where $\mathbf{u}_t$ represents the IMU measurements, and $\mathbf{w}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_t)$ is the process noise.

The EKF update step corrects the IMU pose using landmark observations:

$$\mathbf{z}_t = h(\mathbf{T}_t, \mathbf{m}) + \mathbf{v}_t. \tag{8}$$

The measurement function $h(\mathbf{T}_t, \mathbf{m})$ projects landmarks into the stereo-camera frame, and $\mathbf{v}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_t)$ represents observation noise.

By tuning the noise parameters $\mathbf{Q}_t$ and $\mathbf{R}_t$, the trajectory estimate can be optimized. The estimated landmarks help correct the IMU positions, improving overall SLAM performance.

III. Technical Approach

A. IMU Localization via EKF Prediction

We implement an Extended Kalman Filter (EKF) prediction step to estimate the pose $\mathbf{T}_t \in SE(3)$ of the IMU over time t. The prediction step is based on the SE(3) kinematics equations and utilizes the linear velocity $\mathbf{v}_t$ and angular velocity $\boldsymbol{\omega}_t$ measurements from the IMU.

The process model for the EKF prediction step is derived from the SE(3) kinematics. The predicted pose $\mathbf{T}_{t+1}$ is computed using the current pose $\mathbf{T}_t$, the linear velocity $\mathbf{v}_t$, the angular velocity $\boldsymbol{\omega}_t$, and the time interval τ_t between consecutive measurements. The twist vector $\boldsymbol{\zeta}$ is constructed from the linear and angular velocities:

$$\boldsymbol{\zeta} = \begin{bmatrix} \mathbf{v}_t \\ \boldsymbol{\omega}_t \end{bmatrix}$$

The predicted pose $\mathbf{T}_{t+1}$ is then computed using the exponential map:

$$\mathbf{T}_{t+1} = \mathbf{T}_t \cdot \exp(\tau_t \cdot \boldsymbol{\zeta})$$

The covariance matrix $\boldsymbol{\Sigma}_{t+1}$ is predicted using the adjoint representation of the twist vector $\boldsymbol{\zeta}$. The adjoint representation $\text{ad}_{\boldsymbol{\zeta}}$ is computed as:

$$\text{ad}_{\boldsymbol{\zeta}} = \begin{bmatrix} [\boldsymbol{\omega}_t]_\times & [\mathbf{v}_t]_\times \\ \mathbf{0} & [\boldsymbol{\omega}_t]_\times \end{bmatrix}$$

where $[\cdot]_\times$ denotes the skew-symmetric matrix representation. The predicted covariance $\boldsymbol{\Sigma}_{t+1}$ is then given by:

$$\boldsymbol{\Sigma}_{t+1} = \exp(-\tau_t \cdot \text{ad}_{\boldsymbol{\zeta}}) \cdot \boldsymbol{\Sigma}_t \cdot \exp(-\tau_t \cdot \text{ad}_{\boldsymbol{\zeta}})^\top + \tau_t \cdot \mathbf{W}$$

where $\mathbf{W}$ is the process noise covariance matrix.

The EKF prediction step is implemented as follows:

- Initialize the pose $\mathbf{T}_0$ as the identity matrix $\mathbf{I}_4$ and the covariance matrix $\boldsymbol{\Sigma}_0$ as $\boldsymbol{\Sigma}_0 = T_\Sigma \cdot \mathbf{I}_6$, where T_Σ is a scaling factor.
- For each time step t:
  - Compute the twist vector $\boldsymbol{\zeta}$ from the linear velocity $\mathbf{v}_t$ and angular velocity $\boldsymbol{\omega}_t$.
  - Compute the predicted pose $\mathbf{T}_{t+1}$ using the exponential map.
  - Compute the predicted covariance $\boldsymbol{\Sigma}_{t+1}$ using the adjoint representation and the process noise covariance matrix $\mathbf{W}$.
  - Append the predicted pose $\mathbf{T}_{t+1}$ to the list of predicted poses.

The output of the EKF prediction step is a sequence of predicted poses $\mathbf{T}_{\text{pred}}$ representing the estimated trajectory of the IMU over time.

B. Feature detection and matching

To prepare feature tracks for landmark mapping, we perform feature detection and tracking using stereo camera images. The goal is to obtain pixel coordinates $\mathbf{z}_t \in \mathbb{R}^{4\times M}$ of detected visual features, where M is the total number of features across all frames. Each column of $\mathbf{z}_t$ corresponds to a specific feature, formatted as:

$$\mathbf{z}_t = \begin{bmatrix} l_{x,1} & l_{x,2} & \cdots & l_{x,M} \\ l_{y,1} & l_{y,2} & \cdots & l_{y,M} \\ r_{x,1} & r_{x,2} & \cdots & r_{x,M} \\ r_{y,1} & r_{y,2} & \cdots & r_{y,M} \end{bmatrix},$$

where:

- $l_{x,j}$ and $l_{y,j}$ are the x and y pixel coordinates of the j-th feature in the left camera frame.
- $r_{x,j}$ and $r_{y,j}$ are the x and y pixel coordinates of the j-th feature in the right camera frame.

If the j-th feature is not present in the t-th frame, the corresponding entries are set to -1.

Feature detection is performed using the Shi-Tomasi corner detection algorithm, which identifies strong corners in the left camera image. Let $\mathbf{I}_t^{\text{left}}$ be the left camera image at time t. The feature detection process can be expressed as:

$$\mathbf{P}\_t^{\text{left}} = \text{goodFeaturesToTrack}(\mathbf{I}\_t^{\text{left}}, M, \text{quality\_level}, \text{min\_distance}),$$

where:

- $\mathbf{P}_t^{\text{left}} \in \mathbb{R}^{2\times M}$ contains the pixel coordinates of the detected features in the left image.
- M is the maximum number of features to detect.
- quality_level and min_distance are parameters controlling the quality and spacing of the detected features.

Stereo matching is performed to find correspondences between the left and right camera frames. Optical flow is used to track the features detected in the left image to the right image. Let $\mathbf{I}_t^{\text{right}}$ be the right camera image at time t. The optical flow computation is given by:

$$\mathbf{P}_t^{\text{right}}, \text{status}, \text{err} = \text{calcOpticalFlowPyrLK}(\mathbf{I}_t^{\text{left}},$$

$\mathrm{I}_t^{\text{right}}, \mathbf{P}_t^{\text{left}}, \text{win\_size}, \text{max\_level}),$
where:

- $\mathbf{P}_t^{\text{right}} \in \mathbb{R}^{2\times M}$ contains the pixel coordinates of the corresponding features in the right image.
- status is a binary vector indicating whether the flow was successfully computed for each feature.
- err contains the error for each feature.
- win_size and max_level are parameters controlling the optical flow computation.

Temporal feature tracking is performed to track features in the left image across consecutive time steps. Let $\mathbf{I}_{t+1}^{\text{left}}$ be the left camera image at time $t+1$. The optical flow computation for temporal tracking is given by:

$$\mathbf{P}_{t+1}^{\text{left}}, \text{status}, \text{err} = \text{calcOpticalFlowPyrLK}(\mathbf{I}_t^{\text{left}},$$

$\mathrm{I}_{t+1}^{\text{left}}, \mathbf{P}_t^{\text{left}}, \text{win\_size}, \text{max\_level}),$
where:

- $\mathbf{P}_{t+1}^{\text{left}} \in \mathbb{R}^{2\times M}$ contains the pixel coordinates of the tracked features in the left image at time $t+1$.

The feature matrix $\mathbf{z}_t$ is constructed by combining the pixel coordinates from the left and right camera frames. For each feature j, the corresponding entries in $\mathbf{z}_t$ are:

$$\mathbf{z}_t[:, j] = \begin{cases} \begin{bmatrix} l_{x,j} & l_{y,j} & r_{x,j} & r_{y,j} \end{bmatrix}^T, & \text{if feature } j \text{ is visible in frame } t, \\ \begin{bmatrix} -1 & -1 & -1 & -1 \end{bmatrix}^T, & \text{otherwise.} \end{cases}$$

This approach ensures accurate and efficient feature tracking, and later useful for landmark mapping in visual SLAM.

C. Landmark Mapping via EKF Update

To estimate the landmark positions $\mathbf{m} \in \mathbb{R}^{3\times M}$ observed in the images, we employ an Extended Kalman Filter (EKF) with the landmark positions as the state. The EKF update step is performed using visual observations $\mathbf{z}_t$ to track the mean $\boldsymbol{\mu}_{\text{landmark}}$ and covariance $\boldsymbol{\Sigma}_{\text{landmark}}$ of the landmarks. The landmarks are assumed to be static, so no prediction step is required for them.

Initialization: The landmark positions are initialized with a mean $\boldsymbol{\mu}_{\text{landmark}} \in \mathbb{R}^{3\times M}$ and a covariance matrix $\boldsymbol{\Sigma}_{\text{landmark}} \in \mathbb{R}^{3M\times 3M}$. The covariance matrix is initialized as a diagonal matrix with large values to represent high uncertainty in the initial landmark estimates:

$$\boldsymbol{\mu}_{\text{landmark}} = \mathbf{0}, \quad \boldsymbol{\Sigma}_{\text{landmark}} = \mathbf{I}_{3M} \cdot \sigma_{\text{init}},$$

where $\sigma_{\text{init}} = 100$ is the initial uncertainty.

For each new landmark observed in the stereo images, triangulation is performed to estimate its 3D position. Given the left and right image points $\mathbf{p}_{\text{left}}$ and $\mathbf{p}_{\text{right}}$, the intrinsic matrices $\mathbf{K}_{\text{left}}$ and $\mathbf{K}_{\text{right}}$, and the extrinsic matrices $\mathbf{T}_{\text{left}}$ and $\mathbf{T}_{\text{right}}$, the landmark position $\mathbf{m}_j \in \mathbb{R}^3$ is computed using the following steps:

1. Convert the image points to homogeneous coordinates:

$$\mathbf{p}_{\text{left, homo}} = \begin{bmatrix} \mathbf{p}_{\text{left}} \\ 1 \end{bmatrix}, \quad \mathbf{p}_{\text{right, homo}} = \begin{bmatrix} \mathbf{p}_{\text{right}} \\ 1 \end{bmatrix}.$$

2. Compute the projection matrices for the left and right cameras:

$$\mathbf{P}_{\text{left}} = \mathbf{K}_{\text{left}} \cdot (\mathbf{T}_{\text{left}})[:3], \quad \mathbf{P}_{\text{right}} = \mathbf{K}_{\text{right}} \cdot (\mathbf{T}_{\text{right}})[:3].$$

3. Construct the matrix $\mathbf{A}$ for triangulation:

$$\mathbf{A} = \begin{bmatrix} \mathbf{p}_{\text{left, homo}}[0] \cdot \mathbf{P}_{\text{left}}[2] - \mathbf{P}_{\text{left}}[0] \\ \mathbf{p}_{\text{left, homo}}[1] \cdot \mathbf{P}_{\text{left}}[2] - \mathbf{P}_{\text{left}}[1] \\ \mathbf{p}_{\text{right, homo}}[0] \cdot \mathbf{P}_{\text{right}}[2] - \mathbf{P}_{\text{right}}[0] \\ \mathbf{p}_{\text{right, homo}}[1] \cdot \mathbf{P}_{\text{right}}[2] - \mathbf{P}_{\text{right}}[1] \end{bmatrix}.$$

4. Perform Singular Value Decomposition (SVD) on $\mathbf{A}$:

$$\mathbf{U}, \boldsymbol{\Sigma}, \mathbf{V}^T = \text{SVD}(\mathbf{A}).$$

5. Extract the triangulated point in homogeneous coordinates:

$$\mathbf{m}_j^{\text{homogeneous}} = \mathbf{V}^T[-1].$$

6. Convert to Cartesian coordinates:

$$\mathbf{m}_j = \frac{\mathbf{m}_j^{\text{homogeneous}}[:3]}{\mathbf{m}_j^{\text{homogeneous}}[3]}.$$

For each time step t, the EKF update step is performed using the visual observations $\mathbf{z}_t$ and the predicted IMU trajectory $\mathbf{T}_t$. The steps are as follows:

1. **Observation Model**: The observation model for the i-th feature is given by:

$$\mathbf{z}_{t,i} = h(\mathbf{T}_t, \mathbf{m}_j) + \mathbf{v}_{t,i},$$

where $h(\mathbf{T}_t, \mathbf{m}_j)$ projects the landmark $\mathbf{m}_j$ into the camera frame using the IMU pose $\mathbf{T}_t$, and $\mathbf{v}_{t,i} \sim \mathcal{N}(0, \mathbf{V})$ is the measurement noise.

2. **Jacobian Computation**: The Jacobian $\mathbf{H}_{t,i}$ of the observation model with respect to the landmark position $\mathbf{m}_j$ is computed as:

$$\mathbf{H}_{t,i} = \mathbf{K} \cdot \frac{\partial \pi}{\partial \mathbf{q}} \left(\mathbf{T}_t^{-1}\mathbf{m}_j\right) \cdot \mathbf{T}_t^{-1}\mathbf{P}^T,$$

where $\mathbf{K}$ is the camera intrinsic matrix, $\pi(\cdot)$ is the projection function, and $\mathbf{P}$ is the projection matrix.

3. **Kalman Gain**: The Kalman gain $\mathbf{K}_t$ is computed as:

$$\mathbf{K}_t = \boldsymbol{\Sigma}_{\text{landmark}}\mathbf{H}_t^T \left(\mathbf{H}_t\boldsymbol{\Sigma}_{\text{landmark}}\mathbf{H}_t^T + \mathbf{V}\right)^{-1},$$

where $\mathbf{H}_t$ is the stacked Jacobian matrix for all observed landmarks.

4. **Mean and Covariance Update**: The mean $\boldsymbol{\mu}_{\text{landmark}}$ and covariance $\boldsymbol{\Sigma}_{\text{landmark}}$ are updated as:

$$\boldsymbol{\mu}_{\text{landmark}} = \boldsymbol{\mu}_{\text{landmark}} + \mathbf{K}_t \left(\mathbf{z}_t - h(\mathbf{T}_t, \boldsymbol{\mu}_{\text{landmark}})\right),$$

$$\boldsymbol{\Sigma}\_\text{landmark} = (\mathbf{I} - \mathbf{K}_t\mathbf{H}_t)\boldsymbol{\Sigma}\_\text{landmark}.$$

To manage computational complexity, the visual features are subsampled by selecting only 1 out of every 4 features. This reduces the number of landmarks being processed at each time step, significantly decreasing the size of the state vector and the associated covariance matrix. Mathematically, if the total number of features is N, the subsampled features are given by:

$$\mathbf{z}\_\text{sampled} = \mathbf{z}[:, ::4, :],$$

where $\mathbf{z}$ is the original feature matrix. This sampling strategy ensures that the algorithm remains computationally tractable while still providing sufficient data for accurate landmark estimation.

The covariance matrix $\boldsymbol{\Sigma}\_\text{landmark} \in \mathbb{R}^{3M\times 3M}$ for the landmarks is inherently sparse, as most landmarks are not observed simultaneously. To optimize memory usage and computation, the covariance matrix is stored using sparse matrix representations such as Compressed Sparse Row (CSR) format from the SciPy library. This avoids storing and manipulating a large dense matrix, reducing both memory requirements and computational overhead.

Landmarks are initialized only when they are observed for the first time. This lazy initialization approach avoids unnecessary computations for landmarks that are not visible in the current frame. When a new landmark is detected, its position is estimated using triangulation based on stereo camera observations. The triangulation process involves solving a linear system of equations derived from the camera projection matrices and the observed image points.

A boolean array $\mathbf{vist}\_\text{landmark}$ is used to track whether a landmark has been observed at least once. This array ensures that landmarks are initialized only once and updated only when they are visible in the current frame. This approach avoids redundant computations and ensures that the algorithm focuses on refining the positions of visible landmarks.

D. Visual-inertial SLAM

The Visual-Inertial SLAM algorithm combines IMU-based pose prediction and landmark-based pose correction to estimate the IMU pose $\mathbf{T}_t \in SE(3)$ and landmark positions $\mathbf{m} \in \mathbb{R}^{3\times M}$. Below is the detailed formulation for the IMU prediction and IMU pose correction steps.

The IMU provides linear velocity $\mathbf{v}_t \in \mathbb{R}^3$ and angular velocity $\boldsymbol{\omega}_t \in \mathbb{R}^3$ measurements. The continuous-time kinematics of the IMU pose $\mathbf{T}(t)$ is given by:

$$\dot{\mathbf{T}}(t) = \mathbf{T}(t)\hat{\boldsymbol{\zeta}}(t),$$

where $\boldsymbol{\zeta}(t) = [\mathbf{v}(t)^T, \boldsymbol{\omega}(t)^T]^T \in \mathbb{R}^6$ is the generalized velocity in the body frame, and $\hat{\boldsymbol{\zeta}}(t)$ is its skew-symmetric matrix representation. In discrete time, the pose update is:

$$\mathbf{T}_{t+1} = \mathbf{T}_t \exp(\tau_t \hat{\boldsymbol{\zeta}}_t),$$

where τ_t is the time interval between consecutive measurements, and $\exp(\cdot)$ is the exponential map for $SE(3)$.

The EKF prediction step for the IMU pose is:

$$\boldsymbol{\mu}_{t+1|t} = \boldsymbol{\mu}_{t|t} \exp(\tau_t \hat{\boldsymbol{\zeta}}_t),$$

$$\boldsymbol{\Sigma}_{t+1|t} = \exp(-\tau_t \hat{\boldsymbol{\zeta}}_t)\boldsymbol{\Sigma}_{t|t} \exp(-\tau_t \hat{\boldsymbol{\zeta}}_t)^T + \tau_t \mathbf{W},$$

where:

- $\boldsymbol{\mu}_{t|t} \in SE(3)$ is the mean of the IMU pose at time t.
- $\boldsymbol{\Sigma}_{t|t} \in \mathbb{R}^{6\times 6}$ is the covariance of the IMU pose at time t.
- $\mathbf{W} \in \mathbb{R}^{6\times 6}$ is the process noise covariance matrix.

The IMU pose is corrected using visual observations from the stereo camera. The observation model for the i-th feature is:

$$\mathbf{z}_{t,i} = h(\mathbf{T}_t, \mathbf{m}_j) + \mathbf{v}_{t,i},$$

where:

- $h(\mathbf{T}_t, \mathbf{m}_j)$ projects the landmark $\mathbf{m}_j$ into the camera frame using the IMU pose $\mathbf{T}_t$.
- $\mathbf{v}_{t,i} \sim \mathcal{N}(0, \mathbf{V})$ is the measurement noise.

The Jacobian of the observation model with respect to the IMU pose $\mathbf{T}_t$ is:

$$\mathbf{H}_{t+1,i} = -\mathbf{K}_s \frac{d\pi}{d\mathbf{q}} \left(\mathbf{T}_{t+1}^{-1}\mathbf{m}_j\right) \mathbf{T}_{t+1}^{-1}\mathbf{m}_j^{\circ},$$

where:

- $\mathbf{K}_s$ is the stereo camera calibration matrix.
- $\frac{d\pi}{d\mathbf{q}}$ is the derivative of the projection function $\pi(\cdot)$.
- $\mathbf{m}_j^{\circ}$ is the skew-symmetric matrix representation of $\mathbf{m}_j$.

The EKF update step for the IMU pose is:

1. Compute the Kalman gain:

$$\mathbf{K}_{t+1}^{\text{IMU}} = \boldsymbol{\Sigma}_{t+1|t}^{\text{IMU}} \mathbf{H}_{t+1}^T (\mathbf{H}_{t+1} \boldsymbol{\Sigma}_{t+1|t}^{\text{IMU}} \mathbf{H}_{t+1}^T + \mathbf{V})^{-1},$$

where:

- $\mathbf{H}_{t+1}$ is the stacked Jacobian matrix for all observed features.
- $\mathbf{V}$ is the measurement noise covariance matrix.

2. Update the IMU pose mean:

$$\boldsymbol{\mu}_{t+1|t+1}^{\text{IMU}} = \boldsymbol{\mu}_{t+1|t}^{\text{IMU}} + \mathbf{K}_{t+1}^{\text{IMU}}(\mathbf{z}_{t+1} - h(\mathbf{T}_{t+1}, \boldsymbol{\mu}_t)),$$

where $\mathbf{z}_{t+1}$ is the vector of observed feature coordinates.

3. Update the IMU pose covariance:

$$\boldsymbol{\Sigma}_{t+1|t+1}^{\text{IMU}} = (\mathbf{I} - \mathbf{K}_{t+1}^{\text{IMU}} \mathbf{H}_{t+1}) \boldsymbol{\Sigma}_{t+1|t}^{\text{IMU}}.$$

1. **IMU Prediction**: - Predict the IMU pose $\mathbf{T}_{t+1}$ using the IMU measurements $\mathbf{v}_t$ and $\boldsymbol{\omega}_t$. - Compute the predicted mean $\boldsymbol{\mu}_{t+1|t}$ and covariance $\boldsymbol{\Sigma}_{t+1|t}$.

2. **IMU Pose Correction**: - Use the predicted IMU pose $\mathbf{T}_{t+1}$ to project the landmarks into the camera frame. - Compute the innovation $\mathbf{z}_{t+1} - h(\mathbf{T}_{t+1}, \boldsymbol{\mu}_t)$ and the Kalman gain $\mathbf{K}_{t+1}^{\text{IMU}}$. - Update the IMU pose mean $\boldsymbol{\mu}_{t+1|t+1}^{\text{IMU}}$ and covariance $\boldsymbol{\Sigma}_{t+1|t+1}^{\text{IMU}}$.

3. **Iterate**: - Repeat the prediction and correction steps for each time step t.

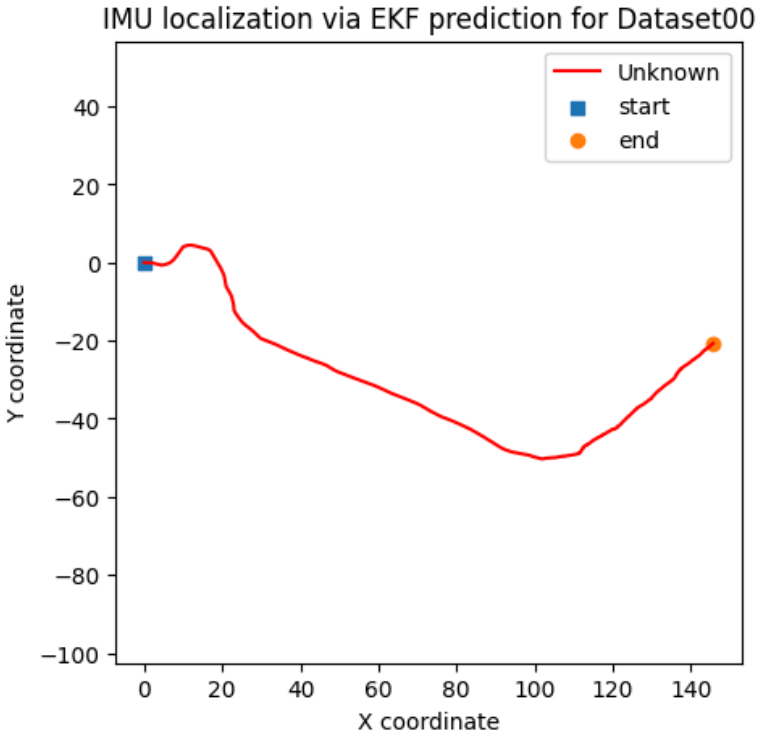
The performance of the Visual-Inertial SLAM algorithm depends on the tuning of the process noise $\mathbf{W}$ and measurement noise $\mathbf{V}$. These parameters control the trade-off between trusting the IMU predictions and the visual observations. Proper tuning ensures accurate trajectory estimation and landmark mapping.

IV. RESULTS

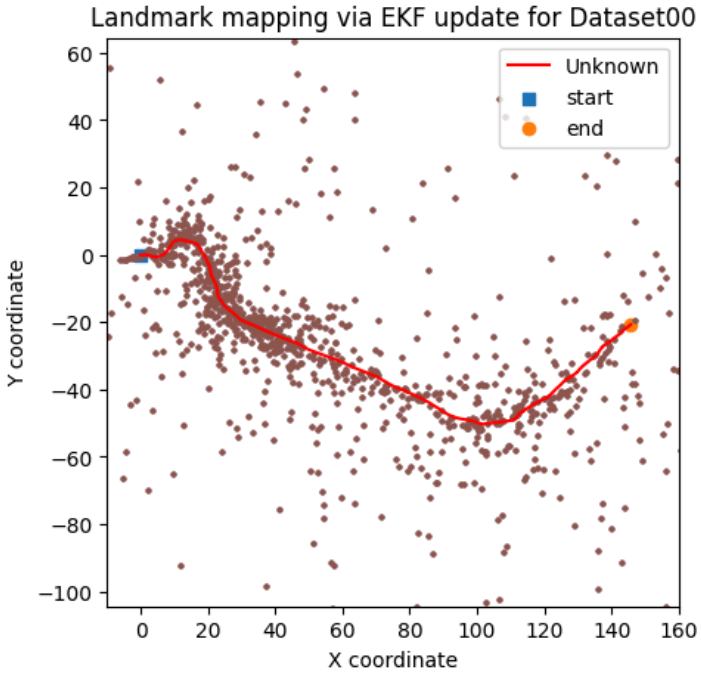
RESULTS

Dataset-00

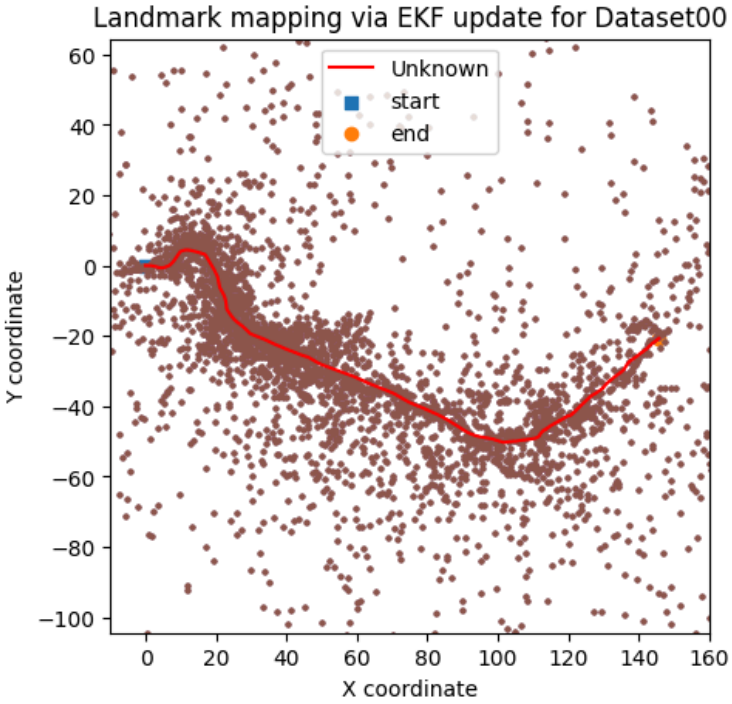
A. IMU localization via EKF prediction



B. Landmark mapping via EKF update



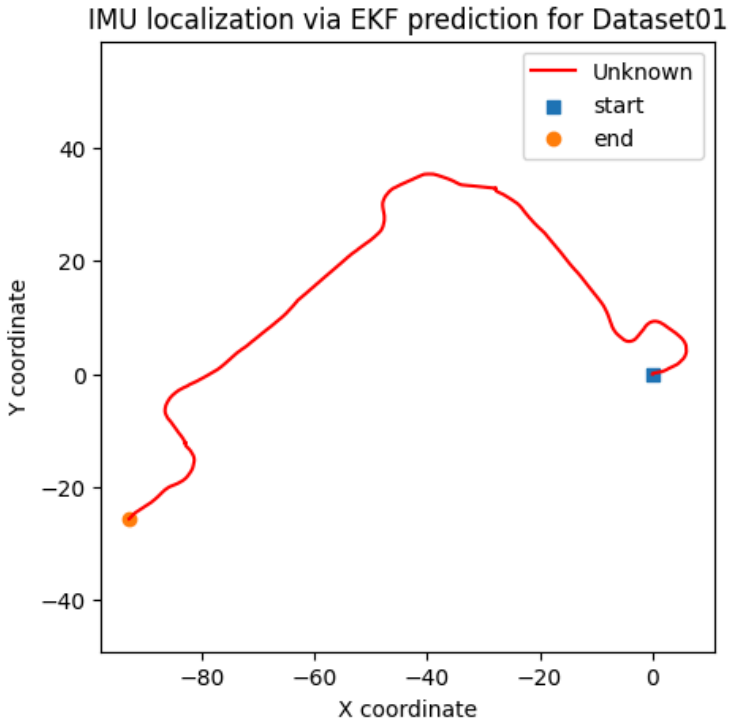
B. Landmark mapping via EKF update (complete landmarks no filtering)



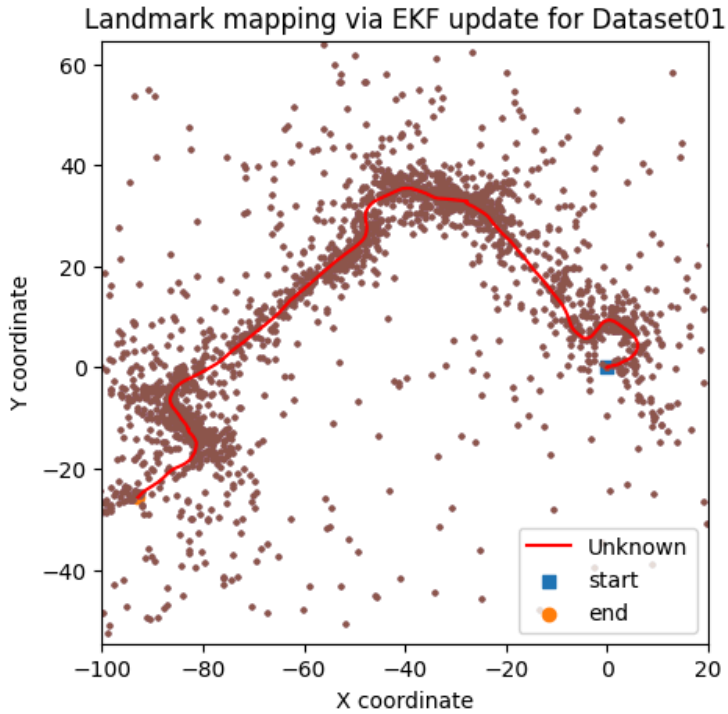
C. Visual-inertial SLAM

Dataset-01

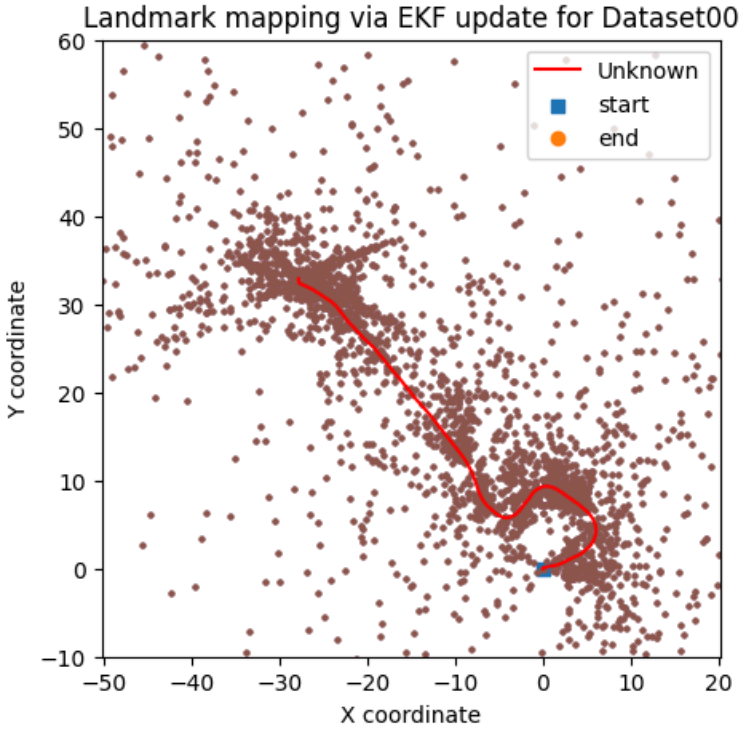
A. IMU localization via EKF prediction



B. Landmark mapping via EKF update



B. Landmark mapping via EKF update (complete landmarks no filtering)
(it is taking lot of time to run on whole features so did 38 percent of trajectory and stopped but in filtered landmarks case (above) complete trajectory is drawn in the top)



C. Visual-inertial SLAM