# Visual-Inertial SLAM via Extended Kalman Filter on SE(3)

Anderson Compalas
*Department of Electrical and Computer Engineering*
*University of California San Diego*
La Jolla, California
acompalas@ucsd.edu

***Abstract*—We implement a visual-inertial simultaneous localization and mapping (SLAM) system using an Extended Kalman Filter (EKF) operating on the Lie group SE(3). The system fuses IMU velocity measurements with stereo camera observations of point landmarks to jointly estimate the robot trajectory and a sparse 3D landmark map. We decompose the full SLAM problem into three tasks: (1) IMU-only dead reckoning via SE(3) kinematics, (2) landmark mapping with a fixed trajectory, and (3) full visual-inertial SLAM with a joint pose-landmark state. For the optional extra credit, we implement feature detection and tracking on dataset02 using Shi-Tomasi corner detection and Lucas-Kanade optical flow. Results are presented on three datasets collected by Clearpath Jackal robots navigating MIT's campus.**

***Index Terms*—Visual-Inertial SLAM, Extended Kalman Filter, SE(3), Stereo Camera, IMU, Landmark Mapping**

## I. Introduction

Simultaneous Localization and Mapping (SLAM) is a fundamental problem in mobile robotics: a robot must estimate its own pose while building a map of the environment, without prior knowledge of either. Visual-inertial SLAM addresses this by fusing two complementary sensors. An IMU provides high-frequency motion measurements but accumulates drift over time. A stereo camera provides geometric observations of the environment that can correct this drift by re-observing previously mapped landmarks. Together they enable robust, drift-corrected trajectory estimation.

In this project we implement VI-SLAM using an EKF operating on SE(3), the group of rigid-body transformations. The approach has four components: (1) an EKF prediction step that propagates the IMU pose using body-frame SE(3) kinematics; (2) optional feature detection and tracking for dataset02, producing stereo pixel tracks using Shi-Tomasi corners and Lucas-Kanade optical flow; (3) a landmark mapping step that estimates the 3D positions of visual features using stereo observations with the IMU trajectory fixed; and (4) a full VI-SLAM step that jointly estimates the pose and landmark map, using stereo observations to correct IMU drift in real time.

We validate the system on three datasets collected by Clearpath Jackal ground robots on MIT's campus, demonstrating consistent trajectory improvement over IMU dead reckoning across all datasets.

## II. Problem Formulation

### A. State, Inputs, and Outputs

The robot state at time $t$ consists of the IMU pose $T_t = {}^{W}T_{I,t} \in SE(3)$ and the positions of $M$ static landmarks $\{\mathbf{m}_j\}_{j=1}^{M} \subset \mathbb{R}^3$ in the world frame. The inputs are: IMU body-frame linear velocity $\mathbf{v}_t \in \mathbb{R}^3$ and angular velocity $\boldsymbol{\omega}_t \in \mathbb{R}^3$; stereo pixel observations $\mathbf{z}_{t,j} = [u_L, v_L, u_R, v_R]^\top \in \mathbb{R}^4$ (set to $[-1,-1,-1,-1]^\top$ when landmark $j$ is not visible); UNIX timestamps $\tau_t$ yielding variable timesteps $\Delta\tau_t = \tau_{t+1} - \tau_t$; and calibration matrices $K_L, K_R \in \mathbb{R}^{3\times3}$ and extrinsics ${}^{I}T_L, {}^{I}T_R \in SE(3)$. The desired outputs are the pose trajectory $\{T_t\}_{t=0}^{T}$ and landmark positions $\{\mathbf{m}_j\}$.

### B. IMU Kinematics

The pose evolves under body-frame twist $\mathbf{u}_t = [\mathbf{v}_t^\top, \boldsymbol{\omega}_t^\top]^\top$ according to the SE(3) kinematic equation:

$$\dot{T}(t) = T(t)\,\hat{\mathbf{u}}(t), \quad \hat{\mathbf{u}} = \begin{bmatrix} \hat{\boldsymbol{\omega}} & \mathbf{v} \\ \mathbf{0}^\top & 0 \end{bmatrix} \in \mathfrak{se}(3) \tag{1}$$

Discretized with variable timestep $\Delta\tau_t$:

$$T_{t+1} = T_t\,\exp(\Delta\tau_t\,\hat{\mathbf{u}}_t) \tag{2}$$

### C. Stereo Observation Model

For landmark $\mathbf{m}_j$ with homogeneous form $\bar{\mathbf{m}}_j = [\mathbf{m}_j^\top, 1]^\top$, the predicted stereo observation through the left and right cameras is:

$$\hat{\mathbf{z}}_{t,j} = \begin{bmatrix} K_L\,\pi({}^{O}T_I \cdot T_t^{-1} \cdot \bar{\mathbf{m}}_j) \\ K_R\,\pi({}^{O}T_{I,R} \cdot T_t^{-1} \cdot \bar{\mathbf{m}}_j) \end{bmatrix}_{[u,v]} \in \mathbb{R}^4 \tag{3}$$

where $\pi(\mathbf{q}) = \mathbf{q}/q_3$ is the perspective projection map and ${}^{O}T_I$ is the IMU-to-optical-frame transform.

### D. EKF Structure

We represent pose uncertainty via a perturbation on the Lie algebra: $T_t = \mu_t \exp(\widehat{\delta\boldsymbol{\xi}})$, $\delta\boldsymbol{\xi} \sim \mathcal{N}(\mathbf{0}, \Sigma_T)$. The full SLAM state and covariance are:

$$\mathbf{x} = \begin{bmatrix} \delta\boldsymbol{\xi} \\ \mathbf{m}_1 \\ \vdots \\ \mathbf{m}_M \end{bmatrix} \in \mathbb{R}^{6+3M}, \quad \Sigma \in \mathbb{R}^{(6+3M)\times(6+3M)} \tag{4}$$

The EKF alternates between a prediction step that propagates $(\mu_T, \Sigma)$ forward using the IMU kinematics, and an update step that corrects $(\mu_T, \Sigma)$ using stereo observations $\mathbf{z}_{t,j}$.

## III. Technical Approach

### A. Task 1: IMU Localization via EKF Prediction

*1) Mean Propagation:* The pose mean is propagated using the discretized SE(3) kinematics, implemented via the provided `axangle2pose` utility:

$$\mu_{t+1|t} = \mu_{t|t} \exp(\Delta\tau_t \, \hat{\mathbf{u}}_t) \tag{5}$$

The timestep $\Delta\tau_t = \tau_{t+1} - \tau_t$ is computed directly from the UNIX timestamps at each step, handling the variable sample rate ($\approx 30$ Hz with slight jitter) without any resampling.

*2) Covariance Propagation:* Linearizing the perturbation dynamics around the mean gives:

$$\dot{\delta\boldsymbol{\mu}} = -{}^{\curlywedge}\mathbf{u}\,\delta\boldsymbol{\mu} + \mathbf{w}, \quad \mathbf{w} \sim \mathcal{N}(\mathbf{0}, W) \tag{6}$$

where ${}^{\curlywedge}\mathbf{u} \in \mathbb{R}^{6\times 6}$ is the adjoint operator (`axangle2adtwist`):

$${}^{\curlywedge}\mathbf{u} = \begin{bmatrix} \hat{\boldsymbol{\omega}} & \hat{\mathbf{v}} \\ \mathbf{0} & \hat{\boldsymbol{\omega}} \end{bmatrix} \tag{7}$$

The discrete covariance update is:

$$\Sigma_{t+1|t} = A_t \, \Sigma_{t|t} \, A_t^\top + \Delta\tau_t \, W, \quad A_t = \exp(-\Delta\tau_t \, {}^{\curlywedge}\mathbf{u}_t) \tag{8}$$

where $A_t$ is computed via `scipy.linalg.expm`.

*3) Process Noise Tuning:* The process noise is diagonal:

$$W = \text{diag}\big(\sigma^2_{\text{trans}}\mathbf{I}_3, \; \sigma^2_{\text{rot}}\mathbf{I}_3\big) \tag{9}$$

The TAs provided a starting point of $\sigma^2 = 0.1$ on Piazza. This encodes confidence in the IMU: large $W$ makes the filter distrust the IMU and accept large camera corrections; small $W$ makes it trust the IMU and accept only small corrections. With $\sigma^2 = 0.1$ the Kalman gain in Task 4 grew large enough that single noisy observations caused pose jumps exceeding 1 radian, leading to immediate divergence. After systematic reduction in orders of magnitude we found $\sigma^2_{\text{trans}} = \sigma^2_{\text{rot}} = 10^{-6}$ produces stable trajectories across all three datasets. This value treats the IMU as near-perfect, using camera observations for small drift corrections only.

### B. Task 2: Feature Detection and Tracking (Extra Credit)

For dataset02, which provides no pre-computed features, we implement a stereo feature tracker producing a $(4, M, T)$ array where missing observations are $[-1, -1, -1, -1]^\top$.

*1) Shi-Tomasi Corner Detection:* Every $N_{\text{detect}} = 5$ frames, new corners are detected in the left grayscale image using `goodFeaturesToTrack` with quality threshold $q = 0.01$ and minimum spacing of 10 pixels. New detections within 10 pixels of an existing track are discarded.

*2) Temporal Tracking (Task 2b):* Tracks are propagated from frame $t-1$ to $t$ using Lucas-Kanade pyramidal optical flow via `calcOpticalFlowPyrLK` with a $15 \times 15$ pixel window and 2 pyramid levels. Features leaving the image boundary are dropped.

*3) Stereo Matching (Task 2a):* For each active track in the left image at time $t$, the right-image correspondence is found by running `calcOpticalFlowPyrLK` from the left image to the right image. Two sanity checks are applied before storing the match:

1) **Positive disparity**: $d = u_L - u_R > 0$. For a forward-facing stereo rig, a 3D point at finite depth always projects further right in the left image. A non-positive disparity indicates a bad match.
2) **Epipolar constraint**: $|v_L - v_R| \leq 2.0$ px. The rig is near-rectified (confirmed by the near-identity rotation block in the extrinsics), so valid matches have nearly identical $v$ coordinates.

### C. Task 3: Landmark Mapping via EKF Update

*1) Camera Projection Model:* The project specification notes the stereo rig may not be perfectly horizontal and recommends projecting landmarks separately through each camera rather than using the compact $K_s$ formulation. We follow this recommendation. Per the instructor's Piazza clarification, `extL_T_imu` represents ${}^{I}T_L$ (camera-to-IMU), so the IMU-to-optical transform is:

$${}^{O}T_I = {}^{o}T_r \cdot ({}^{I}T_L)^{-1} \tag{10}$$

where ${}^{o}T_r$ rotates from the IMU body convention to the optical frame convention.

*2) Landmark Jacobian:* The Jacobian of the stereo observation with respect to $\mathbf{m}_j$ is:

$$H^{\text{lm}}_{t,j} = \begin{bmatrix} K_L^{[2]} \, \frac{d\pi}{d\mathbf{q}}(\mathbf{q}_L)^{[3]} \, {}^{O}T_I \, T_t^{-1} \, P^\top \\ K_R^{[2]} \, \frac{d\pi}{d\mathbf{q}}(\mathbf{q}_R)^{[3]} \, {}^{O}T_{I,R} \, T_t^{-1} \, P^\top \end{bmatrix} \in \mathbb{R}^{4\times 3} \tag{11}$$

where $K^{[2]}$ is the top 2 rows of $K$, $(\cdot)^{[3]}$ is the top 3 rows of `projectionJacobian`, and $P = [\mathbf{I}_3 \; \mathbf{0}] \in \mathbb{R}^{3\times 4}$.

*3) EKF Update:* Since landmarks are static there is no prediction step. The lecture formulation uses a full joint covariance $\Sigma \in \mathbb{R}^{3M\times 3M}$. We use a block-decoupled approximation with independent $3\times 3$ per-landmark covariances. This deviates from the lecture but is deliberate: the full matrix requires $O(M^2)$ storage and $O(M^3)$ updates per timestep, which is intractable for $M \sim 1000$. The project specification explicitly suggests sparse matrix techniques to manage this complexity. The per-landmark update is:

$$S_j = H^{\text{lm}}_{t,j} \, \Sigma_j \, {H^{\text{lm}}_{t,j}}^\top + V \tag{12}$$

$$K_j = \Sigma_j \, {H^{\text{lm}}_{t,j}}^\top \, S_j^{-1} \tag{13}$$

$$\boldsymbol{\mu}_j \leftarrow \boldsymbol{\mu}_j + K_j \, (\mathbf{z}_{t,j} - \hat{\mathbf{z}}_{t,j}) \tag{14}$$

$$\Sigma_j \leftarrow (\mathbf{I}_3 - K_j \, H^{\text{lm}}_{t,j}) \, \Sigma_j \tag{15}$$

where $V = \sigma^2_{V,3} \, \mathbf{I}_4$ is the observation noise covariance with $\sigma^2_{V,3} = 4.0$ px$^2$ (TA suggested range: 1–2 px$^2$; we use 4.0 in Task 3 to account for the linearization error from the fixed, potentially drifted, IMU trajectory).

*4) Landmark Initialization:* Each landmark is initialized via stereo triangulation at its first valid observation. With disparity $d = u_L - u_R$ and baseline $b$:

$$Z = \frac{f_u \cdot b}{d}, \quad X = \frac{(u_L - c_u)\, Z}{f_u}, \quad Y = \frac{(v_L - c_v)\, Z}{f_v} \tag{16}$$

Initial covariance is $\Sigma_j = \mathbf{I}_3$, reflecting equal uncertainty across all dimensions.

*5) Self-Introduced Robustness Filters:* The lecture formulation does not specify handling of degenerate geometry or bad observations. We introduce three filters:

**Minimum disparity gate** ($d_{\min} = 2.0$ px): The formula $Z = f_u b/d$ becomes numerically unstable as $d \to 0$. We reject $|d| < 2.0$ px.

**Maximum depth gate** ($Z_{\max} = 50.0$ m): Depth uncertainty scales as $1/d^2$, so far landmarks are very noisy. Empirically, without this gate the landmark cloud contained outliers at 100–200 m; 50 m captures the useful environment structure in these datasets.

**Innovation gate** ($\gamma_3 = 50.0$ px): If $\|\mathbf{z}_{t,j} - \hat{\mathbf{z}}_{t,j}\| > \gamma_3$ the update is skipped. A large residual indicates either a mismatched feature or a poorly initialized landmark. Applying the update would drag the landmark toward an incorrect position.

### D. Task 4: Visual-Inertial SLAM

*1) Joint Covariance and Prediction:* The joint covariance $\Sigma \in \mathbb{R}^{(6+3M)\times(6+3M)}$ is stored as a SciPy CSR sparse matrix. The TA confirmed on Piazza that this joint covariance must be used even in the prediction step. The joint prediction uses:

$$F = \begin{bmatrix} A_t & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_{3M} \end{bmatrix}, \quad Q = \begin{bmatrix} \Delta\tau_t W & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \tag{17}$$

$$\Sigma_{t+1|t} = F\, \Sigma_{t|t}\, F^\top + Q \tag{18}$$

consistent with the lectures.

*2) Pose Jacobian:* Using the $\odot$ operator, which satisfies $\hat{\boldsymbol{\xi}}\, \mathbf{s} = \mathbf{s}^{\odot}\, \boldsymbol{\xi}$ for $\mathbf{s} \in \mathbb{R}^4$ under the $[\mathbf{v}; \boldsymbol{\omega}]$ convention:

$$\mathbf{s}^{\odot} = \begin{bmatrix} s_4 & 0 & 0 & 0 & s_3 & -s_2 \\ 0 & s_4 & 0 & -s_3 & 0 & s_1 \\ 0 & 0 & s_4 & s_2 & -s_1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{19}$$

The pose Jacobian per observation is shown below, with $\mathbf{s} = T^{-1}\bar{\mathbf{m}}_j$:

$$H_j^{\text{pose}} = \begin{bmatrix} -K_L^{[2]}\, \frac{d\pi}{d\mathbf{q}}(\mathbf{q}_L)^{[3]}\, {}^{O}T_I\, \mathbf{s}^{\odot} \\ -K_R^{[2]}\, \frac{d\pi}{d\mathbf{q}}(\mathbf{q}_R)^{[3]}\, {}^{O}T_{I,R}\, \mathbf{s}^{\odot} \end{bmatrix} \in \mathbb{R}^{4\times 6} \tag{20}$$

This was verified numerically against finite differences to within $5 \times 10^{-3}$.

*3) Joint Measurement Update:* All $N$ visible initialized landmarks are stacked into $H \in \mathbb{R}^{4N\times(6+3M)}$ with both pose and landmark blocks filled. The EKF update is:

$$S = H\, \Sigma\, H^\top + \mathbf{I}_N \otimes V \tag{21}$$

$$K = \Sigma\, H^\top S^{-1} \tag{22}$$

$$\delta\mathbf{x} = K\, (\mathbf{z} - \hat{\mathbf{z}}) \tag{23}$$

*4) Covariance Update: Joseph Form:* The lectures use the standard form $\Sigma \leftarrow (\mathbf{I} - KH)\Sigma$. We instead use the Joseph stabilized form:

$$\Sigma \leftarrow (\mathbf{I} - KH)\, \Sigma\, (\mathbf{I} - KH)^\top + K\, (\mathbf{I}_N \otimes V)\, K^\top \tag{24}$$

followed by $\Sigma \leftarrow \frac{1}{2}(\Sigma + \Sigma^\top)$. This deviates from the lectures but is necessary for numerical stability: the standard form only preserves positive semi-definiteness when $K$ is exactly optimal, but floating-point inversion of $S$ introduces errors that cause $\Sigma$ to become indefinite over time. The Joseph form remains positive semi-definite even with approximate $K$, which is critical as the state dimension grows.

*5) Pose Mean Update and Landmark Augmentation:* The pose correction is applied as a right perturbation on SE(3):

$$\mu_T \leftarrow T_{\text{pred}}\, \exp(\widehat{\delta\boldsymbol{\xi}}), \quad \delta\boldsymbol{\xi} = (\delta\mathbf{x})_{1:6} \tag{25}$$

New landmarks are initialized on-the-fly using the corrected pose and appended to the state with zero cross-covariances:

$$\Sigma \leftarrow \begin{bmatrix} \Sigma & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_{3n_{\text{new}}} \end{bmatrix} \tag{26}$$

This deviates from the ideal augmentation step but is a standard EKF-SLAM approximation, computing the true cross-covariance at initialization is expensive and the approximation is conservative rather than incorrect.

*6) Self-Introduced Robustness Filters for Task 4:* **Tighter innovation gate** ($\gamma_4 = 20.0$ px): Tighter than Task 3's 50 px because in Task 4 a bad observation corrupts the pose directly through the cross-covariance block, not just the landmark.

**Pose correction gate** ($\gamma_\xi = 0.3$): If $\|\delta\boldsymbol{\xi}\| > 0.3$ after computing $\delta\mathbf{x} = K(\mathbf{z} - \hat{\mathbf{z}})$, the entire update is skipped for that timestep; $T_{\text{cur}} = T_{\text{pred}}$ and $\Sigma$ is left unchanged. The EKF linearization assumes $\delta\boldsymbol{\xi}$ is small; corrections above 0.3 (roughly 17 or 0.3 m in twist norm) indicate a degenerate observation or near-singular $S$. In our experiments, all legitimate corrections were below 0.1, making 0.3 a reliable boundary.

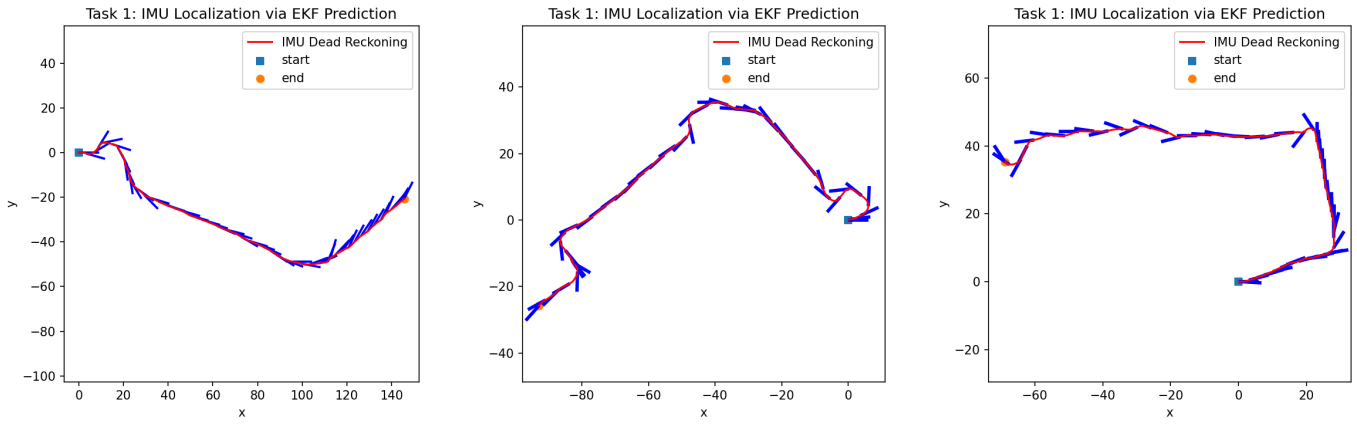
The observation noise for Task 4 is $\sigma_{V,4}^2 = 2.0$ px$^2$ (within the TA-suggested range of 1–2 px$^2$), smaller than Task 3's 4.0 px$^2$ since Task 4 continuously corrects the pose, reducing linearization error.

*7) Summary of All Parameters:* Table I summarizes all parameters. † = self-introduced; ‡ = TA-suggested starting point, then tuned.

## IV. Results

### A. Task 1: IMU Dead Reckoning

Fig. 1 shows the IMU-only EKF prediction trajectories for all three datasets. The paths are smooth and self-consistent, with orientation arrows confirming reasonable heading estimates. Dataset00 traces a curved path approximately 150 m long; dataset01 follows a longer arc; dataset02 follows an L-shaped path. These results confirm that the SE(3) kinematic integration and variable-timestep handling are correct.




(a) Dataset00 (b) Dataset01 (c) Dataset02

Fig. 1: Task 1: IMU dead reckoning via EKF prediction. Red = trajectory, blue arrows = estimated orientation.

TABLE I: Parameter Summary

| Parameter | Symbol | Value | Task | Origin |
|---|---|---|---|---|
| Trans. process noise‡ | $\sigma^2_{\text{trans}}$ | $10^{-6}$ | 1,4 | TA+tuned |
| Rot. process noise‡ | $\sigma^2_{\text{rot}}$ | $10^{-6}$ | 1,4 | TA+tuned |
| Obs. noise T3‡ | $\sigma^2_{V,3}$ | 4.0 px$^2$ | 3 | TA+tuned |
| Obs. noise T4‡ | $\sigma^2_{V,4}$ | 2.0 px$^2$ | 4 | TA+tuned |
| Subsample T3 | $N_{s,3}$ | 5 | 3 | Empirical |
| Subsample T4 | $N_{s,4}$ | 20 | 4 | Empirical |
| Min. disparity† | $d_{\min}$ | 2.0 px | 3,4 | Self |
| Max. depth† | $Z_{\max}$ | 50.0 m | 3,4 | Self |
| Innov. gate T3† | $\gamma_3$ | 50.0 px | 3 | Self |
| Innov. gate T4† | $\gamma_4$ | 20.0 px | 4 | Self |
| Pose correction gate† | $\gamma_\xi$ | 0.3 | 4 | Self |

### B. Task 2: Feature Detection and Tracking (Extra Credit)

Fig. 2 shows feature tracks for dataset02 at $t = 15$, $\Delta t = 100$, matching the format of Fig. 2 in the project specification. Top row: stereo matches — both images show blue (left camera) and red (right camera) feature positions for the same set of landmarks, with green lines meant to show the stereo displacement (not clearly visible here due to the small stereo baseline of $\approx 9$ cm, which produces sub-pixel to few-pixel disparities at typical scene depths). Bottom row: temporal tracks from $t = 15$ (blue) to $t = 115$ (red) with green displacement lines, showing the robot's POV as it turns right toward the building. A total of 19,559 unique feature tracks were computed across 6,393 timesteps.

### C. Task 3: Landmark Mapping

Fig. 3 shows the estimated landmark $x$-$y$ positions overlaid on the fixed IMU trajectory for all three datasets. The landmark clouds follow each trajectory closely; the forward-facing stereo camera observes features in the corridor of environment the robot passes through. The $x$-$y$ positions are well-localized. As noted in the project specification, $z$-coordinate estimates are poor due to the robot's limited vertical motion, and are not shown. Residual outliers (scattered points far from the trajectory) are primarily from border features with unstable optical flow that survive the 50 m depth gate.

### D. Task 4: Visual-Inertial SLAM

Fig. 4 shows the full VI-SLAM results. In all three datasets the SLAM trajectory (blue) visibly corrects the IMU-only trajectory (red), confirming the pose update is active.

On dataset00, the SLAM system reduces a downward $y$-bias in the dead reckoning, producing a tighter path that ends closer to the origin. On dataset01, the correction pulls the trajectory inward along the arc; the SLAM estimate is more compact than the IMU-only path, suggesting the IMU overestimates curvature. The larger deviation on dataset01 compared to dataset00 reflects the longer trajectory and greater accumulated IMU drift. On dataset02, the SLAM trajectory follows the L-shape more precisely, with the corrected trajectory better aligned with the landmarks.

### E. Discussion

*1) What Worked Well:* The SE(3) kinematic integration and separate left/right camera projection model both produced consistent results. The three self-introduced filters (minimum disparity, maximum depth, innovation gate) dramatically improved Task 3 landmark quality. Without the 50 m depth cap, dataset01 had severe outliers at 100–200 m. The Joseph form covariance update and symmetry enforcement in Task 4 were essential for sustained numerical stability; without them, $\Sigma$ lost positive semi-definiteness within hundreds of timesteps.

*2) Challenges:* The main challenge was EKF divergence in Task 4 with the TA-suggested $\sigma^2 = 0.1$. The Kalman gain grew large enough that single noisy observations caused pose jumps exceeding 1 radian. Reducing to $\sigma^2 = 10^{-6}$ and adding the pose correction gate $\gamma_\xi = 0.3$ resolved this. The resulting corrections are visible but modest. A more principled approach would calibrate IMU noise empirically from static data.

The dense Joseph form update densifies the sparse $\Sigma$ at each step, making computation scale with state dimension.


Fig. 2: Task 2: Feature detection and tracking on dataset02. Top: stereo matches at $t = 15$ (blue = left camera, red = right camera). Bottom: temporal tracks from $t = 15$ (blue) to $t = 115$ (red). Green lines show displacement in both rows.

Subsampling landmarks (T4: subsample $= 20$, giving $\sim$300-400 landmarks) kept state dimension below $\sim$800 and runtime manageable.

*3) Approximations and Limitations:* Task 3 uses per-landmark $3 \times 3$ covariance blocks rather than the full joint $3M \times 3M$ matrix. Task 4 initializes new landmarks with zero cross-covariances. Both are standard EKF-SLAM approximations that do not affect the correctness of updates for landmarks that have already been initialized.

## V. Conclusion

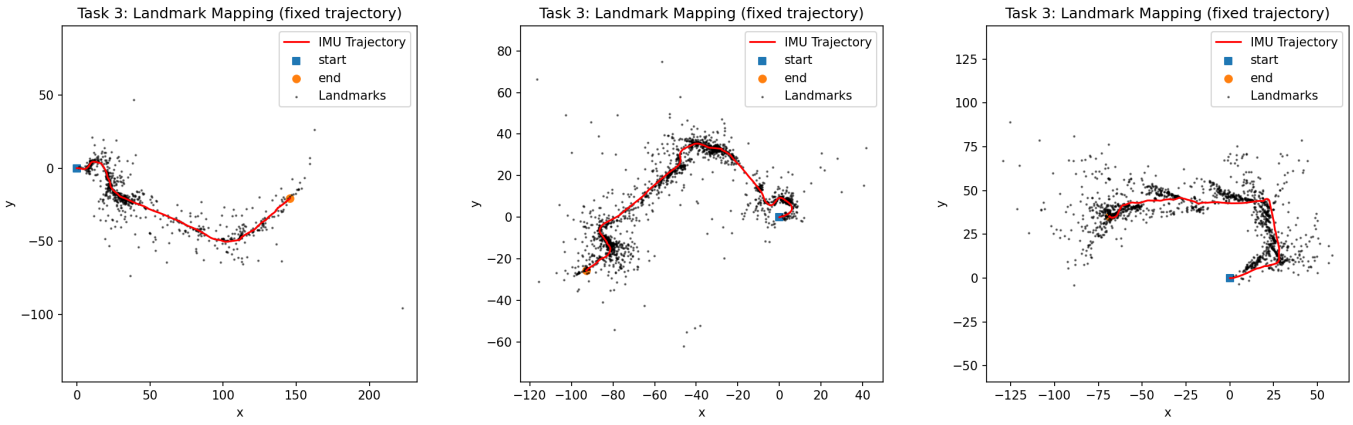
We implemented a complete VI-SLAM system using an EKF on SE(3), validated on three real-world datasets. The system integrates IMU kinematics, estimates 3D landmark positions from stereo observations, and applies joint pose-landmark EKF updates to correct IMU drift. For the extra credit we implemented a full stereo feature tracker on dataset02 using Shi-Tomasi detection and Lucas-Kanade optical flow, producing 19,559 feature tracks. The SLAM update consistently improves upon IMU dead reckoning across all datasets. Key practical contributions include the Joseph form covariance update, the pose correction gate, and the depth and innovation gating scheme, all of which were necessary to achieve stable filter behavior on real sensor data.

## Acknowledgments

We thank Professor Nikolay Atanasov and the ECE 276A teaching assistants for providing the project framework, datasets, calibration data, and Piazza clarifications on the extrinsic convention and noise parameter starting points.

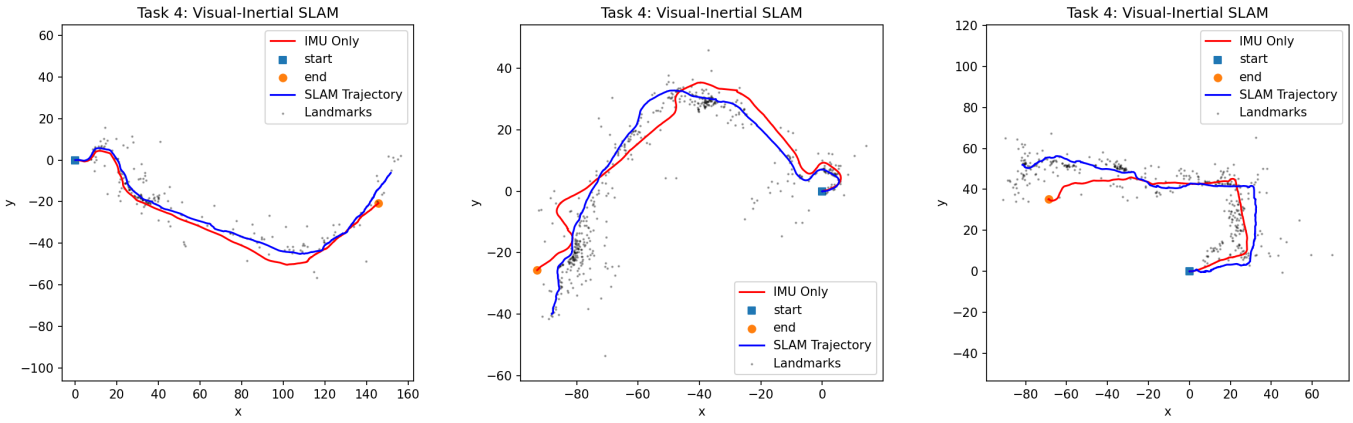
## References

[1] N. Atanasov, "ECE 276A Lecture 12: Visual-Inertial SLAM," University of California San Diego, 2026.
[2] N. Atanasov, "ECE 276A Lecture 10: Extended and Unscented Kalman Filter," University of California San Diego, 2026.
[3] J. Shi and C. Tomasi, "Good features to track," in *Proc. IEEE CVPR*, 1994.
[4] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Proc. IJCAI*, 1981.




(a) Dataset00 (b) Dataset01 (c) Dataset02

Fig. 3: Task 3: Landmark mapping with fixed trajectory. Red = IMU trajectory, black dots = estimated landmark $x$-$y$ positions in the world frame.




(a) Dataset00 (b) Dataset01 (c) Dataset02

Fig. 4: Task 4: Visual-Inertial SLAM. Red = IMU-only trajectory, blue = SLAM-corrected trajectory, gray dots = estimated landmark $x$-$y$ positions in the world frame.

## APPENDIX

### A. Project Structure

```
ECE276A_PR3/
  code/
    main.py              # Driver script (Tasks 1-4)
    ekf_predict.py       # Task 1: IMU EKF prediction
    ekf_mapping.py       # Task 3: Landmark mapping
    ekf_slam.py          # Task 4: Visual-inertial
    SLAM
    feature_tracker.py # Task 2: Feature detection/
    tracking
    pr3_utils.py         # Provided utilities
  data/
    dataset00/
    dataset01/
    dataset02/
```

### B. Dependencies

```
pip install numpy scipy matplotlib transforms3d
    opencv-python
```

### C. Running the Code

**Run all tasks on a dataset:**

```
cd code
python main.py --dataset 00
python main.py --dataset 01
python main.py --dataset 02
```

For dataset02, features are automatically computed on first run and cached to `dataset02_features_computed.npy`. Subsequent runs load from cache. To force recomputation:

```
python main.py --dataset 02 --compute-features
```

**Skip specific tasks for testing:**

```
python main.py --dataset 00 --no-task3
python main.py --dataset 00 --no-task4
```

**Key tunable parameters:**

```
python main.py --dataset 00 \
    --W_trans 1e-6 --W_rot 1e-6 \
    --V 2.0 --V_t3 4.0 \
    --max_depth 50.0 --min_disp 2.0 \
    --innov_gate_t4 20.0 --xi_gate 0.3
```

All figures saved as `task{1,2,3,4}_dataset{00,01,02}.png` in `code/`.