Project 3: Visual-Inertial SLAM

Justin Volheim
Electrical and Computer Engineering
Department UCSD
jvolheim@ucsd.edu

Abstract

This project addresses the implementation of visual-inertial simultaneous localization and mapping (SLAM) using an extended Kalman filter (EKF) framework. Leveraging synchronized data from an inertial measurement unit (IMU) and a stereo camera. Our objective is to achieve accurate pose estimation of the IMU and mapping of static landmarks. The proposed approach integrates EKF prediction and update steps to accommodate IMU localization and landmark mapping, culminating in a robust visual-inertial SLAM algorithm from basic principles that is easily generalized for practical use.

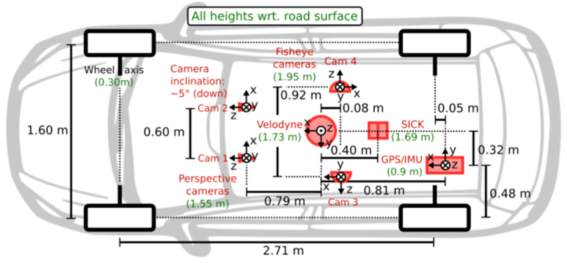
Introduction

Simultaneous Localization and Mapping (SLAM) stands as a pivotal challenge in robotics, necessitating the concurrent estimation of a robot's pose and the environment's map solely from onboard sensor observations. This challenge applies to a wide-reaching list of applications from autonomous cars to industrial cleaners. With the proliferation of SLAM methodologies ranging from particle filters to neural networks, our focus lies on an extended Kalman filter (EKF) approach augmented with visual-inertial measurements. This report outlines our endeavor to implement visual-inertial SLAM, leveraging data from an IMU and stereo camera. By combining IMU-based localization with landmark mapping from visual features, we aim to furnish a comprehensive solution for real-time environment perception. Our methodology will be applied to several real-world datasets in this report that will demonstrate its effectiveness and practical applications.

Problem Formulation

The task of this project is to utilize the IMU and stereo camera landmarks to estimate the trajectory and map the landmark locations using an EKF methodology. This problem can be broken down into 5 sections for easy understanding (Data, Platform Localization, Landmark Localization, Mapping, SLAM).

Data: The data utilized in this task were collected from the following platform which is a car constraining an IMU and stereo camera setup.



This data was then preprocessed for our use to be formatted in the following way.

Landmark pixels ->
(4, landmarks, timesteps)

An example of these landmarks can be seen here.

With the IMU measurements as follows.

Linear velocity -> (3, timesteps)
Angular velocity -> (3, timesteps)

These 3 data samples are correlated together using their associated timesteps.

Platform Localization: The localization of the platform in the world frame will be achieved using the prediction step of the EKF methodology and the IMU values. Four our task the following equation can be used to estimates the next pose based on current pose and the IMU values.

$$T_{t+1|t} = T_{t|t} * \exp\left(\tau_t * \hat{u}_t\right)$$

Where $T_{t|t}$ represent the current pose, $T_{t+1|t}$ represents the next pose, and û is as follows for w, v being the angular and linear velocity respectively.

$$\hat{u}_t = \begin{bmatrix} \widehat{w}_t & v_t \\ 0^T & 0 \end{bmatrix}$$

$$\widehat{w_t} = \begin{bmatrix} 0 & -w_3 & w_2 \\ w_3 & 0 & -w_1 \\ -w_2 & w_1 & 0 \end{bmatrix}$$

Lastly for this equation the value τ_t represents the difference in time between poses.

Mapping: The task of mapping the environment as we traverse a path can be accomplished by estimating the locations of the landmarks in the world frame given a path. To do this we can convert the stereo pixel values to the world frame and update their locations using the EKF update step. To convert stereo pixel values to the camera frame first we can use the following equations. First we separate each stereo camera observation into the pixels from each image as so.

$$z_1 = \begin{bmatrix} u_L \\ v_L \\ 1 \end{bmatrix}, \qquad z_2 = \begin{bmatrix} u_R \\ v_R \\ 1 \end{bmatrix}$$

Now that we have these values, we can calculate the following

$$a = R^T p - e_3^T R^T p z_2$$

$$b = R^T z_1 - e_3^T R^T z_1 z_2$$

Finally, we have the xyz points in the camera frame given from the following equation

$$M = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \frac{a^T a}{a^T b} z_1$$

Now that we have these values in the camera frame, we can use the following to transition them to the world frame using the following equation.

$$Landmark_{world} = wTc \mathbin{@} M$$

With these landmark locations we can use the following EKF update equations to improve our estimate. Where μ_{t+1} represents the predicted landmark locations and Σ_{t+1} represents the updated covariances between the landmarks.

$$\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_t + K_{t+1}\left(\mathbf{z}_{t+1} - \tilde{\mathbf{z}}_{t+1}\right)$$
$$\Sigma_{t+1} = (I - K_{t+1}H_{t+1})\Sigma_t$$

With these equations we will now describe what each of the values are. The μ_t value represents the current estimated locations of the landmarks in the world frame. z_t represents the raw pixel coordinates at time t from the stereo camera. H_t is defined as follows.

$$H_{t+1,i,j} = \begin{cases} K_s \frac{d\pi}{d\mathbf{q}} \left(oT_I T_{t+1}^{-1} \underline{\mu}_{t,j} \right) oT_I T_{t+1}^{-1} P^\top, & \text{if } \Delta_t(j) = i, \\ \mathbf{0}, & \text{otherwise} \end{cases}$$

Where, T_{t+1}^{-1} is the transformation from the world frame to the camera frame. oTi is the transformation from the camera frame to the optical frame. $P = [I\ 0] \in R^{3x4}$.

$$K_s := \begin{bmatrix} fs_u & 0 & c_u & 0 \\ 0 & fs_v & c_v & 0 \\ fs_u & 0 & c_u & -fs_u b \\ 0 & fs_v & c_v & 0 \end{bmatrix}$$

$$\frac{d\pi}{d\mathbf{q}}(\mathbf{q}) = \frac{1}{q_3} \begin{bmatrix} 1 & 0 & -\frac{q_1}{q_3} & 0 \\ 0 & 1 & -\frac{q_2}{q_3} & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & -\frac{q_4}{q_3} & 1 \end{bmatrix} \in \mathbb{R}^{4\times 4}$$

The delta function associates landmarks to the timesteps that they are observed in and $\mu_{t,j}$ is the landmark observed at that timestep. The Kalman gain equation is as follows. Where V a constant representing the noise.

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

The pixel estimate z is.

$$\tilde{\mathbf{z}}_{t+1,i} = K_s \pi \left(oT_I T_{t+1}^{-1} \underline{\mu}_{t,j} \right) \in \mathbb{R}^4 \qquad \text{for } i = 1, \ldots, N_{t+1}$$

Where, N_{t+1} is the number of observed landmarks for that timestep.

SLAM: To create a SLAM algorithm with EKF we simply need to build an equation that combines the previous two outlines into one. In doing this we will predict the location of the platforms then update its location while updating the locations of the landmarks. Here I will show a single iteration step for the slam algorithm which would simply need to be iterated over all timesteps to create the full SLAM algorithm.

Loop begin:

The first step of the algorithm is to run the prediction step for the landmarks and pose as shown in these equations defined in previous sections.

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

$$M = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \frac{a^T a}{a^T b} z_1$$

We also need to update our sigma matrix as follows.

$$\Sigma_{t+1|t} = \begin{bmatrix} \Sigma_{t|t}^{MM} & \Sigma_{t|t}^{MI} F^\top \\ F \Sigma_{t|t}^{IM} & F \Sigma_{t|t}^{II} F^\top + W \end{bmatrix}$$

Where W is (6x6) noise $\Sigma^{MI} \epsilon R^{3M\ x\ 6}$, $\Sigma^{IM} \epsilon R^{6\ x\ 3M}$, $\Sigma^{II} \epsilon R^{6\ x\ 6}$, $\Sigma^{MM} \epsilon R^{3M\ x\ 3M}$ and F is defined as follows.

$$F = exp(-\tau \hat{\hat{u}}_t)$$

With these initializations we need to create the EKF update step. The first step is to create the combined jacobian for the poses and landmarks defined as H in the following equation.

$$H_{t+1} = [H_{map}, H_{loc}] \epsilon R^{3Nt\ x\ 3M+6}$$

Where H_{map} is the jacobian defined in the mapping section and H_{loc} is defined here $H_{loc} = H$.

$$H = \begin{bmatrix} H_{t+1,1} \\ \vdots \\ H_{t+1,N_{t+1}} \end{bmatrix}$$

.

$$H_{t+1,i} = -K_s \frac{d\pi}{d\mathbf{q}} \left({}_O T_I \boldsymbol{\mu}_{t+1|t}^{-1} \underline{\mathbf{m}}_j \right) {}_O T_I \left(\boldsymbol{\mu}_{t+1|t}^{-1} \underline{\mathbf{m}}_j \right)^{\odot} \in \mathbb{R}^{4\times 6}$$

$$\begin{bmatrix} \mathbf{s} \\ 1 \end{bmatrix}^{\odot} := \begin{bmatrix} I & -\hat{\mathbf{s}} \\ 0 & 0 \end{bmatrix} \in \mathbb{R}^{4\times 6}$$

Where the values are defined the same as in the mapping section except for the landmarks which are called m in this equation and the pose is μ_t.

Now that we have H_{t+1}, we can define the Kalman gain using that H and this equation described in the mapping section.

$$K_{t+1} = \Sigma_{t+1|t} H_{t+1}^{\top} \left(H_{t+1} \Sigma_{t+1|t} H_{t+1}^{\top} + I \otimes V \right)^{-1}$$

Using this new Kalman gain we can get the updated covariances using this equation.

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

We can also get the update values utilizing this equation.

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

Where the z values are defined the same as in the mapping section.
With this new estimated update parameter, A from EKF we can split it into two parts and update both the pose and landmarks respectively. The split is simply that the landmark update is the top 3M and the bottom 6 are the pose update.

$$\mathrm{A} = [m, u \in R^{6\,x\,1}]^T \in R^{3M+6\,x\,1}$$

The equation to update the pose is as follows

$$T_{t+1|t+1} = T_{t+1|t} * \exp\,(\tau_t * \hat{u})$$

The landmarks update

$$\mu_{t+1\,|t+1} = \mu_{t+1\,|t} + m$$

: *Loop End*

Running this loop for all time steps will result in the SLAM EKF algorithm.

Technical Approach

In this technical approach we will give a more concrete overview of how the equations in the problem formulation utilized and specifically are applied to our project.

Time complexity reduction:
As the algorithms discussing for this project can take an exceptional amount of time, we have opted to utilize two methods outside of standard code optimization to improve the speed. The first of these methods was to utilize sparse matrix tools which allow for faster computations when dealing with matrixes like ours which have a majority of zero values. This is very beneficial in landmark mapping but becomes less impactful for the SLAM algorithm as the sparsity is greatly reduced over time. The second method we employed which cut the runtime of the smaller dataset ie(3) from 15 hours down to 30 min was to down sample the landmarks. This is possible because we only need roughly 10 -20 landmarks per iteration to accurately predict and update the values. For our datasets we have chosen to down sample uniformly to 1500 landmarks for dataset 3 and 3000 for dataset 10.

IMU localization via EKF prediction:
To utilize the IMU values for localization we will first stack them such that û = [v, w] ^T with this new stacked (6 x Number of timesteps) matrix we can use

the following equation defined in the problem formulation section.

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

To utilize this equation, we will initialize the first pose with the identity matrix and loop over all timesteps until we have the poses in the world frame at each timestep. This list of pose values represents the odometry path that the car took.

Landmark mapping via EKF update:

To create a mapping utilizing the landmark pixel values from the stereo camera we will use the following equation defined in the problem formulation section to transform the pixel values to the camera frame.

$$M = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \frac{a^T a}{a^T b} z_1$$

With initializations of R = I and p=[b,0,0]^T

Now that we have the landmarks in the camera frame, we will assume that the poses calculated in the previous section are correct in order to convert the landmarks from the camera frame to the world frame using the following equation.

$$Landmark_{world} = wTc\ @\ Landmark_{camera}$$

Using these new landmark locations, we can iterate over the following equations defined in the problem formulation section in a loop for all timesteps to achieve an EKF update step.

$$\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_t + K_{t+1}(\mathbf{z}_{t+1} - \tilde{\mathbf{z}}_{t+1})$$
$$\Sigma_{t+1} = (I - K_{t+1}H_{t+1})\Sigma_t$$

The resultant μ values after all iterations will be the updated landmark locations.

Visual-inertial SLAM:

To utilize all the data and combine it into a working slam algorithm we will utilize the equations shown here in a loop for all time steps Note: all equations and variables are defined in problem formulation they are shown here simply for reference.

Step 1 Initialize noise:

The noise utilized in our results uses V=4 and W = diag([1e-4,1e-4,1e-4,1e-8,1e-8,1e-8])

Loop for all time steps:

Step 2 Compute predictions:

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

$$M = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \frac{a^T a}{a^T b} z_1$$

Initializations defined in previous step

Step 3 Update sigma:

$$\Sigma_{t+1|t} = \begin{bmatrix} \Sigma_{t|t}^{MM} & \Sigma_{t|t}^{MI} F^\top \\ F\Sigma_{t|t}^{IM} & F\Sigma_{t|t}^{II} F^\top + W \end{bmatrix}$$

Step 4 Compute Jacobian:

$$H_{t+1} = [H_{map}, H_{loc}] \epsilon R^{3Nt\ x\ 3M+6}$$

Step 5 Compute Kalman Gain:

$$K_{t+1} = \Sigma_{t+1|t} H_{t+1}^\top \left(H_{t+1}\Sigma_{t+1|t}H_{t+1}^\top + I \otimes V\right)^{-1}$$

Step 6 Update sigma:

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

Step 7 Get update metric:

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

Step 8 Split update metric:

$$\mathrm{A} = [m, u \in R^{6\,x\,1}]^T \in R^{3M+6\,x\,1}$$

Step 9 Compute update for landmarks and poses:

$$T_{t+1|t+1} = T_{t+1|t} * \exp(\tau_t * \hat{u})$$

$$\mu_{t+1\,|t+1} = \mu_{t+1\,|t} + m$$

Step 10 Loop back to step 2 for next iteration.

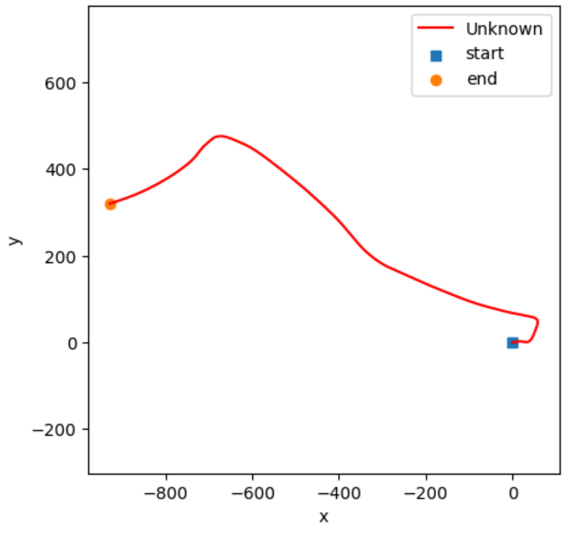
Once all iterations have been complete the result will be an updated trajectory with new landmark locations that better approximate the actual path and environment. Note this algorithm is very sensitive to the noise values in W so if you're seeing poor results tuning those values will likely improve the result.

Results

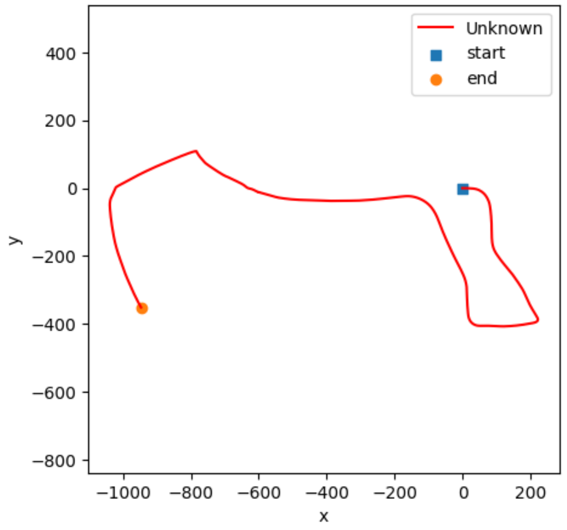
The results in this section show the outputs for each section of the technical approach on two different datasets.

IMU localization via EKF prediction: For the EKF prediction the following two graphs show the estimated path of travel for the platform with respect to dataset 3 and 10 respectively.

Dataset 3:



Dataset 10:

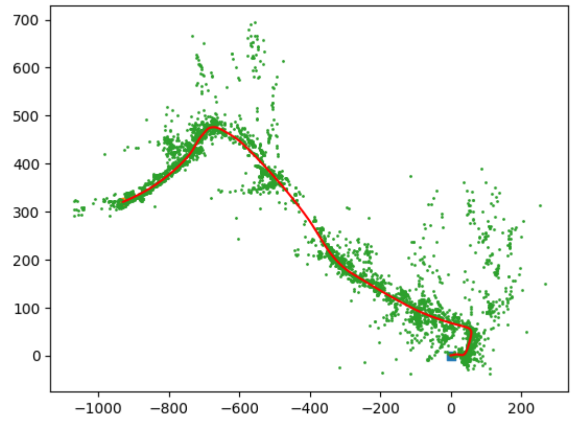


Analysis: Based on comparing these path estimates to the video it does a decent job at estimating the localization of the car throughout its movement.

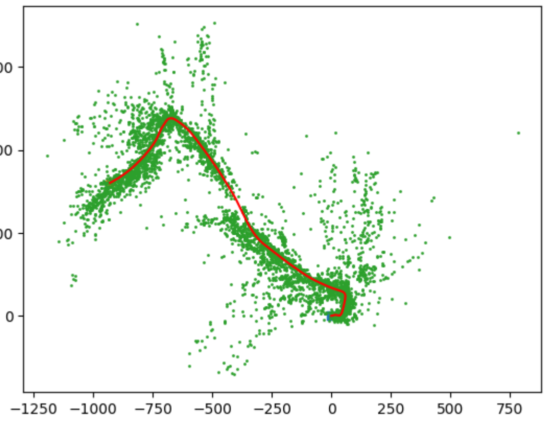
Landmark mapping via EKF update: For this section the following graphs show the landmarks prior to the EKF update and after the update respectively.

Dataset 3:

Prior to EKF update

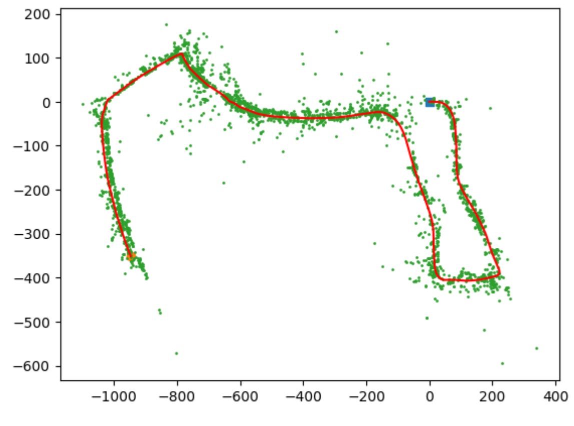


Post EKF update

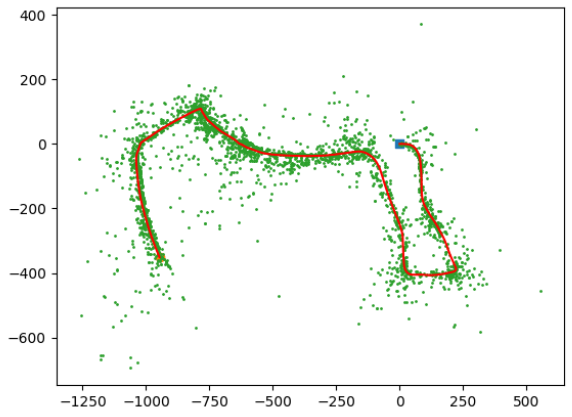


Dataset 10:

Prior to EKF update



Post EKF update

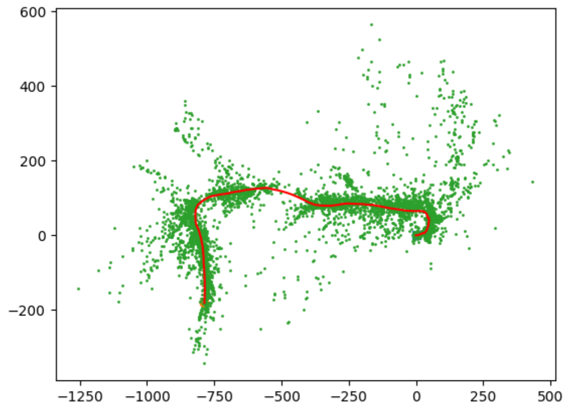


Analysis: Based on these plots we can see that the points which were clustered tightly to the path have been spread out slightly. This aligns with our assumptions for the path traversal landmark locations from the videos.

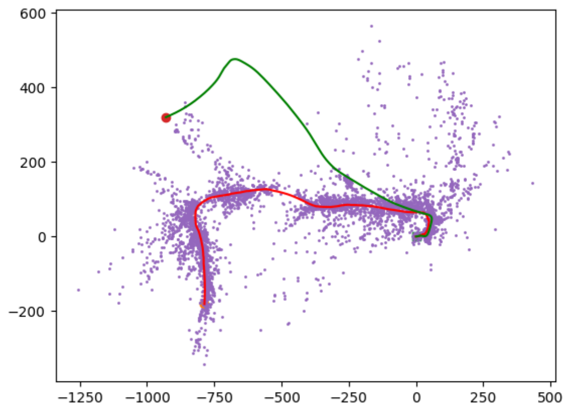
Visual-inertial SLAM: For this section the images show the new slam generated path and landmark locations using EKF slam. As well as an image containing the old path for reference.

Dataset 3:

SLAM Estimates

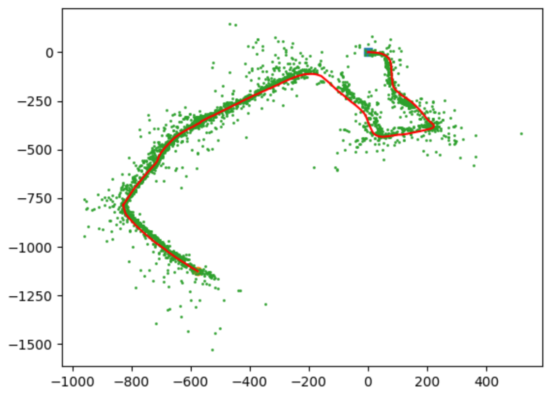


Reference Path

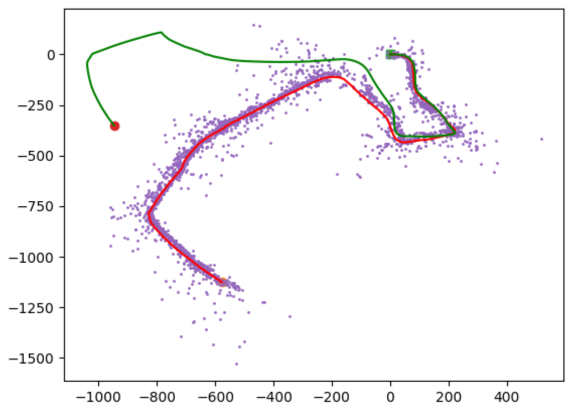


Dataset 10:

SLAM Estimates



Reference Path



Analysis: Based on the new path in dataset 3 when comparing it to the video it does a much better job at approximating the path. However, when comparing the video for dataset 10 it seems to not have handled the sharp corner displayed in the original path well. This is likely due to insufficient tuning of the weights.

Conclusion

In conclusion, our implementation of visual-inertial SLAM using an extended Kalman filter (EKF) framework has demonstrated notable results on two different datasets. The technical approach, consisting of IMU localization via EKF prediction, landmark mapping via EKF update, and visual-inertial SLAM, has provided valuable insights into robotic perception and mapping.

The results reveal the efficacy of our approach in estimating the platform's path of travel and mapping landmarks. Analysis of the IMU localization via EKF prediction shows decent localization estimates aligned with observed movement in the videos. Landmark mapping via EKF update demonstrates the dispersion of points after the update, consistent with assumptions about landmark locations relative to path traversal. Additionally, visual-inertial SLAM outputs new path estimates and landmark locations, with dataset 3 showing improved path approximation compared to the predictions from part 1, while dataset 10 exhibits challenges, possibly due to insufficient parameter tuning.

These findings underscore the potential of visual-inertial SLAM for real-world applications, while also highlighting areas for further refinement and optimization. By addressing these challenges, future iterations of visual-inertial SLAM systems can achieve enhanced accuracy and robustness, advancing the capabilities of robotics and autonomous systems.