ECE 276A PR3: VI-SLAM

Eric Megrabov (A12177906)

***Abstract*—The focus of this project is to implement Visual Inertial Simultaneous Localization and Mapping (also known as VI-SLAM) using the extended Kalman Filter by processing raw odometry in the form of IMU measurements and landmarks in a map in the form of stereo camera images. A car performs some movement and the objective is to map its environment while determining where the car lies in that environment.**

I. Introduction

As the name implies, the purpose of Simultaneous Localization and Mapping is to develop and update a map of an environment while at the same time trying to determine where the robot is in that map. In this case, Visual-Inertial SLAM is a subset of SLAM in which measurements of inertia are given for a car and stereo camera images are provided to represent points in the environment. The problem lies in the fact that it is difficult to know where the robot is without knowing the map, and it is difficult to know the map without knowing where the robot is. To solve this problem, one strategy is to use an Extended Kalman Filter, also known as EKF. As a result, the use of poses must be incorporated (also known as hypotheses of the car's location) in order to do a chain of prediction and updating until convergence is found. SLAM is extraordinarily useful for a plethora of robotics tasks, such as autonomous or self-driving cars, ocean mapping research, drone swarms, and many others.

II. Problem Formulation

In this project, the input is provided as raw measurements from a car equipped with an IMU and stereo camera.

- The IMU measurements are provided in the form of linear velocity v_t and angular velocity ω_t, both of which are in $\mathbb{R}^3$.
- Stereo camera images are provided in the form [ul, ur, vl, vr] as pixel coordinates $z_t \in \mathbb{R}^{4xM}$ where M is the number of landmarks in the dataset. If landmark i is not observed at a given time t, then $z_{i,t}$=[-1, -1, -1, -1].
- Time stamps are provided in UNIX standard.
- Intrinsic calibration consisting of stereo baseline b and camera calibration matrix:

$$K = \begin{bmatrix} fs_u & 0 & c_u \\ 0 & fs_v & c_v \\ 0 & 0 & 1 \end{bmatrix}$$

- Extrinsic calibration ${}_{camera}T_{IMU} \in SE(3)$

The objective is to determine where the car is and determine a map of the environment. Similarly to programming assignment 2, this can be approached by maximizing the joint distribution:

$$p(\mathbf{x}_{0:T}, \mathbf{m}, \mathbf{z}_{0:T}, \mathbf{u}_{0:T-1}) = \underbrace{p_{0|0}(\mathbf{x}_0, \mathbf{m})}_{\text{prior}} \prod_{t=0}^{T} \underbrace{p_h(\mathbf{z}_t \mid \mathbf{x}_t, \mathbf{m})}_{\text{observation model}} \prod_{t=1}^{T} \underbrace{p_f(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \mathbf{u}_{t-1})}_{\text{motion model}}$$

However, there are some new assumptions to be made for the Extended Kalman Filter. The prior $\mathbf{p_{0|0}}$ are Gaussian, observation model $\mathbf{p_h}$ is affected by Gaussian noise, and the motion model $\mathbf{p_f}$ is affected by Gaussian noise. The process noise and as measurement noise are independent of each other and of the state as well as across time. The joint distribution **p** is over the car states, observations, controls, and the map of the world. The variables in the probability mass function are as follows:

- $x_{0:T}$: defines car state at steps 0 to T
- m: represents the map of the world
- $z_{0:T}$: defines observations at steps 0 to T
- $u_{0:T-1}$: defines control inputs at steps 0 to T-1

Since we want to employ a Kalman Filter but our observation model and motion model are not necessarily linear, we must use an Extended Kalman Filter, which forces the problem to be linear by using a first order Taylor series approximation to simulate a linear motion and observation model. The equations for this will be explained in further detail in the Technical Approach section.

III. Technical Approach

The project code was written using Python 3, mainly in Jupyter Notebook.

A. IMU Localization

In this part of the project, we want to split VI-SLAM into a separate part that consists of only localizing the car based on the IMU information using the EKF prediction step. For IMU localization, we want to estimate the inverse IMU pose U_t in SE(3). To begin, I set U_t to be modeled by a Gaussian prior distribution centered around a mean of zero, with a diagonal covariance with small entries along the diagonal. The prior is:

$$U_t | \mathbf{z}_{0:t}, \mathbf{u}_{0:t-1} \sim \mathcal{N}(\boldsymbol{\mu}_{t|t}, \Sigma_{t|t}) \text{ with } \boldsymbol{\mu}_{t|t} \in SE(3) \text{ and } \Sigma_{t|t} \in \mathbb{R}^{6\times 6}$$

The motion model is described as:

$$U_{t+1} = \exp\left(-\tau\left((\mathbf{u}_t + \mathbf{w}_t)\right)^{\wedge}\right) U_t \qquad \mathbf{u}_t := \begin{bmatrix} \mathbf{v}_t \\ \boldsymbol{\omega}_t \end{bmatrix} \in \mathbb{R}^6$$

Note that τ is time discretization, which can be calculated by taking time t and subtracting time t-1 from it. The u_t term

encompasses the kinematics terms for linear and angular velocity.

By splitting the pose kinematics $\dot{T} = -(\hat{\mathbf{u}} + \hat{\mathbf{w}})T$ into nominal and perturbation kinematics using a small perturbation in se(3), we obtain:

$$\text{nominal}: \quad \dot{\boldsymbol{\mu}} = -\hat{\mathbf{u}}\boldsymbol{\mu}$$
$$\text{perturbation}: \quad \dot{\delta\boldsymbol{\mu}} = -\hat{\mathbf{u}}\delta\boldsymbol{\mu} + \mathbf{w} \qquad \hat{\mathbf{u}} := \begin{bmatrix} \hat{\omega} & \hat{\mathbf{v}} \\ 0 & \hat{\omega} \end{bmatrix} \in \mathbb{R}^{6\times 6}$$

which separates the effect of noise from motion. This allows the motion model to be rewritten in terms of the nominal kinematics of the mean of T_t :

$$\boldsymbol{\mu}_{t+1|t} = \exp(-\tau\hat{\mathbf{u}}_t)\,\boldsymbol{\mu}_{t|t}$$
$$\delta\boldsymbol{\mu}_{t+1|t} = \exp\left(-\tau\hat{\mathbf{u}}_t\right)\delta\boldsymbol{\mu}_{t|t} + \mathbf{w}_t$$

Ultimately, this yields the prediction step for the next mean and covariance to help with localization of the vehicle in the environment:

$$\boldsymbol{\mu}_{t+1|t} = \exp(-\tau\hat{\mathbf{u}}_t)\,\boldsymbol{\mu}_{t|t}$$
$$\Sigma_{t+1|t} = \mathbb{E}[\delta\boldsymbol{\mu}_{t+1|t}\delta\boldsymbol{\mu}_{t+1|t}^{\top}] = \exp\left(-\tau\hat{\mathbf{u}}_t\right)\Sigma_{t|t}\exp\left(-\tau\hat{\mathbf{u}}_t\right)^{\top} + W$$

where

$$\mathbf{u}_t := \begin{bmatrix} \mathbf{v}_t \\ \omega_t \end{bmatrix} \in \mathbb{R}^6 \quad \hat{\mathbf{u}}_t := \begin{bmatrix} \hat{\omega}_t & \mathbf{v}_t \\ \mathbf{0}^{\top} & 0 \end{bmatrix} \in \mathbb{R}^{4\times 4} \quad \hat{\mathbf{u}}_t := \begin{bmatrix} \hat{\omega}_t & \hat{\mathbf{v}}_t \\ 0 & \hat{\omega}_t \end{bmatrix} \in \mathbb{R}^{6\times 6}$$

These equations are what will be used in order to perform a repeated prediction step which will allow us to estimate the pose $T_t \in SE(3)$. Also note that the hat map operator is used extensively in these equations. This operator performs the following transformation of a vector in $\mathbb{R}^3$ to a skew-symmetric matrix:

$$[\mathbf{x}]_{\times} := \begin{bmatrix} 0 & -x_3 & x_2 \\ x_3 & 0 & -x_1 \\ -x_2 & x_1 & 0 \end{bmatrix}$$

From here, the process of implementing localization is mostly straightforward given the equations necessary for determining the required mean and covariance. First, I obtain the time discretization τ by subtracting timestep t-1 from timestep t. Then, I obtain each necessary term and perform the respective calculations required in order to get the mean and covariances. Note that for plotting, it is necessary to invert U_t since this provides the pose in the IMU frame, but we are plotting in the world frame. Please refer to the results section to see the trajectory for a few different datasets.

B. Landmark Mapping

In this part of the project, we assume that the IMU trajectory that was created in part A is completely correct and now simply estimate where the landmarks are. Here, we use an EKF with the unknown landmark position **m** which is modeled as a Gaussian distribution. The distribution can be described as follows:

$$\mathbf{m} \mid \mathbf{z}_{0:t} \sim \mathcal{N}(\boldsymbol{\mu}_t, \Sigma_t) \text{ with } \boldsymbol{\mu}_t \in \mathbb{R}^{3M} \text{ and } \Sigma_t \in \mathbb{R}^{3M\times 3M}$$

Note the *M* in the dimension indicates the number of total landmarks seen in a dataset. Let N_t be the number of landmarks seen at a given timestep. To determine which landmarks are observed at a timestep, I filter out all landmarks with a reading of [-1, -1, -1, -1] for that timestep. Now, given the values provided by the input, we can create the calibration matrix M for the stereo camera using the following formula:

$$\begin{bmatrix} u_L \\ v_L \\ d \end{bmatrix} = \begin{bmatrix} fs_u & 0 & c_u & 0 \\ 0 & fs_v & c_v & 0 \\ 0 & 0 & 0 & fs_u b \end{bmatrix} \frac{1}{z} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \qquad \begin{bmatrix} x \\ y \\ z \end{bmatrix} = {}_oR_r R^{\top}(\mathbf{m} - \mathbf{p})$$

In addition, we know that disparity is:

$$d = u_L - u_R = \frac{1}{z} fs_u b,$$

so scalar z can be obtained using:

$$z = \frac{1}{ul - ur} fs_u b$$

Since we are given vector **z** = [ul, vl, ur, vr] from the stereo camera input, the system of equations can be solved above to obtain:

$$x = \frac{z(ul - cu)}{fs_u}, \quad y = \frac{z(vl - cv)}{fs_v}$$

Let's call q = [x, y, z, 1]$^{\top}$.
Now we can obtain m, which is the landmark in the world frame, by using $m = U_t * ({}_{camera}T_{IMU})^{-1} * q$.

After obtaining m, assign the mean to be equal to this m if the landmark and set a small initial covariance if it has never been observed before. However, if it *has* been observed before, I perform the update step, which is performed with the following equations. Note that the prior mean and covariance for the mapping are:

$$\boldsymbol{\mu}_t \in \mathbb{R}^{3M} \text{ and } \Sigma_t \in \mathbb{R}^{3M\times 3M}$$

The predicted observation is:

$$\tilde{\mathbf{z}}_{t,i} := M\pi\left({}_O T_I U_t \underline{\boldsymbol{\mu}}_{t,j}\right) \in \mathbb{R}^4 \qquad \text{for } i = 1, \ldots, N_t$$

To get this, it is necessary to use the canonical projection:

$$\pi(\mathbf{q}) := \frac{1}{q_3}\mathbf{q} \in \mathbb{R}^4$$

The Jacobian of the predicted observation above with respect to m (corresponding to landmark j) evaluated at $\mu_{t,j}$ is:

$$H_{t,i,j} = \begin{cases} M\frac{d\pi}{d\mathbf{q}}\left({}_O T_I U_t \underline{\boldsymbol{\mu}}_{t,j}\right) {}_O T_I U_t P^{\top} & \text{if observation } i \text{ corresponds to landmark } j \text{ at time } t \\ \mathbf{0} \in \mathbb{R}^{4\times 3} & \text{otherwise} \end{cases}$$

To obtain H, it is necessary to use the derivative of the projection, which is:

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

In addition, P is

$$P := \begin{bmatrix} I & \mathbf{0} \end{bmatrix} \in \mathbb{R}^{2\times 3}$$

Finally, the update step equations are:

$$K_t = \Sigma_t H_t^\top \left(H_t \Sigma_t H_t^\top + I \otimes V\right)^{-1}$$
$$\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_t + K_t\left(\mathbf{z}_t - \tilde{\mathbf{z}}_t\right)$$
$$\Sigma_{t+1} = (I - K_t H_t)\Sigma_t$$
$$I \otimes V := \begin{bmatrix} V & & \\ & \ddots & \\ & & V \end{bmatrix}$$

Where K_t is the Kalman Gain and V is a Gaussian noise matrix that is initialized to a high value as this yielded the best results. Note that the Jacobian H is very large ($4N_t$ x 3k, where k is the number of observed landmarks up to the current time step) and the covariance is also very large (3k x 3k), but since the landmarks are independent, we can use the property of a diagonal matrix to simply do inversion on each block matrix rather than the entire, extremely large matrix. This allows for much faster calculation in the update step.

These are chained in succession over all timesteps and all landmarks seen per timestep until each landmark has a predicted location in the world frame.

C. VI-SLAM

In the final part of this project, the objective is to put together the IMU prediction step from part a and the landmark update step from part b, as well as the IMU update step using the stereo camera images, which would allow the creation of a Visual-Inertial SLAM algorithm. In the mapping case, the landmark states were completely independent. Now, there is correlation between the landmark states because the observations are correlated via the IMU state. As a result, making an observation would provide information about the position of a given landmark which would also describe the pose of the camera, thus causing cross correlation terms to appear and thus breaking the independence assumption. To counteract this, we can rewrite the problem to be a joint one as follows.

$x_t' = \begin{bmatrix} P \\ L \end{bmatrix} \sim N(\begin{bmatrix} \mu_{pt} \\ \mu_{Lt} \end{bmatrix}, \begin{bmatrix} \Sigma_{PP} & \Sigma_{LP} \\ \Sigma_{LP}^T & \Sigma_{LL} \end{bmatrix})$ where P corresponds to car pose and L corresponds to landmarks.

New predict step (only for localization because the landmarks are stationary so it is unnecessary for the mapping):

$$x_{t+1}' = \underbrace{\begin{bmatrix} \exp(-\tau\hat{u}_t) & I \end{bmatrix}}_{F} \begin{bmatrix} P \\ L \end{bmatrix}$$

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

$$Q = \frac{dF}{dx} = \begin{bmatrix} \exp(-\tau\hat{u}_t) & 0 \\ 0 & I \end{bmatrix}$$

$$\Sigma_{pt+1|t} = Q\Sigma Q + \begin{bmatrix} W & 0 \\ 0 & 0 \end{bmatrix}$$

$$\Sigma_{pt+1|t} = \begin{bmatrix} \exp(-\tau\hat{u}_t)\Sigma_{PP}\exp(-\tau\hat{u}_t)^T + W & \exp(-\tau\hat{u}_t)\Sigma_{LP} \\ \exp(-\tau\hat{u}_t)^T{\Sigma_{LP}}^T & \Sigma_{MM} \end{bmatrix}$$

The update steps for mapping, which are the same as before, are:

$$K_t = \Sigma_t H_t^\top \left(H_t \Sigma_t H_t^\top + I \otimes V\right)^{-1}$$
$$\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_t + K_t\left(\mathbf{z}_t - \tilde{\mathbf{z}}_t\right)$$
$$\Sigma_{t+1} = (I - K_t H_t)\Sigma_t$$

And the newly-introduced update steps for localization are:

$$K_{t+1|t} = \Sigma_{t+1|t} H_{t+1|t}^\top \left(H_{t+1|t}\Sigma_{t+1|t}H_{t+1|t}^\top + I \otimes V\right)^{-1}$$
$$\mu_{t+1|t+1} = \exp\left(\left(K_{t+1|t}(\mathbf{z}_{t+1} - \tilde{\mathbf{z}}_{t+1})\right)^\wedge\right)\mu_{t+1|t}$$
$$\Sigma_{t+1|t+1} = (I - K_{t+1|t}H_{t+1|t})\Sigma_{t+1|t}$$
$$H_{t+1|t} = \begin{bmatrix} H_{1,t+1|t} \\ \vdots \\ H_{N_{t+1},t+1|t} \end{bmatrix}$$

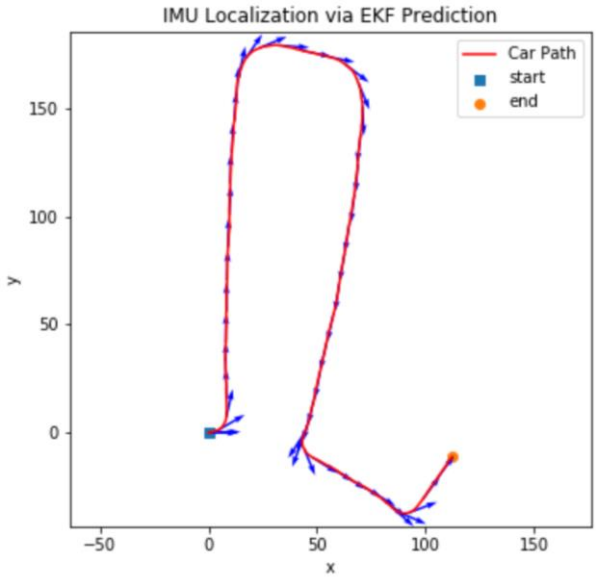
Now that the plethora of required equations have been outlined, it is possible to perform VI-SLAM. First, perform the predict step using the newly acquired predict equations to get the mean and covariance for the localization. From there, the landmarks are processed as before, but now, there are cross covariance terms. As a result, it becomes impossible to perform block inversion along the diagonal because the non-diagonal terms are no longer zero. Therefore, a very large covariance matrix and Jacobian must be maintained. This increases runtimes to be prohibitively long. To counteract this, I significantly downsampled the amount of landmarks that are used in the calculation.

The original covariance matrix is 3Mx3M. The way I downsampled was I preserved the top 100 features that appeared most frequently. This ensures that highly-informative landmarks are used. However, it got rid of too many landmarks that were not seen much, such as several along straight trajectories. As a result, I also chose to use every 10th landmark in the set. So, for the first dataset, there were originally 3950 landmarks in total, causing the covariance matrix to be of size 11850x11850. After downsampling, there are 495 used landmarks. The covariance matrix becomes 1485x1485, which has about 64 times less elements. From here, the respective update steps are performed for localization as well as mapping using the equations listed. The results can be seen in the following section.

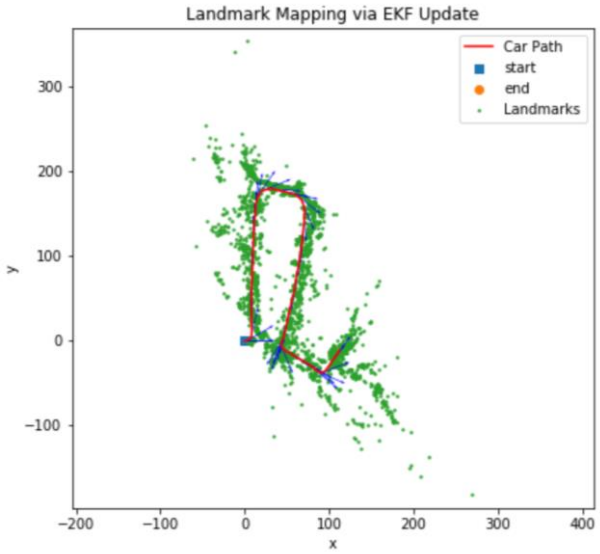
IV. Results

Here, I will discuss the outcome of my localization using predict, mapping using update, and VI-SLAM. One important result observed was that EKF generalized decently well with much more resistance to change in hyperparameter tuning.

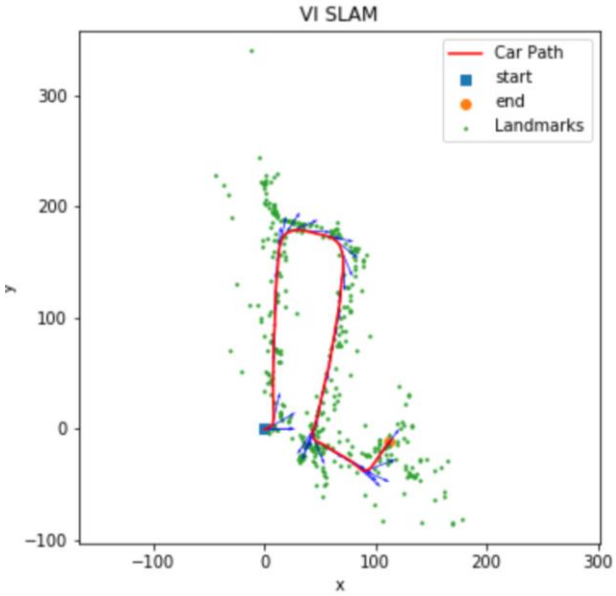
A. Dataset 22



There is a smooth trajectory, as expected. The car drives along a clean path with a few turns. After watching the playback of the video, this path follows very closely to the camera data provided. The term for noise is an identity matrix.

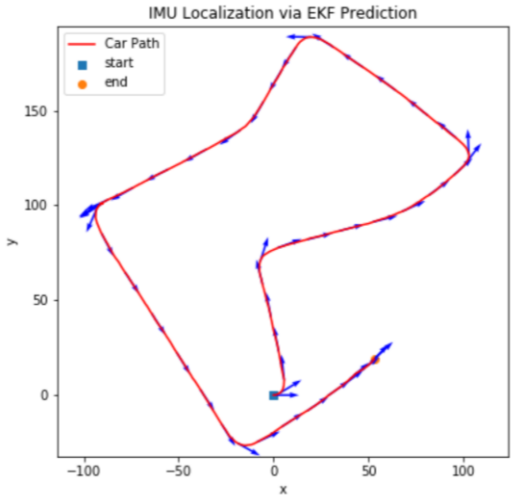


The landmarks are in expected locations here. All landmarks that were available in the given dataset are present since this is not downsampled. Note that there are many landmarks along the road, but many extend far past the trajectory at turns in which the street opens up. This indicates that the car is looking down a street, but turning a corner, which is a good sanity check to make sure that landmark mapping is behaving as expected. The term for noise is a diagonal with nonzero elements 1e6. When an element was seen for the first time, the covariance was set to 1e-4.

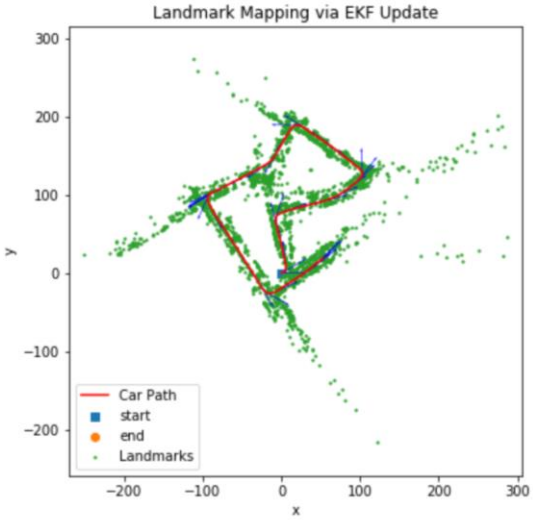


Unfortunately, my SLAM results were not very different from my naïve localization overlaid with my naïve mapping. Many landmarks are missing due to downsampling at a rate in which the top 100 most frequently observed features are included along with every 10th feature. This is likely due to either an issue with my implementation or possibly weight initialization. The noise and covariance initializations used here were the same as in parts a and b. All parameters and initializations in this dataset are also used for the remaining datasets as changing them did not affect results very much.

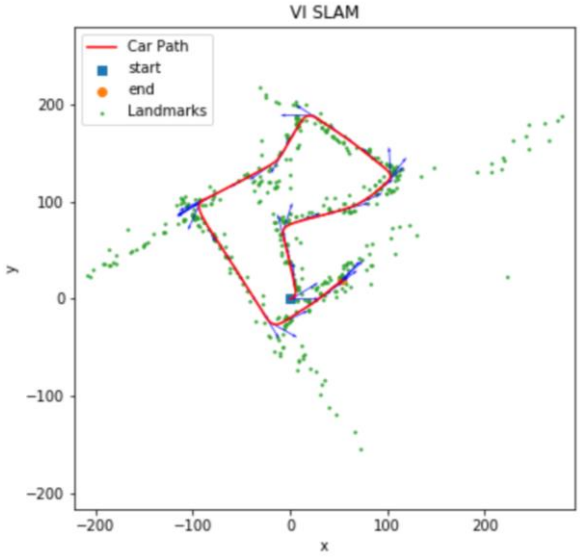
B. Dataset 27



Here, the trajectory looks as expected. The vehicle drives along a street and makes several turns, coming back to almost the same location that it started at.

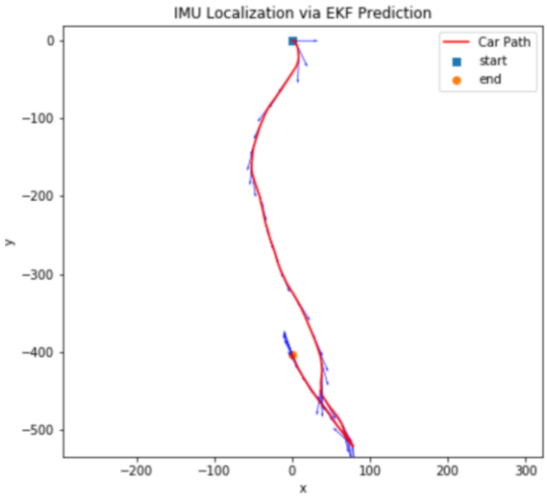


Similarly to dataset 22, there are many landmarks along the road because the car drives between buildings. All landmarks that were available in the given dataset are present since this is not downsampled. Note that there are many landmarks along the road, but others extend far past the trajectory at turns. As expected, there are several landmarks that extend past the trajectory, which happens along turns since the camera observes down a road lane as the car changes direction.

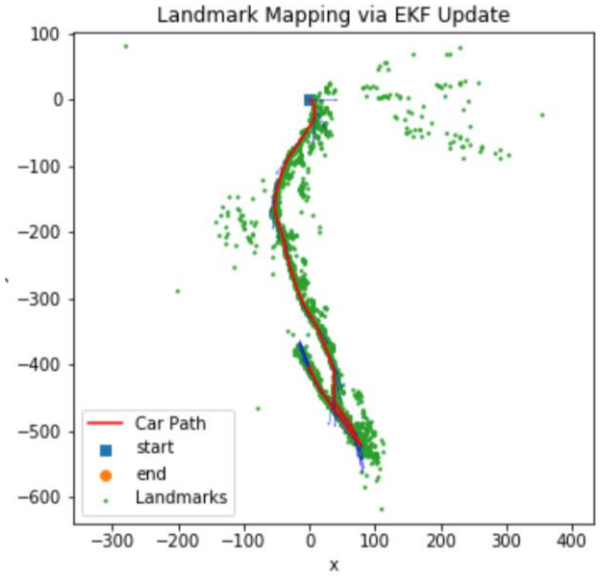


Again, my VI-SLAM did not look very different from the separate localization and mapping. In addition, the landmarks are downsampled for execution speed.

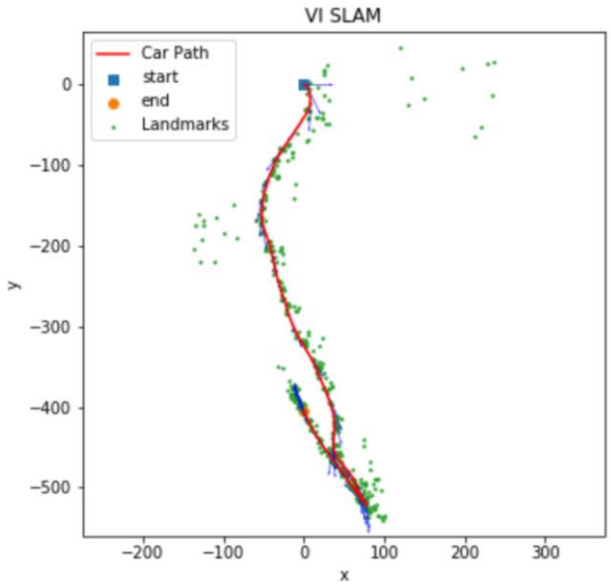
C. Dataset 34



In the provided camera video, the car starts at the top (with relation to the plot above) and moves downwards. Towards the end of the path, the car makes an abrupt turn, heads along a separate path, and stops. This is very consistent with the path that the IMU localization shows above.



There is a large spurt of landmarks at the start which is consistent with the video because the car faces a different direction and turns. These are seen in the top-right corner of the graph. The remainder of the path is very narrow, which is why the landmarks are extremely close to the trajectory path, again consistent with the video.



As before, this simply looks like my separate localization and mapping with downsampled landmarks.

D. Conclusion

The Extended Kalman Filter prediction steps alone can be used to obtain a good estimate of the trajectory of an object, while the update steps alone can be used to obtain a good estimate of the map. The results of using an Extended Kalman Filter compared to a particle filter are very interesting to observe. The EKF was much more resistant to changes in hyperparameter tuning, to start. In addition, the EKF uses a deterministic way to make predictions and updates for both localization and mapping. When the particle filter was used, there was a great deal of randomness involved in the process due to random initializations for particles. In addition, the EKF generalized better to different datasets than the particle filter. Both algorithms are extremely slow and have prohibitively long runtimes without either complex optimization techniques or the use of downsampling in which less data is used than what is given. In the future, I would like to improve my VI-SLAM algorithm so that I can obtain a more accurate trajectory and potential loop closure for dataset 27.

ACKNOWLEDGMENT

I used Professor Atanasov's slides extensively throughout the report as well as the TA's office hours. Thank you for the excellent resources provided throughout the duration of this project.