Project #3

Sensing & Estimation in Robotics

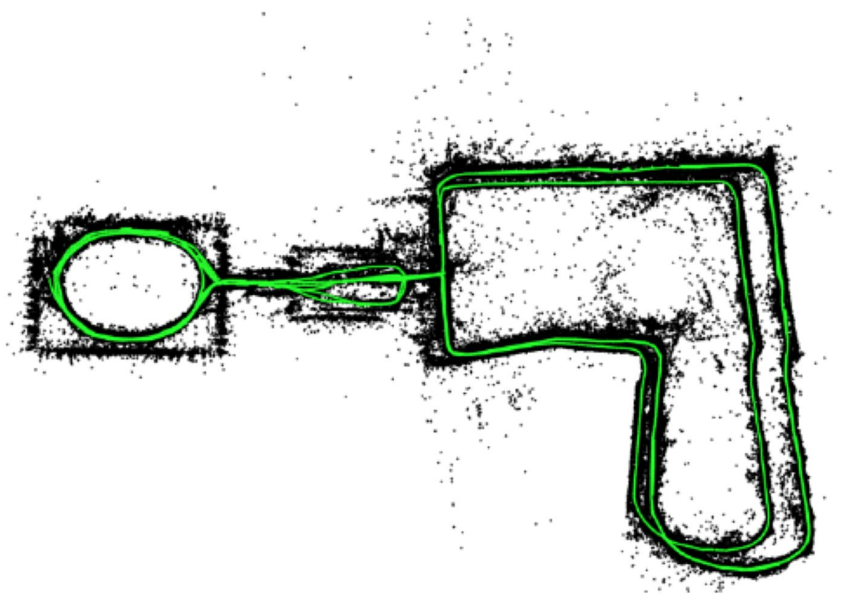
VISUAL INERTIAL SLAM

Arshia Zafari
(A11167578)
ECE 276A
3/13/19

INTRODUCTION

The way intelligent systems perceive and understand the world around them has been an ongoing advancement in robotic technology, some of the key developments being the adoption of filtering techniques for sensing and localization. Simultaneous Localization and Mapping (SLAM) is a process by which a robot constructs a map of its unknown environment while simultaneously keeping track of it's own location within the map. We have previously seen one implementation of SLAM using particle filtering to track the features of an environment. We will now implement a more complex filtering method in 3 dimensions called the Extended Kalman Filter. EKF is the nonlinear extension of the Kalman filter which keeps track of the estimated state of a system and the variance or uncertainty of the estimate. The estimate is predicted and updated using a linearized version of a non-linear state transition model and measurement observations.

This report will explore the implementation of SLAM application on two datasets and a test set using an Extended Kalman Filter, similar to the example shown in **Figure (1)**. Using synchronized measurements from an IMU and a stereo camera mounted on a car, as well as the intrinsic camera calibration and the extrinsic calibration between the two sensors, we will implement 3D localization of the IMU solely using prediction based on SE(3) kinematics. Then we will map landmarks using updates of visual observations of the world. And finally we will combine the two processes to update the IMU position as well as the map based on the stereo camera observation model to obtain a complete visual-inertial SLAM algorithm.



Figure(1). Visual-Inertial Localization and Mapping

PROBLEM FORMULATION

We will implement visual-inertial SLAM in three parts: localization of the IMU pose, visual mapping of landmark positions, and visual-inertial odometry.

Consider the IMU localization via EKF prediction first. Given the IMU measurements $\{u_t\}_{t=0}^{T}$ with control input as the linear and angular velocity of the IMU given by $u_t = [v_t\ \omega_t]^T$, we will estimate the inverse IMU pose $T_t = {}_W T_{I,t}^{-1}$ over time in SE(3) kinematics. Essentially, these inverse poses will be transformations from any point in the world to the IMU.

Next we will consider landmark mapping via EKF update. Assuming the inverse poses of the IMU that we calculated previously to be true, and given visual feature observations $\{z_t\}_{t=0}^{T}$ we will estimate the homogeneous coordinates $m \in \Re^{4\times M}$ in the world frame of the landmarks that generated the visual observations. The visual feature observations are in the form of pixel coordinates of images taken from a stereo camera, where the visual features have already precomputed correspondences between the left and the right camera frames. For this application, each feature of interest in our map is characterized as a normal Gaussian distribution $\sim N(\mu_{t|t},\ \Sigma_{t|t})$.

Finally, combining the two for full visual-inertial odometry, we will use the IMU measurements $\{u_t\}_{t=0}^{T}$ as well as the visual feature observations $\{z_t\}_{t=0}^{T}$ to estimate the inverse IMU pose in SE(3) kinematics and update our map over time.

TECHNICAL APPROACH

To begin, we must have a general understanding of the kinematics space we will be working with. SE(3) is a Special Euclidean group which forms the mathematical basis of rigid body transformations and velocities respectively. All possible transformations between a homogeneous coordinate in world frame to the body frame of a given rigid body is the combination of a translation and a rotation given in **Eq.(1)**:

$$[x\ \ y\ \ z\ \ 1]^T \ \Rightarrow\ T = \begin{bmatrix} R & p \\ 0 & 1 \end{bmatrix} \tag{1}$$

where R is the 3-dimensional extrinsic rotation of pitch, yaw, and roll, and p is the displacement in x, y, and z.

IMU Localization via EKF Prediction

As stated previously, the Extended Kalman Filter is the nonlinear version of the Kalman filter which linearizes about an estimate of the current mean and covariance of a state. In our application we will be using EKF of our inertial and visual measurements to predict and update the position of our vehicle in the world for each time step.

For EKF prediction, start with an initial Gaussian prior with mean and covariance at time t given by **Eq.(2)**:

$$T_{t|z_{0:t},\, u_{0:t-1}} \sim N(\mu_{t|t},\ \Sigma_{t|t}) \tag{2}$$

where the mean $\mu_{t|t} \in SE(3)$ and covariance $\Sigma_{t|t} \in \Re^{6\times 6}$. With time discretization τ, control input u_t and control input noise $w_t \sim N(0, W)$, we can use the following motion model in **Eq.(3)**:

$$T_{t+1} = exp((\tau(-u_t + w_t))^{\wedge})T_t \tag{3}$$
$$u_t = [v_t\ \omega_t]^T \ \in \Re^6$$

Note that u_t is negative since T_t is the inverse pose of the IMU. The component w_t specifically represents the noise in both linear and angular velocity. However, since we are dealing with noise we must consider what happens with the pose kinematics with perturbation. Therefore we can rewrite the motion model in terms of nominal kinematics of the mean of T_t and zero-mean perturbation kinematics as in **Eq.(4)**:

$$\mu_{t+1|t} = exp(-\tau \hat{u}_t)\mu_{t|t} \tag{4}$$

$$\Sigma_{t+1|t} = exp(-\tau \widehat{u}_t)\Sigma_{t|t} exp(-\tau \widehat{u}_t)^T + \tau^2 W \tag{5}$$

where $\hat{u}_t$ belongs to the space of twist matrices in se(3), and $\widehat{u}_t$ represents the hat-map in perturbed kinematics shown below:

$$\hat{u}_t = \begin{bmatrix} \hat{\omega}_t & v_t \\ 0 & 0 \end{bmatrix} \in \mathfrak{R}^{4\times 4} \qquad \hat{\omega}_t = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix} \in \mathfrak{R}^{3\times 3} \qquad \widehat{u}_t = \begin{bmatrix} \hat{\omega}_t & \hat{v}_t \\ 0 & \hat{\omega}_t \end{bmatrix} \in \mathfrak{R}^{6\times 6}$$

The 3-dimensional hat-map $\hat{\omega}_t$ belongs to the space of skew-symmetric matrices in so(3). With this set-up, the prediction step becomes quite simple. We initialize the mean of our IMU pose in SE(3) using the identity matrix I_3 as our rotation matrix and $[0\ \ 0\ \ 0]^T$ as our initial displacement. Using functions to implement the various hat-map transformations, we can predict the inverse pose of the IMU by computing the mean from **Eq.(4)** for each time step in a compact way, this process is all done in the "predict_mean()" function. Since only the mean gives a description of the pose of the IMU, computing the covariance $\Sigma_{t+1|t}$ is not necessarily needed for this step. Using the inverse poses for all time steps, we can show the actual trajectory in the world by taking the inverse of our inverse pose in SE(3) given by **Eq.(6)**:

$$T^{-1} = \begin{bmatrix} R^T & -R^T p \\ 0 & 1 \end{bmatrix} \tag{6}$$

The EKF prediction step code can be found in "hw3_main.py." See *Results* for the trajectory of the car in the world using the EKF prediction for both datasets and the test set.

Landmark Mapping using EKF Update

The EKF update step becomes a little more involved. We now assume that the predicted IMU trajectory from the prediction step above is correct and focus on estimating the landmark positions. We will take unknown landmark positions $m \in \mathfrak{R}^{4\times M}$ (in homogeneous coordinates) as a state and perform EKF update step after every visual observation z_t from the stereo camera in order to keep track of the mean and covariance of m. These landmarks were previously chosen and tracked using an optical flow algorithm. Note that we are assuming that the landmarks are static so it is not necessary to implement a prediction step. Moreover, since the sensor does not move sufficiently along the z-axis, the estimation for the coordinate of the landmarks will not be very good, so we only need to focus on estimating their xy coordinates. **Figure(2)** and **Figure(3)** show an example from each dataset, the observed features through each camera, the left camera reading the blue point and the right camera reading the red point and the correspondence between them marked by the green line.

Figure(2). Visual observation from dataset 27

Figure(3). Visual observation from dataset 42

Each landmark will have a prior Gaussian distribution similar to before, but now with $\mu_t \in \Re^{4\times M}$ and covariance $\Sigma_t \in \Re^{3M\times 3M}$. We will take the mean and covariance of each landmark and compute the predicted observation based on the known correspondences in **Eq.(7)**:

$$\hat{z}_{t,i} := M\pi({}_O T_I T_t \mu_{t,j}) \quad \in \Re^4 \tag{7}$$

for each visible landmark $i = 1, \ldots, N_t$ in a given time frame. M represents the stereo-camera calibration matrix which contains information of the intrinsic parameters of the left and right cameras. The transformation to the optical frame ${}_O T_I$ is given by the variable "cam_T_imu" in our code, and the transformation into the IMU frame ${}_I T_t$ is taken from our prediction step. We then compare the landmark observation with our true observations $z_t \in \Re^{4\times N_t}$ and update the means and covariances by **Eq.(8)** and **Eq.(9)**:

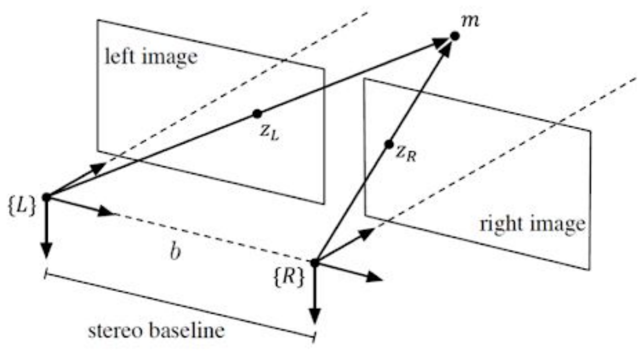
$$\mu_{t+1} = \mu_t + DK_t(z_t - \hat{z_t}) \tag{8}$$
$$\Sigma_{t+1} = (I - K_t H_t)\Sigma_t \tag{9}$$

where D is the dilation matrix, K_t is the Kalman gain at time t, and H_t is the Jacobian of the observation $\hat{z}_{t,i}$ with respect to landmark m_j evaluated at its mean $\mu_{t,j}$ at time t given by the following equations **Eq.(10)** and **Eq.(11)**:

$$K_t = \Sigma_t H_t^T (H_t \Sigma_t H_t^T + I \otimes V)^{-1} \tag{10}$$
$$H_{i,j,t} = M \tfrac{d\pi}{dq}({}_O T_I T_t \mu_{t,j})_O T_I T_t D \tag{11}$$

for a given measurement noise $v_t \sim N(0, V)$ in units of pixels. **Eq.(11)** holds only for observations i that correspond to landmark j at time t, it is $0 \in \Re^{4\times3}$ otherwise.

In summary, what is essentially happening is for each time frame we look at only the visible landmarks for that time step (the visible features will be those that are not represented as $z_t = [-1\ -1\ -1\ -1]^T$ in the stereo camera frame for a given time). So we take the current valid landmark coordinates represented by the mean μ_t in the world, bring them into the stereo-camera frame and use that as our observation $\hat{z}_t$. Then we compute the Kalman gain and Jacobian of the observation at that time step and use those to update our mean and covariance for the next time step μ_{t+1} and Σ_{t+1} respectively.



Figure(4). Stereo Camera model

Now one important thing to consider is, what is the best way to initialize these landmarks in the world? We cannot simply initialize them all at the origin since we don't have enough data for each landmark to allow the positions to converge. Therefore we must use another method. For landmark initialization, we will actually use the true observations z_t in the stereo-camera frame and back-project them into the world to get a more reasonable coordinate of the landmark m in the world similar to the visualization of the stereo camera model in **Figure(4)**. The observations are not always going to be accurate so the positions will still be updating, just not on the first exposure.

Since initialization only matters when we see a landmark, we will be doing this back projection only once per landmark, ie. only at the first time we see it, and continue the update process normally described before. Each observation represents the pixel coordinates in the left and right frames given by $z_t = [u_L\ v_L\ u_R\ v_R]^T$. The coordinates are not just integers in order to maintain subpixel resolution. To get the observation into the world frame, we only need to look at one camera, we will be looking at the left camera. Using **Eq.(12)** with disparity $d = 1$, and camera intrinsics $fs_u b =- 379.8145$, we can compute the third coordinate z for the π operator.

$$d = u_L - u_R = \tfrac{1}{z} f s_u b \tag{12}$$

Then we can use the intrinsic calibration matrix K_l for the left camera and invert the transformations to get the observation in the world frame as in **Eq.(13)**

$$\mu_t = ({}_O T_I T_t)^{-1} (K_l \pi)^{-1} z_t \tag{13}$$

With proper initializations, we will be able to converge to a true landmark position in the world more quickly and efficiently. As far as the other hyperparameters, namely the initial covariance Σ_0 and noise variance of pixel noise V, those can be experimented with since they are design parameters. We will see their importance in the *Results* section. The algorithm itself treats the update equations as large block matrices, but for simplicity of implementation in the code, only the valid landmarks are indexed and individually updated (the code runs quite fast, between 1-2 seconds even with the individual indexing using for-loops). See *Results* for the converged landmark positions in the world around the car's trajectory using the EKF update for both datasets and the test set.

Visual-Inertial SLAM

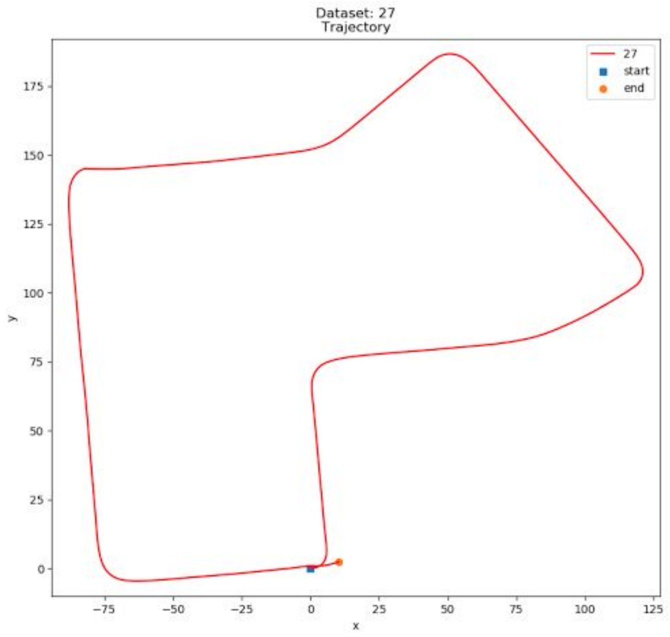
This complete visual-inertial SLAM algorithm combines the processes of EKF prediction and update described previously. Based on the stereo camera calibration matrix M, extrinsics $oT_I \in SE(3)$, coordinate landmark positions $m \in \Re^{4\times M}$ and new observation $z_{t+1} \in \Re^{4\times N_t}$, we will compute the predicted observation based on the predicted pose $\mu_{t+1|t}$ and known correspondences in **Eq.(14)**:

$$\hat{z}_{t+1,i} := M\pi({}_OT_I\mu_{t+1,j}m_j) \quad \in \Re^{4\times N_t} \qquad \textbf{(14)}$$

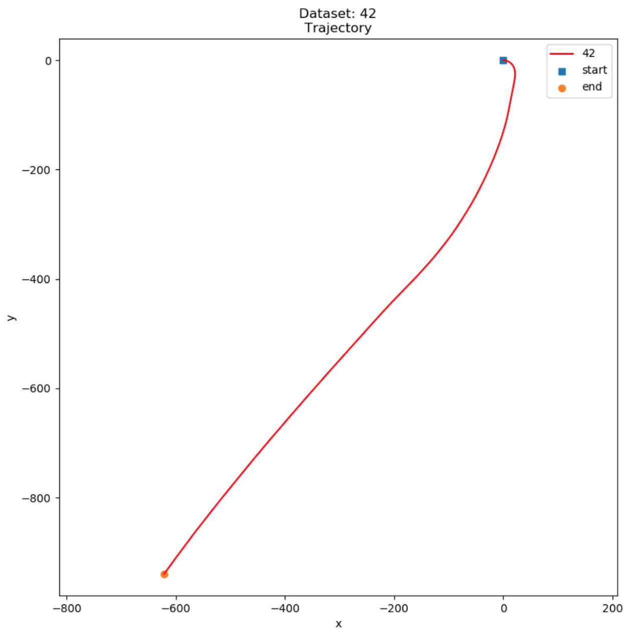
for visible features $i = 1, \ldots, N_t$ in time t. We will in essence predict and update the pose of the IMU with respect to the world as well as update the map landmarks within the world using a slightly different application of the same equations discussed previously, thus completing the visual-inertial SLAM algorithm

RESULTS

The figures below show the trajectory of the car in the datasets using the EKF prediction step. Since the prediction step computes the inverse poses of the IMU, ie. the transformations from the world to the IMU, the plots below show the inverse of the inverse pose, ie. the IMU to the world.



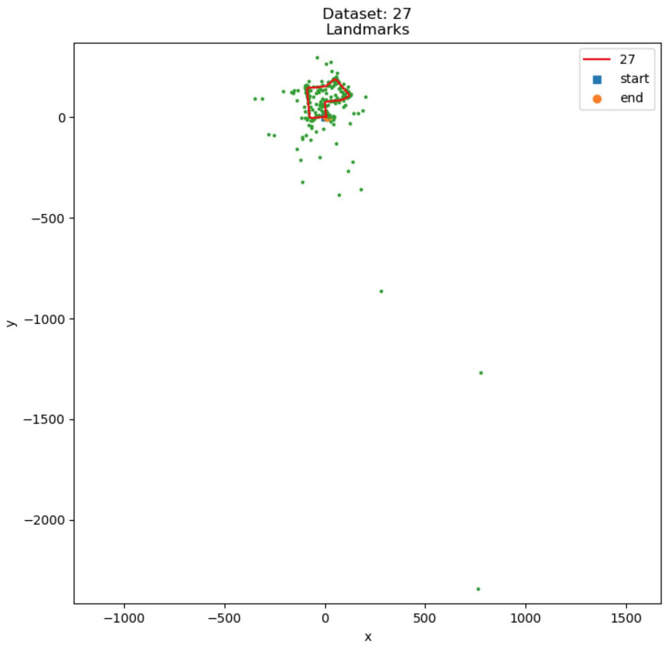
Figure(5a). Trajectory for dataset 27



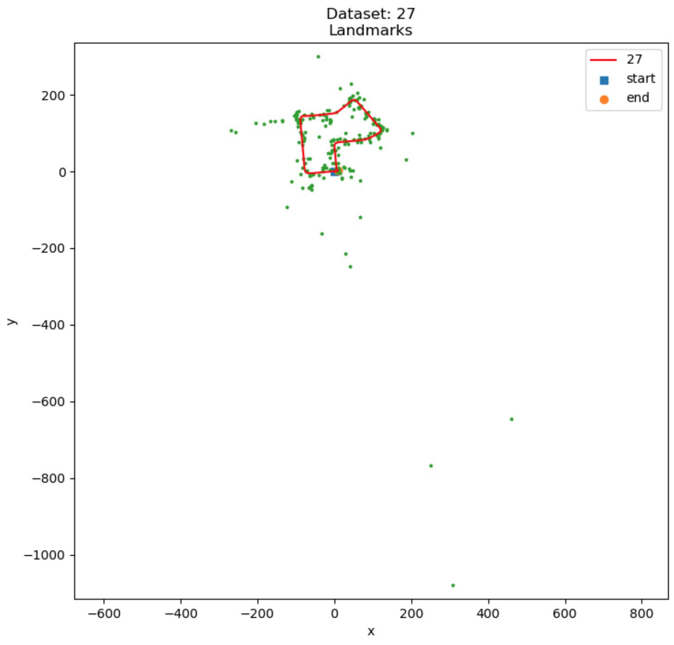
Figure(5b). Trajectory for dataset 42

We can see that the EKF prediction step was successful. Looking at dataset 27, we see a car traverse the streets of a neighborhood, making a right turn and four left turns. Dataset 42 shows the car driving south-west along a highway.

Now lets see the landmarks mapped using EKF update. First we will initialize each feature using an initial identity covariance $\Sigma_0 = I_3$ and pixel variance of $V = 4 \times I_4$, meaning the observations of pixel coordinates could be around ± 2 pixels off. Looking at **Figure(6a)** for dataset 27 with the landmarks marked in green, we can see there are quite a few outliers, especially feature 183 around the bottom right of the map which seems to have an xy coordinate of around (750,-2400).



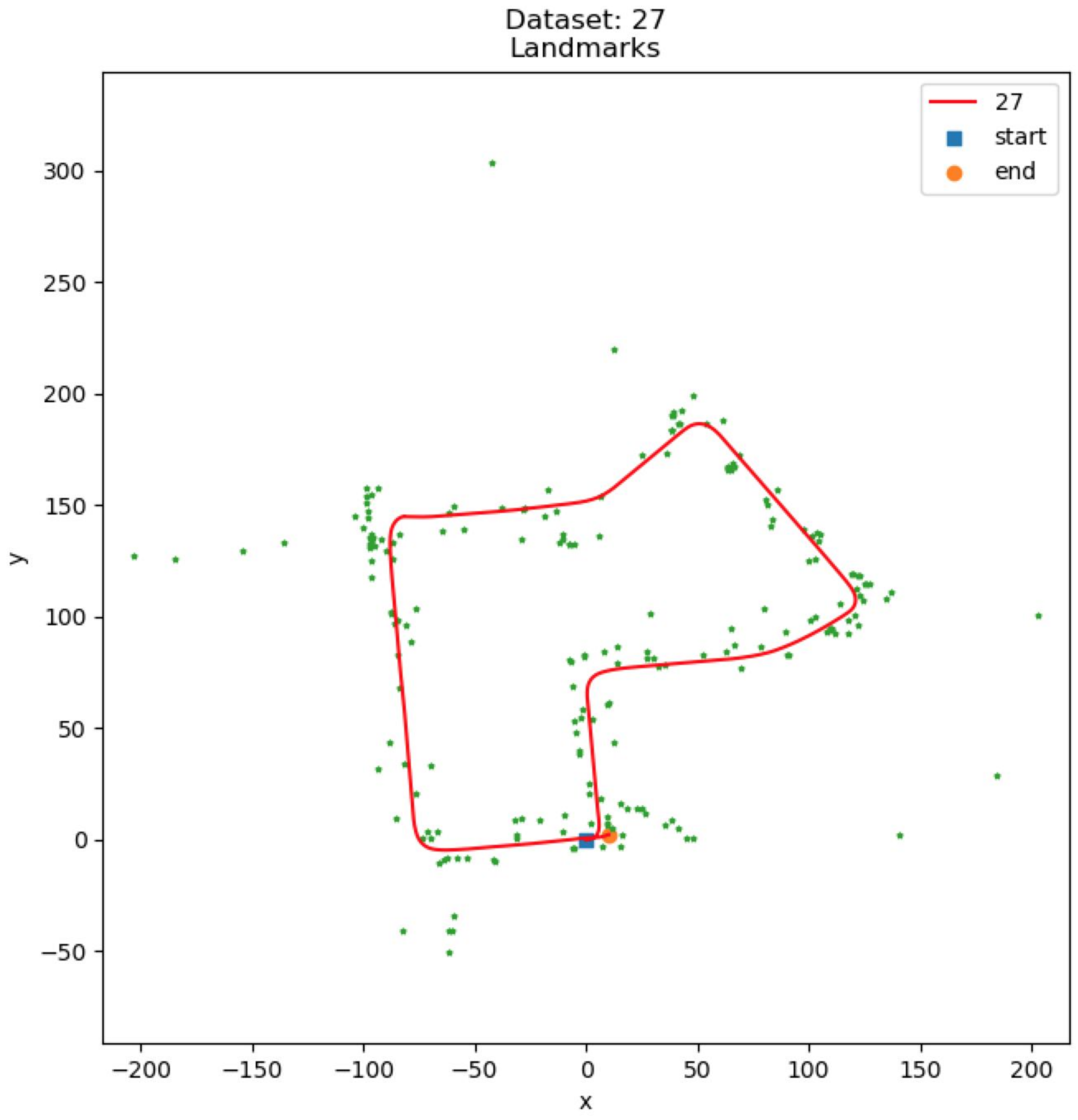
Figure(6a). Landmarks with noise variance = 4



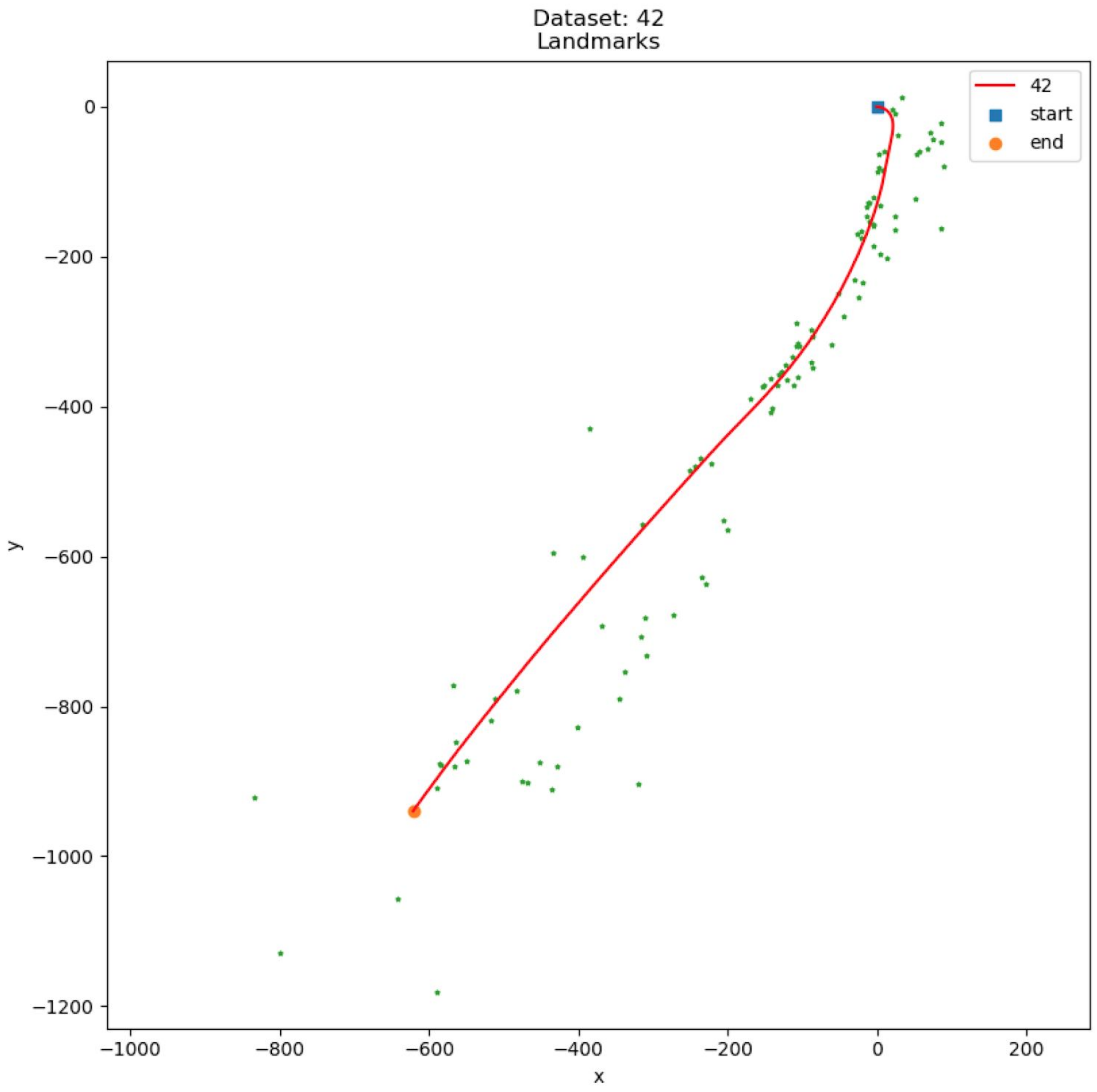
Figure(6b). Landmarks with noise variance = 100

To combat this, we should allow a higher variance, say ± 10 pixels. Sometimes for specific time instances, the measurements for some features can be skewed, and this has a lot to do with the state at that particular time. There could have been things like sun exposure, vibration of the vehicle, or a power issue which can cause a landmark reading to be unreasonably far, so increasing the variance will allow us to mitigate the effects of the outlying measurements. **Figure(6b)** shows the the EKF update on the same dataset but with $V = 100 \times I_4$. The exact same landmark, feature 183 is brought closer now at (300,-1200).

We can also decrease the covariance to further combat those outliers. Setting a small initial covariance increases the confidence in our model, that the features are static and that our estimates don't change much between measurements. **Figure(7)** and **Figure(8)** show the final landmarks using $\Sigma_t = 0.001 \times I_3$ and $V = 100 \times I_4$.

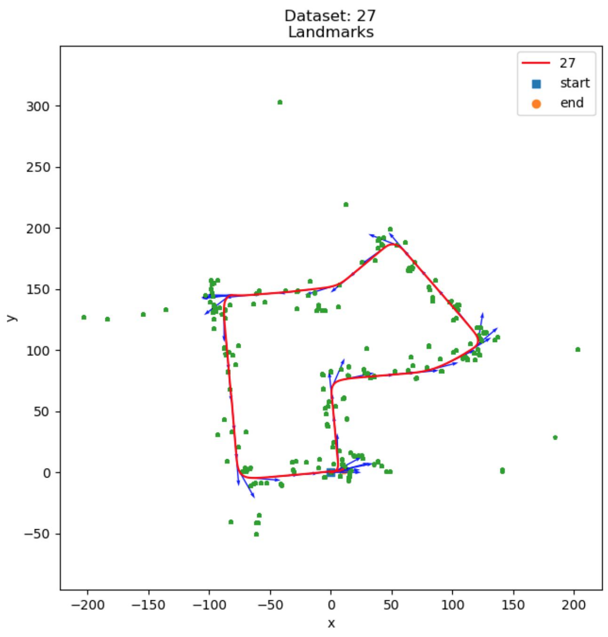


Figure(7). Final mapped landmarks, dataset 27

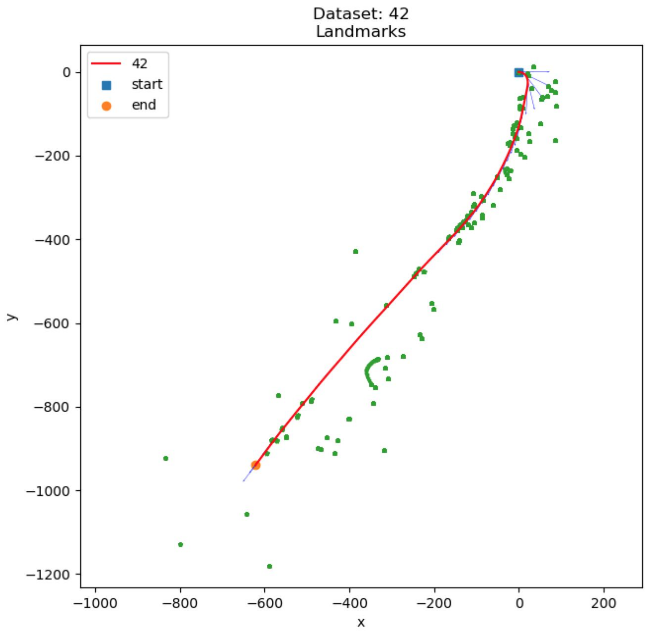


Figure(8). Final mapped landmarks, dataset 42

We can see that the EKF update step was successful, the landmarks seem to follow both trajectories quite well and there are no unreasonable outliers. Lets see how exactly the map locations converged by plotting all the locations for all timesteps, shown in **Figure(9a)** and **Figure(9b)**:

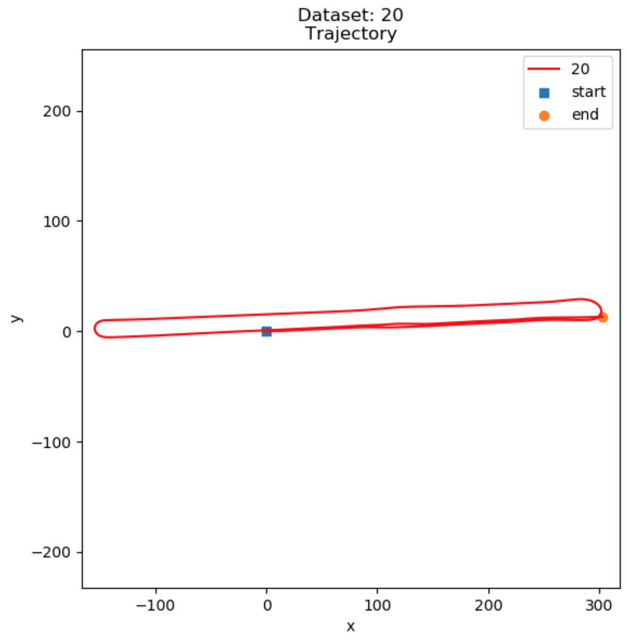


Figure(9a). Convergence of landmarks, dataset 27

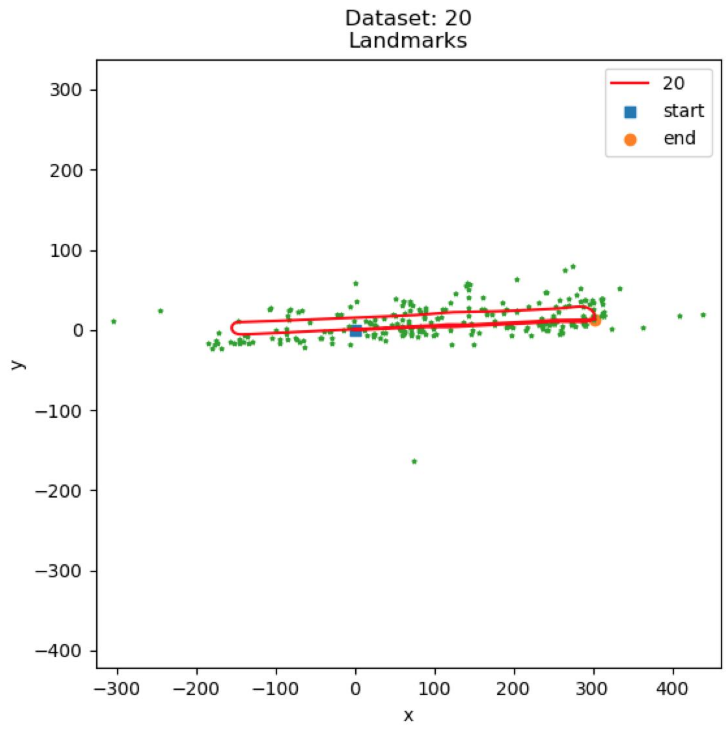


Figure(9b). Convergence of landmarks, dataset 42

The plots above show the converged landmark positions (at the final time epoch for each dataset). It looks like most of them were pretty spot on using the true observation as the initial pose. We can visualize the update process for a few landmarks, for example in **Figure(9b)** for dataset 42, one of the landmarks near the straight-away part of the trajectory seems to be updating its position and reach the converged location in **Figure(8)**. The rest seem pretty spot on, showing how the observations from the stereo camera were quite accurate and the hyperparameters chosen fits the design of this data well. Let's look at the results on the test set:

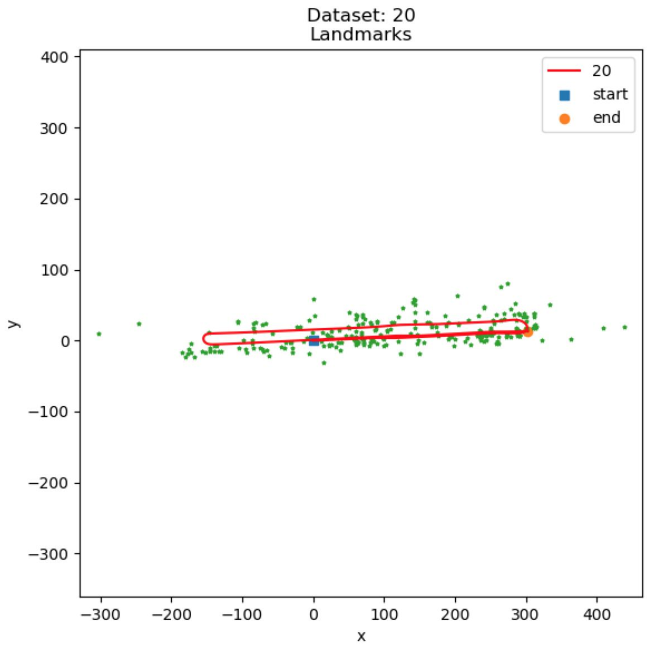


Figure(10). Trajectory for test set



Figure(11). Final mapped landmarks for test set

The results perform quite well, in the video the vehicle drives up and down a narrow street. There is just one landmark that seems to be a bit off, feature 167 at around (80,-165). Like before this can be fixed by increasing the variance of the observation noise since that specific measurement might have something wrong with it. The mapping is fixed with a higher noise of $V = 1000 \times I_4$ below:



Figure(12). Landmarks with noise variance = 1000

This is equivalent to a noise of roughly ± 32 pixels off, however considering the images from the camera are 384×1232 pixels, a measurement that skewed would be from a faulty result of the optical flow algorithm that chose the feature or the intrinsics of the camera sensor are quite poor. Since the other measurements seem decent, the problem most likely comes from our data. We just have to accept that the model may not be perfect.

Unfortunately there wasn't enough time to implement the visual-inertial SLAM, as it was also extra credit.

CONCLUSION:

This project has shown that using an Extended Kalman Filter and measurements from both visual and odometry sensors, we are able to map the features of an unknown environment and localize our vehicle within it. We first looked at the EKF prediction step by estimating the inverse pose of the IMU for a given time step. Our results showed the trajectory of the IMU with respect to the world and it follows both datasets and the test set quite well. We also saw that using EKF update we can map a converged location of a landmark position in a map. Our results showed that with a specific covariance and noise variance, we can map features of an unknown environment pretty quickly and efficiently. There will always be some outliers caused by imperfections of our state, but we've seen that it can be mitigated by increasing the variance of our noise up to a reasonable value. Unfortunately there wasn't enough time to do the actual visual-inertial SLAM but the tools are all here to combine the prediction and update processes to track the vehicle in an updated map of the world.

In reality, we want to design intelligent systems that can withstand the real world, but the real world is never just idle all the time, it is always changing and features within the map will always be changing. Therefore it's important to note the various types of SLAM besides EKF, such as Fast SLAM and Sub-Mapping SLAM for other applications. Using the fundamentals of EKF and these other algorithms, the SLAM application can be expanded to multi-agent SLAM or SLAM using only visual data from a single camera. In all, the results from EKF shown here demonstrate just how useful the algorithm is in tracking and mapping in 3-dimensions and just how much potential there is for SLAM with further advancements in robotic sensing technology.