Visual Inertial SLAM

Jared Leitner (jjleitne@eng.ucsd.edu)

ECE276A: Sensing & Estimation in Robotics

I. Introduction

Simultaneous localization and mapping (SLAM) is a process used for determining a robot's position in the world while simultaneously constructing a map of the surrounding environment. This is accomplished using a series of on-board sensors that provide the robot information regarding its state in the environment. The number and type of sensors used will vary depending on the application, however, it is important that SLAM can be implemented when limited to a certain set of sensors.

This project is titled visual-inertial SLAM because the data comes from an inertial measurement unit (IMU) and a stereo camera, which comprises of a left and right camera. The IMU provides linear and rotational velocity measurements, while the stereo camera provides images and depth information. Specific landmarks in the images are precomputed using a visual feature extraction technique, and the left and right camera pixels of these landmarks are provided and used for implementing SLAM. These optical features are time aligned with the IMU data.

An extended Kalman filter (EKF) is used to implement the SLAM process, and consists of a non-linear motion model and a Gaussian distribution to keep track of the robot's state over time. This project consists of two parts: The first step involves implementing the EKF prediction step using SE(3) kinematics to estimate the robot's pose over time. SE(3) refers to a group of matrices that represent rigid body poses or transformations. In the second step, an EKF update step is used for determining the positions of the landmarks in the world coordinate frame. The stereo camera baseline and calibration matrix are provided along with the transformation from the IMU to left camera frame.

The rest of the paper is structured as follows: Section II describes the visual inertial SLAM problem in a mathematical formulation. Section III details the approach taken to localize the robot and map the landmarks. Finally, Section IV discusses the results.

II. Problem Formulation

The EKF prediction step uses the motion model in order to estimate the mean and covariance of the robot's state, given the current control input. This step is defined as follows, where f is the non-linear motion model of the robot and u_t is the current control input[1]. The noise is set to 0 for determining the predicted mean $\mu_{t+1|t}$. F_t is the derivative of the motion model with respect to the state, and Q_t is the derivative of the motion model with respect to the noise. W is the covariance of the noise for the motion model.

[1] Some of the equations are taken from Professor Atanasov's lecture slides.

$$\boldsymbol{\mu_{t+1|t}} = f(\boldsymbol{\mu_{t|t}}, u_t, 0) \qquad \Sigma_{t+1|t} = F_t \Sigma_{t|t} F_t^T + Q_t W Q_t^T$$

These formulas are used for the first part of this project in order to keep track of the robot over time. In this case, $\mu_{t|t} \in SE(3)$ is the inverse IMU pose with respect to the world frame. Instead of the IMU pose, the inverse IMU pose is tracked because this transformation is required later when implementing the landmark update step. The following equation represents the prediction step for the mean using SE(3) kinematics, where u_t is a 6x1 vector containing the 3-dimensional linear and rotational velocities.

$$\boldsymbol{\mu}_{t+1|t} = \exp\left(-\tau \hat{u}_t\right) \boldsymbol{\mu}_{t|t} \qquad u_t := \begin{bmatrix} v_t \\ \omega_t \end{bmatrix}$$

The exponent maps the control input vector into the SE(3) space which can be matrix multiplied with the current pose in order to get the new predicted pose. τ represents the time step between IMU readings and it is negative because the inverse IMU pose is being tracked. In order to calculate this exponent, Rodriguez's formula is implemented as follows:

$$\exp(\hat{\boldsymbol{\theta}}) = I + \left(\frac{\sin \|\boldsymbol{\theta}\|}{\|\boldsymbol{\theta}\|}\right) \hat{\boldsymbol{\theta}} + \left(\frac{1 - \cos \|\boldsymbol{\theta}\|}{\|\boldsymbol{\theta}\|^2}\right) \hat{\boldsymbol{\theta}}^2$$

$$\hat{\boldsymbol{\theta}} = -\tau \hat{u}_t$$

In order to determine the trajectory of the robot only the prediction step is carried out, meaning the predicted mean will be used as the robot's new position. Since the predicted mean does not depend on the covariance, this does not have to be accounted for. The mean in this case is the inverse IMU pose, so the inverse of the mean is instead plotted to obtain the pose of the IMU in the world frame.

The second part of this project involves implementing an EKF update step in order to determine the positions of the landmarks in the world frame. In this case, the mean $\mu_{t|t} \in R^{4\,x\,M}$ corresponds to the positions of M landmarks using homogenous coordinates and the covariance $\Sigma_{t|t} \in R^{3Mx3M}$. The update equations for both the mean and covariance are as follows:

$$\boldsymbol{\mu}_{t+1} = \boldsymbol{\mu}_t + DK_t\left(\mathbf{z}_t - \hat{\mathbf{z}}_t\right) \qquad \Sigma_{t+1} = (I - K_t H_t)\Sigma_t$$

The Kalman gain K_t is defined:

$$K_t = \Sigma_t H_t^T \left(H_t \Sigma_t H_t^T + I \otimes V\right)^{-1}$$

$$H_{i,j,t} = \begin{cases} M \frac{d\pi}{d\mathbf{q}} \left({}_O T_I \, T_t \boldsymbol{\mu}_{t,j}\right) {}_O T_I \, T_t D \\ \\ \mathbf{0} \in \mathbb{R}^{4\times 3} \end{cases}$$

In the update mean equation, $\hat{z}_t$ corresponds to the predicted landmark coordinates and is determined using the observation model for the stereo camera defined as:

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

The components of this observation model are:

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

$$\pi := canonical\ projection\ function$$

$$T_o^I := transformation\ from\ camera\ to\ image\ frame$$

$$T_t := current\ inverse\ IMU\ pose$$

This model determines the left and right pixel values of a landmark given its position in the world frame. All the components of *M* are provided by the stereo baseline and intrinsic calibration matrix.

For the update step, both the mean and covariance need to be accounted for because the mean update depends on the covariance. The next section describes how these steps were implemented using Python.

III. Technical Approach

In order to obtain a trajectory for the robot, the inverse IMU pose is initialized as the identity matrix. This corresponds to starting at the origin with zero offset. Starting with this pose, the first linear and rotational velocity reading is fed into the EKF prediction function *predict_EKF*, which returns the new predicted pose. This function constructs $\hat{u}$ and uses Rodriguez's formula described in the previous section to calculate the exponential. The exponential is then right multiplied by the current pose in order to obtain the next pose. The covariance is not kept track of, as the trajectory does not depend on it.

This is carried out incrementally for all IMU readings, and each predicted pose is stored in a 3-dimensional matrix and the inverse of the pose is stored in a separate matrix. The inverse of the pose corresponds to the IMU pose in the world frame, and these intermediate inverse poses are required in order to plot the trajectory. This entire process can be seen in the *hw3_main.py* file under part (a). The trajectories for each dataset are displayed in the following section.

The second part of this project involves finding the location of each landmark in the world frame. This data is presented in a 4x*M*x*T* matrix, where *M* is the number of landmarks and *T* is the number of time steps. If a landmark was not viewed for a given time step, its pixel values are set to -1, taking care of the data association process.

The function *construct_M* is used to obtain the stereo camera matrix. In order to determine the initial positions of each landmark, their pixel values must be transformed into the world frame.

This is carried out in a series of transformations, the first being from the image to camera frame and the second from camera to world frame. The image to camera frame transformation is calculated by solving this system of linear equations:

$$\begin{bmatrix} u_L \\ v_L \\ u_R \\ v_R \end{bmatrix} = \underbrace{\begin{bmatrix} fs_u & 0 & c_u & 0 \\ 0 & fs_v & c_v & 0 \\ fs_u & 0 & c_u & -fs_u b \\ 0 & fs_v & c_v & 0 \end{bmatrix}}_{M} \frac{1}{z} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

This results in:

$$z = \frac{f_{s_u} b}{u_L - u_R} \quad , \quad x = \frac{(u_L - c_u)z}{f_{s_u}} \quad , \quad y = \frac{(v_L - c_v)z}{f_{s_v}}$$

Using these equations, the landmark coordinates in the camera frame can be determined from the landmark pixel values. The camera to world frame transformation consists of the camera to IMU transformation (this is the inverse of the provided IMU to camera transformation) and the IMU to world transformation. The IMU to world transformation changes for each time step, however, these poses were saved during the EKF prediction step mentioned earlier. Using these transformations, the initial landmark positions are found and the results are shown in the next section.

In order to update the landmark positions, the EKF update step is implemented. Each landmark is updated independently, instead of all at once as described in the theoretical section of this report. This is allowable because it is assumed that each landmark is independent, and therefore there is no information loss by updating them independently.

Each landmark is iterated over, and only considered valid when its values do not equal -1. The mean and covariance are updated according to the previous section, and the final position for each landmark is stored. In order to carry out this update step, the Jacobian of the observation model must be found. The derivative of the projection function with respect to the camera frame coordinates is as follows:

$$\frac{d\pi}{d\boldsymbol{q}} = \begin{pmatrix} \frac{1}{q_z} & 0 & \frac{-q_x}{q_z^2} & 0 \\ 0 & \frac{1}{q_z} & \frac{-q_y}{q_z^2} & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{-1}{q_z^2} & \frac{1}{q_z} \end{pmatrix}$$

This is carried out in the function *jacobian_of_pi*. The final landmark positions are shown in the following section.

IV. Results and Discussion

The following plots display the trajectories for the two training datasets. These trajectories appear to closely match the route taken by each car based on the videos provided. A high quality IMU is used for data collection, resulting in accurate poses using only the EKF prediction step.

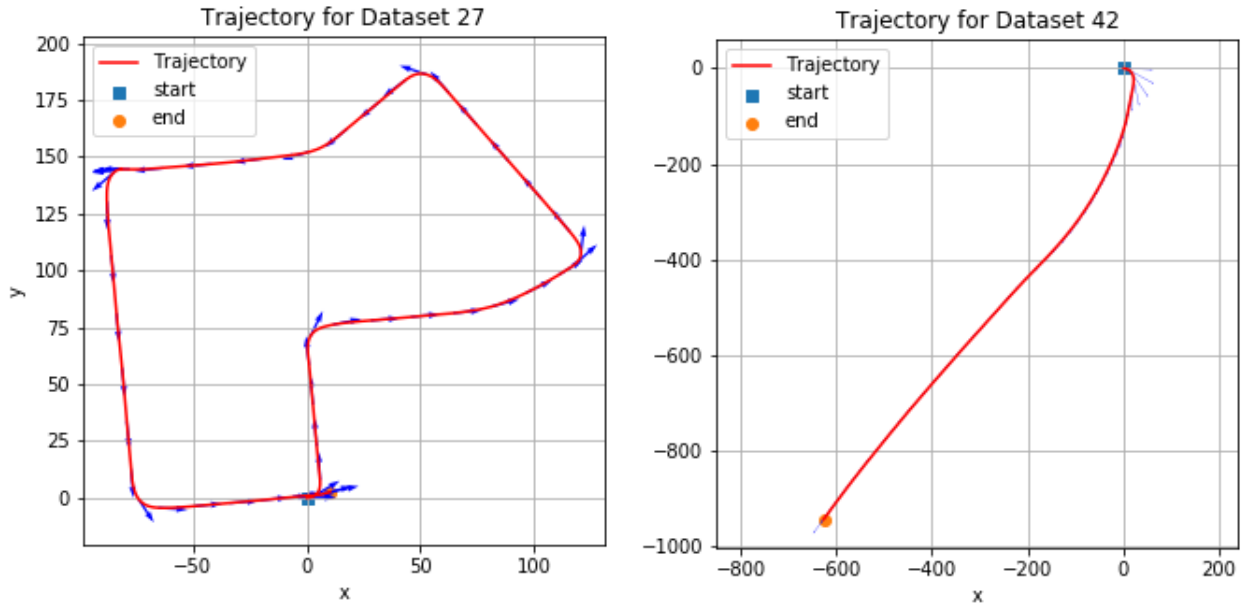

Figure 1: Trajectories for Datasets 27 and 42

Figure 2 displays the initial landmark positions compared to the final landmark positions for dataset 27.

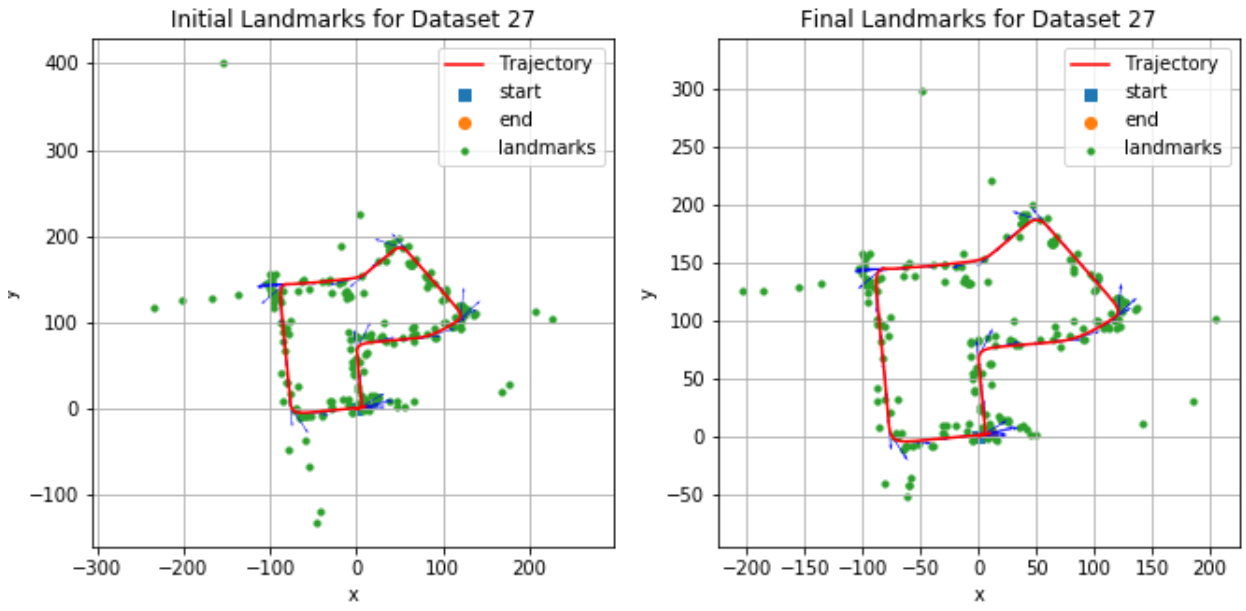

Figure 2: Initial vs. final location of landmarks for Dataset 27

Evidently, the final positions for some of the landmarks are different than their initial positions. It seems that the landmarks that are initially further away from the trajectory end up moving closer to the trajectory. This is a positive result, as the final landmark positions are more accurate than the initial positions, indicating the EKF update step has worked. The landmarks that are further away from the trajectory path are located further down the corresponding street, which can be seen from the video.

Figure 3 displays the same relation between the initial and final landmark positions for dataset 42. Again, the final landmark positions appear to be more accurate than the initial positions.

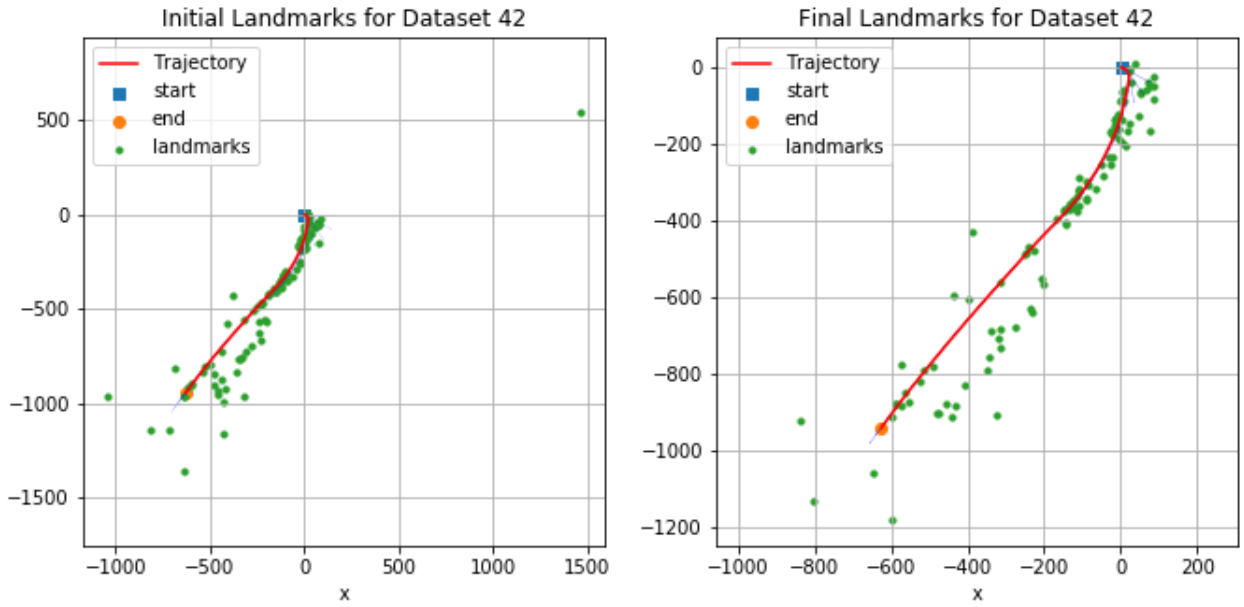

Figure 3: Initial vs. final location of landmarks for Dataset 42

The next figure shows the trajectory for the test data set. Based on the video, the car drives down a straight street, makes a U-turn, and then another U-turn. This closely matched the predicted trajectory.

Figure 4: Trajectory for test dataset

Figure 5 displays the initial vs. final landmark positions for the test data set. Once again, the final landmark positions appear to be more accurate as they are closer to the trajectory path.

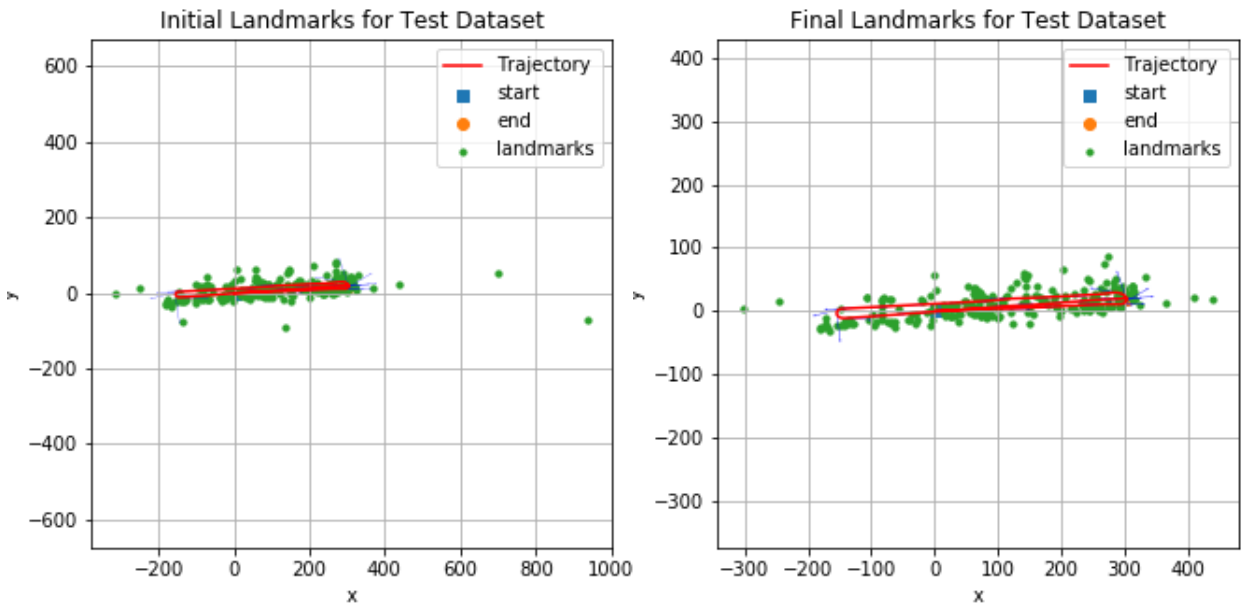

Figure 5: Initial vs. final location of landmarks for test dataset

Overall, it appears that the EKF prediction step for the inverse IMU pose and the EKF update step for the landmark positions are performing well. At this stage, the localization and mapping steps are occurring independently. In order to carry out the full SLAM problem, these steps would need to occur simultaneously. That would require the combination of the IMU pose and landmark positions into one state, and a modified prediction and update step to accommodate for the larger state.