Visual-Inertial SLAM with Extended Kalman Filter

Abhiram Iyer
University of California, San Diego
abiyer@ucsd.edu

***Abstract*—This paper presents an approach to using the Extended Kalman Filter (EKF) and synchronized data from an IMU (inertial measurement unit) and stereo camera to perform Visual-Inertial SLAM (Simultaneous Localization and Mapping). The IMU and stereo camera data is collected from a car that travels through various outdoor streets.**

I. Introduction

The SLAM formulation, which is employed frequently in navigation tasks, is used to simultaneously map an environment and localize an agent's position within this map.

In this paper, we present an approach to this formulation given synchronized IMU measurements and stereo camera data that are collected from a vehicle that moves through several streets. Specifically, we use the Extended Kalman Filter to estimate the car's trajectory while simultaneously estimating the position of the obstacles it encounters (i.e. the map).

II. Problem Formulation

The inertial measurement unit returns both linear velocity $v_t \in \mathbb{R}^3$ and angular velocity $w_t \in \mathbb{R}^3$ which is collected in the body frame of the IMU. The stereo camera returns pixel coordinates $z_t \in \mathbb{R}^{4xM}$ which are coordinates with respect to the left and right cameras (thus, 4 dimensions). There are a total of *M* observed "landmark" features through all time steps.

Because the motion and observation models in this particular problem are nonlinear, we use the Extended Kalman Filter to linearize the models using Taylor series approximations. The EKF approach estimates a Gaussian distribution where the mean of this distribution is the predicted location for the car and assumes that the models are affected by Gaussian noise.

Before implementing Visual-Inertial SLAM, we first try the dead reckoning approach. We implement the EKF prediction step to estimate the inverse IMU pose $U_t \in SE(3)$ given control $u_t = [v_t^\top, w_t^\top]^\top$. That is, estimate $U_t|u_t$, where U_t describes the position of the car at time step *t*. Separately at first, we implement the EKF update step with the assumption that U_t is known. Given the left camera intrinsic matrix *K*, the stereo camera baseline (the distance between the cameras) *b*, the extrinsic matrix from the IMU to the same left camera, and the landmark features collected from the stereo camera (z_t), we estimate the coordinates of the landmarks *m* in the world frame. That is, estimate $m|K, b, {}_{camera}T_{IMU}, z_t$.

After a baseline dead reckoning approach is implemented, we implement Visual-Inertial SLAM by combining the EKF prediction step given the IMU data, EKF update step given the landmark features from the stereo camera, and EKF update step given the IMU data. That is, estimate

$$p(m, U_t|K, b, {}_{camera}T_{IMU}, z_t, u_t) \tag{1}$$

by finding the optimal Gaussian distribution given $K, b, {}_{camera}T_{IMU}, z_t, u_t$.

In summary, create an algorithm that uses IMU measurements and stereo camera measurements to generate a map that the car observes and plot its trajectory within this environment.

III. Technical Approach

A. Localization Only

First, we implement the EKF prediction step to estimate the inverse IMU pose $U_t \in SE(3)$. This inverse IMU pose describes the car's location in the environment. The prediction step is done once per time step.

Given $u_t = [v_t^\top, w_t^\top]^\top$, we calculate $\hat{u}$:

$$\hat{u} = \begin{bmatrix} \hat{w}_t & v_t \\ 0^\top & 0 \end{bmatrix} \in \mathbb{R}^{4\times 4} \tag{2}$$

where $\hat{w}_t$ is the hat map of the the w_t vector:

$$\hat{w}_t = \begin{bmatrix} 0 & -w_3 & w_2 \\ w_3 & 0 & -w_1 \\ -w_2 & w_1 & 0 \end{bmatrix} \tag{3}$$

The mean of this EKF predict step represents the transformation ${}_{IMU}T_{world}$. We initialize the mean to be a *4x4* the identity matrix and update it as follows with a time discretization value τ:

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

To update the covariance, we need to define $u_t^{\curlywedge}$:

$$u_t{}^{\curlywedge} = \begin{bmatrix} \hat{w}_t & \hat{v}_t \\ 0 & \hat{w}_t \end{bmatrix} \in \mathbb{R}^{6\times 6} \tag{5}$$

Thus, given some Gaussian noise parameterized by covariance *W*:

$$\Sigma_{t+1|t} = exp(-\tau u_t^{\curlywedge})\Sigma_{t|t} exp(-\tau u_t^{\curlywedge})^\top + W \tag{6}$$

This covariance term is not used for localization for now, but will be used during the Visual-Inertial SLAM process.

B. Mapping Only

Like EKF prediction, EKF update is also done once per time step. Given all the landmark features observed by the stereo camera, we first discard the ones that are not observed at the current time step. From this pre-processing step, we receive a $z_t \in \mathbb{R}^{4 \times N_t}$ where N_t is the number of observable features at the current time step *t*.

Next, we must convert all the features, which currently reside in the stereo camera frame, to the world frame. We calculate a disparity measurement between the *u* pixel coordinate of the left camera versus the right camera: $d = u_L - u_R = \frac{1}{Z_o} f_{su} b$. Solving for Z_o (which is the ”z” coordinate in the optical frame), we get $Z_o = \frac{f_{su} b}{u_L - u_R}$. Similarly, we can calculate the optical frame coordinates for ”x” and ”y”: $X_o = Z_o \frac{u_L - c_u}{f_{su}}$ and $Y_o = Z_o \frac{v_L - c_v}{f_{sv}}$ where c_u and c_v are the coordinates of the principal point used to translate the image frame origin. Additionally f_{su} and f_{sv} define the scaling from meters to pixels.

To convert the coordinates from the optical frame (denoted as *o*) to the world frame, we must do the following:

$$m = \mu_{t|t}^{-1} \times {}_{cam}T_{IMU}^{-1} \times optical = {}_{world}T_{IMU} \times {}_{IMU}T_{cam} \times o \tag{7}$$

where $\mu_{t|t}^{-1}$ is from the previous localization step.

Next, we use the formal EKF update step. If a landmark has not been seen before, we initialize its mean to be the identity matrix and covariance to be zero. If a landmark has been seen before, we calculate the Kalman gain (K_t) to update the mean and covariance by first computing the predicted observation $\hat{z_t}$:

$$\hat{z_t} = M\pi({}_{cam}T_{IMU} \times \mu_{t|t}^{-1} \times \mu_{t,landmark}) \tag{8}$$

where

$$q = {}_{cam}T_{IMU} \times \mu_{t|t}^{-1} \times \mu_{t,landmark} \tag{9}$$

$$\frac{d\pi}{dq} = \frac{1}{q_3} \begin{bmatrix} 1 & 0 & -q_1/q_3 & 0 \\ 0 & 1 & -q_2/q_3 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & -q_4/q_3 & 1 \end{bmatrix} \tag{10}$$

$$H_t = M \times \frac{d\pi}{dq} {}_{cam}T_{IMU} \times \mu_{t|t}^{-1} \times P^{\top} \tag{11}$$

$$M = \begin{bmatrix} f_{su} & 0 & c_u & 0 \\ 0 & f_{sv} & c_v & 0 \\ f_{su} & 0 & c_u & -f_{su}b \\ 0 & f_{sv} & c_v & 0 \end{bmatrix} \tag{12}$$

where *P* is a 3x3 identity matrix with a column of zeros at the end. M is the calibration matrix that is constructed from *K* (the left camera intrinsic matrix). H_t is the Jacobian with respect to the landmark features at time step *t*.

The Kalman gain can now be computed:

$$K_t = \Sigma_t H_t^{\top} (H_t \Sigma_t H_t^{\top} + I \otimes V)^{-1} \tag{13}$$

where *V* is the covariance of a Gaussian distribution that is added here as noise. In practice, $I \otimes V$ in effect is the identity matrix with small diagonal elements. We experimented with various noise values (like 1e-5, 1e-3, 1e-2, 1e2, 1e3, and 1e5) but found the regular identity matrix ($I \otimes V = I$) to provide the best results if we initialize covariance to zero.

Given this Kalman gain, the mean and covariance can be updated now given our original observation z_t in the stereo camera frame and predicted observation $\hat{z_t}$ (calculated above):

$$\mu_{t+1} = \mu_t + K_t(z_t - \hat{z_t}) \tag{14}$$

$$\Sigma_{t+1} = (I - K_t H_t)\Sigma_t \tag{15}$$

C. Localization and Mapping: Visual-Inertial SLAM

Because the car’s location depends on the location of the landmark, we must update the covariance term jointly. To do this, we need to first construct a new state vector that considers the vehicle pose and the landmarks jointly as well. This new state vector must be sampled from a Gaussian distribution as follows:

$$\tilde{x_t} \sim N(\begin{bmatrix} \mu_{pose} \\ \mu_{landmark} \end{bmatrix}, \begin{bmatrix} \Sigma_{pose,pose} & \Sigma_{pose,landmark} \\ \Sigma_{landmark,pose} & \Sigma_{landmark,landmark} \end{bmatrix}) \tag{16}$$

We can derive a new matrix that represents the mean for both the car’s pose and the mean of the landmarks together:

$$\tilde{\mu_t} = \begin{bmatrix} \mu_{pose} \\ \mu_{landmark} \end{bmatrix} \tag{17}$$

The following observations are made about the dimensions of these parameters:

- $\mu_{pose} \in \mathbb{R}^{4 \times 4}$
- $\mu_{landmark} \in \mathbb{R}^{M \times 4}$
- $\Sigma_{pose,pose} \in \mathbb{R}^{6 \times 6}$
- $\Sigma_{pose,landmark,} \in \mathbb{R}^{6 \times 3M}$
- $\Sigma_{landmark,pose} \in \mathbb{R}^{3M \times 6}$
- $\Sigma_{landmark,landmark} \in \mathbb{R}^{3M \times 3M}$

The μ_{pose} and $\Sigma_{pose,pose}$ used in the EKF predict step to predict the new mean and covariance are exactly the same as described above in Section A: ”Localization Only”.

For the EKF update portion, we create a new Jacobian that stacks both the Jacobian with respect to the pose and with respect to the landmarks: $H_t = [H_{pose}, H_{landmarks}]$. $H_{landmarks}$ is computed exactly the same as described above in Section B: ”Mapping Only”. However, to compute H_{pose}, we use the same *q* established in Equation (9):

$$H_{pose} = M \times \frac{d\pi}{dq}(q) \times {}_{cam}T_{IMU} \times (\mu_{t+1|t} \times \mu_{landmark})^{\odot} \tag{18}$$

where $\mu_{pose} = \mu_{t+1|t}$. $(\mu_{t+1|t} \times \mu_{landmark}) \in \mathbb{R}^4$ and thus the $\odot$ operator takes the first 3 elements (denoted as *s*) of this homogeneous vector and computes the following:

$$\begin{bmatrix} s \\ 1 \end{bmatrix}^{\odot} = \begin{bmatrix} I & -\hat{s} \\ 0 & 0 \end{bmatrix} \in \mathbb{R}^{4 \times 6} \tag{19}$$

Thus, $H_{pose} \in \mathbb{R}^{4N_t \times 6}$ and $H_{landmark} \in \mathbb{R}^{4N_t \times 3M}$. Concatenated together in H_t, we get that $H_t \in \mathbb{R}^{4N_t \times (3M+6)}$. H_t is calculated at every time step after $t = 0$.

The Kalman gain and covariance can be now updated jointly with this formulation:

$$A_t = H_t \Sigma_t H_t^\top + I \otimes V) \tag{20}$$

$$K_t = \Sigma_t H_t^\top (A_t)^{-1} \tag{21}$$

$$\Sigma_{t+1} = \Sigma_t - K_t A_t K_t^\top \tag{22}$$

Given the same equation for $\hat{z}_t$ that was defined in Equation (8) and the original observations z_t, we can also update the mean for the car's pose and observed landmarks.

For the pose mean update, we use the first 6 rows of K_t that correspond to the pose:

$$\mu_{t+1} = exp(K_t(\hat{z_t} - \hat{z_t})) \times \mu_t = \mu_{pose} \tag{23}$$

For the landmark mean update, we use the last $3M$ rows of K_t that correspond to the landmarks:

$$\mu_{t+1} = \mu_t + K_t(z_t - \hat{z}_t) = \mu_{landmark} \tag{24}$$

D. General Approach

We first started by implementing only localization, followed by separately implementing the mapping portion. This constituted the dead reckoning approach to solving this problem.

Next, we approached the problem with Visual-Inertial SLAM, which used the localization and mapping portions simultaneously. In order to run the code quickly, we downsampled the number of features given using the following metric: count the number of times each feature appears throughout all time steps. Only use the top 600 most frequently occurring features. Additionally, to ensure that there is an even spread of features used, pick every 5th occurring feature in the data and use it as well.

IV. Results

We will now present our results for each training set. The trajectory of the car is plotted in red and the observed landmarks are plotted in green.

A. Localization and Mapping Done Separately

For training set 22, we found the following result:

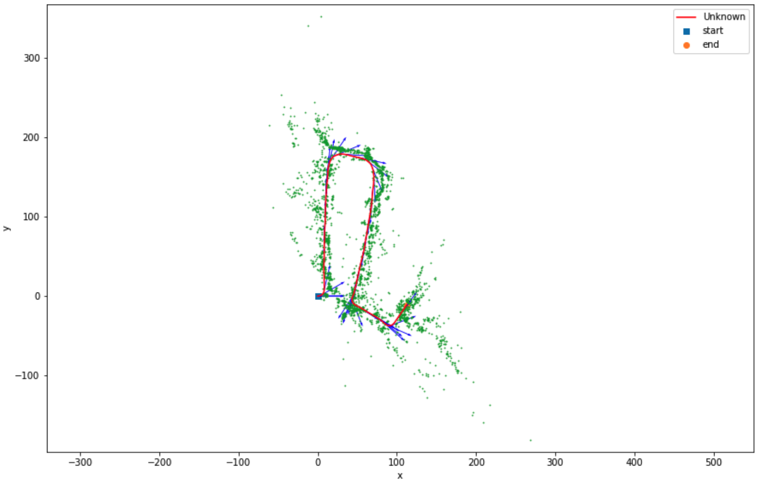

Fig. 1: Separate Localization and Mapping - Training Set 22

The trajectory lines are not completely straight, and the further the car travels through the environment, the more the trajectory starts to visibly bend.

For training set 27, we found the following result:

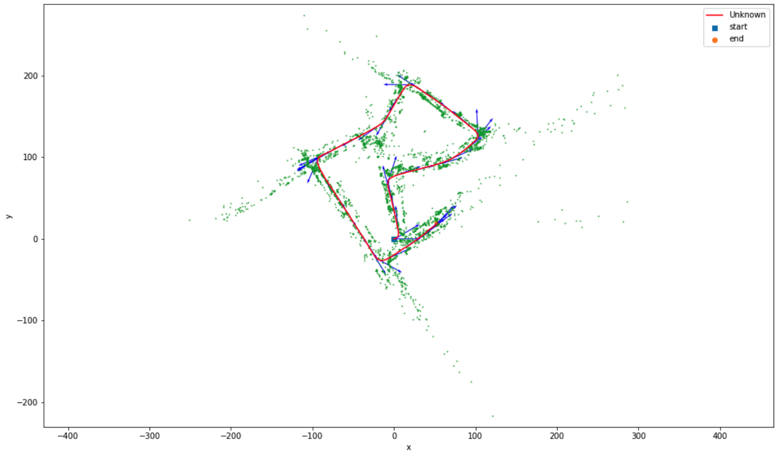

Fig. 2: Separate Localization and Mapping - Training Set 27

The trajectory loop does not close if we separately localize and map the car and environment. Additionally, like training set 22, some of the trajectory lines are very bent, contrary to the actual path that the car takes. The landmarks are closely clumped around the car's turning points, since the stereo camera observes a lot of points in a smaller area during the turn.

For training set 34, we found the following result:

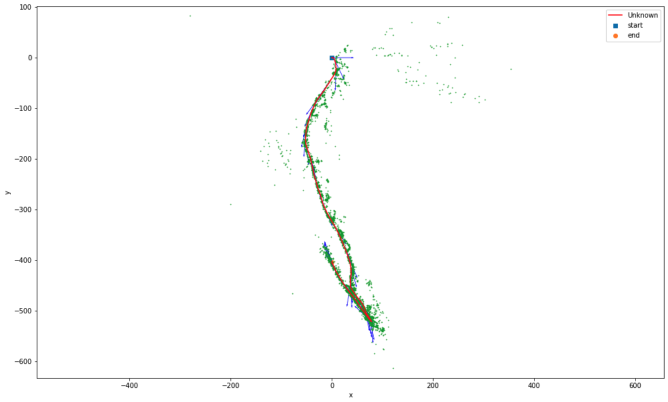

Fig. 3: Separate Localization and Mapping - Training Set 34

The trajectory near the end of the path bends too sharply again and does not accurately depict the path the car actually took. Instead, the trajectory lines come very close to overlapping towards the end when they should be far apart.

B. Visual-Inertial SLAM

All Visual-Inertial SLAM results are found with downsampling. The metric for downsampling is explained above in ”Section D: General Approach” of ”Technical Approach”. The best results were found when the covariance was initialized to zero rather than a small value at the very beginning since we technically do not have any information between the car pose and landmarks. The best results were also found when the noise vectors ”W” and ”V” were identity matrices. Larger noises were tried, but these experiments only produced good results when the covariance initializations were very small (but not zero).

For training set 22, we found the following result with SLAM:

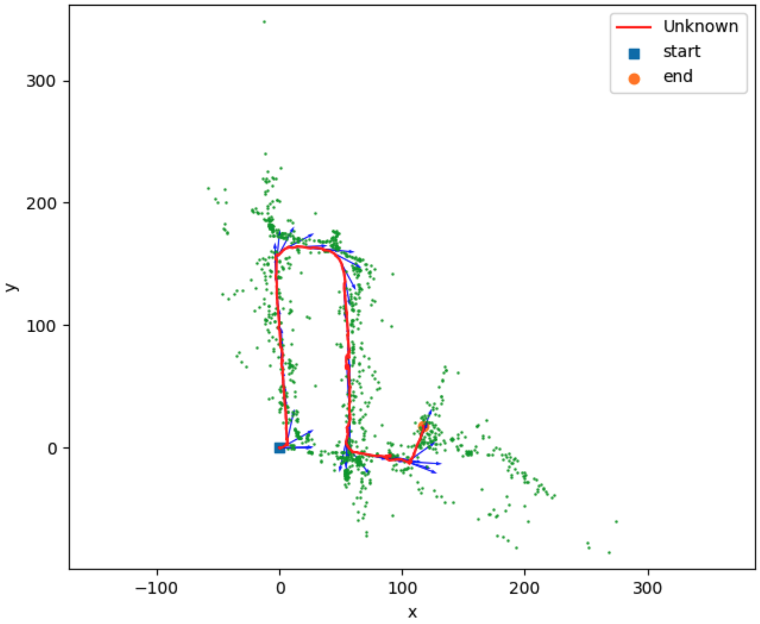

Fig. 4: Visual-Inertial SLAM - Training Set 22

The trajectory lines are much more straight and do not excessively bend when the car makes a turn. This result is better than that obtained by doing localization and mapping separately since there is indeed some correlation between the car’s pose and the observed landmarks’ positions. Additionally, despite downsampling, the landmarks’ positions remain relatively accurate and situated around the car’s trajectory.

For training set 27, we found the following result with SLAM:

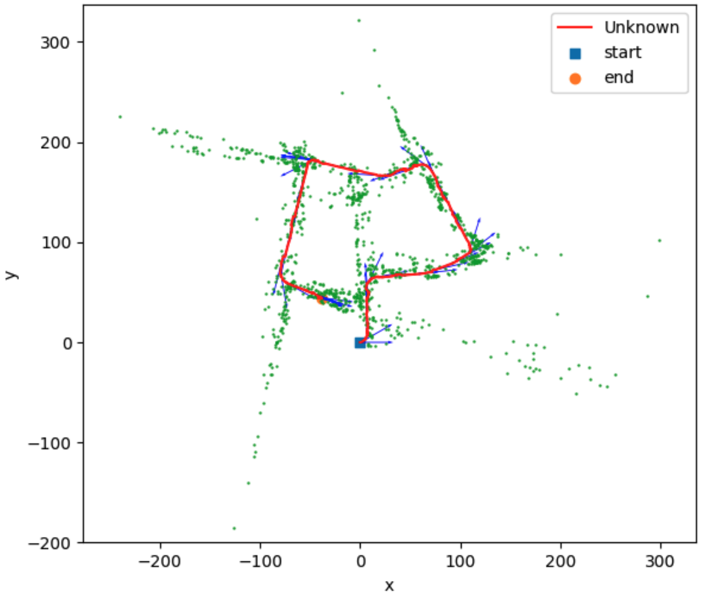

Fig. 5: Visual-Inertial SLAM - Training Set 27

Again, the lines are much more straight at the start of the car's path. However, although the overall trajectory resembles the reference video of the car's path, the final position of the car is not the same as the starting position - there is no loop closure unlike the reference video. Despite tuning the algorithm with various Gaussian noises, the best solution we found was when the noise was the identity matrix. Again, even with downsampling, the position of the landmarks remains relatively accurate.

For training set 34, we found the following result with SLAM:

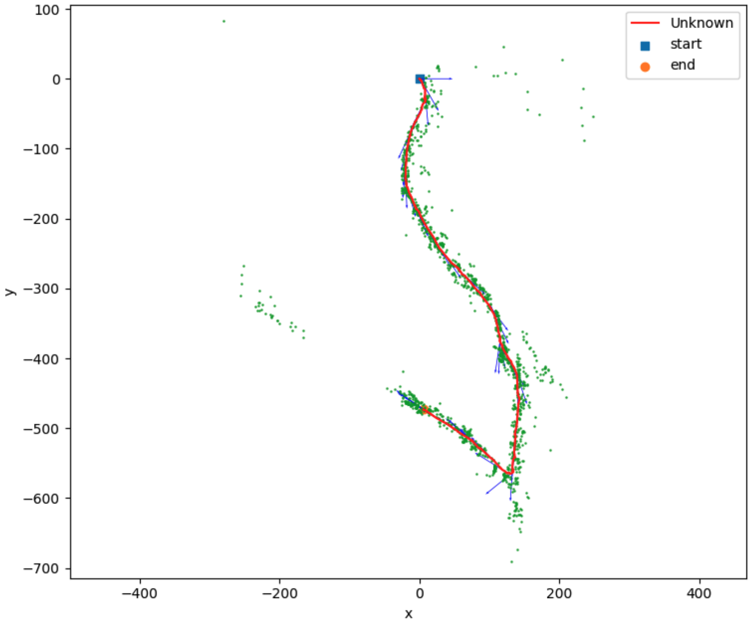

Fig. 6: Visual-Inertial SLAM - Training Set 34

The ending portion of the trajectory is much better compared to the original when localization and mapping are done separately. This trajectory more closely matches the reference video. Additionally, like with training sets 22 and 27, the landmark positions are accurately estimated as they are situated closely around the trajectory.

V. Conclusion

Visual-Inertal SLAM with Extended Kalman Filter produces good results. Extensive testing has showed that downsampling the original number of features is necessary to make the algorithm run in a timely manner. Further work on this project will largely involve trying to make the project more efficient and robust to a larger number of features.

Acknowledgment

I would like to thank Professor Atanasov and the class TA's for their guidance through this project, as well as Joan Sola from UPC's IRI department for documenting his approach to SLAM with EKF. I would also like to thank the NumPy website for their useful open-source documentation.