Visual Inertia SLAM with Extended Kalman Filter

Michael Liu
University of California San Diego
Electrical Computer & Engineering Department
x4liu@eng.ucsd.edu

Abstract

This report discussed the approach of using extended kalman filter to perform robust simultaneous localization and mapping with IMU and a stereo-rig camera sensors. In this work, we implemented the extended kalman filter and performed mapping with the given point correspondence dataset collected from a mobile vehicle.

1. Introduction

With the autonomous vehicle becomes increasingly relevant in real-world application, simultaneous localization and mapping has become one of the important challenges for intelligent system to estimate and interact with its dynamic surroundings. One of the problems with placing system in the real-world is that the data recorded from the robot is often noise, and it requires us to use optimization techniques to effectively eliminate the sensor noise to perform robust estimation. Previous work in solving the problem involving using particle filter or unscented kalman filter to perform robot pose estimation. In this paper, we will implement and discuss on another particular filter: the extended kalman filter, and test the robustness and effect of this filter.

2. Problem Formulation

2.1. Dataset Overview

In this task, we use stereo-camera data and IMU sensor data collected by a moving vehicle to perform estimation for the layout of the map. In this report, we discuss the result of these 3 of the total dataset collected by the vehicle on solving the simultaneous localization and mapping problem, and an overview of these dataset by the number of timestamps collected in the dataset is provided in **Table** 1.

Along with the dataset, we are also given the baseline of the stereo-rig camera $\mathbf{b}$, camera calibration matrix K and the transformation from IMU frame to camera optical frame ${}_{\text{cam}}T_{\text{imu}}$.

Vehicle Dataset	Total Landmark Features	IMU Timesteps
dataset 0022	3220	800
dataset 0027	3950	1106
dataset 0034	4815	1224

Table 1. VIO SLAM Dataset Overview

Figure 1. Example of Stereo-Camera Images from the dataset, dots represent the location of known features observed in the scene

2.2. Simultaneous Localization and Mapping Formulation

Given the dataset above, we formulate the problem as three parts for solving the Simultaneous Localization and Mapping problem.

1. **Robot Pose Localization (Motion Model)**: Given the linear velocity of robot position $v_t = \begin{bmatrix}\delta_x \\ \delta_y \\ \delta_z\end{bmatrix}$ and angular velocity of robot orientation $\omega_t = \begin{bmatrix}\omega_x \\ \omega_y \\ \omega_z\end{bmatrix}$ provided by the IMU sensor data, incorporate sensor noise to predict the next position of the robot orientation ${}_rT_w \in SE3$. Because the robot pose represents the spatial location and orientation in a 3-D space, the pose value needs to satisfies the constrains of being a special euclidean transformation ($SE3$). Here our motion

model formulation is:

$$_{\text{imu}}T^{t+1}_{\text{world}} = \exp\left(\tau \begin{bmatrix} \hat{\omega}_t & \mathbf{v}_t \\ \mathbf{0}^\top & 0 \end{bmatrix}\right) {}_{\text{imu}}T^{t}_{\text{world}} + \mathcal{N}(\mu, \sigma^2)$$

where $\mathcal{N}(\mu, \sigma^2)$ is our assumption of the sensor noise, which is a Gaussian distribution around the predicted value. τ is the absolute difference between the current timestep with the previous timestep.

2. **3D Visual Mapping (Camera Observation Model)**: Given the robot pose estimation ${}_{\text{imu}}T^{t}_{\text{world}} \in SE3$ and a series of observed 3D landmarks positions $\begin{bmatrix} x \\ y \\ z \end{bmatrix}_{\text{world}}$, we want to estimate the corresponding 2D pixel coordinates of these landmarks observed by the current robot pose. Since the landmarks is initialized in the IMU coordinate frames, we will first perform the transformation from the IMU frames into the camera frames with the transformation matrix ${}_{\text{cam}}T_{\text{imu}}$. We can then use the camera projection equation to get the uv-coordinates observed by the left camera $\begin{bmatrix} u_L \\ v_L \end{bmatrix}_{\text{cam}}$. The transformation from the world frame to the IMU frame and from IMU frame into the camera frame is:

$$\begin{bmatrix} X_o \\ Y_o \\ Z_o \\ 1 \end{bmatrix} = {}_{\text{cam}}T_{\text{imu}}\ {}_{\text{imu}}T_{\text{world}} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

The camera projection equation here is:

$$\underbrace{\begin{pmatrix} u_L \\ v_L \\ 1 \end{pmatrix}}_{\text{pixels}} = \underbrace{\begin{bmatrix} fs_u & 0 & c_u \\ 0 & fs_v & c_v \\ 0 & 0 & 1 \end{bmatrix}}_{\text{calibration matrix: } K} \begin{pmatrix} X_o/Z_o \\ Y_o/Z_o \\ 1 \\ 1/Z_o \end{pmatrix}$$

3. **3D Stereo-Camera Mapping (Stereo-Camera Observation Model)**:

With stereo-camera, we can also have a different observation model from 3D camera points to 2D pixel coordinates in both left and right cameras using the stereo-camera projection matrix M, and the observation model equation is as followed:

$$\begin{bmatrix} u_L \\ v_L \\ u_R \\ v_R \end{bmatrix} = \underbrace{\begin{bmatrix} fs_u & 0 & c_u & 0 \\ 0 & fs_v & c_v & 0 \\ fs_u & 0 & c_u & -fs_u b \\ 0 & fs_v & c_v & 0 \end{bmatrix}}_{M} \begin{pmatrix} X_o/Z_o \\ Y_o/Z_o \\ 1 \\ 1/Z_o \end{pmatrix} \tag{1}$$

where b is the baseline between the two stereo-cameras.

3. Technical Approach

In order to perform robust pose and landmark estimations, we will incorporate Extended Kalman Filter to effectively use observations at hands for refining our previous estimation. Our technical approach consists of two major parts: dead-reckoning of the robot pose estimation and visual landmarks and the EKF modification of the dead-reckoning approach.

3.1. Robot Pose Prediction (Dead-reckoning)

In the dead-reckoning of our robot pose estimation, we only use the data from the IMU sensor at each timestep. For computational convenience on our observation model, here we keep track of the inverse of the pose estimation $U^t = ({}_{\text{imu}}T^{t}_{\text{world}})^{-1}$. Given the IMU sensor data $v_t = \begin{bmatrix} \delta_x \\ \delta_y \\ \delta_z \end{bmatrix}$ and $\omega_t = \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix}$, we get our dead-reckoning pose prediction as:

$$U^{t+1} = \exp\left(-\tau \begin{bmatrix} \hat{\omega}_t & \mathbf{v}_t \\ \mathbf{0}^\top & 0 \end{bmatrix}\right) U^t$$

where $\hat{\omega} = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix}$, and τ is the absolute difference between the current timestep to previous timestep.

3.2. 3D Visual Mapping (Initialization)

To initialize our 3D landmarks, we use the stereo-projection equation to back project the observed points from both camera frames to triangulate the 3D landmarks, shown in Figure 2. The steps are state as following:

1. Calculate the disparity $d = u_L - u_R$
2. Calculate the Z_o coordinate in the camera frame, $Z_o = \frac{fs_u b}{d}$.
3. Calculate the X_o and Y_o coordinates in the camera frame, $\begin{bmatrix} X_o \\ Y_o \\ Z_o \end{bmatrix} = K^{-1} \begin{bmatrix} u_L \\ v_L \\ 1 \end{bmatrix} Z_o$
4. Calculate the 3D landmark points:

$$\begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} = \left[U^t\right]^{-1}_{\text{cam}} T^{-1}_{\text{imu}} \begin{bmatrix} X_o \\ Y_o \\ Z_o \\ 1 \end{bmatrix}$$

From the above equation, we can further derive that (this is the equation we implement in the code to avoid

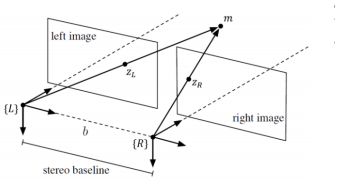

Figure 2. Stereo-Image Project Graph

matrix inversion):

$$X_o = (u_L - c_x)Z_o/f_x$$

$$Y_o = (v_L - c_y)Z_o/f_y$$

5. Initialize the landmark prior $\mu = \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$ and landmark co-variance matrix as $\Sigma = I_{3\times 3}$

3.3. Extended Kalman Filter for Robot Pose Prediction

An EKF is a nonlinear Kalman filter that forces the predicted and updated probability density functions to be Gaussian using a first-order Taylor series approximation to the motion and observation models. Here, our robot pose at time t is $\mu_t = U^t$, and our initial co-variance matrix for pose is $\Sigma_{0|0} = I_{6\times 6}$. Our predict step with EKF for the pose estimation becomes:

$$\begin{aligned} \mu_{t+1|t} &= \exp\left(-\tau \hat{u}_t\right) \mu_{t|t} \\ \Sigma_{t+1|t} &= \exp\left(-\tau u_t^{\curlywedge}\right) \Sigma_{t|t} \exp\left(-\tau u_t^{\curlywedge}\right) + W\tau^2 \end{aligned} \tag{2}$$

where $\mu_{t|t} \in SE(3)$, $\hat{\mathbf{u}}_t := \begin{bmatrix} \hat{\omega}_t & \mathbf{v}_t \\ \mathbf{0}^\top & 0 \end{bmatrix} \in \mathbb{R}^{4\times 4}$ and $\mathbf{u}_t^{\curlywedge} := \begin{bmatrix} \hat{\omega}_t & \hat{\mathbf{v}}_t \\ 0 & \hat{\omega}_t \end{bmatrix} \in \mathbb{R}^{6\times 6}$. And the prediction step is where we take into the account of IMU sensor data to update the pose estimation for our vehicle.

3.4. Extended Kalman Filter for Landmark Update

Once the prediction step is accomplished, we need to update the observed landmark positions with both our predicted pose estimation and our camera observation. This is the part where we use extensively the pre-defined camera observation model.

$$\hat{z}_{t,i} = M\pi\left({}_{\text{cam}}T_{\text{imu}} U^t \mu_{t,j}\right), \pi(\mathbf{q}) = \frac{\mathbf{q}}{q_3} \in \mathbb{R}^4$$

In this equation, M is the stereo-matrix defined in Equation 1, and U^t is the predicted pose estimation of the vehicle. Our observation model gives us $z_{t,i} \in \mathbb{R}^4$, which contains the estimation of pixel coordinates in both stereo-cameras. However, this estimation is purely based on our existing knowledge about the robot pose and landmark position, and it doesn't take into the account of the ground-truth pixel coordinates observed in the camera scene. To perform an optimization step using the ground-truth pixel coordinates to update the landmark position, we will first construct a Jacobian matrix H, which will allow us to calculate the Kalman gain K.

The construction of matrix $H \in \mathbb{R}^{4N\times 3M}$, where N is the number of observed features at time t and M is all the previously observed landmark 3D positions up to time t. It is reasonable to think that over time M will begin to grow very large. Our H is a very sparse and block diagonal matrix. The definition of H is as follow:

$$H_{i,j,t} = M\frac{d\pi}{dq}\left(T_{oi} T_t \mu_{t,j}\right) P, P = \begin{bmatrix} I_3 \\ 0^T \end{bmatrix}$$

where

$$\frac{d\pi}{dq}(q) = \frac{1}{q_3}\begin{bmatrix} 1 & 0 & -\frac{q_1}{q_3} & 0 \\ 0 & 1 & -\frac{q_2}{q_3} & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & -\frac{q_4}{q_3} & 1 \end{bmatrix}$$

Here $H_{i,j,t}$ represents the ith observation in time t corresponding with jth landmark, and the block is filled with a matrix of 4×3. Once the H is calculated, we can then calculate the Kalman gain K, where we define V as our measurement noise ($\sigma^2 = 1$):

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

Using Kalman gain K, we can then update all the **observed** landmark mean position and co-variance matrix with observed camera feature coordinates z_t:

$$\begin{aligned} \mu_{t+1} &= \mu_t + K_t\left(z_t - \hat{z}_t\right) \\ \Sigma_{t+1} &= \left(I - K_t H_t\right)\Sigma_t \end{aligned}$$

3.5. Extended Kalman Filter for Pose/Landmark Joint Update

Since we want to perform joint update on both landmark position and robot pose estimation with observed pixel feature coordinates, we need to modify our above matrix H to include the cross-correlation between pose and landmarks. Our modified H has a shape of $4N \times 3M + 6$, with the 6 new columns recording the Jacobian of the cross-correlation between pose and observations. The block for Jacobian of cross-correlation between pose and observations is defined as:

$$H_{i,t+1|t} = M\frac{d\pi}{d\mathbf{q}}\left({}_{\text{cam}}T_{\text{imu}}\boldsymbol{\mu}_{t+1|t}\mathbf{m}_j\right){}_{\text{cam}}T_{\text{imu}}\left(\boldsymbol{\mu}_{t+1|t}\mathbf{m}_j\right)^{\odot} \in \mathbb{R}^{4\times 6} \tag{3}$$

where $\begin{bmatrix} \mathbf{s} \\ 1 \end{bmatrix}^{\odot} = \begin{bmatrix} I & -\hat{\mathbf{s}} \\ 0 & 0 \end{bmatrix} \in \mathbb{R}^{4\times 6}$, $\mathbf{m}_j$ represents the jth landmark of the ith observation at time t, and $\boldsymbol{\mu}_{t+1|t}$

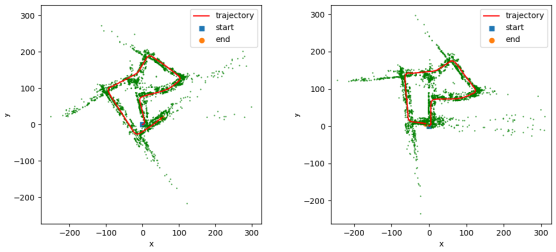

Figure 3. Comparison between dead-reckoning with the EKF update

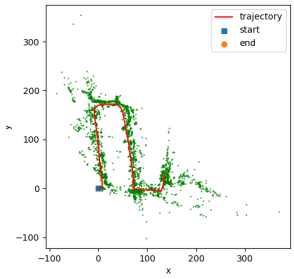

Figure 4. EKF SLAM Results on Dataset **0022**

represents the predicted pose of the vehicle. We will use the new K derived from new H as our Kalman gain. And our update for the pose is done through using the first 6 columns of the matrix as $K_{t+1|t}$ in the equation as follow:

$$\boldsymbol{\mu}_{t+1|t+1} = \exp\left(\left(K_{t+1|t}\left(\mathbf{z}_{t+1} - \hat{\mathbf{z}}_{t+1}\right)\right)^{\wedge}\right)\boldsymbol{\mu}_{t+1|t}$$
$$\Sigma_{t+1|t+1} = \left(I - K_{t+1|t}H_{t+1|t}\right)\Sigma_{t+1|t}$$

4. Results

In Figure 3, we have demonstrated the effectiveness of EKF filters applied to the sensing and estimation of the robot localization and mapping. We can see that with dead-reckoning alone the graph has significantly more drift compared with the map trajectory using EKF. We have also applied EKF filters in all the provided dataset: **dataset 0022** (Figure 4), **dataset 0027** (Figure 5) and **dataset 0034** (Figure 6). We have also demonstrated the evolution of map trajectory with **dataset 0022** (Figure 4), **dataset 0027** (Figure 8), and **dataset 0034** (Figure 6).

5. Discussion

In this section, we discussed the importance of two aspects of the EKF which controls the stability of the systems: the noise standard deviation applied in the motion and observation models and the number of recalled landmarks during the joint update step of pose and landmark estimation.

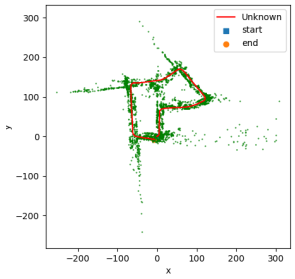

Figure 5. EKF SLAM Results on Dataset **0027**

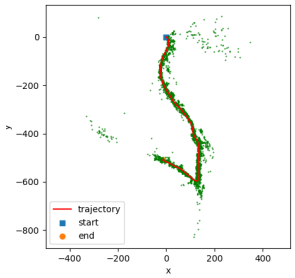

Figure 6. EKF SLAM Results on Dataset **0034**

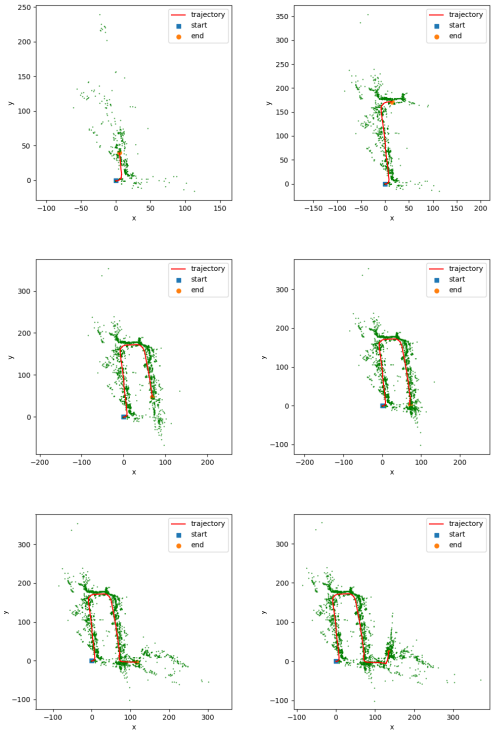

Figure 7. EKF SLAM Evolution on Dataset **0022**

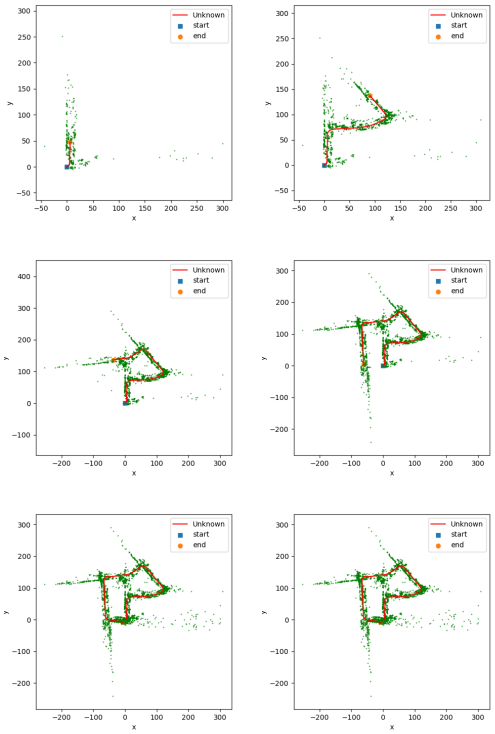

Figure 8. EKF SLAM Evolution on Dataset **0027**

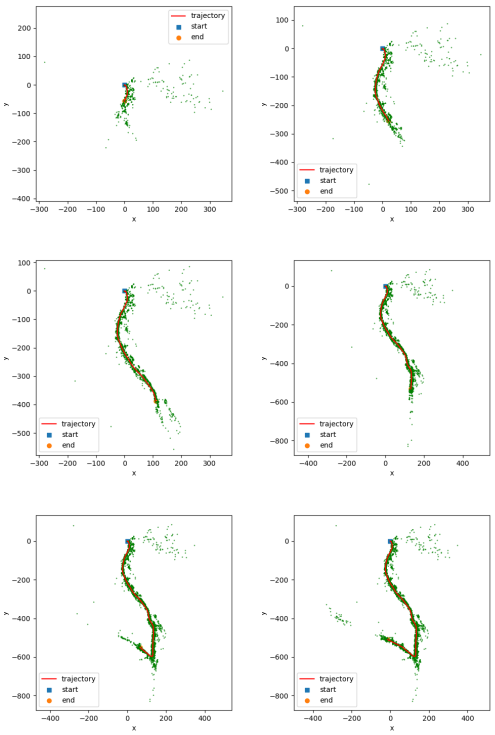

Figure 9. EKF SLAM Evolution on Dataset **0034**

5.1. Gaussian Noise in Motion Model

Because EKF leverage the first order of the Taylor series, if the noise is too large or too small, the system tends to diverge very quickly. This can be shown in the matrix inversion as singularity appears during the update pose if the system diverges. In our experiment, the system tends to remain stable around $0.1 < \sigma^2 < 10$, and a slightly too large or too small of the noise variance σ^2 is likely to diverge much later in the timestep. This rate of divergence is also related to the number of landmarks that the Jacobian matrix H tracks at time t, and large H tends to diverge easier than small H matrix.

5.2. Number of Recalled Features

As we have mentioned in our technical approach that the Jacobian $H \in \mathbb{R}^{4N \times 3M+6}$ tends to grow large due to increasing history of landmarks it tracks. However, the inversion of this large matrix can often make the system run very slow. In order to perform near real-time estimation of the trajectory using EKF, I used a variable r as the number of recalled landmarks I track in H that are unobserved in the current frame. During each timestep, the system will based on the latest observed time of the landmark to select the most recently updated r unobserved landmarks to include in the Jacobian matrix H. This works particularly effective in decreasing the computation complexity of our system, because landmarks observed in the early stage of prediction rarely affects the pose and landmark estimation later in the map. It is also important to see that as r grow large, the system will be able to use a larger bank of landmarks to track the current pose and landmark estimation, and this usually applies when the vehicle is moving slow and previous landmarks don't become irrelevant too quickly. In our cases, vehicle in the dataset tends to move in a high speed and thus, the result of $r = 150$ doesn't differ much from $r = 500$, as most of the unobserved landmarks becomes irrelevant very fast.