Kalman Filter based SLAM and Visual Tracking

Yusi Chen

October 9, 2021

Contents

1 Introduction

Visual-inertial SLAM is a problem where only visual (camera) measurements and inertial measurements (IMU) are used to do simultaneously localization and mapping (SLAM). In this project, we are provided with synchronized measurements from an IMU and a stereo camera. Then we want to use a Extended Kalman Filter (EKF) to do mapping and localization.

2 Problem Formulation

2.1 SLAM and Extended Kalman filter (EKF)

SLAM is such an estimation problem that given external output $u_{0:T}$ and observation $z_{0:T}$, we want to estimate the best current robot state x_T and external map m_T. Under Markov assumption, we could expression the joint pdf as Equation **??** where p_h and p_f are observation model and motion model respectively. To maximize the data likelihood given parameters (MLE), we aim to solve the optimization problem of $\max_{x_{0:T},m} \log p(x_{0:T}, m, z_{0:T}, u_{0:T-1})$. According to Equation 1, we want to find $x_{0:T}$ and m that maximize $p_h(z_t|x_t, m)$ and $p_f(x_t|x_{t-1}, u_{t-1})$ at every step t under a given observation and motion model.

$$\begin{aligned} p(x_{0:T}, m, z_{0:T}, u_{0:T-1}) &= p_{0|0}(x_0, m) \prod_{t=0}^{T} p_h(z_t|x_t, m) \prod_{t=1}^{T} p_f(x_t|x_{t-1}, u_{t-1}) \\ \log p(x_{0:T}, m, z_{0:T}, u_{0:T-1}) &= \sum_{t=0}^{T} \log p_h(z_t|x_t, m) + \sum_{t=1}^{T} \log p_f(x_t|x_{t-1}, u_{t-1}) \end{aligned} \tag{1}$$

For a Bayesian filter (extended Kalman filter in this case), we keep track of $p_{t|t}(x_t) := p(x_{t+1}|z_{0:t}, u_{0,t-1})$ so that we could find the best x_t and m. Under Markov assumption and Bayes rule, we could keep track of it by Equation **??**. In addition, this equation could be decomposed into prediction (the integral term) and update (the whole term). So we could compute $p_{t+1|t}(x)$ and $p_{t+1|t+1}(x)$ iteratively to do that.

$$\begin{aligned} p_{t+1|t+1}(x_{t+1}) &:= p(x_{t+1}|z_{0:t+1}, u_{0:t}) \\ &= \frac{p_h(z_{t+1}|x_{t+1})}{p(z_{t+1}|z_{0:t}, u_{0:t})} \int p_f(x_{t+1}|x_t, u_t) p_{t|t}(x_t) dx_t \\ \text{Prediction} : p_{t+1|t}(x) &:= \int p_f(x_{t+1}|x_t, u_t) p_{t|t}(x_t) dx_t \\ \text{Update} : p_{t+1|t+1}(x) &:= \frac{p_h(z_{t+1}|x_{t+1})}{p(z_{t+1}|z_{0:t}, u_{0:t})} p_{t+1|t}(x) \end{aligned} \tag{2}$$

Kalman filter is a Bayes filter with Gaussian prior, linear motion and observation model and Gaussian independent noise. Therefore, all state variables in a Kalman filter could be represented as Gaussian distribution. Extended Kalman filter could deal with more general, nonlinear motion model and observation model. EKF linearizes motion model and observation model around previous state mean and then forces the posterior distribution to be Gaussian distribution.

2.2 Visual mapping

For the mapping problem, we have the following assumptions:

- The inverse IMU pose $T_t =_w T_{I,t}^{-1} \in SE(3)$ over time is known from localization steps.
- Data association $\pi_t : \{1, ..., M\} \rightarrow \{1, ..., N_t\}$ stipulating which landmarks were observed at each time t is known or provided by an external algorithm

- The landmarks are static, i.e. it is not necessary to consider a motion model or a prediction step

Under the assumption of EKF, all state variables are represented as Gaussian distribution. Therefore, the distribution of map m given previous observations $z_{0:t}$ is $m|z_{0:t-1} \sim \mathcal{N}(\mu_t, \Sigma_t)$ with $\mu_t \in \mathbb{R}^{4\times M}$ and $\Sigma_t \in \mathbb{R}^{3M\times 3M}$. Then given a new observation z_t, our goal is to compute the updated distribution of $m|z_{0:t} \sim \mathcal{N}(\mu_{t+1}, \Sigma_{t+1})$.

2.3 Location prediction and update

In the location-only problem, we have the following assumptions:

- Landmark coordinates in the world frame $m \in \mathbb{R}^{4\times M}$ are known;
- Data association $\pi_t : \{1, ..., M\} \to \{1, ..., N_t\}$ stipulating which landmarks were observed at each time t is known or provided by an external algorithm.

In this part, we are given the IMU measurements $u_{0:T}$ with $u_t := [v_t, \omega_t]^T$ and visual feature observations $z_{0:T}$. Then we want to estimate the inverse IMU pose $T_t :=_W T_{I,t}^{-1} \in SE(3)$ over time.

3 Technical Approach

3.1 Location prediction and motion model

Given prior distribution on the inverse IMU pose $T_t|z_{0:t}, u_{0:t-1} \sim \mathcal{N}(\mu_{t|t}, \Sigma_{t|t})$ with $\mu_{t|t} \in SE(3)$ and $\Sigma_{t|t} \in \mathbb{R}^{6\times 6}$, and IMU measurement $u_t = [v_t, \omega_t]^T$ of linear and angular velocities, we now need to update $\mu_{t+1|t}$ and $\Sigma_{t+1|t}$ given the motion model shown in Equation 3:

$$T_{t+1} = \exp((\tau(-u_t + w_t))\hat{})T_t, \tag{3}$$

where $w_t \sim \mathcal{N}(0, W)$ is the model noise $\in \mathbb{R}^6$ and τ is the time discretization.

To linearize the motion model, let's separate the pose into mean pose $\mu_{t+1|t}$ and a perturbation ξ. Then the prediction equations are given as follows:

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

$$\xi_{t+1|t} = \exp(-\tau u_t^{\curlywedge})\xi_{t|t} + \tau w_t, \tag{5}$$

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

The EKF prediction step is then given as:

$$\begin{aligned} \mu_{t+1|t} &= \exp(-\tau \hat{u}_t)\mu_{t|t} \\ \Sigma_{t+1|t} &= \mathbb{E}[\xi_{t+1|t}\xi_{t+1|t}^T] \\ &= \exp(-\tau u_t^{\curlywedge})\Sigma_{t|t}\exp(-\tau u_t^{\curlywedge})^T + \tau^2 W \end{aligned} \tag{6}$$

3.2 Map update and observation model

Given the prior mapping distribution $m|z_{0:t} \sim \mathcal{N}(\mu_t, \Sigma_t)$ with $\mu_t \in \mathbb{R}^{4\times M}$ and $\Sigma_t \in \mathbb{R}^{3M\times 3M}$, we now need to update the mean μ_{t+1} and covariance Σ_{t+1} assuming the observation model:

$$z_{t,j} = h(T_t, m_j) + v_t = M\pi({}_OT_I T_t m_j) + v_t \tag{7}$$

where j is the j-th observed feature, T_t is the current estimate of the IMU pose, $\pi(q) = \frac{1}{q_3}q \in \mathbb{R}^4$ is the projection function, and $v_t \sim \mathcal{N}(0, V) \in \mathbb{R}^4$ is the observation noise.

To do the update step in Kalman filter, we first need the innovation term, which is defined as follows:

$$\Delta z_{t,j} = z_{t,j} - M\pi({}_OT_I T_t \mu_{t,j}) \in \mathbb{R}^{4\times 1}, \tag{8}$$

where $z_{t,j}$ is the pixel coordinates of all the current feature observations j and if we stack $\Delta z_{t,j}$ together in a column vector, we could have $\Delta z_t \in \mathbb{R}^{4N_t\times 1}$.
Then to linearize observation model, we want to compute the Jacobian matrix $H_t \in \mathbb{R}^{4N_t\times 3M}$ as $\partial z_{t,j}/\partial m$. We could use the first-order Taylor series to derive this:

$$\begin{aligned} z_{t,j} &= M\pi({}_OT_I T_t(\mu_{t,j} + D\delta_{t,j})) \\ &\approx M\pi({}_OT_I T_t \mu_{t,j}) + M\frac{d\pi}{dq}({}_OT_I T_t \mu_{t,j}){}_OT_I T_t D\delta_{t,j} \\ H_{i,j} &= M\frac{d\pi}{dq}({}_OT_I T_t \mu_{t,j}){}_OT_I T_t D \in \mathbb{R}^{4\times 3} \end{aligned} \tag{9}$$

where $D = [I_3, 0]^T \in \mathbb{R}^{4\times 3}$, and $\delta_{t,j}$ denotes a small perturbation of position of the lankmark j. Then we could stack $H_{i,j}$ on the diagonal to get a big Jacobian matrix $H_t \in \mathbb{R}^{4N_t\times 3M}$. H_t is a block diagonal matrix.
With the above terms defined, we can then perform the EKF update step as follows:

$$\begin{aligned} K_t &= \Sigma_t H_t^T (H_t \Sigma_t H_t^T + I \otimes V)^{-1} \\ \mu_{t+1} &= \mu_t + DK_t\Delta z \\ \Sigma_{t+1} &= (I - K_t H_t)\Sigma_t \end{aligned} \tag{10}$$

where $I \otimes V$ defines a block diagonal matrix with N_t subblock V's on the diagonal.

3.3 Location update

Given the result from the prediction step $T_{t+1}|z_{0:t}, u_{0:t+1} \sim \mathcal{N}(\mu_{t+1|t}, \Sigma_{t+1|t})$ with $\mu_{t+1|t} \in SE(3)$ and $\Sigma_{t+1|t} \in \mathbb{R}^{6\times 6}$, we now need to update the mean and covariance of the pose assuming the observation model:

$$z_{t+1,j} = h(T_{t+1}, m_j) + v_{t+1} = M\pi({}_OT_I T_{t+1} m_j) + v_{t+1} \tag{11}$$

Since in this step, we want to linearize the observation model in terms of T_{t+1} instead of m. Therefore, the Jacobian matrix $H_t \in \mathbb{R}^{4N_t\times 6} = \partial z_{t+1,j}/\partial T$. We used the first-order Taylor series to derive this:

$$\begin{aligned} z_{t+1,j} &= M\pi({}_OT_I \exp(\hat{\xi}_{t+1|t+1})\mu_{t+1|t}m_j) \\ &\approx M\pi({}_OT_I(I+\hat{\xi}_{t+1|t+1})\mu_{t+1|t}m_j) \\ &= M\pi({}_OT_I\mu_{t+1|t}m_j +_O T_I(\mu_{t+1|t}m_j)^{\odot}\xi_{t+1|t+1}) \\ &\approx M\pi({}_OT_I\mu_{t+1|t}m_j) + M\frac{d\pi}{dq}({}_OT_I\mu_{t+1|t}m_j)_OT_I(\mu_{t+1|t}m_j)^{\odot}\xi_{t+1|t+1} \\ H_j &= M\frac{d\pi}{dq}({}_OT_I\mu_{t+1|t}m_j)_OT_I(\mu_{t+1|t}m_j)^{\odot} \in \mathbb{R}^{4\times 6} \end{aligned} \tag{12}$$

where $\xi_{t+1|t+1} \in \mathbb{R}^{6\times 1}$ denotes a small pose perturbation of IMU, and the $(\cdot)^{\odot}$ maps a vector $r \in R^4$ to $r^{\odot} = ([s,\lambda]^T)^{\odot} = [[\lambda I, -\hat{s}],[0,0]] \in \mathbb{R}^{4\times 6}$. If we stack H_j vertically, we could obtain the big Jacobian matrix $H_t \in \mathbb{R}^{4N_t\times 6}$.

With the above terms defined, we can then perform the EKF update step as follows:

$$\begin{aligned} K_{t+1|t} &= \Sigma_{t+1|t}H_{t+1|t}^T(H_{t+1|t}\Sigma_{t+1|t}H_{t+1|t}^T + I\otimes V)^{-1} \\ \mu_{t+1|t+1} &= \exp((K_{t+1|t}\Delta z)^{\wedge})\mu_{t+1|t} \\ \Sigma_{t+1|t+1} &= (I - K_{t+1|t}H_{t+1|t})\Sigma_{t+1|t} \end{aligned} \tag{13}$$

where Δz is defined in Equation 8, $I\otimes V$ defines a block diagonal matrix with N_t subblock V's on the diagonal.

3.4 Camera reverse projection

In this problem, we are given features coordinates in the camera frame: $z = [u_L, v_L, u_R, v_R]^T$ and want to find its corresponding world frame m. According to the stereo camera model, m could be calculated as follows:

$$\begin{aligned} m &= T_t^{-1}[X_o, Y_o, Z_o, 1]^T \\ z &= M\frac{1}{Z_o}[X_o, Y_o, Z_o, 1]^T \\ Z_o &= \frac{fs_u b}{u_L - u_R} \end{aligned} \tag{14}$$

where M is the intrinsic matrix of the stereo camera and it is non-invertible. So in practice, I explicitly calculated X_o and Y_o based on fs_u and fs_v.

4 Results and Discussion

4.1 Prediction-only pose

In this part, I only used IMU readings to impose motion model on the robot. There's no update step involved here. As one may observed from Figure 1, 2 and 3, the prediction-only trajectory is already very reasonable. For example, in dataset 27, the pose trajectory went back to origin and in dataset 20, the pose trajectory shows exactly two driving lanes. This may indicate that the IMU readings are accurate and reliable.

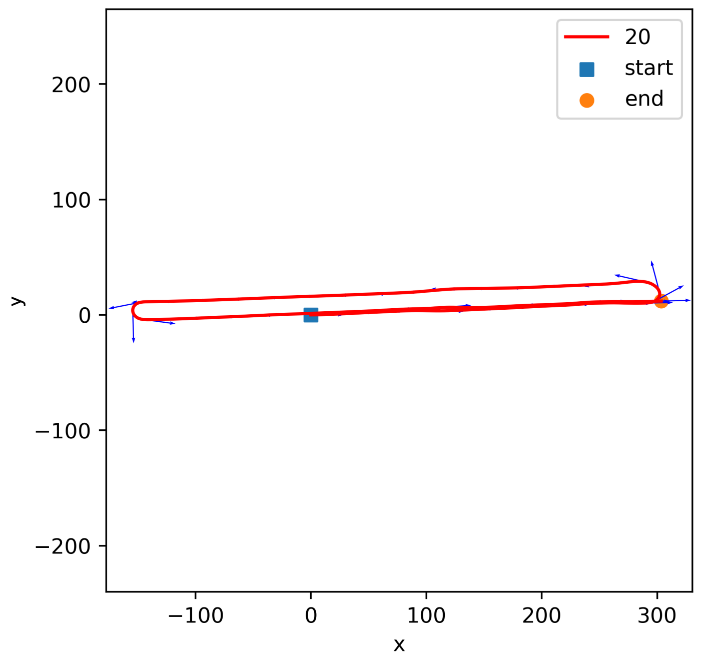

Figure 1: [Dataset20] Prediction-only trajectory in the world frame. Meter as unit

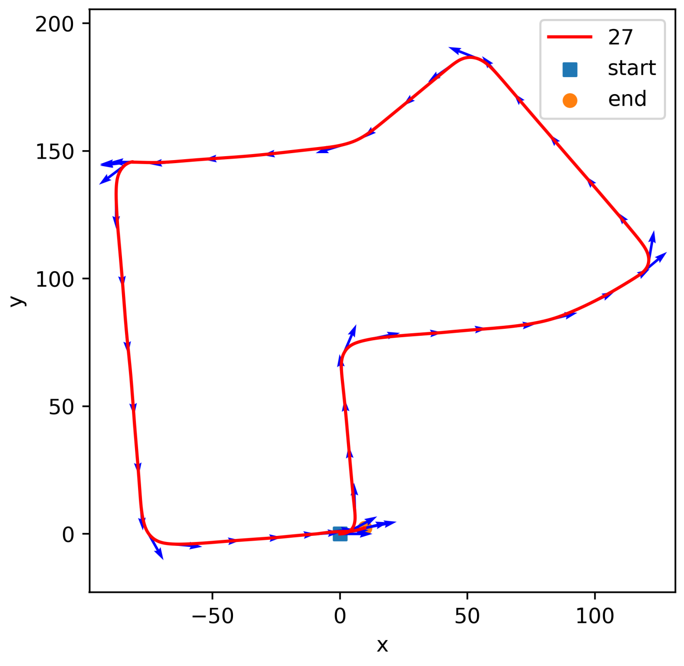

Figure 2: [Dataset27] Prediction-only trajectory in the world frame.

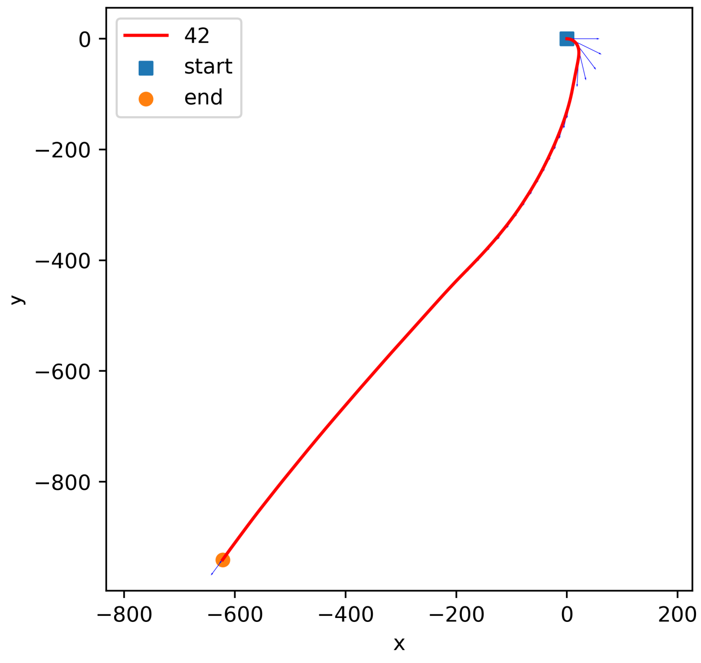

Figure 3: [Dataset42] Prediction-only trajectory in the world frame.

4.2 Visual mapping

In this part, we are assuming that the prediction-only trajectory is correct and use the above trajectory to calculate the feature coordinates in the world frame. I first initialized the map based on the pose where one given feature first appears as shown in Figure 4. Then I updated the feature coordinates based on Equation 10. Results are shown in Figure 5, 6 and 7

4.3 Visual-Inertial SLAM

To update location and map simultaneously, we treat both map $m \sim \mathcal{N}(\mu_1, \Sigma_1)$ and pose $T \sim \mathcal{N}(\mu_2, \Sigma_2)$ as state variable $\sim \mathcal{N}(\mu, \Sigma)$ where $\mu \in \mathbb{R}^{(3M+6)\times 1}$ and $\Sigma \in \mathbb{R}^{(3M+6)\times(3M+6)}$. Since during the calculation of μ_1 and μ_2, the Jacobian matrix with respect to m and T component don't interfere with each other. Therefore, we could still update μ based on Equation 10 and 13. However, in order to calculate Σ, which also account for the cross-corrletion between m and T, we need to compute a bigger Jacobian matrix. The calculation is shown as follows:

$$\begin{aligned} H &= [H_1; H_2] \in \mathbb{R}^{4N_t \times (3M+6)} \\ K &= \Sigma H^T (H \Sigma H^T + I \otimes V)^{-1} \in \mathbb{R}^{(3M+6)\times 4N_t} \\ \Sigma_{t+1} &= (I - KH)\Sigma \end{aligned} \tag{15}$$

where H_1 and H_2 are the Jacobian matrix computed in Equation 10 and 13.

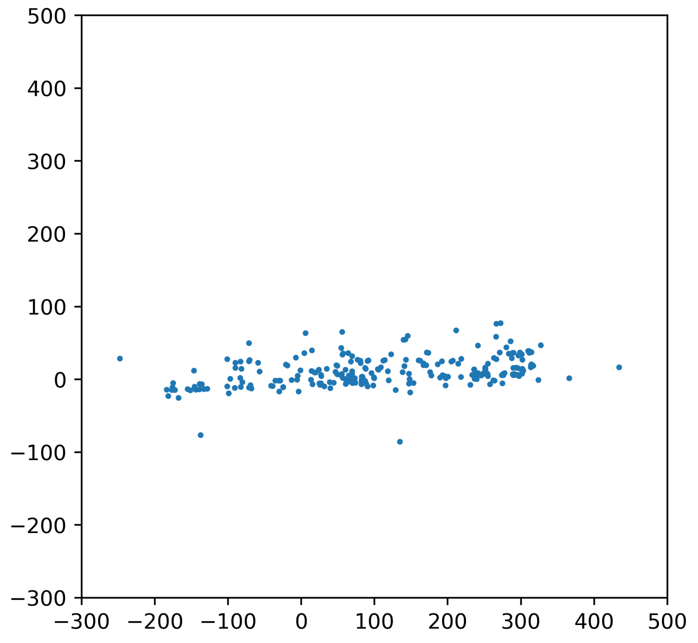

Figure 4: [Dataset20] Initial map of tracked features based on prediction-only trajectory. Each dot represented one tracked feature from the stereo camera

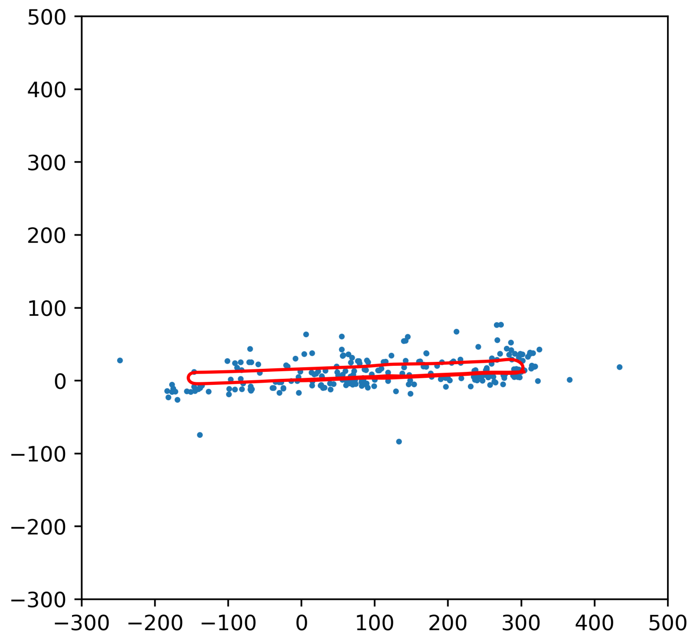

Figure 5: [Dataset20] Visual mapping based on prediction-only trajectory. Red line represented pose trajectory and blue dots are features.

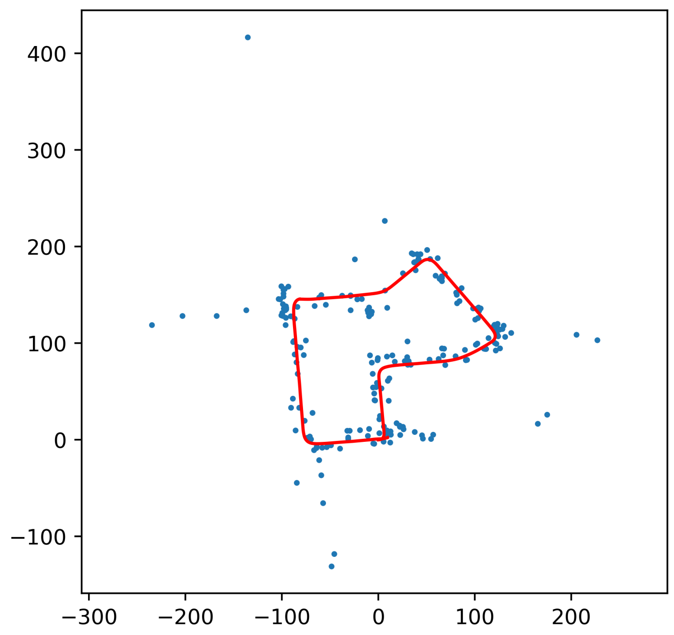

Figure 6: [Dataset27] Visual mapping based on prediction-only trajectory

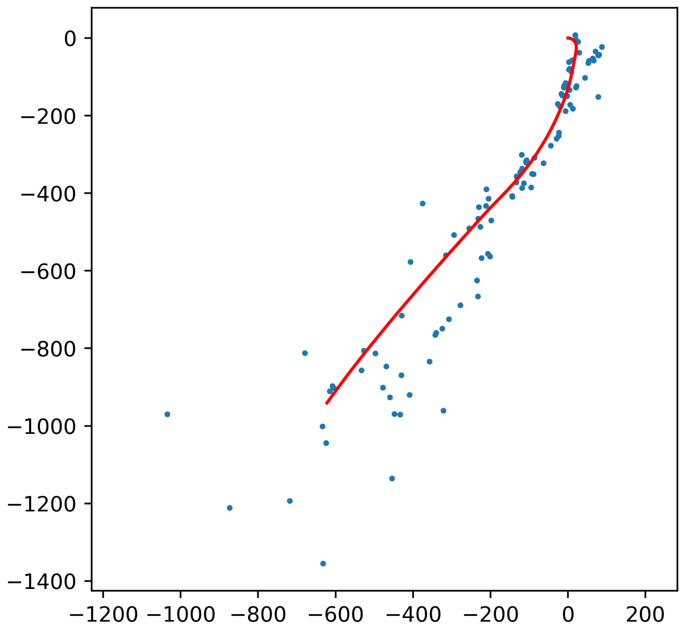

Figure 7: [Dataset42] Visual mapping based on prediction-only trajectory

The preliminary results of visual-inertial SLAM are shown in Figure 8. Surprisingly, the results is worse than prediction-only trajectory. I tried various noise levels (w and v) and Figure 8 is the best result among all the conditions I have tried.
After inspecting the video, I found several non-static features and some unstable features which could jump between frames. So I adopted the algorithm to account for the reliability of IMU and camera measurements.

4.3.1 Reliability of prediction and update

In my first attempt, I filtered out the non-static and unstable features. To be more specific, for one given feature, if its coordinate distance between two time frames are larger than 2 meters (dataset20, Approximately 72km/h). The algorithm is going to discard this observation. After feature filtering, the results are shown in Figure 9. There's improvement on the estimation, but it's still not as good as the prediction-only results.
Then I tried to constrain the location update level. I set a boundary on how much the location update step could modify mean pose. If the updated pose is too far away from the predicted pose, I forced the updated pose to be equal to the predicted pose. In dataset20, about one quarter of updated pose were reset as the predicted pose. Covariance matrix calculation remains the same. In this way, the SLAM was modified further toward prediction step. The results are shown in Figure 10, 11 and 12. For dataset 20, there's slight improvement compared to Figure 9 but still not as good as the prediction-only pose. For dataset 42,

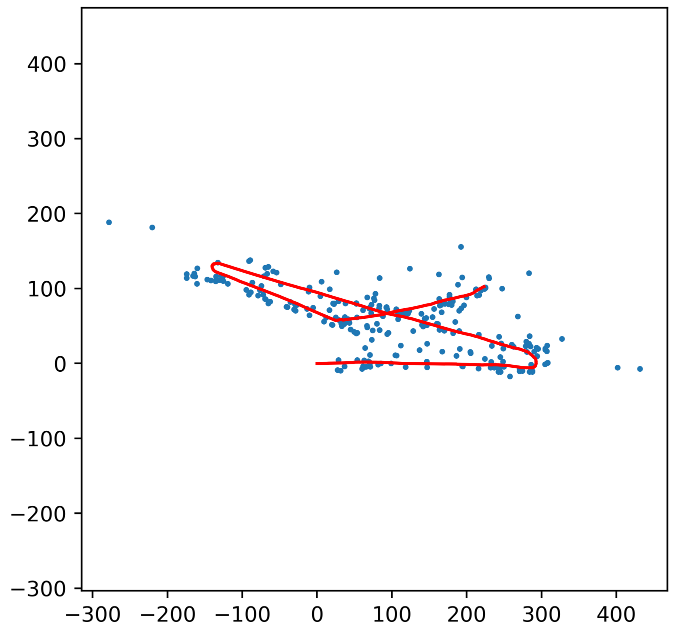

Figure 8: [Dataset20] Preliminary SLAM

there are two turning points which are not observed from the prediction-only trajectory. This may corresponding to the two act of changing lanes in the video. Maybe further parameter fine-tuning or improvement on the feature-tracking algorithm could improve the results.

4.3.2 Map initialization

If I initialized the map based on prediction-only trajectory, as I did in Figure 4, the SLAM results are substantially improved (Figure 13). But we need to be cautious about this initialization because again it is biased to the prediction measurements.

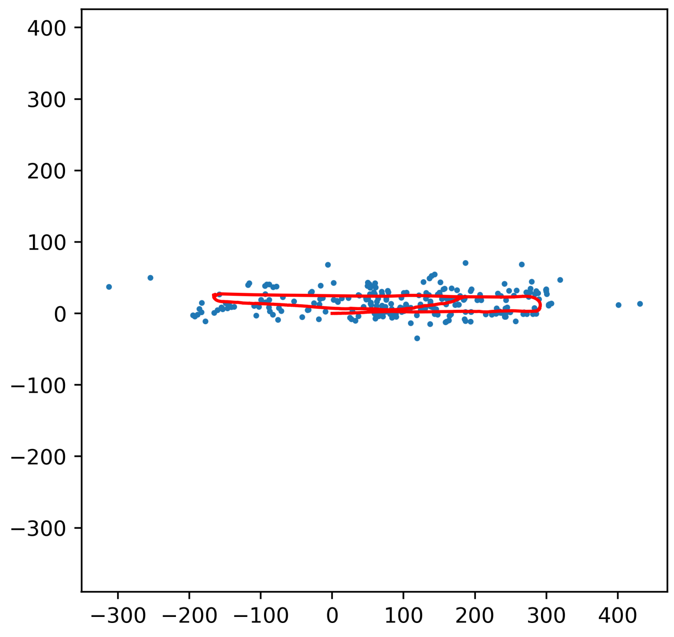

Figure 9: [Dataset20] Feature-filtered SLAM

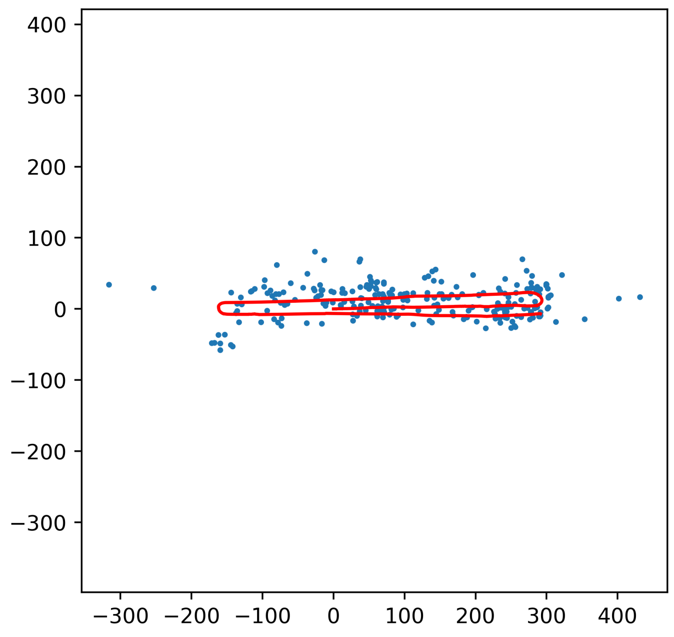

Figure 10: [Dataset20] Location update constrained SLAM

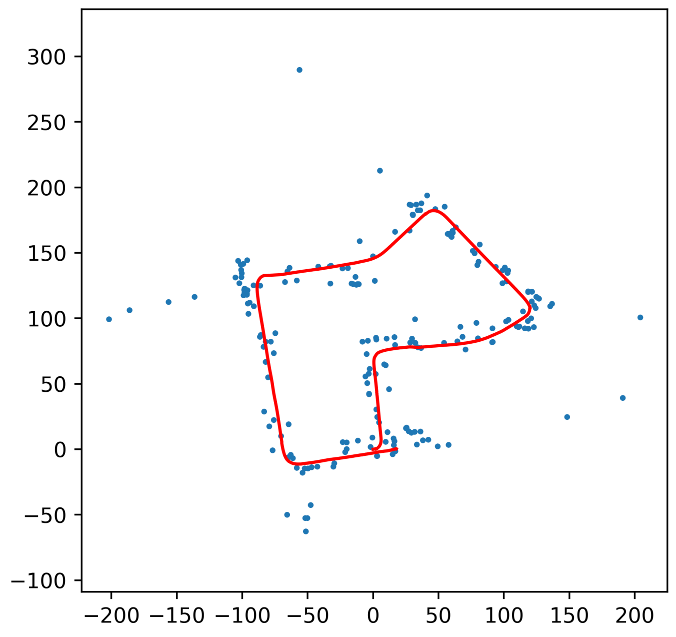

Figure 11: [Dataset27] Location update constrained SLAM

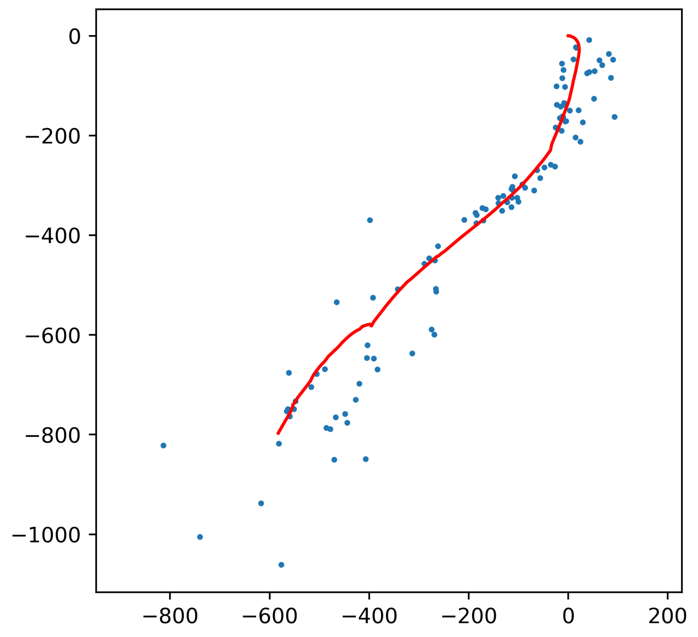

Figure 12: [Dataset42] Location update constrained SLAM

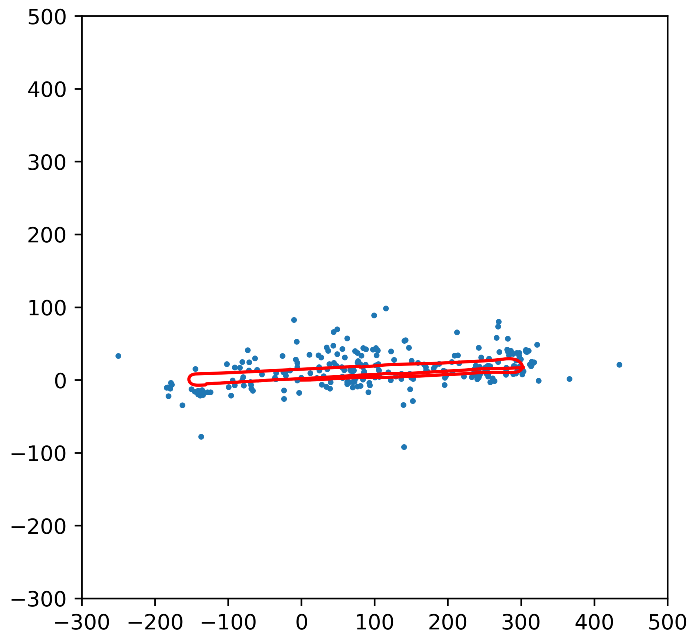

Figure 13: [Dataset 20] SLAM with initialization biased towards prediction