ECE 276 Project: Visual-inertial SLAM using EKF

Chen Du
Department of Electrical Computer Engineering
University of California, San Diego
c6du@eng.ucsd.edu

***Abstract*—This paper presented the methods used in Simultaneous Localization & Mapping (SLAM) based on IMU measurements and stereo camera images. The goal of the project is to use the Extended Kalman Filer (EKF) equations with a prediction step based on SE(3) kinematics and an update step based on the stereo camera observation model to perform localization.**

Keywords—SLAM, EKF, kinematics

I. INTRODUCTION

Simultaneous Localization and Mapping is part of navigation problem for mobile platforms that involves capturing the data from sensors and simultaneously creating map and calculating its location in this map. SLAM can be used in autonomous vehicle which helps to tell where the obstacles are and avoid them. This can also help people estimate a map without doing it in person. This make it possible to use robot doing dangerous tasks instead of human. There are many ways to implement SLAM, mostly based on what kinds of sensors equipped by the robots.

In this project, based on the sensors we have (Inertial Measurement Unit, Stereo Camera) as well as the pre-processed data (landmark positions and appeared time), I estimate the state (position, orientation) of the robot and 3D mapping of the landmarks using Extended Kalman Filter. The SLAM project is separated into three tasks. First, implement the EKF prediction step to estimate the pose of the robot over the time, based on the SE(3) kinematics and the IMU data (velocity, angular velocity). Second, suppose the first task is correct, using EKF to estimate the 3D map of the landmarks. The first two tasks will provide the trajectory of the robot and the 3D map built based on the trajectory. Third, combine the first two tasks and add update step for state estimation and 3D mapping. The third task is able to give a quite good result when the parameters are properly tuned.

In this paper, section II shows how I formulate the SLAM problem. Section III tells the detailed technique approaches for the project. And section IV provides the results of the project.

II. PROBLEM FORMULATION

A. Observation Model

The observation model I use is

$$z_{t,i} = M\pi\left(oTi\cdot\Delta T\cdot iTw\left(\mu_{t,j}+D\delta_{t,j}\right)\right) \tag{1}$$

Where $z_{t,i}$ is the pixel positions of observation i at time j. $u_{t,j}$ is the position of the landmark j in world frame, and known M is the stereo camera calibration matrix. The $\delta_{t,j}$ is the perturbation of landmark j, and ΔT is perturbation of the robot pose.

$$M := \begin{bmatrix} fs_u & 0 & c_u & 0 \\ 0 & fs_v & c_v & 0 \\ fs_u & 0 & c_u & -fs_u b \\ 0 & fs_v & c_v & 0 \end{bmatrix} \tag{2}$$

$$\pi(\mathbf{q}) := \frac{1}{q_3}\mathbf{q} \in \mathbf{R}^4 \tag{3}$$

$$\frac{d\pi}{d\mathbf{q}}(\mathbf{q}) = \frac{1}{q_3}\begin{bmatrix} 1 & 0 & -\frac{q_1}{q_3} & 0 \\ 0 & 1 & -\frac{q_2}{q_3} & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & -\frac{q_4}{q_3} & 1 \end{bmatrix} \tag{4}$$

$$D = \begin{bmatrix} I_3 \\ 0 \end{bmatrix} \tag{5}$$

B. Motion Model

In this project, we are provided with velocity and angular velocity as robot input. To make the problem easier, we use the pose (transformation matrix SE(3)) as the robot state. The motion model is

$$\begin{aligned} wTi_{t+1} &= wTi_t \cdot {}_iTi_{t+1} \\ &= wTi_t \cdot \exp\left(\left(\tau\left(u_t + w_t\right)^{\wedge}\right)\right) \end{aligned} \tag{6}$$

Where $wTi_t \in \mathbf{R}^{4\times4}$ is the transformation matrix from IMU to world at time t. $i_{t+1}Ti_t \in \mathbf{R}^{4\times4}$ is the transformation matrix for IMU from time t+1 to time t. w the input noise. $u_t = [v_t, \omega_t]$ is the input. And

$$\mathrm{se}(3) := \left\{ \hat{\xi} := \begin{bmatrix} \hat{\boldsymbol{\theta}} & \boldsymbol{\rho} \\ 0 & 0 \end{bmatrix} \middle| \xi = \begin{bmatrix} \boldsymbol{\rho} \\ \boldsymbol{\theta} \end{bmatrix} \in \mathbf{R}^6 \right\} \tag{7}$$

$$\mathrm{so}(3) := \left\{ \hat{\boldsymbol{\theta}} := \begin{bmatrix} 0 & -\theta_3 & \theta_2 \\ \theta_3 & 0 & -\theta_1 \\ -\theta_2 & \theta_1 & 0 \end{bmatrix} \middle| \boldsymbol{\theta} = \begin{bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \end{bmatrix} \in \mathbf{R}^3 \right\} \tag{8}$$

(9)

In this case we'll use iTw for easier implementation for EKF, considering the transformation between camera and world frame. The motion model is then

$$\begin{aligned} i_{t+1}Tw &= i_{t+1}Ti_t \cdot i_tTw \\ &= \exp\left(\left(\tau\left(-u_t + w_t\right)^{\wedge}\right)\right)\cdot i_tTw \end{aligned} \tag{10}$$

C. Task 1: IMU Localization via EKF Prediction

For task 1, we only need to implement EKF prediction step. Because it's Gaussian model, we only need to estimate the mean and the covariance. Based on the motion model mentioned and the EKF theory, we have

$$i_{t+1|t}Tw = \exp\left(-\tau \hat{u}_t\right) i_{t|t}Tw \tag{11}$$

$$\begin{aligned}\Sigma_{t+1|t}^{robot} &= E\left[\xi_{t+1|t}\xi_{t+1|t}^{T}\right] \\ &= \exp\left(-\tau \overset{\curlywedge}{u}_t\right)\Sigma_{t|t}^{robot}\exp\left(-\tau \overset{\curlywedge}{u}_t\right)^{T} + \tau^2 W\end{aligned} \tag{12}$$

ξ is perturbation, W the covariance of the input noise, and

$$\overset{\curlywedge}{u}_t := \begin{bmatrix} \hat{\omega}_t & \hat{v}_t \\ 0 & \hat{\omega}_t \end{bmatrix} \in \mathbf{R}^{6\times 6} \tag{13}$$

Since we are not doing update part. We assume

$$i_{t+1|t+1}Tw = i_{t+1|t}Tw \tag{14}$$

$$\Sigma_{t+1|t+1}^{robot} = \Sigma_{t+1|t}^{robot} \tag{15}$$

And iterate this over time to get the trajectory.

D. Task 2: Landmark Mapping via EKF Update

For this task, we assume the trajectory from task one is correct. And use EKF to update the 3D coordinates of the landmarks. We don't implement prediction step of EKF because we regard the features are static.

For the landmarks first time being observed, I use the inverse projection, based on the camera parameters and robot pose, to calculate the landmarks' world frame positions.

For update part, use the observation model and calculate the model Jacobian matrix H. Use the Jacobian matrix to calculate Kalman gain matrix K. And use K to update the position and covariances of the landmarks.

E. Task 3: Visual-Inertial SLAM

This part is the combination of the Task 1 and Task 2 but adding an update part for robot state. For each iteration, the whole process can be expressed as follows

1. Implement prediction step for robot state mentioned in Task 1. Get $i_{t+1|t}Tw$ and $\Sigma_{t+1|t}^{robot}$.
2. For each first-time observed landmarks, use $i_{t+1|t}Tw$ and camera parameters to implement inverse projection mentioned in Task 2, and get the positions of these landmarks in world frame.
3. Implement update part for robot state and mapping at the same time. We need to combine the robot state and landmark positions into one big matrix. Calculating the observation Jacobian matrix for both robot state and landmark position. And update the information at the same time.

Implementation details will be given in the next section.

III. Technical Approach

A. Task 1

This part is detailed described in problem formulation section. Using (11) and (12), we can have predicted state $i_{t+1|t}Tw$ and covariance $\Sigma_{t+1|t}^{robot}$ for the robot. The initial value for covariance is set to $\Sigma_{0|0}^{robot} = I_6 \in \mathbf{R}^{6\times 6}$.

B. Task 2

As mentioned, at time t+1, there may be new observed landmarks, and we need to initialize these points, supposing the robot state prediction step is correct.

Since we already have stereo camera calibration matrix M, pixel positions of the landmarks and transformation matrix from world to camera. Then for the new seen features, using first row and third row from (16). We can get (17)

$$\begin{bmatrix} u_L \\ v_L \\ u_R \\ v_R \end{bmatrix} = \begin{bmatrix} fs_u & 0 & c_u & 0 \\ 0 & fs_v & c_v & 0 \\ fs_u & 0 & c_u & -fs_u b \\ 0 & fs_v & c_v & 0 \end{bmatrix} \frac{1}{z} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{16}$$

$$z_o = \frac{fs_u b}{u_L - u_R} \tag{17}$$

And using first two rows in (16), we have

$$\begin{aligned}\begin{bmatrix} u_L \\ v_L \\ 1 \end{bmatrix} &= \begin{bmatrix} fs_u & 0 & c_u \\ 0 & fs_v & c_v \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_o / z_o \\ y_o / z_o \\ 1 \end{bmatrix} \\ z &= K \cdot \frac{1}{z_o} X_o\end{aligned} \tag{18}$$

K is the calibration matrix of the camera and X_o is the coordinate in optical frame. We can estimate the landmark in world frame using

$$\begin{aligned}\mu_{t+1|t} &= wTo \cdot X_o \\ &= wTi_{t+1|t} \cdot iTo \cdot K^{-1} \cdot z \cdot z_o\end{aligned} \tag{19}$$

Since the value of u_L and u_R are slightly different in the given data. I use the mean of this two to calculate the world frame position.

Then we implement update step for mapping. For this part, the observation model is

$$\begin{aligned} z_{t,i} \approx & \underbrace{M\pi\left(oTi \cdot iTw \cdot \mu_{t,j}\right)}_{\hat{z}_{t,i}} + \\ & \underbrace{M\frac{d\pi}{d\mathbf{q}}\left(oTi \cdot iTw \cdot \mu_{t,j}\right) \cdot oTi \cdot iTw \cdot D\,\delta_{t,j}}_{H_{i,j,t}^{LM}} \end{aligned} \tag{20}$$

$H_t \in \mathbf{R}^{4N_t \times 3M}$ is the observation model Jacobian evaluate at $\mu_{t,j}$, N_t is the observation number at time t and M is the total landmark number. And the element $H_{i,j,t}$ is Jacobian for observation i and landmark j. If landmark j is not detected at time t, $H_{i,j,t}$ is set to 0. The update step is

$$K_t = \Sigma_t^{LM} H_t^T \left(H_t \Sigma_t^{LM} H_t^T + I \otimes V\right)^{-1} \tag{21}$$

$$u_{t+1} = u_t + DK_t\left(z_t - \hat{z}_t\right) \tag{22}$$

$$\Sigma_{t+1}^{LM} = (I - K_t H_t)\Sigma_t^{LM} \tag{23}$$

This step helps us to update the landmark positions and their covariances.

C. Task 3

We need to combine the robot state with the landmark position. So I create a vector of mean

$$\mu_t = \begin{bmatrix} \mu_t^{robot} \in \mathbf{R}^{6\times1} \\ \mu_t^{LM} \in \mathbf{R}^{4M\times1} \end{bmatrix} \tag{24}$$

And matrix of covariance

$$\Sigma = \begin{bmatrix} \Sigma_{robot} \in \mathbf{R}^{6\times6} & \Sigma_{robot\_LM} \in \mathbf{R}^{6\times3M} \\ \Sigma_{LM\_robot} \in \mathbf{R}^{3M\times6} & \Sigma_{LM} \in \mathbf{R}^{3M\times3M} \end{bmatrix} \tag{25}$$

The relation between u^{robot} and iTw is

$$\begin{aligned} iTw &= \exp\left(\left(\mu^{robot}\right)^{\wedge}\right) \\ \mu^{robot} &= \left(\log\left(iTw\right)\right)^{\vee} \end{aligned} \tag{26}$$

And the initialization for Σ is $0.01 I_{6+3M}$.

For step 1, prediction is then performed as

$$\begin{aligned} \mu_{t+1|t}^{robot} &= \left(\log\left(\exp\left(-\tau\hat{u}_t\right) i_{t|t}Tw\right)\right)^{\vee} \\ &= \left(\log\left(\exp\left(-\tau\hat{u}_t\right)\cdot\exp\left(\left(\mu_{t|t}^{robot}\right)^{\wedge}\right)\right)\right)^{\vee} \end{aligned} \tag{27}$$

$$\Sigma_{t+1|t} = \begin{bmatrix} \exp\left(-\tau \overset{\curlywedge}{u}_t\right) & 0 \\ 0 & I \end{bmatrix} \Sigma_{t|t} \begin{bmatrix} \exp\left(-\tau \overset{\curlywedge}{u}_t\right) & 0 \\ 0 & I \end{bmatrix}^T + \tau^2 W \tag{28}$$

Where

$$W = \begin{bmatrix} \Sigma_w \in \mathbf{R}^{6\times6} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \tag{29}$$

Is the covariance for input noise.

For step 2, adding first observed landmark positions, this is exactly the same process as task 2, using function (17) to (19), I'm not going to repeat it here.

For step 3, update part, we are updating the landmark and robot state at the same time.

The observation model for landmark is still the same as (20). And the observation model for the robot is

$$\begin{aligned} z_{t+1,i} &= M\pi\left(oTi\cdot\exp\left(\hat{\xi}_{t+1|t+1}\right)\cdot i_{t+1|t}Tw\cdot\mu_{t+1,j}\right) \\ &\approx \underbrace{M\pi\left(oTi\cdot i_{t+1|t}Tw\cdot\mu_{t+1,j}\right)}_{\hat{z}_{t+1,i}} \\ &\quad + \underbrace{M\frac{d\pi}{d\mathbf{q}}\left(oTi\cdot i_{t+1|t}Tw\cdot\mu_{t+1,j}\right)\cdot oTi\left(i_{t+1|t}Tw\cdot\mu_{t+1,j}\right)^{\odot}}_{H_{i,t+1|t}^{robot}}\cdot\xi_{t+1|t+1} \end{aligned} \tag{30}$$

Where

$$\begin{bmatrix} s \\ \lambda \end{bmatrix}^{\odot} = \begin{bmatrix} \lambda I_3 & -\hat{s} \\ 0 & 0 \end{bmatrix} \tag{31}$$

And $s \in \mathbf{R}^3, \lambda \in \mathbf{R}$

In order to do update at the same time. The observation model Jacobian then is

$$H_{t+1|t} = \begin{bmatrix} H_{1,t+1|t} \\ \vdots \\ H_{N_{t+1},t+1|t} \end{bmatrix} \in \mathbf{R}^{4N_{t+1}\times(6+3M)} \tag{32}$$

And each $H_{1,t+1|t}$ is

$$\begin{aligned} &H_{i,t+1|t} \in \mathbf{R}^{4\times(6+3M)} = \\ &\left[H_{i,t+1|t}^{robot} \in \mathbf{R}^{4\times6}, \mathbf{0} \in \mathbf{R}^{4\times(3j-3)}, H_{i,j,t}^{LM} \in \mathbf{R}^{4\times3}, \mathbf{0} \in \mathbf{R}^{4\times(3M-3j)}\right] \end{aligned} \tag{33}$$

The EKF update step is then calculated as follows

$$K_{t+1|t} = \Sigma_{t+1|t} H_{t+1|t}^T \left(H_{t+1|t}\Sigma_{t+1|t}H_{t+1|t}^T + I \otimes V\right)^{-1} \tag{34}$$

$$\mu_{t+1|t+1}^{robot} = K_{t+1|t}\left(z_{t+1} - \hat{z}_{t+1}\right) + \mu_{t+1|t}^{robot} \tag{35}$$

$$\mu_{t+1|t+1}^{LM} = \text{homogeneous}\left(K_{t+1|t}\left(z_{t+1} - \hat{z}_{t+1}\right) + \mu_{t+1|t}^{robot}\right) \tag{36}$$

$$\Sigma_{t+1|t+1} = \left(I - K_{t+1|t}H_{t+1|t}\right)\Sigma_{t+1|t} \tag{37}$$

Since the mean vector I use is homogeneous while for K is inhomogeneous. So, to implement update, function to transfer from inhomogeneous to homogeneous is necessary.

IV. Results

The followings are results generated from task 1, 2 and 3.

From Fig. 1 to Fig. 3, We can notice that the prediction without update can give pretty good trajectory result. Which means that the IMU data is highly reliable.

Fig. 4 to Fig. 6 shows the difference before and after mapping update. The blue dots are the initialization positions and the yellow dots are the position after updating. The observation noise covariance is set to I*10. Most of the landmarks only have slightly change after update. I also test larger V=I*1000, but the results are quite similar and not likely to notice the difference of V=I*10.

Fig. 7 to Fig. 9 shows the EKF visual-inertial SLAM result using parameters as Table 1

Table 1 Parameters set 1

Parameters	value
V	I * 1e5
Σ_w	I * 1e-6
initial Σ_{robot}	I * 1e-2
Initial Σ_{robot_LM} ; Σ_{LM_robot}	0
Initial Σ_{LM}	I * 1e-2

This give quite good result for all test sets. The input noise is set to be small because the IMU can provide accurate estimation of the velocity and angular velocity. I set observation noise being large because observation is not accurate enough.

If we use parameters set as Table 2, we can get Fig. 10 to Fig. 12, we can notice for test 27. We can still get quite good result. While for test 27 and 42, the trajectory starts to twist. This may because some features are labeled on moving objects and we are still regarding it being static. Since we tune the V to be smaller, we also believe the observation is accurate. The mismatch makes the model to give a wrong prediction of the trajectory. Test 27 still give good result may because the observed landmarks are statics while the robot is moving.

Table 2 Parameters set 2

Parameters	value
V	I * 1e3
Σ_w	I * 1e-6
initial Σ_{robot}	I * 1e-2
Initial Σ_{robot_LM} ; Σ_{LM_robot}	0
Initial Σ_{LM}	I * 1e-2

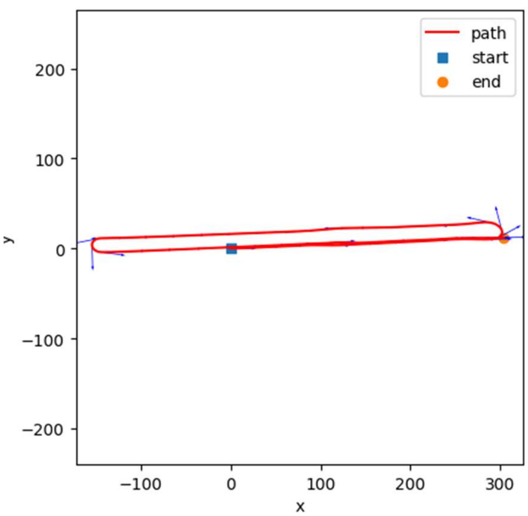

Fig. 1 EKF prediction for test 20

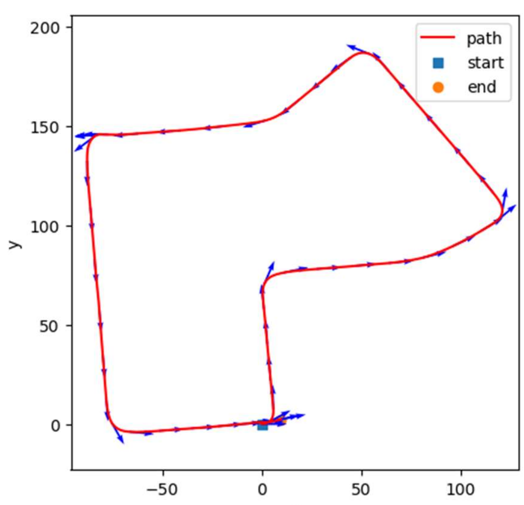

Fig. 2 EKF prediction for test 27

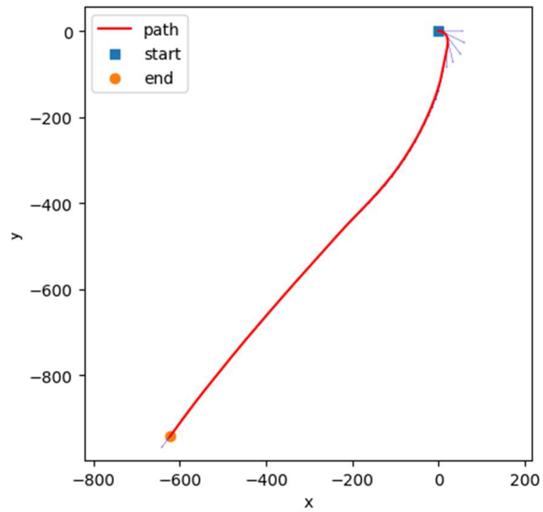

Fig. 3 EKF prediction for test 42

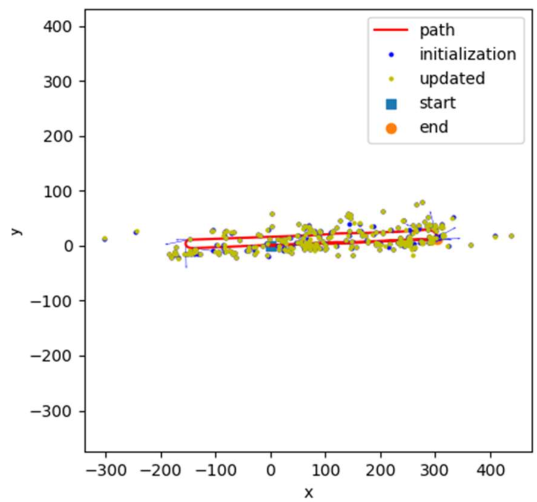

Fig. 4 EKF mapping for test 20

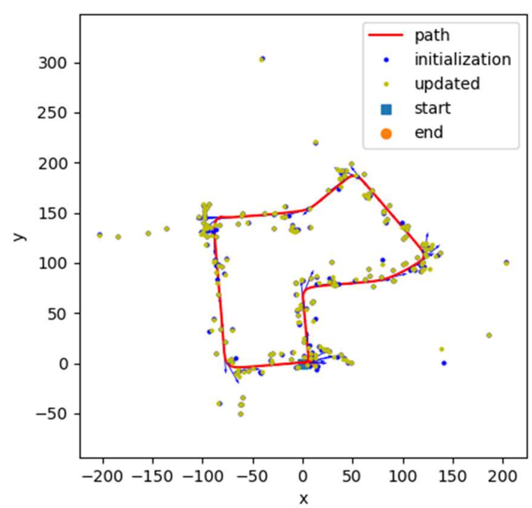

Fig. 5 EKF mapping for test 27

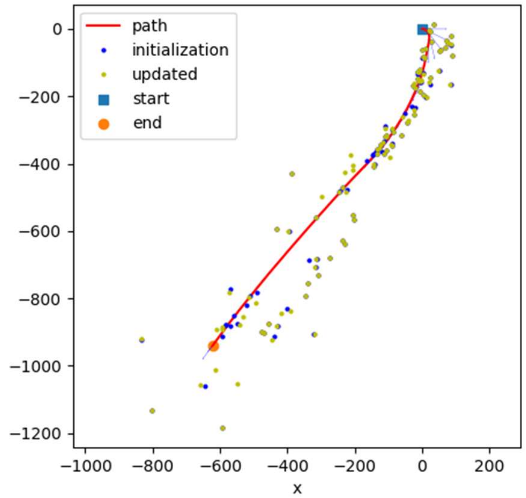

Fig. 6 EKF mapping for test 42

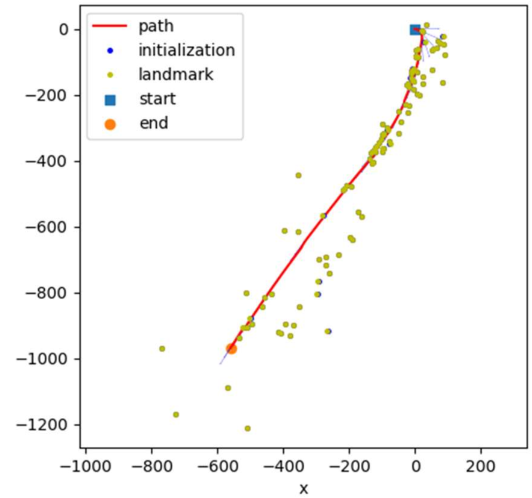

Fig. 9 Visual inertial SLAM with parameter set 1

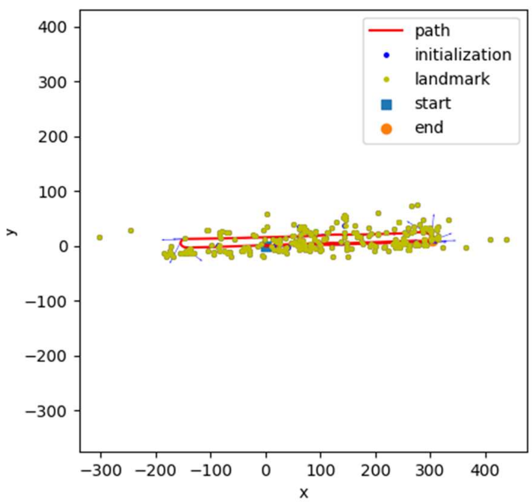

Fig. 7 Visual inertial SLAM with parameter set 1

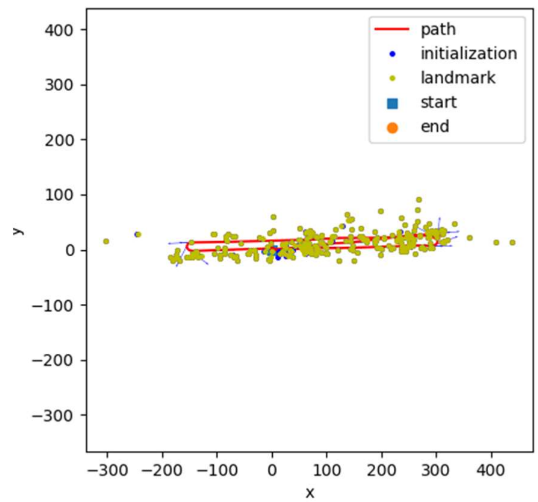

Fig. 10 Visual inertial SLAM with parameter set 2

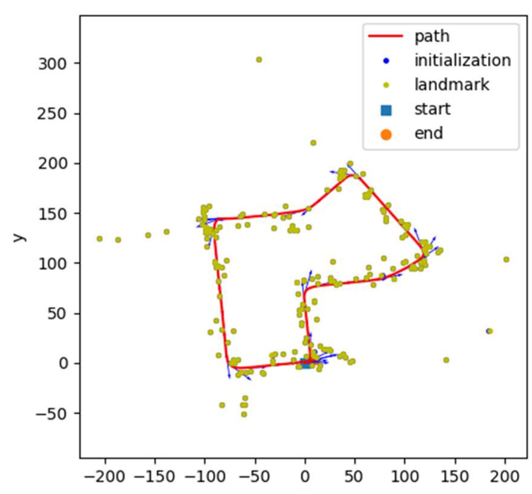

Fig. 8 Visual inertial SLAM with parameter set 1

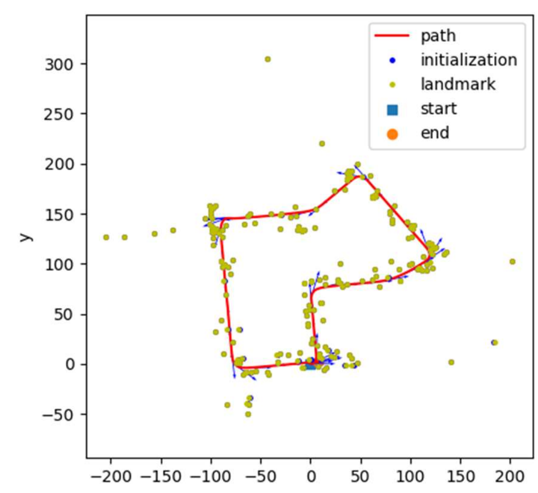

Fig. 11 Visual inertial SLAM with parameter set 2

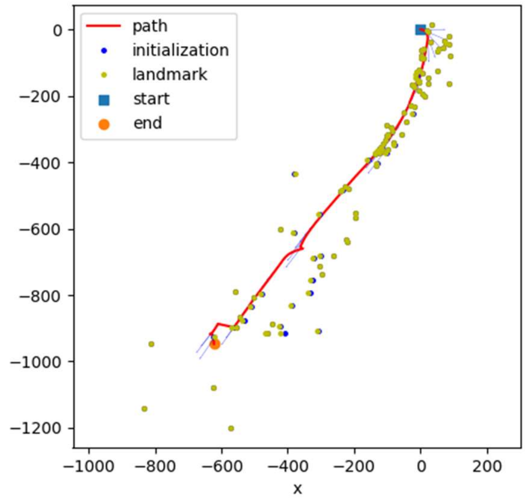

Fig. 12 Visual inertial SLAM with parameter set 2