Visual Inertial SLAM

1st Hao Xiang
department of Electrical and Computer Engineering
UC San Diego
haxiang@ucsd.edu

***Abstract*—This is the report for Hao's Project 3 Visual Inertial SLAM (VI-SLAM). In this report, Hao reviewed the key component of VI-SLAM and the techniques Hao has tried to improve the performance. In this project Visual-inertial SLAM utilize IMU data and camera data to joint estimate the inverse pose of robot and the 3D positions of landmarks.**

***Index Terms*—SLAM, EKF, VIsual-Inertial**

I. Introduction

SLAM aims to build a map from an unknown environment while keeping the trajectory of the agents. Leonard, John etc. [2] first introducted the concept of SLAM. Since then, SLAM has gradually become a key problem in the mobile robotics and has developed several approaches including Graph-SLAM [4], EKF-SLAM [1], Fast-SLAM [3] etc. The defficulty of the SLAM problem comes from the chicken-or-egg situation, i.e. the environment and the location both are unknown while estimating one needs the information of the other unknown one.

SLAM is very useful in many domain like outdoor/indoor robot navigation, building local map/global map as well as terrian mapping in the space. And with the development of deep learning and semantic segmentation, there is also new research on semantic slam.

EKF-based SLAM algorithm uses moment matching idea by leveraging linear approxiation techinques to make the calculation feasible. Due to the Gaussian assumption, EKF can estimate the positions in continuous 3D space while histogram filter based SLAM can only approxiamte the position in finite space. Another approach similar to EKF is Unscented Kalman Filter, it utilize the numerical approximation method to estimate the joint probability.

In this project, I implemented EKF-SLAM for the task of Visual-Inertial SLAM by utilizing the stereo camera and IMU data. To capture the correlations between landmarks and inverse poses, I utilized joint update. Also I tried several techniques like feature selection, z-axis fixation etc.

II. Problem Formulation

Visual Inertial SLAM: Given the IMU data $\mathbf{u}_{1:T}$, sampling time interval $\delta\tau_t$ and visual feature observation $\mathbf{z}_{1:T}$ which is detected by stereo cameras, estimate the inverse IMU pose $U_{1:T}$ and landmark position $\mathbf{m}$, s.t. $U_t, \mathbf{m} = \arg\max_{U_t,\mathbf{m}} p(U_{1:t}, \mathbf{m}|\mathbf{u}_{1:t}, \mathbf{z}_{1:t})$.

Here the substript t indicate the data for current time. $\mathbf{u}_t = \begin{bmatrix}\mathbf{v}_t \\ \omega_t\end{bmatrix}$ is the observed IMU data including linear velocity $\mathbf{v}_t \in \mathcal{R}^3$ and rotational velocity $\mathbf{w}_t \in \mathcal{R}^3$. $\delta\tau_t$ is the difference between time stamps t and time stamps $t-1$. Landmark positions $\mathbf{m}$ is the 3D position of all the landmarks.

In this project, we assume the number of total landmarks are knwon in advance, thus $\mathbf{m} \in \mathcal{R}^{3\times M}$, where M is the total number of features or landmarks we care about. $U_t = T_{IMU,t}^{-1} \in SE(3)$ is the inverse IMU pose for the time stamp t. Also in this project, we assume the data association π_t is also known. Visual feature observations are the pixel coordinates of the feature points in the left and right camera frame, thus $\mathbf{z}_t \in \mathcal{R}^{4\times N_t}$ where N_t is the number of features observed at time frame t. Besides, we also assume the map is static thus no motion model for thie project's mapping.

This problem can be subdevided to Localization and Mapping problem.

Localization: Assume the world frame landmark coordinates $\mathbf{m}$ are known, given the IMU data $\mathbf{u}_{1:t}$ and the feature observations $\mathbf{z}_{1:t}$, we want to estimate the inverse IMU pose U_t s.t. $U_t = \arg\max_{U_t} p(U_t|\mathbf{u}_{1:t}, \mathbf{z}_{1:t}, \mathbf{m})$.

Mapping: Assume the inverse pose $U_{1:t}$ is known, given the visual feature observation $\mathbf{z}_{1:t}$, estimate $\mathbf{m}$ s.t., $\mathbf{m} = \arg\max_{\mathbf{m}} p(\mathbf{m}|\mathbf{z}_{1:t}, U_{1:t})$

III. Technical Approach

A. *Localization EKF Prediction Step*

For the prediction step, assume the prior U_t satisfies

$$U_t|\mathbf{z}_{1:t}, \mathbf{u}_{0:t-1} \sim \mathcal{N}\left(\boldsymbol{\mu}_{t|t}, \Sigma_{t|t}\right) \tag{1}$$

And then we have the motion model as equation.2 ,where $\delta\tau_t$ is the time discretization.

$$U_{t+1} = \exp\left(-\delta\tau_t\left((\boldsymbol{u}_t + \mathbf{w}_t)\right)^{\wedge}\right) U_t \tag{2}$$

Thus for the mean $\boldsymbol{\mu}_{t=1|t} \in SE(3)$ and covariance $\boldsymbol{\Sigma}_{t+1|t} \in \mathcal{R}^{6\times 6}$ their update rules are shown as equation.3

$$\begin{aligned}\boldsymbol{\mu}_{t+1|t} &= \exp\left(-\delta\tau_t \hat{\mathbf{u}}_t\right) \boldsymbol{\mu}_{t|t} \\ \Sigma_{t+1|t} &= \exp\left(-\delta\tau_t \mathbf{u}_t^{\curlywedge}\right) \Sigma_{t|t} \exp\left(-\delta\tau_t \mathbf{u}_t^{\curlywedge}\right)^{\top} + W\end{aligned} \tag{3}$$

where W is the covariance of noise for motion model. After experimenting with several values for the noise W ranging from $10^{-6} - 10^{6}$, I choose the noise as equation.4

$$W = \begin{bmatrix} 10^{-2} & 0 & 0 & 0 & 0 & 0 \\ 0 & 10^{-2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 10^{-2} & 0 & 0 & 0 \\ 0 & 0 & 0 & 10^{-1} & 0 & 0 \\ 0 & 0 & 0 & 0 & 10^{-1} & 0 \\ 0 & 0 & 0 & 0 & 0 & 10^{-1} \end{bmatrix} \tag{4}$$

Since our IMU is better at estimating the linear velocity, thus I gave the corresponding correlation slighly smaller value while giving the one for rotational velocity higher value, meaning higher unconfidence. Also the hat ^ and curly hat ⅄ operations are defined as equation.5

$$\hat{\mathbf{u}}_t := \begin{bmatrix} \hat{\omega}_t & \mathbf{v}_t \\ \mathbf{0}^\top & 0 \end{bmatrix} \in \mathbb{R}^{4\times 4} \quad \overset{\curlywedge}{\mathbf{u}_t} := \begin{bmatrix} \hat{\omega}_t & \hat{\mathbf{v}}_t \\ 0 & \hat{\omega}_t \end{bmatrix} \in \mathbb{R}^{6\times 6} \tag{5}$$

B. Mapping EKF update Step

I first initialize the position of the new landmarks which means it is their first time being observed. And then update the EKF mean and covariance for the all the previously observed landmarks. To be more specifit, for initiliazation, I transferm the points from pixel coordinate to optical frame and then transform to IMU frame via the transformation ${}_oT_I^{-1}$. After that I will transform the points from IMU frame to world frame via the pose of the robot e.i. U_t^{-1}.

$$P_o = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} \frac{Z(u_L - c_u)}{f_{su}} \\ \frac{Z(v_L - c_v)}{f_{sv}} \\ \frac{f_{su}}{u_L - u_R} \end{bmatrix} \tag{6}$$

$$P_w = U_t^{-1} {}_oT_I{}^{-1} P_o \tag{7}$$

For the EKF update step, assume the prior $\mathbf{m}$ satisfies:

$$\mathbf{m} | \mathbf{z}_{1:t} \sim \mathcal{N}(\boldsymbol{\mu}_t, \Sigma_t) \tag{8}$$

Also we have the observation model as equation.9

$$\mathbf{z}_{t,i} = h(U_t, \mathbf{m}_j) + \mathbf{v}_{t,i} := M\pi\left({}_oT_I U_t \underline{\mathbf{m}}_j\right) + \mathbf{v}_{t,i} \tag{9}$$

Thus the update rule for the mean and covariance are shown in equation.10

$$\begin{aligned} K_t &= \Sigma_t H_t^\top \left(H_t \Sigma_t H_t^\top + I \otimes V\right)^{-1} \\ \mu_{t+1} &= \mu_t + K_t\left(z_t - M\pi\left(oT_l U_t \underline{\mu}_t\right)\right) \\ \Sigma_{t+1} &= (I - K_t H_t)\Sigma_t \end{aligned} \tag{10}$$

Here $\pi(\mathbf{q})$ is the projection function as $\pi(\mathbf{q}) = \frac{1}{q_3}\mathbf{q}$. And the $\underline{\mu}_t$ is the homogeneous coordinates of μ_t. Besides the $\frac{d\pi}{d\mathbf{q}}$ is

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

And $H_t \in \mathcal{R}^{4N_t \times 3M}$ is shown in equation.11

$$H_{t,i,j} = \begin{cases} M\frac{d\pi}{d\mathbf{q}}\left(oT_1 U_t \underline{\boldsymbol{\mu}}_{t,j}\right) oT_1 U_t P^\top & \text{If correspondence} \\ \mathbf{0} \in \mathbb{R}^{4\times 3} & \text{otherwise} \end{cases} \tag{11}$$

Here I assume V is a diagonal matrix with the value of 1, thus $I \otimes V$ will also be a diagonal matrix $\in \mathcal{R}^{4N_t \times 4N_t}$ with element of 1.

C. EKF joint update for Mapping and Localization

For part c, I calculate a joint H_t for both mapping and localization and then I calculate the corresponding $K_t\left(z_t - M\pi\left(oT_l U_t \underline{\mu}_t\right)\right)$. And use the first $3M$ row to update mapping's mean/variance and the last 6 rows to update localization's mean/variance. Also I use $K_t\left(z_t - M\pi\left(oT_l U_t \underline{\mu}_t\right)\right)$ to update the joint variance. I will detail the process in the following paragraphs.

Define the means for mapping and localization are $\mu_{map,t}$ and $\mu_{loc,t}$ respectively. Define the joint covariance as $\Sigma_t \in \mathcal{R}^{(3M+6)\times(3M+6)}$. Define the joint $H_t \in \mathcal{R}^{4N_t \times (3M+6)} = \begin{bmatrix} H_{map,t} & H_{loc,t} \end{bmatrix}$, where $H_{map,t} \in \mathcal{R}^{4N_t \times 3M}$ is calculated by using equation.11 like before. And for $H_{loc,t,i} \in \mathcal{R}^{4N_t \times 6}$, I use equation.12.

$$H_{loc,t,i} = M\frac{d\pi}{d\mathbf{q}}\left(oT_l \boldsymbol{\mu}_{loc,t} \underline{\mu}_{map,j}\right) oT_l \left(\boldsymbol{\mu}_{loc,t} \underline{\mu}_{loc,j}\right)^{\odot} \tag{12}$$

where $^\odot$ is defined as equation.13

$$\underline{\mathbf{s}}^{\odot} = \begin{bmatrix} I & -\hat{\mathbf{s}} \\ 0 & 0 \end{bmatrix} \in \mathcal{R}^{4\times 6} \tag{13}$$

Then I use the joint H_t to calculate K_t by using the equation.14

$$K_t = \Sigma_t H_t^\top \left(H_t \Sigma_t H_t^\top + I \otimes V\right)^{-1} \tag{14}$$

And I would calculate the K_t times innovation as equation.15

$$\delta Kz = K_t\left(z_t - M\pi\left(oT_l U_t \underline{\mu}_{map,t}\right)\right) \tag{15}$$

And then I will use the equation.16 to update the mean of mapping and use equation.17 to update the joint covariance. Also I use equation.18 to update the mean for the localization.

$$\mu_{map,t+1} = \mu_{map,t} + \delta Kz[0:3M,:] \tag{16}$$

$$\Sigma_{t+1} = (I - K_t H_t)\Sigma_t \tag{17}$$

$$\mu_{loc,t+1} = \exp\left((\delta Kz[3M:3M+6,:])^{\hat{}}\right)\mu_{loc,t} \tag{18}$$

Also for the prediction step of the localization, I will use the same process but this time only update part of the joint covariance. In other words, I will update $\Sigma_t[3M:3M+6, 3M:3M+6] \in \mathcal{R}^{6\times 6}$. This process can be shown in the following equation.

$$\begin{aligned} &\Sigma_{loc,t+1}[3M:3M+6, 3M:3M+6] = \\ &\exp\left(-\delta\tau_t \mathbf{u}_t^{\curlywedge}\right)\Sigma_{loc,t}[3M:3M+6, 3M:3M+6]\exp\left(-\delta\tau_t \mathbf{u}_t^{\curlywedge}\right)^\top \\ &+ W \end{aligned}$$

D. Techniques

1) Fix z-axis:

As stated in the problem discription, since sensor's movement along the axis is very small, thus we can assume the axis are the same over time. To model this, I tried two approaches

*) Set z-axis of $\mu_{map,t}$ to 0 after the update step of the mapping. But during experiment, I find that this approach would sometimes lead to singular matrix error. I think the

reason for this may due to the fact that we are forcing $z = 0$, thus the y-axis in the optical frame is also zero. According to the stereo camera model as equation.19, we know that $v_L = v_R = c_v$. Thus it would introduce the error when calculating the innovation term since the observations don't have fixed v_L or v_R.

$$\begin{bmatrix} u_L \\ v_L \\ u_R \\ v_R \end{bmatrix} = \begin{bmatrix} fs_u & 0 & c_u & 0 \\ 0 & fs_v & c_v & 0 \\ fs_u & 0 & c_u & -fs_u b \\ 0 & fs_v & c_v & 0 \end{bmatrix} \frac{1}{z} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{19}$$

To solve this problem, I force the v_L and v_R of obervations to be c_v. And the experiment results showed promising performance as figure.1

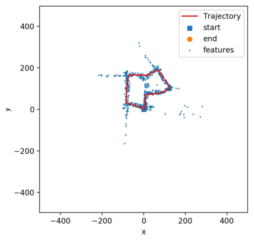


Fig. 1: Results for fix z-axis to 0 on dataset 27.

*) Set the z-axis to be the same as the first observed z-axis value after update step. To be more specific, after calculating the update step for mapping, I choose to set the z-axis value to be the one before the update step so that during the EKF process the estimated z-axis remains fixed. This approach also demonstrate promising results as shown in figure.2.

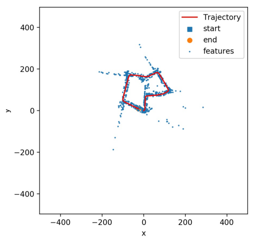


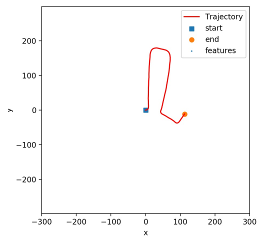
Fig. 2: Results for fix z-axis to the same observed value on dataset 27.

IV. Experiment Results

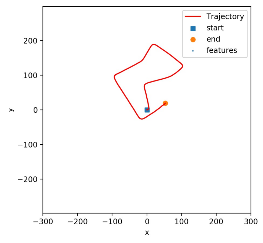
After some experiments, I choose to use 1000 features and the experiments results demonstrate that using all the feature points may not increase the performance while needing long time.

A. IMU-based Localization via EKF Prediction

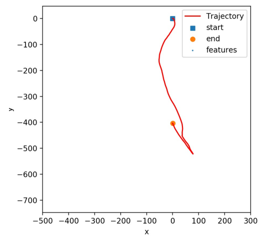
The results for part (a) are shown in figure.3. In part (a), I only do the EKF prediction step for the localization. As we can see as the robot observes the inertial input, the robot would update its mean and covariance. And the trajectory is inverse of the estimated mean. Since there is no update step, thus the robot could not correct its pose. This corresponds to the accumulated error in the image like (c) in figure.3



(a) dataset 22



(b) dataset 27

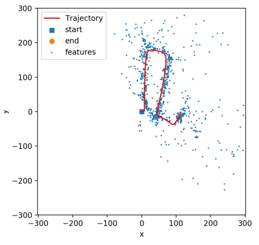


(c) dataset 34

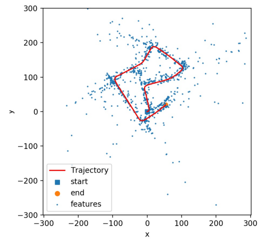
Fig. 3: Part (a) results for dataset 22, 27, 34. Localization EKF prediction step only.

B. Landmark Mapping via EKF Update

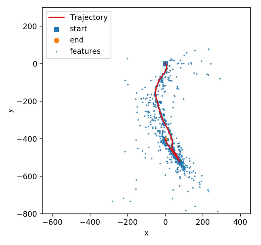
The results for part (b) are shown in figure.4.



(a) dataset 22



(b) dataset 27



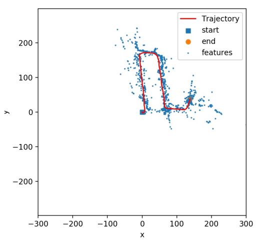
(c) dataset 34

Fig. 4: Part (b) results for dataset 22, 27, 34. The results are for landmark mapping via EKF update step.

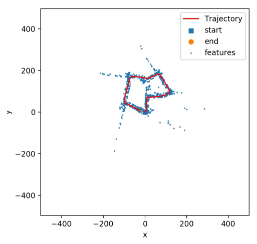
In this part, I implemented both localization prediction step and mapping update step but no joint update. As shown in the figure, the landmark prediction could adjust its position according to the information of the observation model. But since there is no update step for the localization, thus the trajectory is the same as part (a) with accumulated error.

C. *Visual Inertial SLAM*

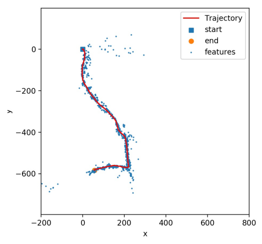
The results for part (c) are shown in the figure.5. In this part, I implemented the full visual inertial slam including localization prediction step and joint update step. Also I have attached the experiment results as a `gif` figure in the zip in the code part. For readers' convience, I also plot the process for SLAM is shown in the feature.7, figure..8, figure.9. (They are attached at the end of the report.)



(a) dataset 22



(b) dataset 27



(c) dataset 34

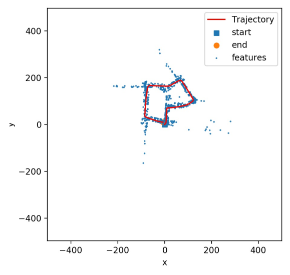
Fig. 5: Part (c) results for dataset 22, 27, 34. The results are for Visual Inertial SLAM including the joint EKF update step and localization prediction step.

As we can see the results are pretty good! For dataset 27, the start point and end point is very close to each other while in part (a) and part (b), the start point and end point are far away from each other. For dataset 22, the line started from forth turning point is parallel to the start line while in part (a) and part (b) the parallel relationship is not satisfied. For dataset 34, the right turn is almost 90 degree while in the part (a) and part(b) the degree is near 180, which is very different from the video. Also the landmark positions become more accurate! Compared with part (a) and (b), the results have far less outliers which are far away from the trajectory.

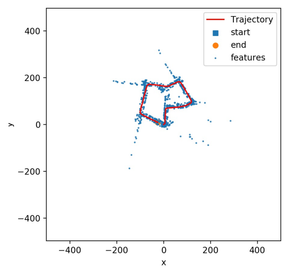
The improvement is largely due to the joint update step. Joint update step could capture the correlations between landmarks and inverse pose, making the estimation more accurate. And during prediction step, robot estimate its inverse pose via motion model while during update step, robot adjust its estimated inverse pose according toe the motion model. The landmark estimation also become more accurate during update step.

D. *Comparison between two different methods for fixing z-axis*

As I mentioned in the technical approach section, I tried two methods to fix z-axis. The first approach is by setting z-axis of mean to 0 and forcing the observed features' v_L and v_R to c_v. The second approach is forcing the z-axis of mean to the first observed value so that z-axis remains the same over time.



(a) Set z to 0



(b) Set z to first observed value

Fig. 6: Comparison between 2 different methods of fixing z-axis.

As shown in the figure.6, both approach could reach satisfying results. But there is a slightly rotation relationship between two methods. This is because for approach 2, even setting the z-axis to the previous value, we would still estimate a small innovation error along this axis, so the model would also try to fit the z-axis a little bit.

E. *Data Association and data processing*

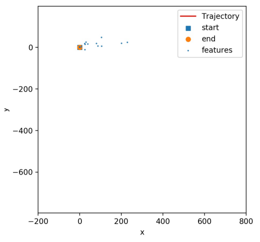
Since the data association is stored, thus I use a numpy boolen array to store the index of the landmark. If the element is true then it is observed before. Otherwise it is new. And I use a boolen mask to store the observed features at current time stamp. If the feature is not [-1,-1,-1], then it is observed at current time stamp. And its corresnpoding value would be True.

F. *Tensor Speed Up*

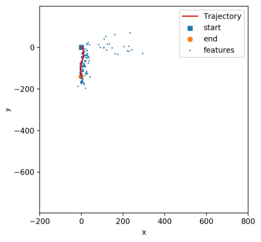
I use `np.tensordot` to perform the tensor multiplication instead of a for loop. So that the joint update could speed up! And when use 1000 features, it takes me less than 2 minites to run the whole dataset.

V. Conclusion

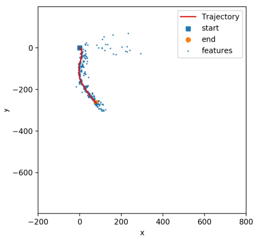
In this project, I implemented Visual Inertial SLAM and achive satisfying results for all three dataset. For better performance, I utilize joint update techniques. And I experiment with 2 fix-z axis techniques. And analysis the methodology behind them.



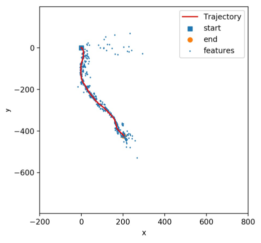
(a) frame 0



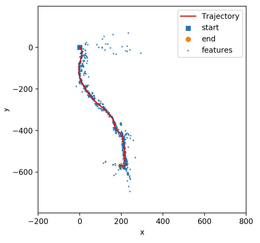
(b) frame 210



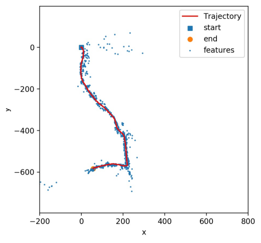
(c) frame 420



(d) frame 720



(e) frame 900



(f) frame 1223

Fig. 7: Process for building SLAM on dataset 34

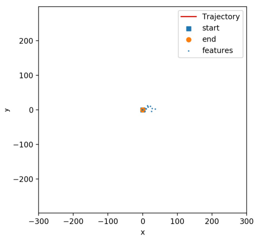
References

[1] Guoquan P Huang, Anastasios I Mourikis, and Stergios I Roumeliotis. Observability-based rules for designing consistent ekf slam estimators. *The International Journal of Robotics Research*, 29(5):502–528, 2010.

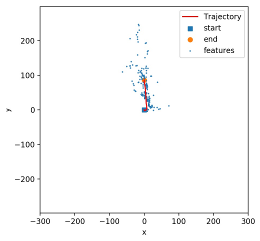
[2] John J Leonard and Hugh F Durrant-Whyte. Mobile robot localization by tracking geometric beacons. *IEEE Transactions on robotics and Automation*, 7(3):376–382, 1991.

[3] Michael Montemerlo, Sebastian Thrun, Daphne Koller, Ben Wegbreit, et al. Fastslam: A factored solution to the simultaneous localization and mapping problem. *Aaai/iaai*, 593598, 2002.

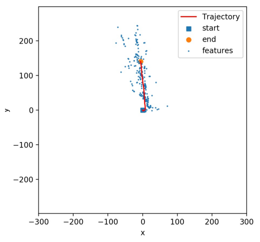
[4] Sebastian Thrun and Michael Montemerlo. The graph slam algorithm with applications to large-scale mapping of urban structures. *The International Journal of Robotics Research*, 25(5-6):403–429, 2006.



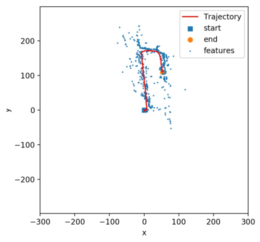
(a) frame 0



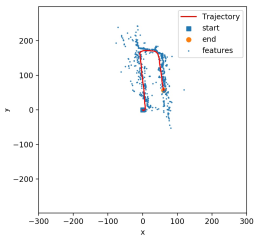
(b) frame 150



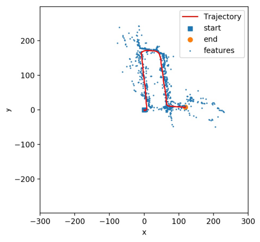
(c) frame 210



(d) frame 420

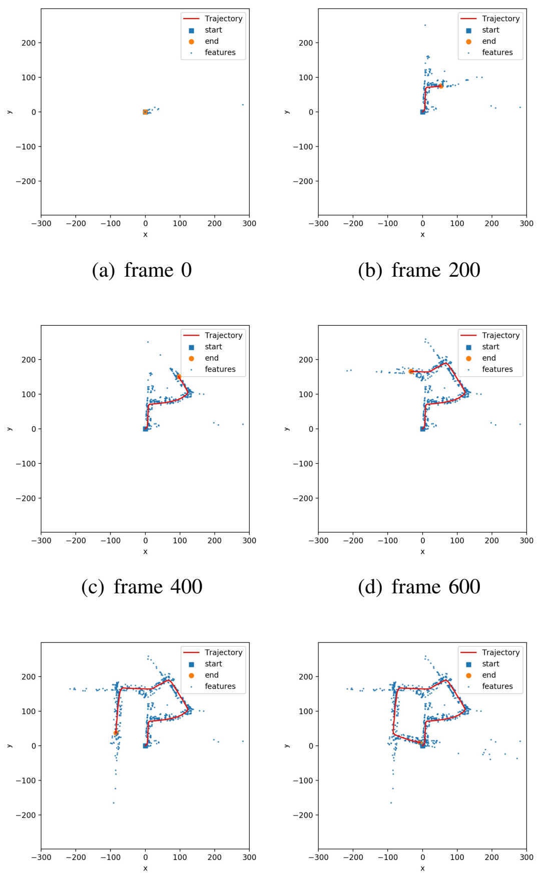


(e) frame 500



(f) frame 720

Fig. 8: Process for building SLAM on dataset 22.



(a) frame 0 (b) frame 200

(c) frame 400 (d) frame 600

(e) frame 900 (f) frame 1105

Fig. 9: Process for building SLAM on dataset 27.