# Visual Inertial SLAM

Sandeep Chintada
A59015527 dchintada@ucsd.edu

***Abstract*—The goal of this project is to demonstrate the application of Extended Kalman Filter for Simultaneous Localization and Mapping (SLAM) in a practical scenario involving car data. To accomplish this, data is collected from various sensors, including stereo cameras and an Inertial Measurement Unit (IMU). The report details the project's methodology and results, outlining the difficulties encountered during the process and the approaches taken to resolve them.**

***Index Terms*—Kalman Filter, Extended Kalman Filter, SLAM, Dead Reckoning, Linearization**

## I. Introduction

Simultaneous Localization and Mapping (SLAM) is a fundamental problem in the field of autonomous systems. It involves the creation of a map of an unknown environment by a robot, while simultaneously estimating its own position within that environment. SLAM provides the basic information required for a robot to perform any task, which includes determining its own location and understanding the surrounding environment. In other words, SLAM answers the questions ”Where am I?” and ”What is around me?”.

The ability to accurately locate oneself and understand the surrounding environment is essential for autonomous robots to operate effectively in real-world scenarios. For example, robots designed for tasks such as search and rescue, exploration, and delivery need to navigate through unknown environments while avoiding obstacles and making efficient decisions based on their surroundings. In these scenarios, SLAM enables the robot to create a map of its environment, localize itself within that map, and plan its movements accordingly.

Bayes Filter is a probabilistic inference technique used in autonomous systems for SLAM. However, its application to SLAM can be computationally intractable due to the large state space and the need to consider all possible hypotheses. To address this, several tractable methods have emerged that aim to approximate the posterior distribution over the robot's pose and environment map given its sensor measurements and control inputs.

Kalman Filter is one such Bayes Filter and it assumes that the state dynamics can be modeled as a linear system, and that the measurement process can be described by another linear system. The key idea behind the Kalman Filter is to represent the probability distribution of the system state using a mean vector and a covariance matrix. These parameters capture the most important statistical information about the state of the system and are sufficient for making optimal predictions and decisions.

The Kalman Filter assumes linear dynamics and Gaussian noise, which may not hold for many real-world systems that are nonlinear. To address this, the Extended Kalman Filter (EKF) is used, which applies the Kalman Filter to a linearized model of the nonlinear system at each time step.

## II. Problem Formulation

### A. Simultaneous Localization and Mapping

The SLAM problem can be described as a probabilistic Markov Chain. Given robot's pose $x_t$ and control input $u_t$ at time t, the pose in the following time step $t+1$ is a probabilistic function:

$$x_{t+1} = f(\mathbf{x_t}, \mathbf{u_t}, \mathbf{w_t}) \sim p_f(|\mathbf{x_t}, \mathbf{u_t}) \tag{1}$$

where $w_t$ is the motion noise $f$ is the motion model. To map the environment, the robot senses obstacles using different sensors like LiDAR, Camera or even Sonar at each time step. We denote this observation as $z_t$ and the state of the environment as $m$. The sensor measurements relate to the state($x_t$) by the observation model as follows:

$$\mathbf{z_t} = h(\mathbf{x_t}, \mathbf{m}, \mathbf{v_t}) \sim p_h(.|\mathbf{x_t}, \mathbf{m}) \tag{2}$$

With the motion and observation model as defined, SLAM is the problem to determine environment $\mathbf{m}$ and the robot pose $\mathbf{x_t}$ from the control inputs $u_{0:t-1}$ and observations $\mathbf{z_{0:t}}$. This can be mathematically formulated as follows:

$$p(\mathbf{m}, \mathbf{x_t}|\mathbf{z_{0:t}}, \mathbf{u_{0:t1}}) \tag{3}$$

We can use the Markov assumptions to induce a factorization of joint probability density function and that results in:

$$p(\mathbf{x_{0:t}}, \mathbf{z_{0:t}}, \mathbf{u_{0:t1}}) = p(\mathbf{x_0}) \prod_{t=0}^{t} p_h \mathbf{z_t}|\mathbf{x_t} \prod_{t=0}^{t-1} p_f \mathbf{x_{t+1}}|\mathbf{x_t}, \mathbf{u_t} \prod_{t=0}^{t-1} p(\mathbf{u_t}|\mathbf{x_t}) \tag{4}$$

We usually assume that the control policy is independent of the state and so we'd get rid of $p(\mathbf{u_t}|\mathbf{x_t})$ term.

### B. IMU Localization

Localization of the robot involves predicting the state of the robot($x_t$) at time $t$. A motion model($f$) is used to predict the future state of a body based on its current state and the actions applied to it. In this project, we assume that the system moves according to the $SE(3)$ kinematic equations as follow:

$$\dot{T(t)} = T(t)\zeta\hat{(t)} \tag{5}$$

The discrete time formulation of the same results in:

$$T_{k+1} = T_k \exp(\tau_k \hat{\zeta_k}) \tag{6}$$

### C. Landmark-based Mapping

The landmark-based mapping generates a map of an environment using uncertain sensor observations and known robot pose. The environment consists of static landmarks represented by their 3D location($m_i$) and their association($\Delta_t$) with the observations is assumed to be known. The robot observes multiple landmarks at each time step, denoted by $\mathbf{z_t}$, and the goal is to estimate the landmark locations based on the robot pose and observations. In this project, we use stereo camera data as input to our observation model. So, the equation relating the state $x_t$ and the observation is given as follows:

$$z_{t,i} = h(T_t, m_j) = K_s \pi({}_O T_I T_t^{-1} \underline{m_j}) + v_{t,i} \tag{7}$$

where $\pi$ is the projection matrix

### D. Sensor Data

Our proposed solution aims to solve the SLAM problem with the data from IMU and a stereo camera setup in the car. From the IMU, we receive synchronized velocity $v_t \in \mathbb{R}^3$ and the angular velocity $\omega_t \in \mathbb{R}^3$ data of the car. The stereo camera images are pre-computed to extract the visual features and find the correspondences ($\Delta$). These features would now be our observations $z_t \in \mathbb{R}^{4M}$ where each row contains x,y pixel co-ordinates in the left and right camera. When a particular feature is not observed, the data is set to $[-1, -1, -1, -1]$. The camera to IMU pose and the calibration matrix of the stereo setup is also known.

## III. Technical Approach

### A. Extended Kalman Filter

The Extended Kalman Filter (EKF) is a nonlinear version of the Kalman Filter, which linearizes about an estimate of the current mean and covariance. The nonlinear Kalman Filter is a Bayes filter with the following assumptions:

1) The prior pdf $p_{t|t}$ is Gaussian
2) Motion and Observation noise is Gaussian
3) Motion and Observation noise are independent of each other and of the state and/or observation

The motion and observation model can be forced to be linear using first-order Taylor series approximation. This approximation of the motion model and the observation models are as follows:

$$\begin{aligned} &f(\mathbf{x_t}, \mathbf{u_t}, \mathbf{w_t}) \\ &\quad\approx f(\mu_{\mathbf{t|t}}, \mathbf{u_t}, \mathbf{0}) + [\frac{df}{dx}(\mu_{\mathbf{t|t}}, \mathbf{u_t}, \mathbf{0})](\mathbf{x_t} - \mu_{\mathbf{t|t}}) \\ &\qquad + [\frac{df}{dw}(\mu_{\mathbf{t|t}}, \mathbf{u_t}, \mathbf{0})](\mathbf{w_t} - 0) \\ &\quad\approx f(\mu_{\mathbf{t|t}}, \mathbf{u_t}, 0) + \mathbf{F_t}(\mathbf{x_t} - \mu_{\mathbf{t|t}}) + \mathbf{Q_t}(\mathbf{w_t} - 0) \end{aligned} \tag{8}$$

where we assume that the noise has zero mean and $\mathbf{F_t}$ $\mathbf{Q_t}$ are the Jacobians evaluated around the mean.

Similarly, the approximation of the observation model would be;

$$\begin{aligned} &h(\mathbf{x_{t+1}}, \mathbf{v_{t+1}}) \approx \\ &\quad h(\mu_{\mathbf{t+1|t}}, 0) + \left[\frac{dh}{dx}(\mu_{\mathbf{t+1|t}}, 0)\right](\mathbf{x_{t+1}} - \mu_{\mathbf{t+1|t}}) \\ &\qquad + \left[\frac{dh}{dv}(\mu_{\mathbf{t+1|t}}, 0)\right](\mathbf{v_{t+1}} - 0) \\ &\approx h(\mu_{\mathbf{t+1|t}}, 0) + \mathbf{H_{t+1}}(\mu_{\mathbf{t+1|t}}, 0) + \mathbf{R_{t+1}}(\mathbf{v_{t+1}} - 0) \end{aligned} \tag{9}$$

where we again assume that the noise $v_t$ has zero mean and $\mathbf{H_t}$ and $\mathbf{R_{t+1}}$ are the Jacobians with respect to the state and the noise respectively. Based on the above linearized equations of the motion and observation model, we can define the prediction and update step of the Extended Kalman filter. Prior:

$$\mathbf{x_t}|\mathbf{z_{0:t}}, \mathbf{u_{0:t-1}} \sim \mathcal{N}(\mu_{\mathbf{t|t}}, \mathbf{\Sigma_{t|t}}) \tag{10}$$

Motion Model:

$$\begin{aligned} x_{t+1} &= f(x_t, u_t, w_t) \\ w_t &\sim \mathcal{N}(0, W) \\ \mathbf{F_t} &= \frac{df}{dx}(\mu_{\mathbf{t|t}}, \mathbf{u_t}, \mathbf{0}) \\ \mathbf{Q_t} &= \frac{df}{dw}(\mu_{\mathbf{t|t}}, \mathbf{u_t}, \mathbf{0}) \end{aligned} \tag{11}$$

Observation Model:

$$\begin{aligned} z_t &= h(x_t, v_t) \\ w_t &\sim \mathcal{N}(0, V) \\ \mathbf{H_t} &= \frac{dh}{dx}(\mu_{\mathbf{t|t-1}}, \mathbf{0}) \\ \mathbf{R_t} &= \frac{dh}{dv}(\mu_{\mathbf{t|t-1}}, \mathbf{0}) \end{aligned} \tag{12}$$

Prediction :

$$\begin{aligned} \mu_{t+1|t} &= f(\mu_{t|t}, u_t, 0) \\ \Sigma_{t+1|t} &= F_t \Sigma_t F_t^T + Q_t W Q_t^T \end{aligned} \tag{13}$$

Update:

$$\begin{aligned} \mu_{t+1|t+1} &= \mu_{t+1|t} + K_{t+1|t}(z_{t+1} - h(\mu_{t+1|t}, 0)) \\ \Sigma_{t+1|t+1} &= (I - K_{t+1}H_{t+1})\Sigma_{t+1|t} \\ K_{t+1|t} &= \Sigma_{t+1|t}H_{t+1}(H_{t+1}\Sigma_{t+1|t}H_{t+1}^T + I \otimes V)^{-1} \end{aligned} \tag{14}$$

### B. EKF based Visual Mapping

For the given landmark based visual mapping, we assume that the map is static and the data association is known as described in problem formulation. Since the map is static, we need not consider the prediction/motion model. Also, since we are only considering mapping, we assume that the pose of the robot is known. The observation model for this scenario is given by: Prior:

$$\begin{aligned} m|z_{0:T} &\sim \mathcal{N}(\mu_t, \Sigma_t) \\ \mu_t &\in \mathbb{R}^{3M} \\ \Sigma_t &\in \mathbb{R}^{3M*3M} \end{aligned} \tag{15}$$

where $\mu_t \in \mathbb{R}^{3M}$ is the expected location of all the landmarks and $\Sigma_t \in \mathbb{R}^{3M*3M}$ is the covariance of the said landmarks. Observation Model:

$$\begin{aligned} z_{t,i} = h(T_t, m_j) = K_s \pi({}_O T_I T_t^{-1} \underline{m_j}) + v_{t,i} \\ v_t \sim \mathcal{N}0, V \end{aligned} \tag{16}$$

where $\underline{m_j}$ is the homogeneous coordinate of the 3D point $m_j$ and $\pi$ is the projection. The projection matrix and its Jacobian are given by:

$$\begin{aligned} \pi(q) &= \frac{1}{q_3} q, \qquad q \in \mathbb{R}^4 \\ \frac{d\pi}{dq} &= \frac{1}{q_3} \begin{bmatrix} 1 & 0 & -\frac{q_1}{q_3} & 0 \\ 0 & 1 & -\frac{q_2}{q_3} & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & -\frac{q_4}{q_3} & 1 \end{bmatrix} \end{aligned} \tag{17}$$

We can stack all the observations $z_{t,i}$ into a single tall matrix rewrite the observation model as follows:

$$\begin{aligned} z_t = h(T_t, m) = K_s \pi({}_O T_I T_t^{-1} \underline{m}) + v_t \\ v_t \sim \mathcal{N}(I \otimes V) \end{aligned} \tag{18}$$

The Jacobian of $\tilde{z}_{t+1,i}$ with respect to $m_j$ evaluated at $\mu_{t,j}$ is given by:

$$H_{t+1,i,j} = K_s \frac{d\pi}{dq}({}_O T_I T_t^{-1} \mu_{t,j}) {}_O T_I T_t^{-1} P^T \tag{19}$$

when observation j corresponds to landmark i and P = $[I0] \in \mathbb{R}^{4\times4}$. Else, it would be set to zero. Now, we have everything we need for our EKF implementation as described by the set of equations in Equation 14.

In the code, we get the distance measurements in the camera frame using the following equations.

$$\begin{aligned} z &= f s_u * b/(u_L - v_L) \\ x &= (u_L - c_u) * z/f s_u \\ y &= (u_R - c_v) * z/f s_v \end{aligned} \tag{20}$$

where $f$ is the focal length and $s_u$ and $s_v$ are the pixel scaling factors and $u_L, v_L, u_R$ are the pixel coordinates of the corresponding feature in the left and right cameras.

At each time step, we initialize the previously unobserved landmarks using the above set of equations to get the camera frame coordinates then transform them to world frame using the known pose. The pixel noise $V$ is set to have a covariance of 3 pixels. To aid in compute resources, only 1 in every 20 features were considered.

### C. EKF-based Visual-Inertial Odometry

The localization problem aims to compute the state(pose $T \in SE(3)$ in this case) of the robot at each time stamp given the IMU measurements $[v_t^T, w_t^T]^T$, map of the environment $m$ and known data association between the landmarks and the measurements.

Since the pose $T$ is not a vector, we define Gaussian distribution over $T$ using a perturbation $\epsilon$ in the Lie algebra i.e.,

$$T = \mu \exp(\epsilon) \qquad \epsilon \sim \mathcal{N}(0, \Sigma) \tag{21}$$

The prediction/motion model is defined as follows: Prior:

$$\begin{aligned} T_t | z_{0:t}, u_{0,t-1} \sim \mathcal{N}(\mu_{t|t}, \Sigma_{t|t}) \\ \mu_{t|t} \in SE(3) \\ \Sigma_{t|t} \in \mathbb{R}^{6\times6} \end{aligned} \tag{22}$$

Motion Model:

$$\begin{aligned} \mu_{t+1|t} &= \mu_{t|t} \exp(\tau_t \overset{\wedge}{u}_t) \\ \delta\mu_{t+1|t} &= \exp(-\tau_t \overset{\curlywedge}{u}_t) \delta\mu_{t|t} + w_t \end{aligned} \tag{23}$$

where

$$\begin{aligned} u_t &= \begin{bmatrix} v_t \\ \omega_t \end{bmatrix} \in \mathbb{R}^6 \\ \overset{\wedge}{u}_t &= \begin{bmatrix} \overset{\wedge}{\omega}_t & v_t \\ 0 & 0 \end{bmatrix} \in \mathbb{R}^{4\times4}, \quad \overset{\wedge}{\omega}_t \in se(3) \\ \overset{\curlywedge}{u}_t &= \begin{bmatrix} \overset{\wedge}{\omega}_t & v_t \\ 0 & \overset{\wedge}{\omega}_t \end{bmatrix} \in \mathbb{R}^{6\times6}, \quad \overset{\wedge}{\omega}_t \in se(3) \end{aligned} \tag{24}$$

EKF Prediction step: With the motion model defined as above, the EKF prediction steps are listed as follows:

$$\begin{aligned} \mu_{t+1|t} &= \mu_{t|t} \exp(\tau_t \overset{\wedge}{u}_t) \\ \Sigma_{t+1|t} &= \mathbb{E}[\delta\mu_{t+1|t} \delta\mu_{t+1|t}^T] \\ &= \exp(-\tau_t \overset{\curlywedge}{u}_t) \Sigma_{t+1|t} \exp(-\tau_t \overset{\curlywedge}{u}_t)^T + W \end{aligned} \tag{25}$$

The observation model remains the same for localization as well i.e, equation 18. But the EKF update steps would vary since the Jacobian would now be computed with respect to the pose and note the mapping point. This Jacobian evaluated at $\mu_{t+1|t}$ is given by:

$$H_{t+1|t,i} = K_s \frac{d\pi}{dq}({}_O T_I T_t^{-1} \mu_{t+1|t,j})({}_O T_I)(\mu_{t+1|t} m_j)^{\circledcirc} \in \mathbb{R}^{4\times6} \tag{26}$$

where $\begin{bmatrix} s \\ 1 \end{bmatrix}^{\circledcirc} = \begin{bmatrix} I & -\overset{\wedge}{s} \\ 0 & 0 \end{bmatrix} \in \mathbb{R}^{4\times4}$
We stack the Jacobians at each observed feature to form a $4N_t \times 6$ matrix and apply the EKF update step as described in Equation 14.

### D. EKF based Visual Inertial SLAM

To estimate the pose of the robot and the state of the environment simultaneously, we merge the EKF update steps of visual inertial odometry problem and the visual mapping problem. First of all, the joint estimated state and covariance are defined as follows:

$$\begin{aligned} \mu &= \begin{bmatrix} \mu_m \\ \mu_p \end{bmatrix} \in \mathbb{R}^{6+3M} \\ \Sigma &= \begin{bmatrix} \Sigma_{mm} & \Sigma_{mp} \\ \Sigma_{mp}^T & \Sigma_{pp} \end{bmatrix} \in \mathbb{R}^{(6+3M)\times(6+3M)} \end{aligned} \tag{27}$$

where $\mu_m and \Sigma_{mm}$ are the estimated location and the covariance of visual mapping alone, and $\mu_p$and$\Sigma_{pp}$ are the estimated poses and its covariance. $\Sigma_{mp}$ is the cross covariance between the two.

Prediction Step: In the prediction step, we update the pose of the robot $\mu_{p,t+1|t}$ and its covariance $\Sigma_{pp,t+1|}$ using equation 25. Then, we update the cross covariance $\Sigma_{mp,t+1|t} =$

$\Sigma_{mp,t|t}F^T$ where F is $\exp\left(-\tau_t \overset{\curlywedge}{u_t}\right)$. It's transpose goes in the bottom left. Since we assume that the map is static, we don't have a prediction step for the map and the previous values move forward.

Update Step: In the update step, we initialize the hitherto unseen landmarks and compute the innovation $z_{t+1} - \tilde{z}_{t+1}$ where $\tilde{z}_{t+1}$ is the predicted observation based on the camera model. Then, we compute the Jacobian $H \in \mathbb{R}^{4N_t \times (3M+6)}$. In the first $4N_t \times 3M$ matrix, we put in the Jacobian of the visual mapping using equation 19 and in the last 6 columns, we place the Jacobian of the visual odometry using Equation 26 i.e.,

$$H_{t+1} = \begin{bmatrix} H_{m,t+1} & H_{p,t+1} \end{bmatrix} \tag{28}$$

The equations for the EKF update step are given as follows for this case.

$$\begin{aligned} K_{t+1|t} &= \Sigma_{t+1|t} H_{t+1} (H_{t+1} \Sigma_{t+1|t} H_{t+1}^T + I \otimes V)^{-1} \\ \mu_{t+1|t+1} &= \begin{bmatrix} \mu_{m,t+1|t} + K_{t+1|t}(z_{t+1} - \tilde{z}_{t+1}) \\ \mu_{p,t+1|t} \exp\left(K_{t+1|t}(z_{t+1} - \tilde{z}_{t+1})\right)^{\wedge} \end{bmatrix} \\ \Sigma_{t+1|t+1} &= (I - K_{t+1} H_{t+1}) \Sigma_{t+1|t} \end{aligned} \tag{29}$$

This combined SLAM is very sensitive to the noise chosen i.e., the motion noise, observation noise and the variance of the pose. In this experiment, covariance of the motion noise(W) is chosen to be $diag(1, 1, 1, 0.1, 0.1, 0.1)$ and that of the observation noise(V) is chosen to be 5 pixels. I also assumed that the initial location is very well known and the variance was chosen to be $diag(10^{-3}, 10^{-3}, 10^{-3}, 10^{-3}, 10^{-3}, 10^{-3})$

## IV. Results

In this section, we will delve into the findings of our project and examine the results in detail. To begin, we will first provide a comprehensive overview of the results and then analyze them in depth. Note that the camera to IMU pose wasn't flipped to match that of the real world. For visual mapping with the features subsampled to 20, it is taking me around 6 mins for dataset 10 and around 1 min for dataset 3. My results in the full visual SLAM went horribly wrong because of some issue in computing the Kalman Gain. The map however tracks the expected path. Reducing the Kalman Gain to a small value just for the pose adjustment shows that the the trajectory is in-fact turning. This tells me that the direction is in fact correct but there's some numerical computation error that has crept in. I've played around with multiple noise initializations but the results didn't change. I attached whatever results I have currently. Also, due to the lack of ground truth data, we can't really assess the performance of our code.

### A. Localization using IMU

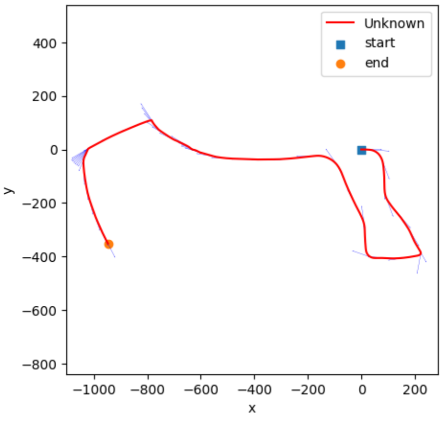

Fig. 1: Dead-Reckoning Trajectory - Dataset 10

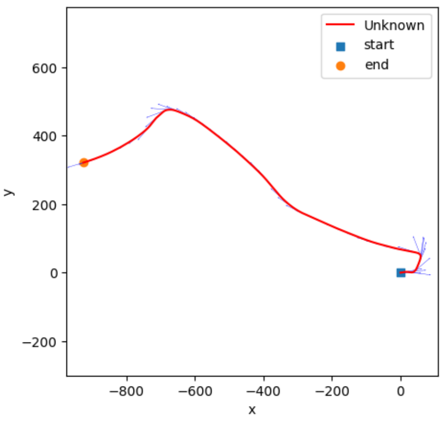

Fig. 2: Dead-Reckoning Trajectory - Dataset 3

### B. Visual Mapping with Dead Reckoning Trajectory

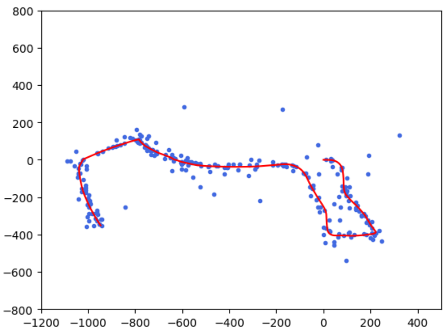

Fig. 3: World Frame coordinates - Dataset 10

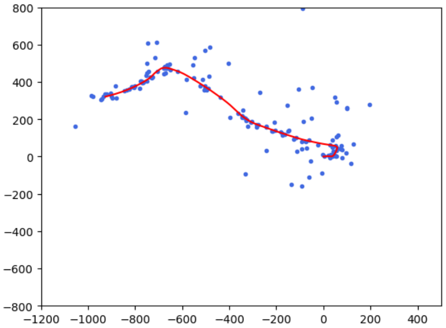

Fig. 4: World Frame coordinates - Dataset 3

### *C. Visual Inertial SLAM*

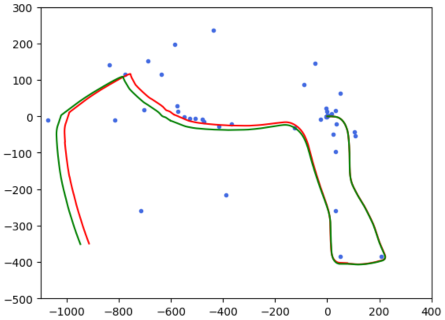

Fig. 5: Dataset 10

This figure was achieved after the Kalman Gain was reduced to 0.01 for the pose. The green line is the dead reckon and the red line is the SLAM trajectory.