Simultaneous Localisation and Mapping with Sparse Extended Information Filters

Evann Courdier Joseph Lam Dmitry Zhukov

January 20, 2017

Introduction

Simultaneous localization and mapping consists in reconstructing a map of environment from robot's observations, while localizing robot on the map at the same time. We assume that robot's observations and its estimations of its own movement are noisy, which leads to an error accumulation over time. To reduce the error we apply filtering methods.

In this work we consider three different filtering methods: a classic extended Kalman filter, an extended information filter and its sparsified version which allows to reduce computational complexity of the filtering at the cost of precision.

Theory

Consider a discrete time problem, where at each time step robot changes its position and observes nearby landmarks. This yields a series of noisy observations of robot's motion $u^t = u_1, ..., u_t$ and a series of noisy observations of landmarks $z^t = z_1, z_2, ...z_t$. Our goal is to estimate posterior distributions $p(x_t, y_t \mid u^t, z^t)$, where x_t corresponds to the robot's position and y_t corresponds to a landmark's position. This is a filtering problem. Figure 1 illustrates this formulation.

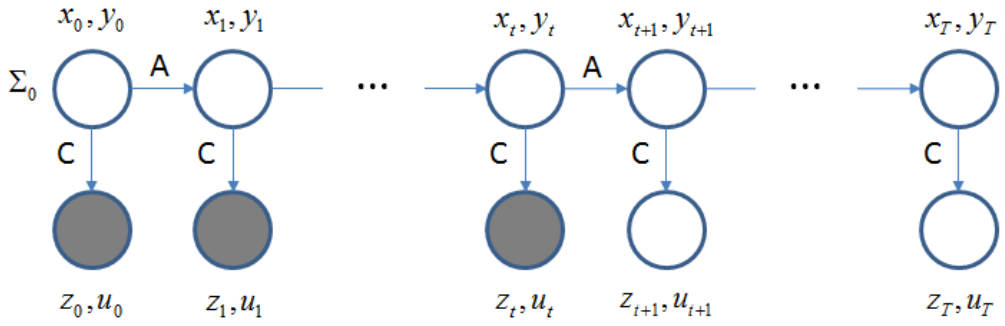

Figure 1: Graphical representation of the SLAM problem at time t. (Dark for observed variables)

At a fixed timestep, the problem reduces to one of factor analysis: the latent variable, which represents the positions, is a continuous random vector. In the framework of factor analysis, the latent variable is modeled as a gaussian. Thus, generalizing this framework to the dynamical case, the gaussian assumption is kept. Then, if the state transitions and the measurements are linear with some noise mixed in, we call such a framework Kalman filtering.

Extended Kalman Filter and Extended Information Filter for SLAM

For each timestep, the Kalman filter gives estimates of the unknown variables, then it updates these estimates with the next measurements, giving more weight to estimates with higher certainty. Since we consider gaussian latent variables, we identify their distributions thanks to their parameters: (μ, Σ). Now, this is the moment parameterization of a Gaussian, and it seems natural to consider its dual representation, the information filter, based on the canonical parameters of a Gaussian given by: $(\xi = \Sigma^{-1}\mu, \Omega = \Sigma^{-1})$. The moment parameterization corresponds to the Kalman filter, whereas the canonical parameterization corresponds to the Information filter.

The extended version gets rid of the linear assumption of state transition and measurement, which are strong assumptions. The key idea is then to linearize back again using Taylor expansion.

Implementation of EKF and EIF

Let us consider the implementation of EKF and EIF together since they are so closely related.

U_t, Z are the covariance matrices of the error terms for odometry and measurements respectively. $A_t = \nabla_\xi g(\mu_{t-1}, u_t)$ originates from the Taylor linearization on g. $C_t = (\frac{\partial h}{\partial x_t}, 0, ..., 0, \frac{\partial h}{\partial y_i}, 0, ..., 0)^T$ is a sparse Jacobian which is only non-zero for the elements describing the robot pose and the feature observed at time t. The Kalman gain K_t gives an optimal change to the estimate given the measurement.

For both algorithm, at each time step, the updates can be decomposed into a motion update and a measurement update corresponding to the dynamics of information as the agent moves and as it detects landmarks.

Algorithm 1 Algorithm EKF SLAM $(\mu_{t-1}, \Sigma_{t-1}, u_t, z_t, c_t)$:

Motion update

$\bar{\mu}_t = g(u_t, \mu_{t-1})$; $\bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + U_t$

Measurement update

$K_t = \bar{\Sigma}_t C_t^T (C_t \bar{\Sigma}_t C_t^T + Z)^{-1}$

$\mu_t = \bar{\mu}_t + K_t(z_t - h(\bar{\mu}_t))$; $\Sigma_t = (I - K_t C_t)\bar{\Sigma}_t$

Algorithm 2 Algorithm EIF SLAM $(b_{t-1}, H_{t-1}, \mu_{t-1}, u_t, z_t, c_t)$:

Motion update

$\bar{H}_t = (I + A_t)\Sigma_{t-1}(I + A_t)^T + S_x U_t S_x^T$

$\bar{b}_t = (\mu_{t-1} + g(u_t, \mu_{t-1}))\bar{H}_t$

Measurement update

$H_t = \bar{H}_t + C_t^T Z^{-1} C_t$

$b_t = \bar{b}_t + (z_t - h(\mu_{t-1}) + C_t^T \mu_{t-1})^T Z^{-1} C_t^T$

Conversion from canonical parameters to moment parameters

$\Sigma_t = H_t^{-1}$; $\mu_t = b_t^T \bar{H}_t$

If we were to compare both extended Kalman filtering (EKF) and extended information filtering (EIF), we would find complementary advantages and disadvantages. But one major drawback of the EIF requires the inversion of the information matrix, whereas the EKF does not involve the inversion of matrices of comparable size. Indeed, the EKF resolves information into a probability distribution every time it is acquired, whereas EIF simply memorizes information it receives and has to perform inference on the accumulated information for the map construction. Such characteristics lead to EKF being inherently adapted for online implementation and EIF for offline implementation. That is the reason why, EKF has been more popular in order to solve the SLAM problem, although it is still computationally expensive. But a sparse version of EIF adapts the algorithm to an online framework, compensating the drawbacks of the original EIF and making it more time efficient than EKF.

Sparsification of the Information matrix

When we look at the information matrix H, we can notice that many components are zero or very low. Thus comes the idea to impose the sparsity of this matrix in order to limit the number of non zero values in it. This property is very interesting since it allows us to update the information matrix much faster. Indeed, since H is sparse, we can use numerical libraries that are optimized for sparse matrices and we can approximate the state estimate μ without having to invert the information matrix.

Below you can see the Sparse Extended Information Filter algorithm for the SLAM Problem as described in [2], here with known data association:

Algorithm 3 Algorithm SEIF SLAM $(b_{t-1}, H_{t-1}, \mu_{t-1}, u_t, z_t, c_t)$:

$\overline{b}_t, \overline{H}_t, \overline{\mu}_t = \text{SEIF\_motion\_update}(b_{t-1}, H_{t-1}, \mu_{t-1}, u_t)$
$\mu_t = \text{SEIF\_update\_state\_estimate}(\overline{b}_t, \overline{H}_t, \overline{\mu}_t)$
$b_t, H_t = \text{SEIF\_measurement\_update}(\overline{b}_t, \overline{H}_t, \mu_t, z_t, c_t)$
$\tilde{b}_t, \tilde{H}_t = \text{SEIF\_sparsification}(b_t, H_t)$

The paper [3] proposes update equations when the information matrix is sparse.

In particular, it states the following lemma : The motion update requires constant time if the mean μ_t is available for the robot pose and all active features. The motion updates are the following:

$$\begin{aligned}
\Delta &= S_x^T A_t S_x \\
\Psi_t &= I + S_x \left((I + \Delta)^{-1} - I \right) S_x^T \\
H'_{t-1} &= \Psi_t^T H_{t-1} \Psi_t \\
\Delta H_t &= H'_{t-1} S_x [U_{t-1} + S_x^T H'_{t-1} S_x]^{-1} S_x^T H'_{t-1} \\
\overline{H}_t &= H'_{t-1} - \Delta H_t \\
b_t &= b_{t-1} - \mu_{t-1}^T \left(\Delta H_t - H_{t-1} + H'_{t-1} \right) + \widehat{\Delta}_t^T \overline{H}_t
\end{aligned} \tag{1}$$

The measurement update step is the same as the one previously written for EIF. They are constant time as long as we use an estimation of μ that does not imply matrix inversion.

The sparsification step is an approximate step that removes active features. By active features, we mean features that have a link with the robot in the information matrix. A new link between the robot and a feature is introduced only when observing that feature, while a new link between two features is created only by the motion update if the features are active. By controlling the number of active features, we therefore limit the number of links between features and ensure some degree of sparsity to the information matrix. We consider in the following that we want to keep a maximum of N active features.

To begin with, we need to split the set of known features in three disjoint sets:

$$Y = Y^+ + Y^0 + Y^- \tag{2}$$

where Y^+, Y^0, Y^- respectively denote the set of active features, the set of active features to become passive, and the set of passive features. To choose this sets, we compute the norm of every robot-feature links and keep only the N links with higher norms.

We then write the posterior:

$$\begin{aligned}
p(x_t, Y | z_t, u_t) &= p(x_t, Y^0, Y^+, Y | z_t, u_t) \\
&= p(x_t | Y^0, Y^+, Y, z_t, u_t)\, p(Y^0, Y^+, Y | z_t, u_t) \\
&= p(x_t | Y^0, Y^+, Y = 0, z_t, u_t)\, p(Y^0, Y^+, Y | z_t, u_t)
\end{aligned}$$

We then drop the dependence on Y_0 and get the following approximate distribution $\tilde{p}$ that minimize the KL divergence to the exact distribution

$$\begin{aligned}\tilde{p}(x_t, Y|z_t, u_t) &= p(x_t|Y^+, Y=0, z_t, u_t)\, p(Y^0, Y^+, Y|z_t, u_t) \\ &= \frac{p(x_t, Y^+|Y=0, z_t, u_t)}{p(Y^+|Y=0, z_t, u_t)}\, p(Y^0, Y^+, Y|z_t, u_t)\end{aligned}$$

From this, it follows that we can compute the sparsified information matrix and the corresponding canonical vector this way:

$$\begin{aligned}\tilde{H}_t &= H_t - H_t' S_{Y_0} \left(S_{Y_0}^T H_T' S_{Y_0}\right)^{-1} S_{Y_0}^T H_t' \\ &\quad + H_t' S_{x,Y_0} \left(S_{x,Y_0}^T H_T' S_{x,Y_0}\right)^{-1} S_{x,Y_0}^T H_t' \\ &\quad - H_t S_x \left(S_x^T H_T S_x\right)^{-1} S_x^T H_t \\ \tilde{b}_t &= b_t + \mu_t(\tilde{H}_t H_t)\end{aligned}$$

For the mean update step, we use a similar approach as the one proposed in the paper, that is, we write it as an optimization problem and then solve it by performing a coordinate ascent. We say that μ is the mode $\hat{\nu}$ of the Gaussian distribution p:

$$\begin{aligned}p(\nu_t) &= \text{const.} \cdot \exp\left(-\frac{1}{2}\nu_t^T H_t \nu_t + b_t^T \nu_t\right) \\ \hat{\nu}_t &= \text{argmax}_\nu\, p(\nu_t) \\ &= \text{argmax}_\nu\, \exp\left(-\frac{1}{2}\nu_t^T H_t \nu_t + b_t^T \nu_t\right) \\ &= \text{argmin}_\nu\, \frac{1}{2}\nu_t^T H_t \nu_t - b_t^T \nu_t\end{aligned}$$

Taking the derivative of this quantity w.r.t ν gives the following update for the coordinate ascent:

$$\forall i,\ \nu_{i,t}^{[k+1]} = H_{i,i,t}^{-1}\left[b_{i,t}^T - \sum_{j \neq i} H_{i,j,t}\, \nu_{j,t}^{[k]}\right] \tag{3}$$

Experiments

In order to test our implementation, we simulate the evolution of the robot in an environment with point landmarks, which positions are defined only by coordinates of their centers, as well as the position of the robot itself. Furthermore, we assume that the robot is capable of distinguishing the landmarks. We have performed experiments with this model in 1- and 2-dimensional cases. As 1-dimensional case is too simple and doesn't introduce any interesting effects that are not observable in 2 dimensions, only the 2-dimensional case is discussed below.

Experimental setup

Robot Modeling and Motion Model

In our 2D model, the robot is located with its coordinates (x, y) in the 2-D space and its orientation α. The robot has therefore a 3-dimensional state vector. At each time step t, the robot moves of a certain distance $d_t = v_t \cdot \Delta_t$ and then rotates of a certain angle $\beta = \dot{\alpha} \cdot \Delta_t$. The linear and angular velocity are bounded

such that the distance d is less than $v_{max} \cdot \Delta_t$ as well as the rotation $|\beta|$ is less than $\dot{\alpha}_{max} \cdot \Delta_t$. The robot's moves take any value between these bounds with equal probability. The real move of the robot is the chosen command to which we add an independent Gaussian noise δ_t with zero mean and known variance. Therefore, EIF and EKF motion updates are written the following way:

$$\begin{pmatrix} x_t \\ y_t \\ \alpha_t \end{pmatrix} = \begin{pmatrix} x_{t-1} \\ y_{t-1} \\ \alpha_{t-1} \end{pmatrix} + \begin{pmatrix} v_t \, \Delta_t \cdot \cos(\alpha_{t-1}) \\ v_t \, \Delta_t \cdot \sin(\alpha_{t-1}) \\ \beta \end{pmatrix} + \delta_t \tag{4}$$

The robot motion is therefore a non-linear function with added Gaussian noise. From these equation, we can compute the jacobian of the motion model that we use in EKF and EIF motion updates:

$$A_t = \begin{pmatrix} 0 & 0 & -v_t \, \Delta_t \cdot \sin(\alpha_{t-1}) \\ 0 & 0 & v_t \, \Delta_t \cdot \cos(\alpha_{t-1}) \\ 0 & 0 & 0 \end{pmatrix} \tag{5}$$

Environment Modeling and Measurement Model

Features in the simulated world are also located thanks to their 2D position coordinates (x, y). We denote θ_i the angle between the x axis and the line joining the robot a detected feature i. The robot we consider has a detection cone represented through a detection radius r and a detection angle γ. The measurement model is the following: When a environment feature enters the robot vision cone, the robot estimates the distance e and the angle $\theta_i - \alpha$ between himself and the feature. These values can then be used to perform the measurement step of the algorithms and thus correct state estimation. Thanks to our model, we have the following feature estimate:

$$h(\xi) = \begin{pmatrix} \sqrt{(x - x_i)^2 + (y - y_i)^2} \\ \arctan(\frac{y - y_i}{x - x_i}) - \alpha \end{pmatrix} \tag{6}$$

And from this we can derive the matrix C_t, which is the transpose of the Jacobian measurement matrix:

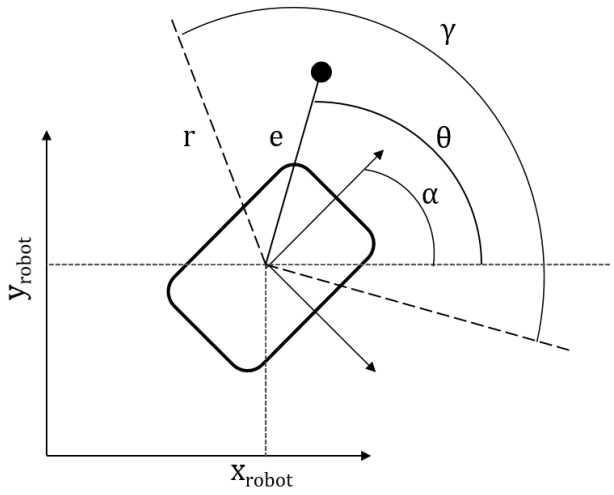

Figure 2: Schematic of the robot

$$C_t^T = \nabla_\xi h = \begin{pmatrix} \frac{x - x_i}{h_1(\xi)} & \frac{y - y_i}{h_1(\xi)} & 0 & 0 & \cdots & 0 & \frac{x_i - x}{h_1(\xi)} & \frac{y_i - y}{h_1(\xi)} & 0 & \cdots \\ \frac{y_i - y}{h_1(\xi)^2} & \frac{x - x_i}{h_1(\xi)^2} & -1 & 0 & \cdots & 0 & \frac{y - y_i}{h_1(\xi)^2} & \frac{x_i - x}{h_1(\xi)^2} & 0 & \cdots \end{pmatrix} \tag{7}$$

Data Association

The application to a real case cannot keep the assumption of perfect knowledge of the number of landmarks and their intrinsic identification. Instead, for each measurement of a landmark, we need to determine the index of the landmark or add a new index if it is the first time the landmark is detected. For this task, let us define a criterion:

With z_t the measured feature vector, and the expected measurement $\hat{z}_t^k$ for each landmark k, and $\Psi_k = H_t^k \; \bar{\Sigma} \; [H_t^k]^T + Q_t$ the innovation covariance.

$$d_M^k = \left(z_t - \hat{z}_t^k\right)^T \Psi_k^{-1} \left(z_t - \hat{z}_t^k\right)$$

As a quadratic distance weighted by Ψ_k, we call this criterion the Mahalanobis distance. Ψ_k determines the rate at which the distance function increases in the different dimensions and it can be identified in the Kalman gain term of 1.

Moreover, d_M^k follows a χ^2 distribution. Thus, once the distance has been computed for every landmark, a threshold d_α expressing a confidence level $1 - \alpha$ can be defined.

Then, we look for all k such that $d_M^k < d_\alpha$. If a unique k passes the test, then we can associate the measurement to that landmark unambiguously. However, if several landmarks correspond validate the criterion, two possible actions have been considered in [2] and [1]. Either we take the landmark corresponding to the smallest distance, or another possibility is to simply reject the observation.

In case there is not any landmark which satisfies the condition, then the observation is considered to be linked to a landmark that had not yet been observed. But instead of directly creating a new landmark, it is possible to increase the stability of our algorithm, thanks to a waiting list which stores potential new landmarks. Each potential landmark in the waiting list corresponds to an observation that has neither matched with a landmark from the list of confirmed landmarks nor the waiting list. These potential landmarks are attributed a counter which increases every time an observation matches with the potential landmark. Then, the potential landmark has to reach a certain threshold with the counter in a limited period of time in order to become a confirmed landmark. Once the counter has been reached or the time is up, the landmark is removed from the waiting list.

Random trajectories simulation

We simulate a random movement of a robot inside a landmark grid. Figure 3 shows 500 steps of random movement and the trajectories estimated with EIF and EKF methods, as well as with a naive approach that doesn't take the noise into account. Information filter and Kalman filter produce the same results.

Figure 3 also illustrates the comportment of filtering in the absence of observed landmarks (when the robot goes outside of the grid). Landmarks are denoted with points, and the circles around landmarks correspond to the radius of observation.

As expected, in the absence of observations filtering methods behavior is similar to a naive estimation.

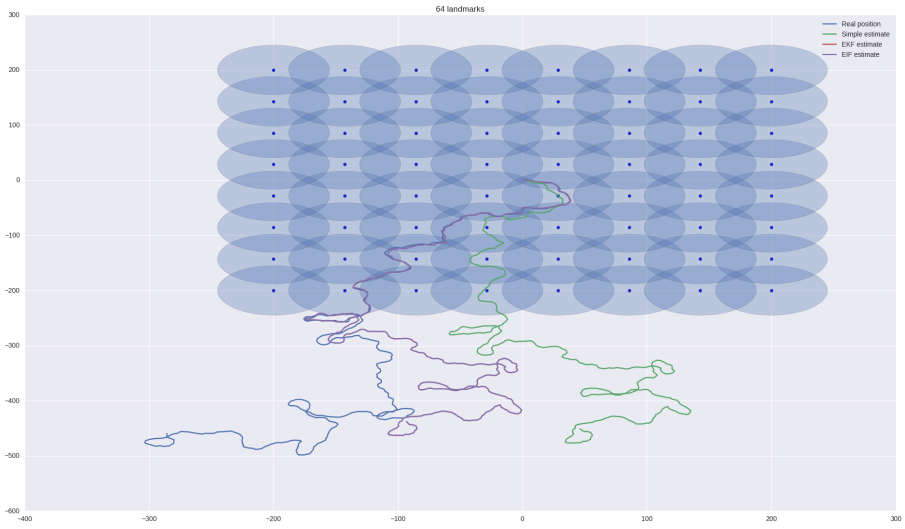

Figure 3: Real and estimated trajectories

Figure 4 shows the error in robot's position estimation. As we can see, a naive estimation accumulate error over time, while filtering methods are capable of keeping the error at a small level. An accumulation of error in the second half of simulation corresponds to leaving the grid.

Figure 5 further illustrates the dependence between the number of observed landmarks and the quality of filtering.

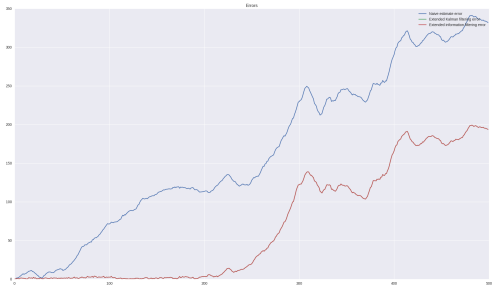

Figure 4: Error of position estimation for each iteration

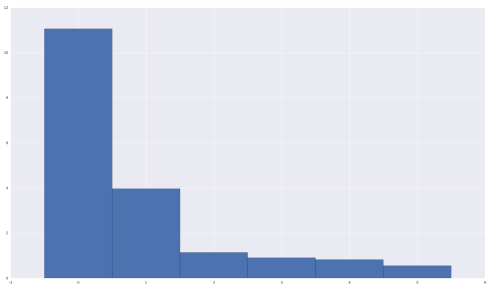

Figure 5: Average error as a function of number of observed landmarks

Simulating closed trajectories

A common way to illustrate the performance of SLAM is to simulate a closed trajectory and see whether a localization method ends up with a position close to the origin. For this task we use a more challenging setup by reducing the observation radius. The red line on figure 6 connects a beginning and the end of an estimated trajectory.

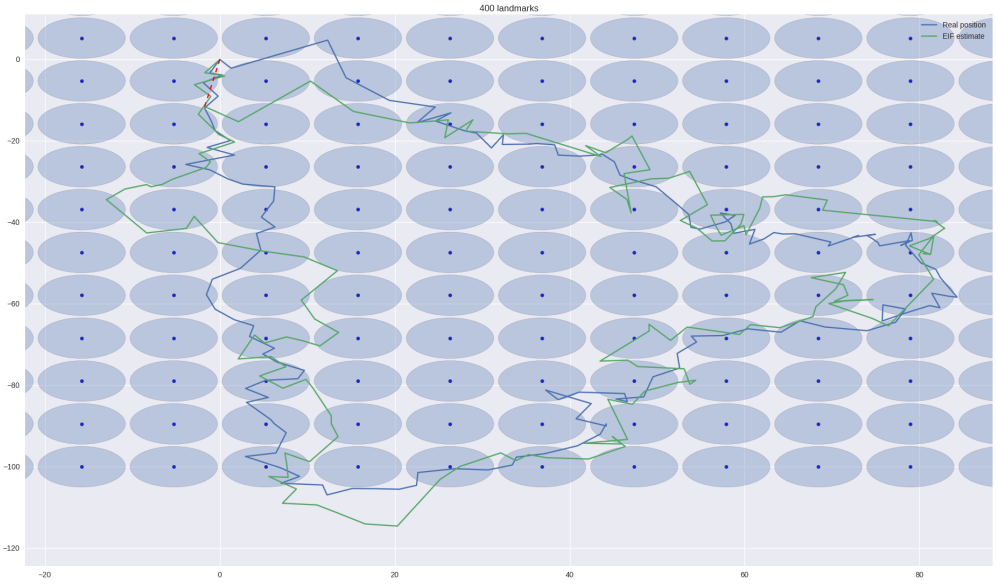

Figure 6: Closed loop simulation for EIF

Simulation for sparse extended information filter

We use the same setting to test SEIF. In the following simulation we restrain the number of active features by 6. The results are shown on figure 7. As we can see, while introducing an additional error due to the sparsification, the SEIF estimate is still capable of fitting well the true trajectory. The main advantage of SEIF is the computation time that proves to be several times smaller than in case of EIF.

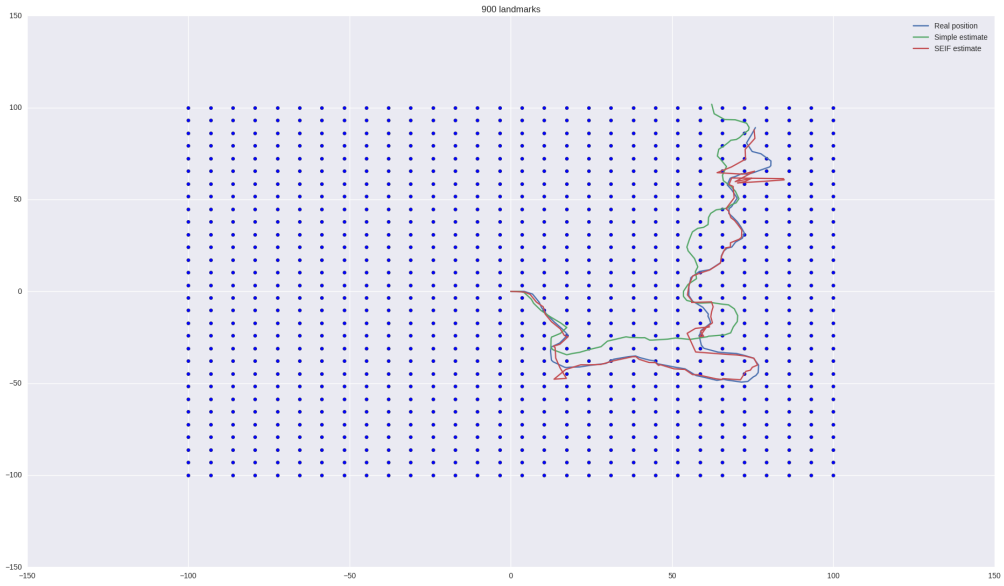

Figure 7: Simulation for SEIF

Conclusion

We have provided an implementation for the famous Extended Kalman Filter, but also for an alternative algorithm, the Extended Information Filter, and its online adaptation exploiting sparsity. All the models have been tested in an artificial environment against a naive estimation baseline. And we managed to get satisfying results with all methods in comparison with this baseline. Further experimentation could be led in order to compare with other methods for SLAM, like FastSLAM, which is one of the most popular methods. Working on this subject has been of great interest to us and we have really appreciated applying the concepts studied in class to the practical problem of Simultaneous Localization And Mapping.

References

[1] MWM Gamini Dissanayake, Paul Newman, Steve Clark, Hugh F Durrant-Whyte, and Michael Csorba. A solution to the simultaneous localization and map building (slam) problem. *IEEE Transactions on robotics and automation*, 17(3):229–241, 2001.

[2] Sebastian Thrun. Probabilistic robotics. *Communications of the ACM*, 45(3):52–57, 2002.

[3] Sebastian Thrun, Yufeng Liu, Daphne Koller, Andrew Y Ng, Zoubin Ghahramani, and Hugh Durrant-Whyte. Simultaneous localization and mapping with sparse extended information filters. *The International Journal of Robotics Research*, 23(7-8):693–716, 2004.