CS272: CS272 Project Report

Implementation of FastSLAM 1.0

Due on Friday, December 11, 2009

Student: Tue-Cuong Dong-Si

Contents

1 Overview

In robotics, simultaneous localization and mapping (SLAM) is a technique used by mobile robots to build a map in an unknown environment while localizing its current location within the map. As the robot moves, it receives observations or measurements from different nearby landmarks and update its belief on its current position as well as map of those landmarks.

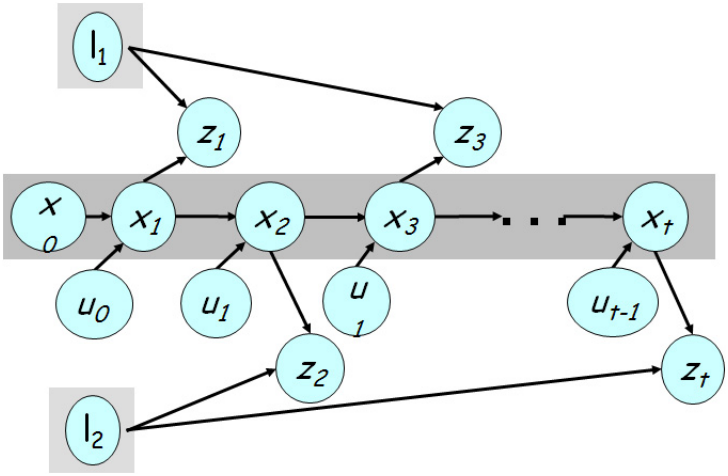


Figure 1: The Bayesian network for SLAM problem. Retrieved from [3].

Figure 1 shows a Dynamic Bayesian Network (DBN) that represents the SLAM problem. As shown in Figure 1, interaction between robot states, landmark positions and their measurements can be represented using graphical models. Thus, solving SLAM problem is essentially solving the inference and learning tasks in those graphical models.

In this project, FastSLAM 1.0 algorithm will be studied and implemented. After implementation, the algorithm will be tested in simulation. A simulated environment with known landmarks will be generated together with noisy measurements for the robot. The performance of the algorithm will be analyzed and compared with the standard EKF solution.

2 Background

In this section, only essential information of EKF-based SLAM and FastSLAM will be reviewed. For detailed formulation of FastSLAM and EKF from probabilistic graphical model perspective and their comparison, interested readers are referred to [3], [2] and [1].

2.1 Extended Kalman Filter

EKF's predecessor, the basic Kalman filter, is limited to linear systems. However, most more complex systems, including SLAM, are nonlinear. The state transition and measurement models in these systems have the forms as

$$\begin{cases} X(k+1) = f(X(k), U(k), W(k)) \\ Z(k+1) = h(X(k), U(k), N(k)) \end{cases} \tag{2.1}$$

where f and h are differentiable functions, and $W(k)$ and $N(k)$ are noises as in basic Kalman filter.

In EKF, the main idea is that the nonlinear system is linearized about the current estimate and covariance, by computing the Jacobians at the current estimate. After linearization, we have a linear systems and we can proceed to apply the Kalman filter formulas to estimate the next state estimate and covariance.

Algorithm 1 Extended Kalman Filter equations

1: PROPAGATION

$$\hat{X}_{k+1|k} = f(\hat{X}_{k|k}, U(k), 0) \tag{2.2}$$

$$\Phi(k) = \nabla_{X_k} f(\hat{X}_{k|k}, U(k), 0) \tag{2.3}$$

$$G(k) = \nabla_{W_k} f(\hat{X}_{k|k}, U(k), 0) \tag{2.4}$$

$$P_{k+1|k} = \Phi(k) P_{k|k} \Phi^T(k) + G(k) Q(k) G^T(k) \tag{2.5}$$

2: UPDATE

$$\hat{Z}_{k+1|k} = h(\hat{X}_{k+1|k}) \tag{2.6}$$

$$r_{k+1|k} = Z(k+1) - \hat{Z}_{k+1|k} \tag{2.7}$$

$$H(k+1) = \nabla_{X_{k+1}} h(\hat{X}_{k+1|k}) \tag{2.8}$$

$$S_{k+1|k} = H(k+1) P_{k+1|k} H^T(k+1) + R(k+1) \tag{2.9}$$

$$K_{k+1|k} = P_{k+1|k} H^T(k+1) S^{-1}_{k+1|k} \tag{2.10}$$

$$\hat{X}_{k+1|k+1} = \hat{X}_{k+1|k} + K_{k+1|k} r_{k+1|k} \tag{2.11}$$

$$P_{k+1|k+1} = P_{k+1|k} - P_{k+1|k} H^T(k+1) S^{-1}_{k+1|k} H(k+1) P_{k+1|k} \tag{2.12}$$

As mentioned, EKF is dominantly used to solve the SLAM problem. Unlike the linear Kalman filter, EKF is not an optimal estimator, and it is liable to errors and divergence, due to *linearization*. However, it can still give reasonable performance and has been applied widely in SLAM, navigation

systems and GPS. It has been arguably considered the *de facto* standard in navigation systems and GPS [7].

2.2 FastSLAM

Although EKF-based approaches gives reasonable performance, they suffers from an $\mathcal{O}(K^2)$ complexity where K being the number of landmarks. This comes from the fact that EKF maintains a square covariance matrix that grows with the number of landmarks. This makes EKF-based approaches not scalable for large map with many landmarks.

Advances in Dynamic Bayesian Network (DBN) and probabilistic inference algorithms lead to many probabilistic methods proposed in robotic mapping and SLAM [4]. FastSLAM [3] is a probabilistic method that primarily attempts to improve the speed and the scalability for the SLAM problem. FastSLAM has an $\mathcal{O}(M \log K)$ complexity with M is the number of particles in its particle filter and could be constant. Thus, for a very large map, FastSLAM has a significant advantage over EKF-based approaches. This is achieved based on the observation that knowledge of the robot's path (all $\mathbf{x}_0$, $\mathbf{x}_1$, ..., $\mathbf{x}_t$ in the Figure 1) will make all the landmark measurements conditionally independent.

In FastSLAM, a finite number of filters are maintained, with one particle filter and K EKFs for each particle in the particle filter. Each of these EKFs are corresponding to a landmark and, thus, it is low-dimensional. New robot pose is generated by the particle filter, and because the pose is known, each EKF can be updated independently. The true robot pose will be determined as the pose with the largest importance weight.

3 2D Planar SLAM - System Formulation

In this section, we will formulate the standard example of 2D planar SLAM [6], to apply for EKF and FastSLAM. The formulas in this section would help to understand better the MATLAB codes.

In 2D SLAM, as shown in Figure 2, the robot vehicle is moving in a 2D planar surface, with its primary system state $\mathbf{p} = (x, y, \phi)$ and controls input as translational and angular velocities, V and ω, respectively.

When the robot moves around in the environment, it observes the environment through one

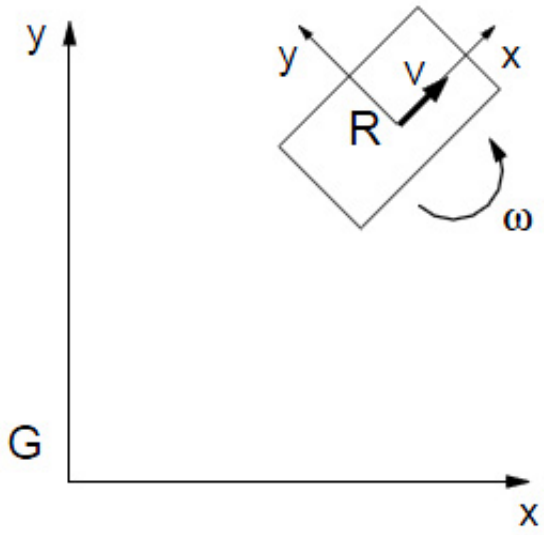


Figure 2: The robot setup in 2D planar SLAM.

or several pre-defined sensors. In this case study, we assume that the sensor returns the *relative position* from a robot to an object, as shown in Figure 3.

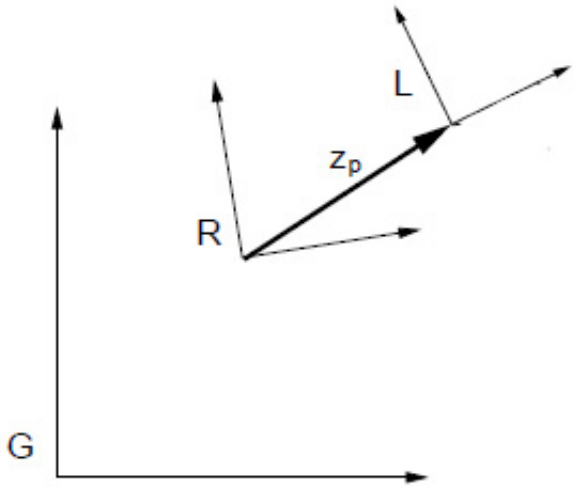


Figure 3: The global, robot and landmark reference frames.

3.1 EKF formulation

In EKF, the system state includes not only robot pose but also the landmarks' position. The process model and measurement model for EKF are derived as

$$\begin{bmatrix} x(k+1) \\ y(k+1) \\ \phi(k+1) \\ x_L(k+1) \\ y_L(k+1) \end{bmatrix} = \begin{bmatrix} x(k) + V(k)\cos\phi(k)\delta t \\ y(k) + V(k)\sin\phi(k)\delta t \\ \phi(k) + \omega(k)\delta t \\ x_L(k) \\ y_L(k) \end{bmatrix} = \begin{bmatrix} x(k) + (V_m(k) - w_V(k))\cos\phi(k)\delta t \\ y(k) + (V_m(k) - w_V(k))\sin\phi(k)\delta t \\ \phi(k) + (\omega_m(k) - w_\omega(k))\delta t \\ x_L(k) \\ y_L(k) \end{bmatrix} \tag{3.1}$$

$$Z(k+1) = \begin{bmatrix} z_x(k+1) \\ z_y(k+1) \end{bmatrix} = \begin{bmatrix} \cos\phi_R & \sin\phi_R \\ -\sin\phi_R & \cos\phi_R \end{bmatrix} (p_L - p_R) + \begin{bmatrix} n_x(k+1) \\ n_y(k+1) \end{bmatrix} \tag{3.2}$$

$$\Phi(k) = \frac{\partial f}{\partial X}\Big|_{\hat{X}(k),U(k),0} = \begin{bmatrix} 1 & 0 & -V_m\delta t\sin\hat{\phi}(k) & 0 & 0 \\ 0 & 1 & V_m\delta t\cos\hat{\phi}(k) & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \Phi_R(k) & 0 \\ 0 & I_{2\times 2} \end{bmatrix} \tag{3.3}$$

$$G(k) = \frac{\partial f}{\partial W}\Big|_{\hat{X}(k),U(k),0} = \begin{bmatrix} -\delta t\cos\hat{\phi}(k) & 0 \\ -\delta t\sin\hat{\phi}(k) & 0 \\ 0 & -\delta t \\ 0 & 0 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} G_R(k) \\ 0 \end{bmatrix} \tag{3.4}$$

$$H = \begin{bmatrix} -(C^T(\hat{\phi}) & -(C^T(\hat{\phi})J(\hat{p}_L - \hat{p}_R) & (C^T(\hat{\phi}) \end{bmatrix} \tag{3.5}$$

where

$${}^R_G C(\phi_R) = {}^G_R C^T(\phi_R) = C^T(\phi_R) = \begin{bmatrix} \cos\phi_R & \sin\phi_R \\ -\sin\phi_R & \cos\phi_R \end{bmatrix}$$

$$J = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$$

For simplicity, the equations only show the case of one landmark. But those equations can be readily augmented to include multiple landmarks.

3.2 FastSLAM formulation

3.2.1 New pose sampling

The control inputs are the measured translational and rotational velocities, $V_m(k)$ and $\omega_m(k)$. They are corrupted by zero-mean white Gaussian noise processes denoted by $w_V(k)$ and $w_\omega(k)$. Therefore, the new pose sampling is derived as:

$$X^{[m]}(k+1) = \begin{bmatrix} x^{[m]}(k+1) \\ y^{[m]}(k+1) \\ \phi^{[m]}(k+1) \end{bmatrix} = \begin{bmatrix} x^{[m]}(k) + V(k)\cos\phi(k)\delta t \\ y^{[m]}(k) + V(k)\sin\phi(k)\delta t \\ \phi^{[m]}(k) + \omega(k)\delta t \end{bmatrix} = \begin{bmatrix} x^{[m]}(k) + (V_m(k) - w_V^{[m]}(k))\cos\phi(k)\delta t \\ y^{[m]}(k) + (V_m(k) - w_V^{[m]}(k))\sin\phi(k)\delta t \\ \phi^{[m]}(k) + (\omega_m(k) - w_\omega^{[m]}(k))\delta t \end{bmatrix} \tag{3.6}$$

where $w_V^{[m]}$ and $w_\omega^{[m]}$ are sampled from $\mathcal{N}(0, \sigma_V^2)$ and $\mathcal{N}(0, \sigma_\omega^2)$, respectively.

3.2.2 Updating landmark estimates and calculating importance weights

In landmark update step, N EKFs attached to each particle are updated, where N is the number of landmarks discovered. The standard EKF update equations in Algorithm 1 are applied here with the following measurement model and Jacobian:

$$Z(k+1) = \begin{bmatrix} \cos\phi_R & \sin\phi_R \\ -\sin\phi_R & \cos\phi_R \end{bmatrix} (p_L - p_R) + \begin{bmatrix} n_x(k+1) \\ n_y(k+1) \end{bmatrix} \tag{3.7}$$

$$H = C^T(\phi_R) = \begin{bmatrix} \cos\phi_R & \sin\phi_R \\ -\sin\phi_R & \cos\phi_R \end{bmatrix} \tag{3.8}$$

The measurement Jacobian is much simpler because the system states for these EKFs are the position of landmarks, and the position of the robot is constant for a particular particle. After

updating the EKFs, the weight of the particle is also updated with the importance weight can be computed as:

$$w_t^{[m]} = \frac{1}{\sqrt{|2\pi S|}} \exp(-\frac{1}{2} r^T S^{-1} r) \tag{3.9}$$

where S and r are the measurement covariance and measurement innovation that are already computed when updating the EKFs.

3.2.3 Importance resampling

Once the particles are assigned weights and normalized, if the effective number of particles are below some threshold value, a new set of particles is drawn from this set with probabilities in proportion to the weights. The resampling process is implemented based on Stochastic universal resampling technique.

3.3 Unknown data association

For unknown data association, in both EKF and FastSLAM implementation, a Maximum Likelihood (ML) data association heuristics is used. It is done by defining two thresholds corresponding to two uncertainty boundaries for each landmark. If a measurement is outside outer boundaries for all discovered landmarks, it is determined as a new landmark. If a measurement is the only one inside the inner boundary of a particular landmark, it is assigned to that landmark. Otherwise, the measurement will be discarded. Although such data association scheme is not optimal [3], it is enough for our simulation.

3.4 Log(N) FastSLAM

One important feature of FastSLAM that is not implemented is the balanced binary tree data structure to store the landmark filters.

In the resampling step, each particle has to be copied and its K landmark filter associated with it has to be copied. However, it is observed that in each update and resampling step, only one landmark filter is updated while others remain intact. Thus, by using the binary tree to store the landmark filters, the updated landmark filter can be retrieved and copied quickly while the rest can be copied quickly by copying the links to subtrees.

This feature is not implemented in this project because the algorithms are implemented in MATLAB. Such method is much less effective in MATLAB, compared with other programming languages such as C. The implementation of binary tree in MATLAB is possible but the resulting code will be distracting and much less clear. In the resampling step, copying the particles is conveniently done with a single command.

4 Experimental results

4.1 Simulation - Known data

The EKF and FastSLAM algorithms are tested in simulated environments. A number of landmarks are pre-defined in this environment and the measurements are generated based on the robot's location. These measurements are corrupted by normal Gaussian noise.

First, we consider a simple case where landmarks are identifiable. In this case, the robot will receive the measurements together with landmark ID associated with those measurements. Performance of EKF and FastSLAM in this environment are shown in Figures 4-7. It can be observed that the performance of EKF and FastSLAM is similar. It fact, if more particles are used, performance of FastSLAM will approach to EKF performance. However, since this map has a small number of landmarks, the advantage of FastSLAM is not obvious. As mentioned earlier, the FastSLAM algorithm is more suited for large-scale map with numerous landmarks.

In this scenario, by changing different parameters, we can observe different performance properties of EKF and FastSLAM. One important property of FastSLAM is that its performance degrade when measurement noise is very low. This is because most particles will be thrown away in the resampling process. By changing control parameters for measurement noise, we can observe this. This problem will be addressed in the modified version of FastSLAM, called FastSLAM 2.0.

4.2 Simulation - Unknown data

In this scenario, the measurements come with no identification information. The algorithms has to associate the given measurement with the most likely landmark based on the current estimates and update accordingly. If the measurement is more likely to belong a new landmark, a new landmark should be added. This process is called data association, and it is significantly harder for both

algorithms. The performance of EKF and FastSLAM in this scenario is shown in Figures 8-11.

We can see that the covariances of landmarks in FastSLAM quickly converge but because of data association rule, this cause many fake landmarks are added into the map. On the other hand, the covariances of landmarks in EKF is significantly larger than FastSLAM, but its environment map is much cleaner than FastSLAM estimates.

4.3 Real-world data

Beyond simulated environments, it is possible to extend the performance test on real world data. The data sets provided by University of Sydney [5] are intended to use for real-time testing. However, the physical configurations of the laser sensor and the robot car is much different from the standard robot model [6]. That means the process propagation model and the measurement model for these data sets are completely different. Although the conversion to such models are mathematically not difficult, it requires most of the codes has to be revised accordingly. Due to time constraints, real-world data testing is not included in this report. However, the code and the processed data for Sydney carpark data set will still be included.

5 Conclusion

In this project, the EKF and FastSLAM algorithm for SLAM problem is implemented and tested in the simulated environment. Real-word data testing is possible, but due to different physical models and time constraints, it cannot be realized.

References

[1] M. Calonder. EKF SLAM vs. FastSLAM A Comparison, 2006.

[2] D. Koller and N. Friedman. Probabilistic Graphical Models: Principles and Techniques, 2009.

[3] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. FastSLAM: A factored solution to the simultaneous localization and mapping problem. In *Proceedings of the National conference on Artificial Intelligence*, pages 593–598. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2002.

[4] K. Murphy. Bayesian map learning in dynamic environments. *Advances in Neural Information Processing Systems*, 12:1015–1021, 2000.

[5] E. Nebot. Experimental data for SLAM. `http://www-personal.acfr.usyd.edu.au/nebot/dataset`.

[6] S. Roumeliotis. Sensing and estimation in robotics. `http://www-users.cs.umn.edu/~stergios/classes/csci5552/index.html`.

[7] Wikipedia. Extended kalman filter. `http://en.wikipedia.org/wiki/Extended_Kalman_filter`.

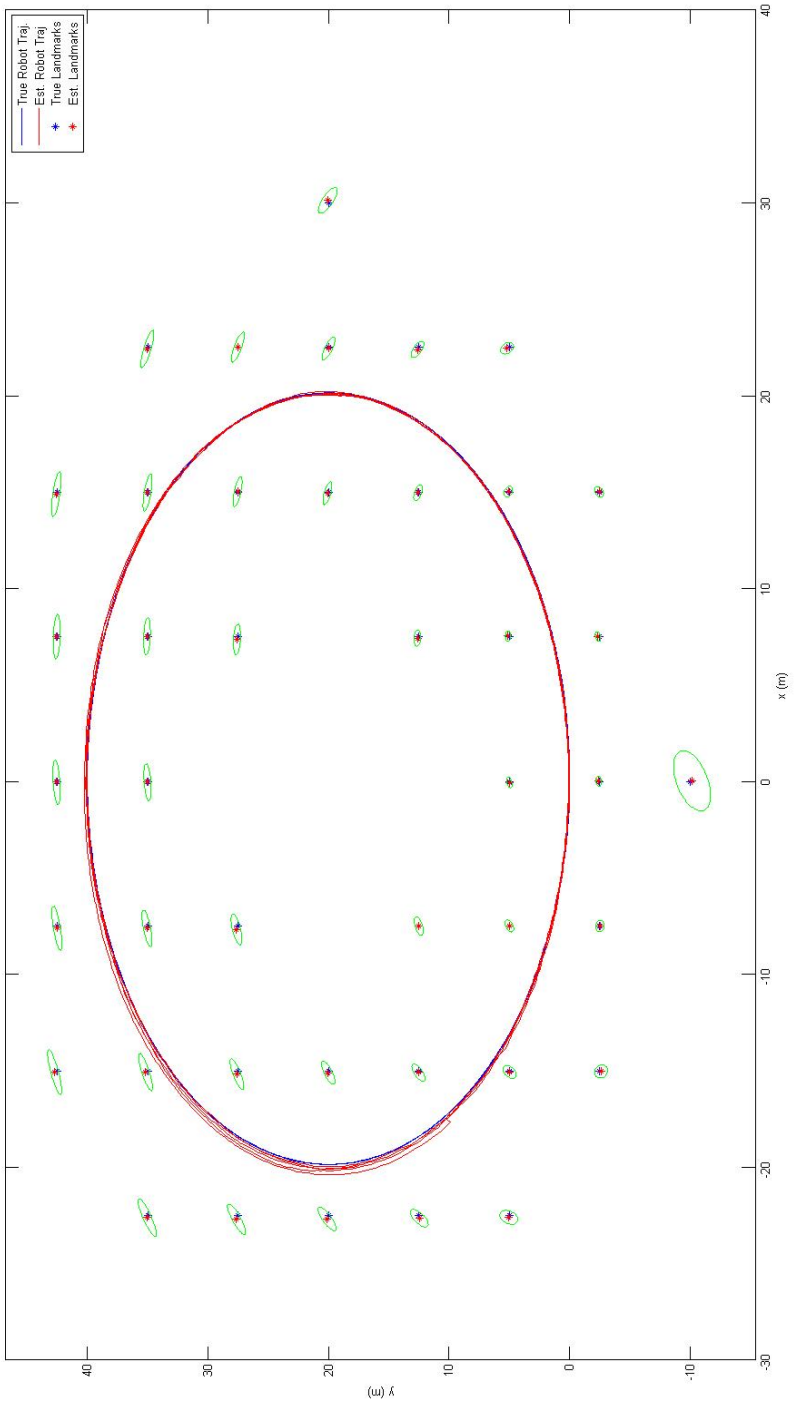


Figure 4: The environment map with EKF estimates. Known data.

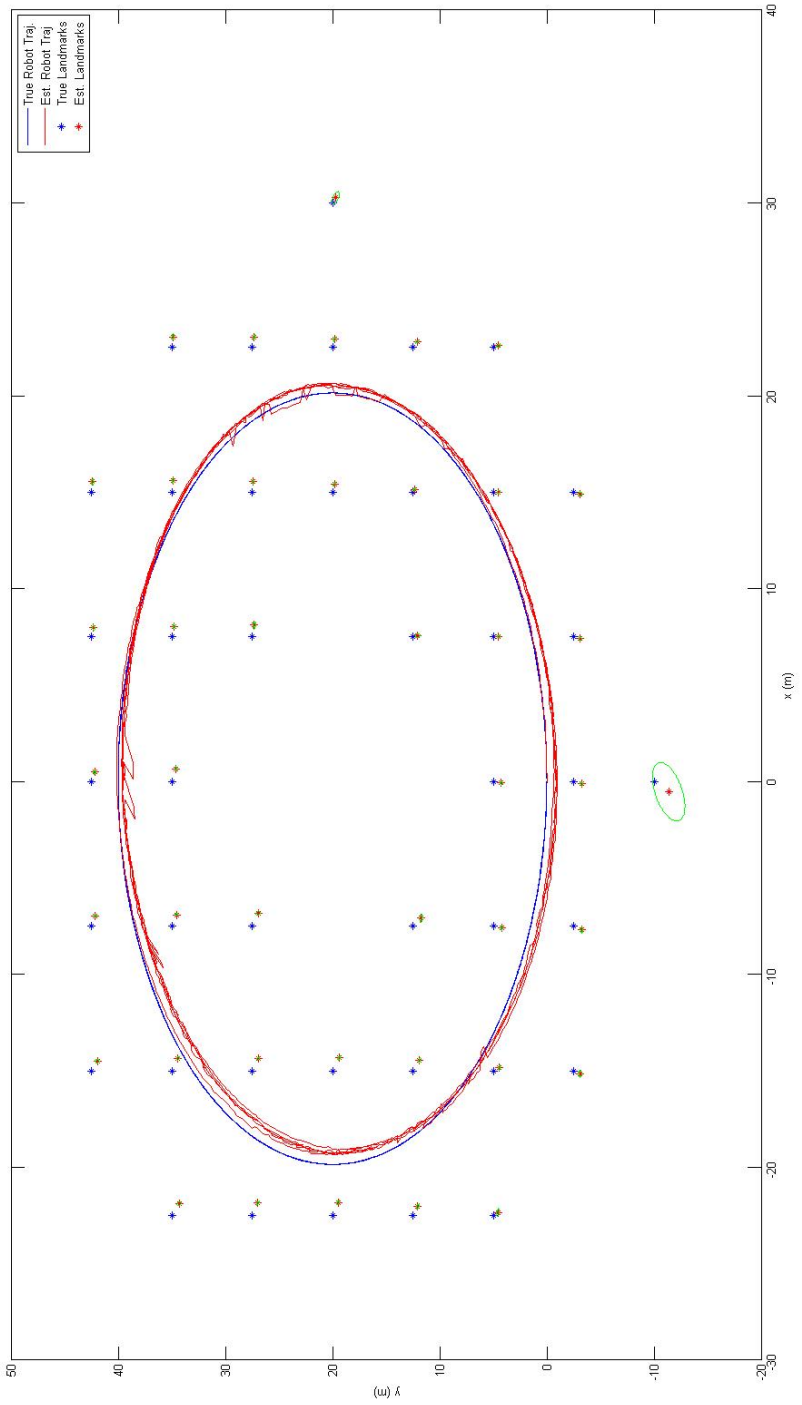
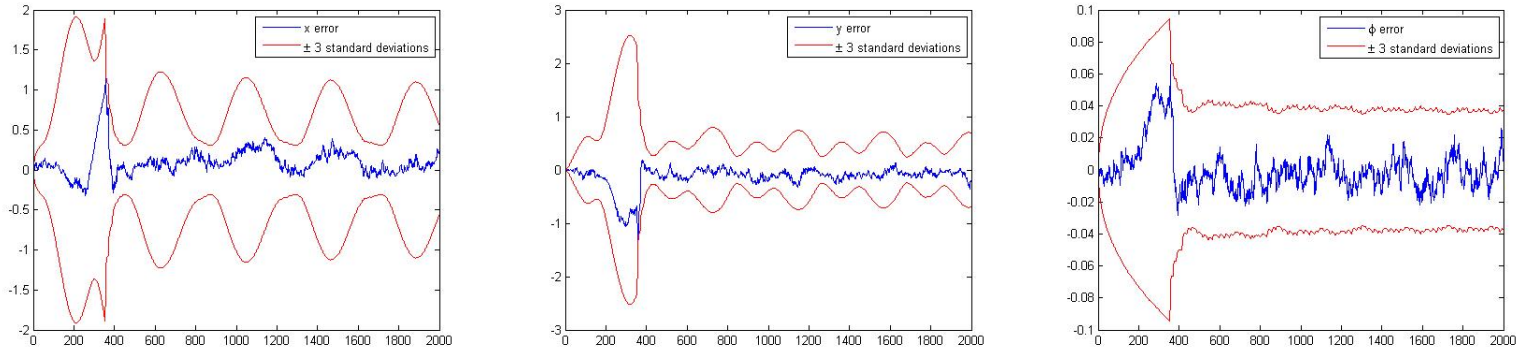
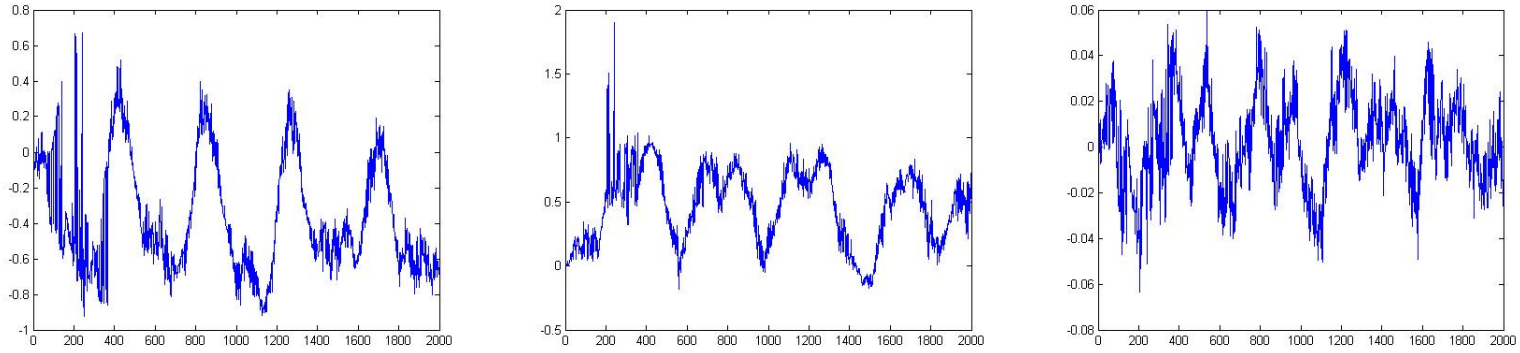


Figure 5: The environment map with FastSLAM estimates. Known data.



(a) Error in x_R (b) Error in y_R (c) Error in ϕ_R

Figure 6: Error in EKF estimates in simulated environment. Known data.



(a) Error in x_R (b) Error in y_R (c) Error in ϕ_R

Figure 7: Error in FastSLAM estimates in simulated environment. Known data.

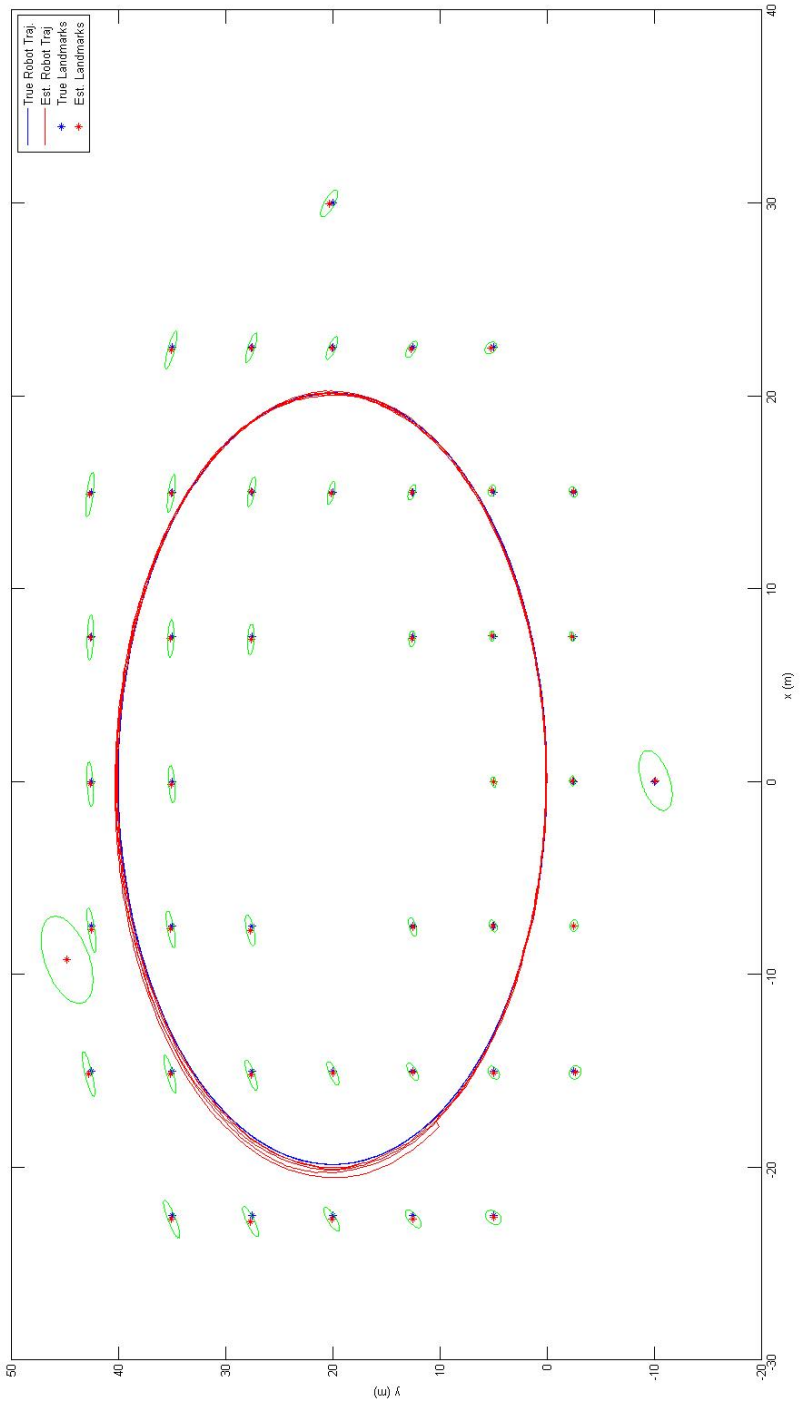


Figure 8: The environment map with EKF estimates. Unknown data.

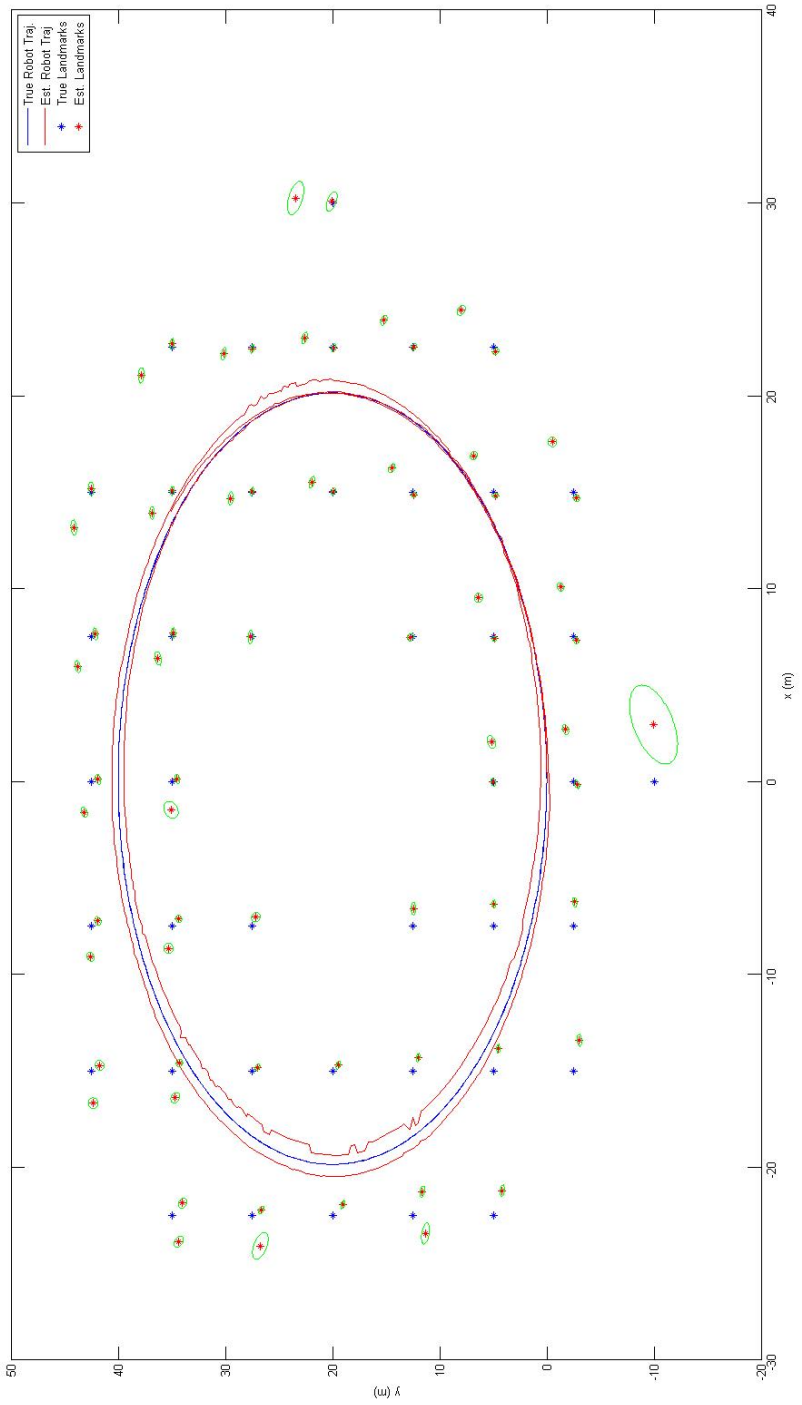
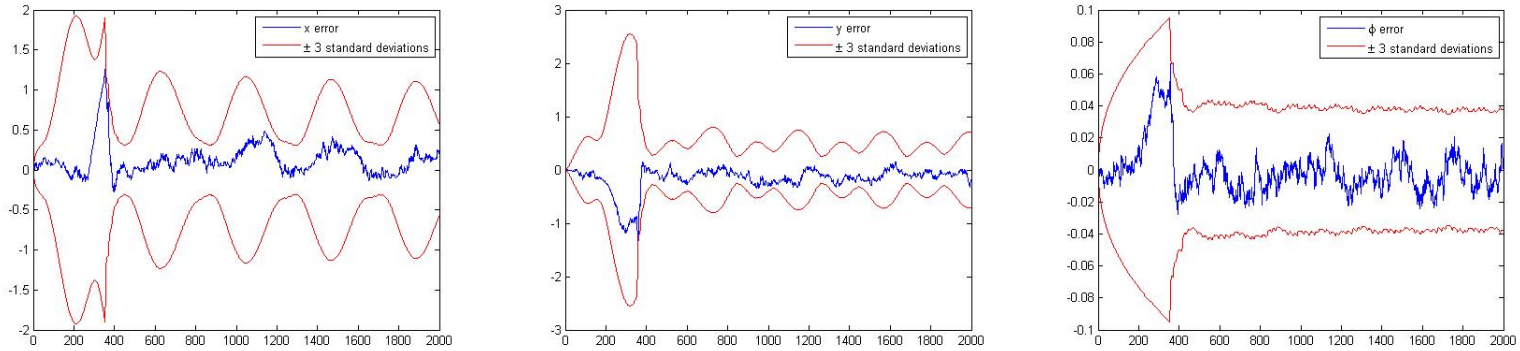
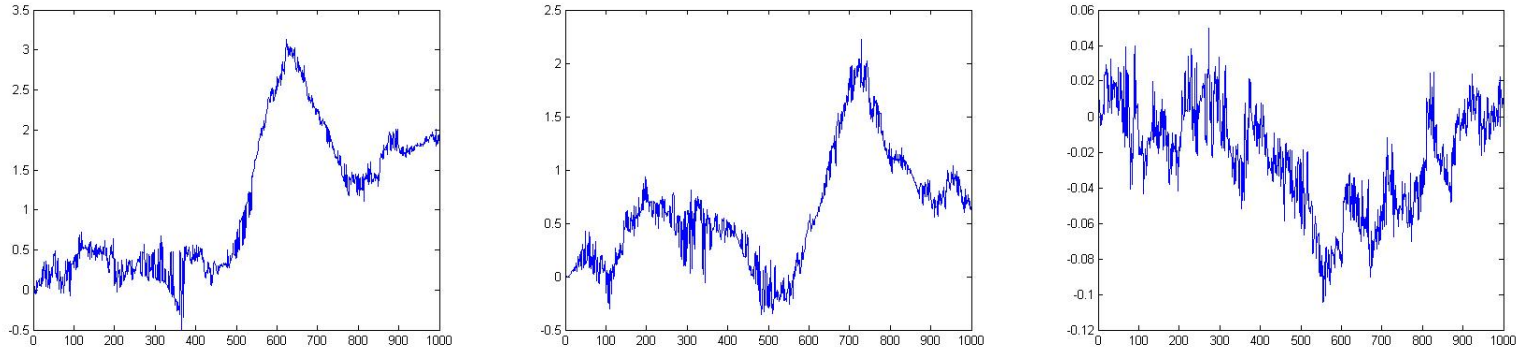


Figure 9: The environment map with FastSLAM estimates. Unknown data.



(a) Error in x_R (b) Error in y_R (c) Error in ϕ_R

Figure 10: Error in EKF estimates in simulated environment. Unknown data.



(a) Error in x_R (b) Error in y_R (c) Error in ϕ_R

Figure 11: Error in FastSLAM estimates in simulated environment. Unknown data.