Universiteit van Amsterdam

Autonomous Mobile Robots

Fourth assignment

Particle Filter Based Simultaneous Localization And Mapping

Ruben Janssen, 10252657
David van Erkelens, 10264019

Department of Computer Science
University of Amsterdam

December 15, 2013

All code published on
`www.github.com/David1209/AMR/assignment4`

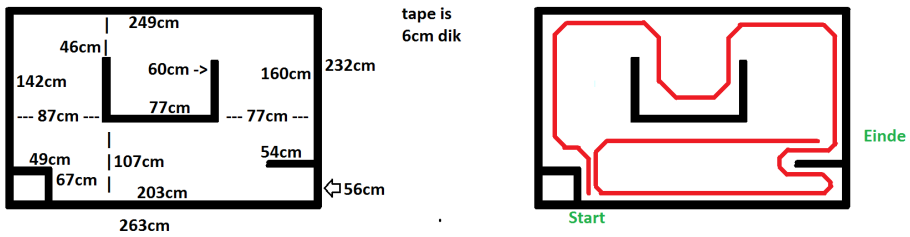
Contents

1 Introduction

In order for an autonomous mobile robot to be able to navigate without a priori knowledge about the map, it has to determine its position whilst building the map. This can be accomplished by using the Simultaneous Localization and Mapping (SLAM) algorithm. One of the variations of SLAM is FastSLAM, a particle filter based algorithm which has a better performance in real time map building than most other variations. FastSLAM scales logarithmically with the number of landmarks in the map, and is therefore much more suitable for real-time applications.

2 Setup

A NXT robot has been equipped with an omnidirectional camera. Using this NXT, a dataset of images has been made. To localize and map the environment, this dataset is used instead of a live video feed. Even though using the dataset is less accurate, the FastSLAM algorithm should be able to compensate for its uncertainty. To simulate a basic environment, 'walls' are represented by tape and from the point of view of the robot, 60 snapshots have been made that form the dataset.



(a) Environment measures (b) Approximate route

Figure 1: Experiment environment

Also, for testing the FastSLAM implementation, a log of odometry data and observable features is used.

3 FastSLAM

To implement FastSLAM, a pre-recorded log is used. The input of the SLAM algorithm consists of odometry data and observable features. Here, corners are used as landmarks, detected as endpoint segments of lines extracted from data using the omnidirectional camera. The corner detection algorithm can be reduced to two steps: line extraction and endpoint collection.

3.1 Line extraction

A snapshot from the omnidirectional camera describes a 2D slice of the environment. Points in such a snapshot are specified in the polar coordinate system $(pi, \theta i)$, with the origin at the location of the camera. In order to fit a line through the points, a line is expressed in polar coordinates:

$$x \cos \alpha + y \sin \alpha = r \tag{1}$$

In which $-\pi < \alpha < \pi$ is the angle between the line and the x-axis, $r >= 0$ is the distance of the line to the origin and (x, y) are the Cartesian coordinates of the point lying on the plane.

In order to identify the best line segments for the obtained measurements, the Split-and-Merge algorithm is employed. The endpoints from the lines returned by the Split-and-Merge algorithm are used as endpoints, and therefore the corners detected.

3.2 Parameters

The FastSLAM approach estimates the pose of the robots and the features using a particle filter. Each particle represents a possible robots position and its map of the environment. A few parameters of the SLAM algorithm can be tweaked, in order to improve performance of the algorithm:

1. **The number of particles** - saved as `PARAMS.NPARTICLES`: this influences the number of particles in the cloud. More particles gives an higher chance of a particle representing the exact position of the robot, but requires more computational power and also introduces more wrong particles.

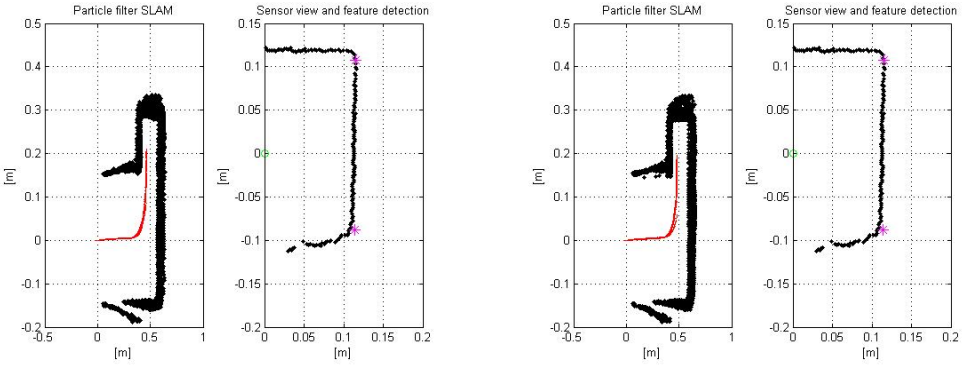
2. **The variance of the odometry** - saved as `sigmaX` and `sigmaTH`: this indicates the possible error value for the odometry of the robot. `sigmaX` indicates the error in the speed, while `sigmaTH` indicates the error in rotation. Higher values for these parameters make the size of the particle cloud bigger, and therefore the locations of the walls in the map less certain.

3. **The variance of the range sensor** - saved as `sigmaR` and `sigmaB`: this indicates the possible error value for the range sensor. Higher values for these parameters make the locations of the walls less certain.

Results of tweaking the parameters can be seen in figure 3.

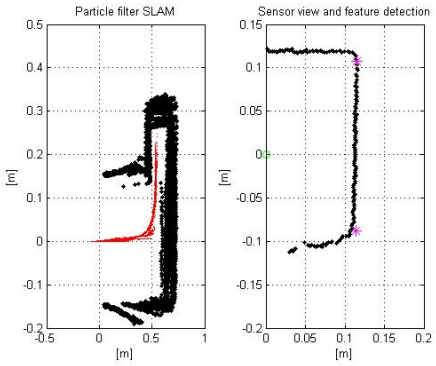
4 Experiments

When running the algorithm on the data set provided, it returned the following result:



(a) Normal result

(b) More particles



(c) Using a higher σ

Figure 2: Different parameters

4.1 Results

When comparing the images in figure 3, it can be seen that increasing the uncertainty in figure 2c increases the size of both the particle cloud and the possible locations of the walls. Furthermore, increasing the number of particles in figure 2b, make the cloud of particles denser but not necessarily wider. Therefore is seems a good practice to keep the particle cloud at a reasonable size, while trying to reduce the σs as much as possible.

5 Working with a photo dataset

5.1 Logging the data

To build the map of the environment using FastSLAM, the robot keeps track of its movements and environment by reading its encoder and taking photos of its environment respectively. Using this information, FastSLAM is able to locate the robots position and map its environment. In case of a dataset of photos without encoder logs, this becomes troublesome as the movements of the robot are not known exactly. Therefore, these movements have to be estimated by hand.

5.2 Parameters

Given that the robot movements are estimated by hand, the uncertainty of the movements increases. As a result, it becomes harder for FastSLAM to localize the robot and map the environment. Because the noise of the movements increase, the robots location is less precise and more uncertain. To compensate for this, the deviations σ_x and σ_θ increase, resulting in a less accurate localization of the robot.

In the used photo dataset however, only σ_x increases significantly. In this dataset, it is relatively more difficult to estimate the exact distance moved by the robot, than to estimate the difference in orientation. This is because almost all orientation movements by the robot are 45 degrees and can be estimated relative to the environment. To estimate the distance however, it cannot be estimated easily on the surroundings, as all distances are not exactly known.

5.3 Experiments

To examine the influence of the increased σ value, two tests were done with different σ_x values.

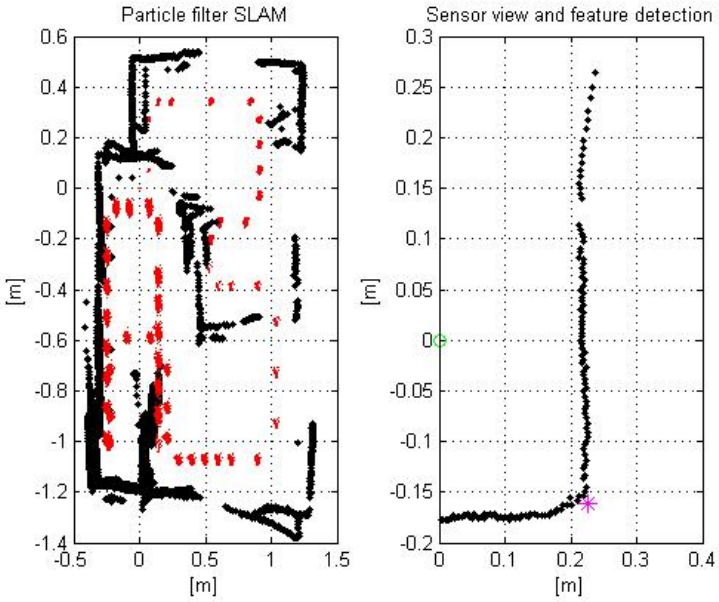


Figure 3: σ_x 0.003 meter

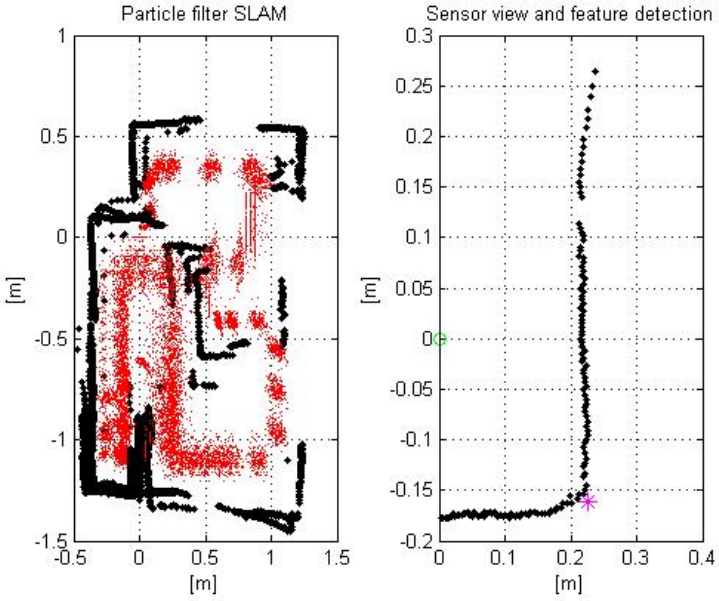


Figure 4: σ_x 0.02 meter

5.4 Results

Comparing figure 3 and figure 4, it can be stated that a higher σ_x results in a less consistent mapping of the environment. In figure 4, several walls, especially in the middle, are located at different positions while this is less the case in figure 3. A high σ_x results in more uncertain approximations of the environment.

6 Conclusion

Both experiments indicate that the most important factor is σ. It is key to the success of any SLAM implementation. A high σ will lead to inaccurate estimations and cause the robot to localize its positions and map its surroundings unsuccessfully. To improve σ, the odometry and captured images should be as noise free as possible.