USING 3D DATA FOR MONTE CARLO LOCALIZATION IN COMPLEX INDOOR ENVIRONMENTS

Oliver Wulf, Bernardo Wagner

Institute for Systems Engineering (RTS/ISE), University of Hannover, Germany

Mohamed Khalaf-Allah

Institute of Communications Engineering (ANT), University of Hannover, Germany

ABSTRACT

With this paper we present the integration of a 3D laser range sensor into a Monte Carlo Localization (MCL) system. Having the detailed environment perception of the 3D sensor, robust localization in difficult indoor scenes is possible. The presented localization system is able to handle moving people and other unmapped obstacles. At the same time it is possible to use a simplified world-model that consists of 2D walls only. This kind of map is easy to generate and independent of displaced furniture. Following the description of the localization system, real-world experiments on position tracking and global localization will demonstrate the functionality of our approach.

1. INTRODUCTION

Self-localization is one of the main requirements for autonomous mobile robots. It can be defined as the task of estimating the robot's pose given a map of the environment, the observation of the internal state and world perceptions. The mobile robot localization problem has many types [2]. The most important are *position tracking* and *global localization*. In the former, the initial pose of the robot is known and the task is to compensate incremental errors in the robots odometry. Knowledge about the initial pose is not available in the global localization problem. Consequently the robot has to determine its pose from scratch. This problem is more difficult because the robot has to handle multiple and distinct hypotheses.

Over the past years many different solutions for these localization problems have been presented. One algorithm that performs very good is the Monte Carlo Localization (MCL) [4] [5]. The advantages of this probabilistic algorithm are the possibility to represent non-linearity and the ability to handle multiple hypotheses. The MCL algorithm has been successfully tested within different environments and with a number of different sensors including 2D laser scanners [10], cameras [6] and omnidirectional cameras [7].

The localization problem is harder to solve in real-world environments with dynamic objects (e.g., people) and obstacles that are not mapped. An even bigger problem is furniture that can be assumed to be static in the first place but is subject to change over long-term operation.

For this reason the Institute for Systems Engineering is working on the integration of 3D laser sensors into mobile robot navigation. In [11] we presented a SLAM algorithm that utilizes a 3D sensor to build line-feature maps of highly cluttered indoor environments. The results show that 3D sensors are not only useful for outdoor-navigation [3] [8] but also for navigation in difficult indoor environments (see Fig. 1).

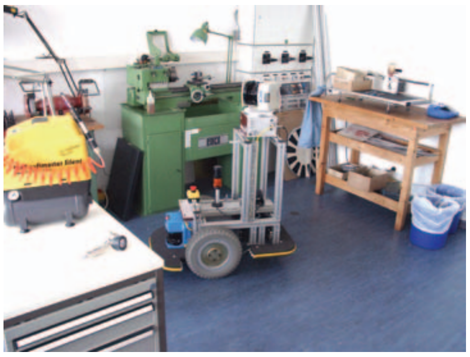

Figure 1: The experimental robotic platform ERIKA in the RTS workshop.

With this paper we present a localization system that integrates a 3D laser range senor into MCL. Thus we can demonstrate that the use of 3D sensors allows robust localization in real-world environments and that it is possible to use simplified world-models at the same time.

Section 2 describes the theoretical background of the used Monte Carlo Localization. In Section 3 the implementation details are described. The focus of this section lies on the integration of the 3D sensor and the used world-model. The functionality of the presented approach is demonstrated by experimental results on position tracking and global localization.

2. MONTE CARLO LOCALIZATION

Monte Carlo Localization (MCL) or particle filtering is a recursive Bayes filter that estimates the posterior belief distribution of a robot's pose given a map of the environment and sensor data of motion and perception. The posterior is approximated by a set of samples drawn from the belief distribution. Each sample represented by a vector contains a Cartesian coordinate, an orientation and an associated number called *importance factor*, which represents the probability of being at the location.

2.1. Bayes Filters

Bayes Filters assume that the environment is *Markovian*, i.e. past and future data are conditionally independent if the current state is known. The key idea of Bayes filtering is to estimate a posterior probability density over the state space conditioned on the sensor data. This posterior distribution is called the *belief* and is denoted by

$$\mathrm{Bel}\ (s_t) = p\ (s_t \mid d_{0\ldots t}, m) \qquad (1)$$

where *Bel* (s_t) is the robot's belief state at time t, s_t is the state at time t, $d_{0\ldots t}$ denotes the data delivered from time 0 up to time t, and m is the model of the environment, i.e. a map. Two types of sensor data can be distinguished: *perceptual data*, i.e. data that characterizes the momentary situation such as laser range scans or camera images, and *odometry data*, i.e. data that delivers information about robot motion. The former is referred to as *observations* and the later as *actions*. Assuming that observations and actions data arrive in an alternating sequence expression (1) can be rewritten as

$$\mathrm{Bel}\ (s_t) = p\ (s_t \mid o_t, a_{t-1}, o_{t-1}, a_{t-2},\ldots, o_0, m) \qquad (2)$$

The desired posterior is estimated *recursively* by applying Bayes rule and the theorem of total probability [1] to expression (2), and exploiting the Markov assumption twice. Hence, we obtain the following recursive equation

$$Bel(s_t) = \eta \;\; p(o_t \mid s_t, m) \int p(s_t \mid s_{t-1}, a_{t-1}, m) Bel(s_{t-1}) dx_{t-1} \qquad (3)$$

In the context of mobile robot localization this equation is often referred to as *Markov localization*, which is the recursive update equation in Bayes filter. Combined with the initial belief, it defines a recursive estimator for the state of a partially observable system and acts as the basis for MCL algorithms. To implement equation (3), we need to specify two conditional, usually time-invariant densities. The first is $p(s_t \mid s_{t-1}, a_{t-1}, m)$, which is called *motion model*, and characterizes the effect of actions *a* on the robot's pose. The second is $p(o_t \mid s_t, m)$ which is called *perceptual model* or *sensor model*.

2.2. MCL Algorithm

In the case of mobile robot localization, the state space representation is *continuous*. Therefore, implementing equation (3) is not an easy matter, especially if efficiency is taken into account. The basic idea of MCL is to represent the belief *Bel(s)* by a set of *n* weighted samples (or particles) distributed according to *Bel(s)*:

$$Bel(s) \approx \{s^{(i)}, w^{(i)}\}_{i=1,\ldots,n} \qquad (4)$$

where each $s^{(i)}$ is a sample of the random variable *s* (the hypothesized pose of the robot). The parameters $w^{(i)}$ are non-negative numeric values and called *importance factors*, which sum up to 1, but it is worthy to mention that this is not strictly required in particle filtering. These importance factors determine the weight or the importance of each sample. The continuous belief *Bel(s)*, thus, is approximated by a discrete probability function defined by the sample set. The initial sample set represents the initial knowledge or belief $Bel(s_0)$ about the state of the dynamical system. In position tracking, the initial belief is a set of samples drawn from a narrow Gaussian distribution centered on the start position of the robot, where each sample has the same uniform initial weight n^{-1}. In the global localization problem, the initial belief is represented by a set of samples distributed uniformly over the whole possible poses in the robot's working environment. This initial belief also is annotated by the uniform importance factor n^{-1}. The recursive update of the basic MCL is realized in three steps, computing equation (3) from right to left:

Draw a random sample s_{t-1} from the current belief *Bel* (s_{t-1}), with a likelihood given by the importance factors of the belief *Bel* (s_{t-1}).

Prediction Phase: for this sample s_{t-1}, predict a successor pose s_t, according to the motion model $p(s_t \mid s_{t-1}, a_{t-1}, m)$.

Update Phase: assign a preliminary (non-normalized) importance factor $p(o_t \mid s_t, m)$ to this sample and add it to the new sample set representing *Bel* (s_t).

Repeat the three steps n times. Finally, normalize the importance factors in the new sample set $Bel(s_t)$ so that they sum up to 1. The sample set converges to the true posterior $Bel(s_t)$ as n reaches infinity; with a convergence speed $O(1/\sqrt{n})$ [10].

3. IMPLEMENTATION

3.1. Combining 3D Perception with a 2D World Model

Having the full 3D measurement (240x181 points scanned in 3.2s) it would be possible to compare all points of the 3D point cloud to a 3D world model in the Update Phase (step 3) of the MCL algorithm. But in praxis this full 3D MCL has got a number of drawbacks. The main problem is the required 3D world model that is harder to build than a 2D world model. Another disadvantage is the necessity to compute the likelihoods of the beams (e.g., 16290 per sample). Especially in flat indoor environments, where the elevation and the roll and pitch angles are known, this complexity seems unreasonable.

Our solution to this problem is an intelligent reduction of the 3D point cloud before it is compared to a 2D map. The map that will be described in subsection 3.2 contains only vertical walls represented by 2D lines. For this reason it is necessary to find the 3D measurement points that represent these walls. Assuming that walls are vertical it is sufficient to use only one wall point per vertical raw scan. The information of all other points on the same wall is redundant.

A robust heuristic that extracts these virtual 2D scans from 3D point clouds is presented in [11]. Following this procedure the distance between the sensor and the wall in direction of the 2D raw scan can be calculated by projecting all points onto the horizontal plane. The point with the maximum horizontal distance to the sensor is then taken to represent the virtual 2D scan. As it can be seen in Fig. 3, this simple heuristic shows good results for closed indoor environments. The walls are represented correctly as long as they are visible from the sensors position. In case of open doors and windows the heuristic will select the most distant visible point. In case of open doors this points typically represent walls in the next room or on the corridor. In case of windows the return cannot be clearly matched. Experiments show that the heuristic selects objects outside the window, false points from total reflection or the window frame itself. Due to these effects doors and windows are modeled open in the 2D map.

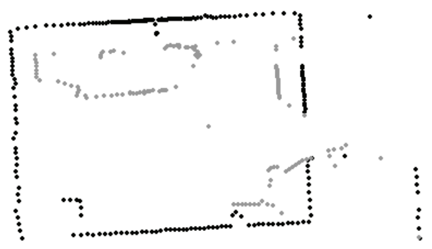
Figure 2: Virtual 2D Scan (black) in comparison to a regular 2D Scan in 20cm height (grey). Scene from (Fig.1)

3.2. The simplified World Model

In our implementation we used a line feature map as a world model to represent walls, which are the dominant feature in any indoor environment. All lines in the map are measured by hand using a tape measure. Such maps can also be generated using available metric plans of interesting working areas, e.g., factories, stores, homes, etc. Using the SLAM technique described in [11], a line feature map of an indoor environment can also be generated automatically during an initialization run. The map of the environment is stored in the AutoCAD dxf format and is loaded when the localization module is turned on.

The map is used in the update phase of the MCL to calculate the expected range measurements (circular 2D scans) at a given pose. These range measurements are then compared to the actual range measurements (virtual 2D scans delivered by the 3D laser scanner system) to determine the weight of the pose (sample) according to the sensor model described in [9]. Expected range measurements are determined by calculating the distances to the nearest lines (walls) in the map at the given position. This is done by solving the following equation system

$$\begin{pmatrix} x_1^{(i)} \\ y_1^{(i)} \end{pmatrix}_{i=1\ldots L} + a \begin{pmatrix} x_2^{(i)} - x_1^{(i)} \\ y_2^{(i)} - y_1^{(i)} \end{pmatrix}_{i=1\ldots L} = \begin{pmatrix} x_3^{(j)} \\ y_3^{(j)} \end{pmatrix}_{j=1\ldots n} + b \begin{pmatrix} \cos\Theta^{(j)} \\ \sin\Theta^{(j)} \end{pmatrix}_{j=1\ldots n} \qquad (5)$$

where b is the desired range measurement, (x_1, y_1) and (x_2, y_2) are the Cartesian coordinates of the start and end points of a line in the map, a is the line scalar and (x_3, y_3, Θ) is the given pose. This is repeated with every line in the map where L is the number of lines. Θ is increased in regular steps (according to the angular resolution of the 3D laser scanner) to provide the circular range measurements in a 2D plane. The above procedure is repeated with every sample, where n is the number of samples used.

4. EXPERIMENTAL RESULTS

Several test runs and numerous offline simulations with real data have been performed to find suitable parameter settings and to test the functionality of the 3D MCL system. Figure 3 shows the results of one test run at our lab. During this experiment the mobile robot traveled a total distance of 217m with an average speed of 0.7m/s.

All statistical tests are calculated offline in a Matlab environment and all numerical results printed in this paper are average values of 10 runs. As there is no ground truth available, explicit errors can only be measured at the end of an experiment. The localization accuracy during the run can only be tested qualitatively by plotting the laser raw data at the estimated poses.

During all experiments two persons were continuously moving around the robot in order to provide dynamic obstacles. Furthermore, other employees appear occasionally in the sensing vicinity of the robot, providing additional random corruption to the perception of the environment. These arrangements ensure the validation of our approach under real-world working conditions.

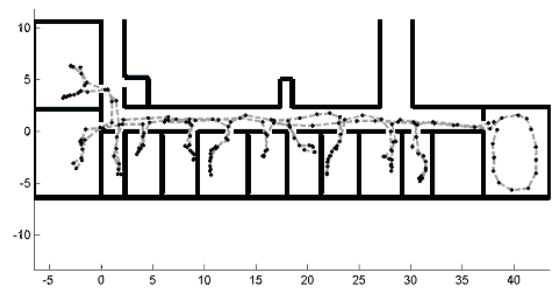

Figure 3: A test run at the Institute for Systems Engineering. The mobile robot travelled a path of 217m (grey) with an average speed of 0.7m/s.

4.1. Position Tracking

Within the position tracking experiments the initial position of the robot is known. For this reason all samples are initialized at this start position. During the experiment the localization has to show its ability to compensate the odometry drift. The experiments show that accurate position tracking is possible over a great range of parameters. A lot of offline experiments have been performed to show the break off points.

One important parameter for MCL is the number of samples as it linearly affects the processing time. Our experiments show that the localization works even using only 50 samples. But to achieve good accuracy all following position tracking experiments are calculated with 400 samples.

Another parameter that effects the processing time linearly is the number of range measurements that are used to calculate the sample weight. As it can be seen in Figure 5 it is not necessary to calculate all 240 virtual 2D scan points of a slow 3D scan (3.2s). Only in experiments with less then 20 scan points the robot may get lost. Knowing that it becomes clear that even the fast 3D scan (1.2s, 4° horizontal angle resolution) can provide sufficient information for accurate localization.

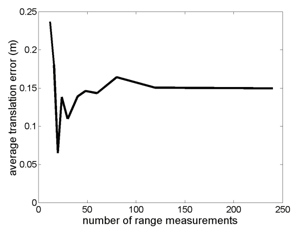

Figure 4: The effect of the number of range measurements processed in the update phase

The range deviation is used to describe the laser sensor model. Even though the accuracy of the 3D laser sensor is about 5cm the deviation of the sensor model σ_{laser} needs to be set to greater values. As it can be seen in Figure 6 values between 0.5m and 2m allow accurate position tracking. This pessimistic sensor model tolerates errors in the world model and allows recovering from positioning errors. Only values of more than 0.2% of the maximal sensor range lead to big localization errors.

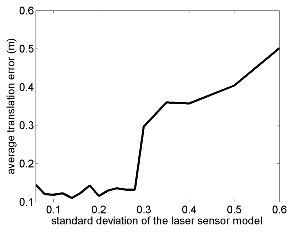

Figure 5: The influence of the laser sensor model on the position tracking results. The working range of the sensor is 20m. A standard deviation of 0.1 corresponds to 20

4.2. Global Localization

As it is known from other MLC implementations the number of samples that is needed for global localization is higher than the number of samples in position tracking. In our experiments sample numbers between 2000 and 10000 have been tested. The best results have been gained with 6000 or more samples evenly spread over the 500m² test environment (see Fig. 6).

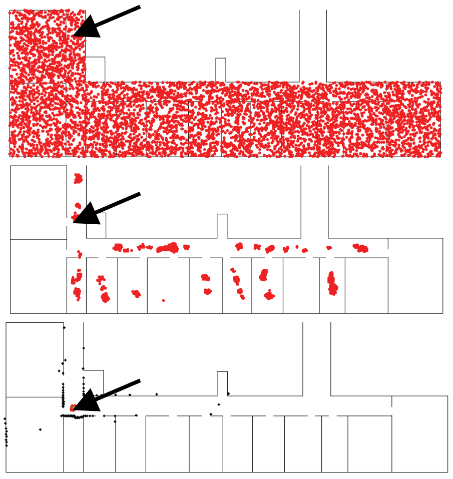

Figure 6: Global localization with 6000 samples.
Particle distribution at the beginning (upper), after 10 steps (middle) and after 19 steps (lower).
The arrow is pointing towards the true robot position.

During our experiments it turned out that the deviation of the sensor model σ_{laser} has a major influence on the global localization results. Unlike position tracking, where a wide range of values for σ_{laser} are possible, correct global localization is only possible with a precise sensor model (see Fig. 7). This effect can be lead back to the simplified world-model. As there is no furniture mapped, different rooms of approximately the same size look very similar in a virtual 2D scan. This ambiguity can only be solved by seeing through doors and windows or by passing from one room to another.

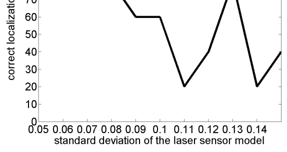

Figure 7: The influence of the laser sensor model on the global localization results.

4.3. Computational Performance

Experimental results show that the performance of the localization algorithm is strongly affected by the time that is needed to calculate the expected sensor measurement. This time is linearly depended of the number of samples, the number of range measurements per scan and the number of lines in the environment. In our application with 44 lines and 24 range measurements, the processing time for global localization (6000) samples is 1.3s and less then 100ms for position tracking. All execution times are measured on the onboard Pentium III 700MHz computer.

4.4. Comparison of MCL with 2D and 3D data

To find out if the 3D sensor is a worthwhile extension to an indoor navigation system we made experiments that compare MLC with 2D and 3D data. To carry out these experiments, we tested the MCL algorithm and the same line-feature world model first with regular 2D data and afterwards with virtual 2D data. The regular 2D data was acquired with an IBEO 360° laser scanner mounted on a Pioneer2 robot at 45cm height. The virtual 2D data was acquired with a 3D laser scanner as described in this paper.

During these experiments we found out that there are no significant differences in the localization results for simple indoor environments like a corridor. Both systems give accurate localization results and are able to cope with a certain amount of dynamic obstacles and non-modeled static objects. On one hand small objects are gated out of the virtual 2D scan, but on the other hand a small amount of clutter does not affect the pure 2D localization.

Differences appear in more complex indoor environments like a workshop (Fig. 1 & 2) or a lab (Fig. 8). In this kind of environments plane 2D data contains mainly furniture, people and other obstacles. Wall segments that are necessary for localization are more and more occluded. In contrast to that, the 3D sensor is still able to see sufficient parts of the walls. In many cases this leads to localization errors when only using the 2D scanner. A typical example can be seen in Figure 9. In this case only few wall segments can be seen. The big planar structure on the bottom of the figure results from a cupboard. As this feature is not included in the simplified world model, the scan is wrongly matched to a wall. In similar cases with only a few corresponding features the robot even got lost. There results are quite contrary to the experiments with 3D data. Having the virtual 2D scan, the good visibility of landmarks makes localization in difficult environments as accurate and robust as it is known from simple environments.

Comparing the results from localization with 2D and 3D data it can be stated that both systems give equally good results in simple indoor environments. In more difficult environments the 3D system has got its advantages. To achieve similar results the world model that is used from the 2D system would need a higher level of detail. That means that the generation of such maps is more costly and may not even be possible in case of dynamic or semi-static environments.

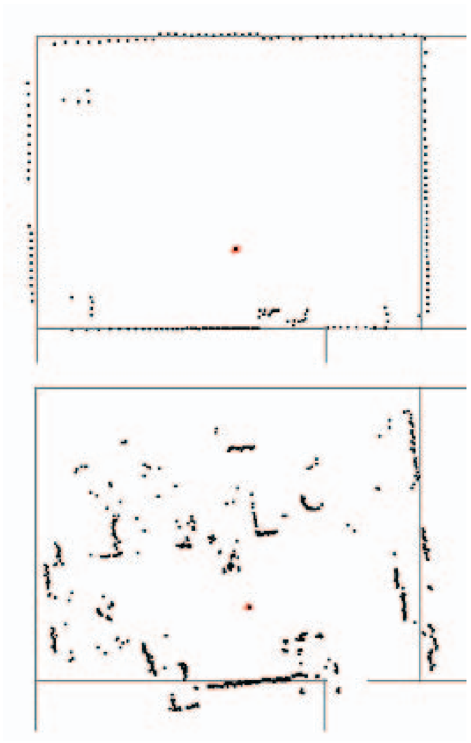
Figure 8: Localization in a lab environment. Comparison between MCL with 3D laser range data (top) and regular 2D laser range data (bottom).

5. CONCLUSION

With this paper we presented a novel localization system for mobile robots in indoor environments. The system is based on the well-known Monte Carlo Localization. In contrast to other implementations, using 2D laser scanner or camera, the described system uses a 3D laser range sensor for environment perception.

It was shown that it is not necessary to process the full 3D raw data and a 3D world model to make use of the 3D information in indoor environments. In our approach the 3D point cloud is reduced to a virtual 2D scan that contains mainly walls. Due to this reduction the computational complexity was not increased compared to MCL with 2D laser range sensors.

Experimental results show that the rich information content of the 3D sensor data allowed robust localization even in cluttered and populated rooms. On the other hand it was possible to reduce the complexity of the world model. The used map was a 2D line feature map that can be generated manually or from existing floor planes. As there is no movable furniture included in the map, the manual generation is simplified and the long-term use is possible.

6. REFERENCES

[1] Y. Shalom, X. Li, and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation: Theory, Algorithms and Software*, Wiley, New York, 2001.

[2] J. Borenstein, H. R. Everett, and L. Feng, *Navigating Mobile Robots: Systems and Techniques*, A.K. Peters, Wellesley, MA, 1996.

[3] C. Brenneke, O. Wulf, and B. Wagner, Using 3D Laser Range Data for SLAM in Outdoor Environments, In *International Conference on Intelligent Robots and Systems (IROS)*, Las Vegas, USA, 2003.

[4] F. Dellaert, D. Fox, W. Burgard, and S. Thrun, Monte Carlo Localization for Mobile Robots, In *International Conference on Robotics and Automation (ICRA), 1999*.

[5] A. Doucet, N. de Freitas, and N. Gordon, *Sequential Monte Carlo in Practice*, Springer, New York, 2001.

[6] D. Fox, W. Burgard, F. Dellaert, and S. Thrun, Monte Carlo Localization: Efficient Position Estimation for Mobile Robots, In *American Association for Artificial Intelligence, 1999*.

[7] H. Gross, A. Koenig, H. Boehme, and C. Schroeter, Vision-based Monte Carlo Self-localization for a Mobile Service Robot Acting as Shopping Assistant in a Home Store, In *International Conference on Intelligent Robots and Systems (ICRA)*, Lausanne, Switzerland, 2002.

[8] S. Lacroix, A. Mallet, D. Bonnafous, G. Bauzil, S. Fleury, M. Herrb, and R. Chatila, Autonomous Rover Navigation on Unknown Terrains: Functions and Integration, In *International Journal of Robotics Research*, vol. 21, issue 10, 2002.

[9] S. Thrun, Probabilistic Algorithms in Robotics, *AI Magazine*, vol. 21(4), 2000.

[10] S. Thrun, D. Fox, W. Burgard, and F. Dellaert, Robust Monte Carlo Localization for Mobile Robots, In *Artificial Intelligence*, vol. 128, 2001.

[11] O. Wulf, K. O. Arras, H. I. Christensen, and B. Wagner, 2D Mapping of cluttered Indoor Environments by means of 3D Perception, In *International Conference on Robotics and Automation (ICRA)*, New Orleans, USA, 2004.