in: Proc. Int. Workshop on Dynamic Perception 2002, Bochum, pp.171-176, IOS Press, Infix 2002

Panoramic View Based Monte Carlo Self-localization for Mobile Robots Operating in Real-world Environments

H.-M. Gross, H.-J. Boehme, Ch. Schroeter, and A. Koenig

Ilmenau Technical University, Department of Neuroinformatics, Germany
Horst-Michael.Gross@tu-ilmenau.de

Abstract. We present a novel panoramic view based robot localization approach which utilizes the Monte Carlo Localization (MCL) [1], a Bayesian filtering technique based on a discrete density representation by means of particles. We show how omidirectional imaging can be combined with the MCL-algorithm to globally localize and track a mobile robot given a taught graph-based representation of the operation area. To demonstrate the reliability of our approach, we present promising experimental results in the context of a challenging robotics application, the self-localization of a mobile service robot acting as shopping assistant in a very regularly structured, maze-like and crowded environment, a home store.

1 Introduction and motivation

Self-localization is the task of estimating the pose (position and orientation) of a mobile robot given a map of the environment and a history of sensor readings and executed actions. This includes both the ability of globally localizing the robot from scratch, as well as tracking the robot's position once its location is known. The localization problem is one of the fundamental problems in mobile robot navigation and many solutions have been presented in the past including approaches employing Kalman filtering, grid-based Markov localization, or Monte Carlo Methods [3]. The current state-of-the-art localization methods often use laser range finders or sonar, but these sensor modalities tend to be easily confused in environments with very regular topology, e.g. a supermarket or a home store with a great number of hallways of equal width, length and geometrical structure. Because of this maze-like topology, self-localization methods based on laser or sonar can produce numerous ambiguities complicating or preventing a quick self-localization or re-localization in case of a complete loss of positioning. In contrast, vision-based systems do not show these limitations, but supply a much greater wealth of information about the 3D-structure of the hallways and racks. For example, the filling of the goods racks gives the hallways a characteristic appearance, especially with respect to color or texture. Because of this, we expected to defuse the localization problem drastically by development of an approach for view-based localization that combines omnidirectional imaging with the probabilistic Monte Carlo Localization (MCL) [1].

2 Omnivision-based MCL

The Monte Carlo Localization (MCL) method underlying our omnivision-based localization approach is a version of Markov localization [6], a family of probabilistic approaches for approximating a multi-modal density distribution coding the robot's belief

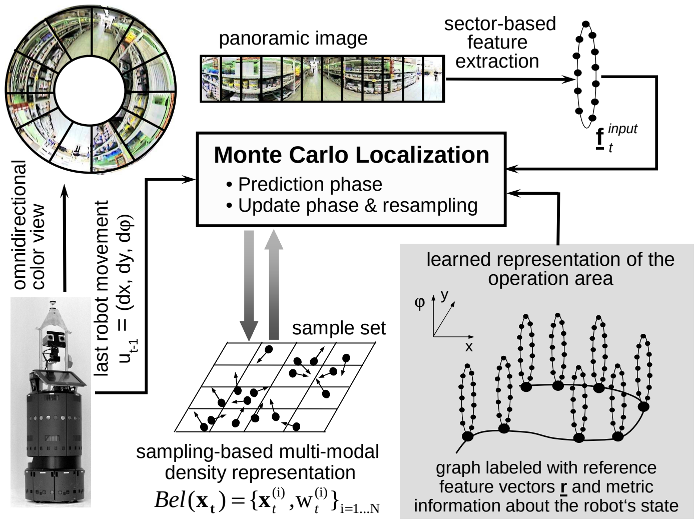
panoramic image
sector-based feature extraction
$\underline{\mathbf{f}}_t^{input}$
Monte Carlo Localization
• Prediction phase
• Update phase & resampling
omnidirectional color view
last robot movement
$u_{t-1} = (dx, dy, d\varphi)$
sample set
sampling-based multi-modal density representation
$Bel(\mathbf{x}_t) = \{\mathbf{x}_t^{(i)}, w_t^{(i)}\}_{i=1...N}$
learned representation of the operation area
φ
y
x
graph labeled with reference feature vectors $\underline{\mathbf{r}}$ and metric information about the robot's state

Figure 1. General idea of our omniview-based Monte Carlo Localization. The approach is based on a graph-based representation of the operation area. The nodes of the graph are labeled with both view-based visual features and metric information about the pose of the robot (position and heading direction in a world-centered reference frame) at the moment of the node insertion.

$Bel(\underline{\mathbf{x}}_t)$ for being in state $\underline{\mathbf{x}}_t = (x, y, \varphi)_t$ in its state space. x and y are the robot's position coordinates in a world-centered Cartesian reference frame, and φ is the robot's heading direction. The key idea of MCL is to represent the belief $Bel(\underline{\mathbf{x}}_t)$ by a set S_t of N weighted samples distributed according to $Bel(\underline{\mathbf{x}}_t)$: $S_t = \{\underline{\mathbf{x}}_t^{(i)}, w_t^{(i)}\}_{i=1..N}$. Here each $\underline{\mathbf{x}}_t^{(i)}$ is a sample, and the $w_t^{(i)}$ are non-negative numerical weighting factors called importance factors. Because the sample set constitutes a discrete approximation of the continuous density distribution, the MCL approach is computationally efficient, it places computation just “where needed”.

The general idea of our view-based Monte Carlo Localization is illustrated in Fig. 1. In our approach, we use a graph-based representation of the operation area by a set of visual reference vectors $\underline{\mathbf{r}}(x, y, \varphi)$ extracted from the respective panoramic views at positions x, y in heading direction φ (Fig. 1, bottom right). The graph is constructed on-the-fly when manually joy-sticking the robot through the hallways of the store. During this training, omnidirectional images are captured from the environment and associated with the corresponding locations. For this purpose, in addition to the feature vectors extracted from the omnidirectional images, the nodes of the graph are labeled with metric information about the pose $\underline{\mathbf{x}} = (x, y, \varphi)$ of the robot at the moment of the node insertion. A new node (reference point) with importance for the representation is inserted, either if the Euclidian position distance to other reference points in a local Ω-vicinity or if the Euclidian feature distance between the current feature vector $\underline{\mathbf{f}}_t^{input}$ and the feature vectors $\underline{\mathbf{r}}_\Omega(x, y, \varphi)$ of these reference points are larger than given values. However, the labeling of the graph nodes with odometric data about the pose of the robot necessitates an efficient correction of odometry because of the increasing error over time, especially concerning the orientation angle. To attenuate this effect, we utilize a

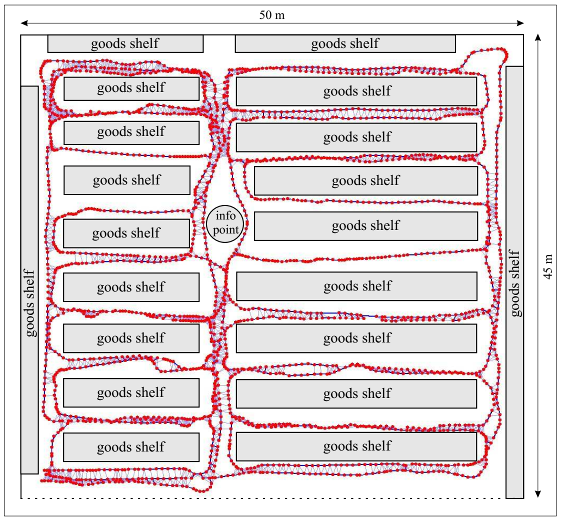

Figure 2. Topological map of the operation area in the home store. The size of the area is 50×45 meters, the graph consists of 2007 reference points (marked as dots) labeled with visual feature vectors and odometric data about the pose (position and orientation) of the robot at the moment of node insertion. The total distance travelled to learn this map was about 1000 meters.

specific feature of our market floor that shows a very regular structure caused by tiles that are uniquely oriented across the whole market area. For details of our vision-based odometry correction see [2]. We utilized this odometry correction method for learning a large-scale graph representation of the operation area as shown in Fig. 2 and achieved a very small absolute position error of about $60cm$ after a total distance of 1000 meters.

Feature extraction: Both during map-building and self-localization, the omnidirectional image is transformed into a panoramic image (see Fig. 1, top). Each panoramic image is first partitioned into a fixed number of non-overlapping sectors (typ. 10) each covering a part of the panoramic field of view. The following criteria determined the selection of appropriate features to describe the present scene: 1) To allow for an on-line localization, the calculation of the features should be as easy and efficient as possible. 2) The features should include the orientation of the robot as prerequisite to estimate the heading direction of the robot. 3) The feature description should allow for an easy generation of expected observations for unknown positions and orientations of the robot. 4) The features should be largely insensitive against partial occlusion of the environment, such as caused by people in the vicinity of the robot. Considering these criteria and the requirements of other omnivision-based localization approaches published recently, e.g. [4, 5], we decided to implement the simplest feature extraction method possible. Thereto, for each sector of the panoramic image, the mean RGB-color value is determined. This way, for each node in the graph a reference feature vector $\underline{\mathbf{r}}(x, y, \varphi)$ consisting of a small number of mean RGB-values has to be learned.

The localization algorithm: In analogy to the MCL algorithm presented in [1], our omniview-based MCL proceeds in two phases: In the *Prediction phase (robot motion)*, the sample set computed in the previous iteration (or during random initialization) is moved according to the last movement of the robot u_{t-1} (Fig. 1, left). The *motion model* $p(x_t|x_{t-1}, u_{t-1})$ describes how the position of the samples changes using information u_{t-1} from odometry. This way, MCL generates N new samples that approximate the expected density distribution of the robot's pose after the movement u_{t-1}. To determine the expected observations $\underline{\mathbf{f}}_t^{(i)}$ of the moved samples, our approach requires interpolations both in state and feature space because of the coarse graph representation and the chosen feature coding. For each sample $s^{(i)}$, we first interpolate linearly between the reference feature vectors $\underline{\mathbf{r}}(x, y, \varphi)$ of the two reference nodes closest to the respective sample position $\underline{\mathbf{x}}_t^{(i)}$. After this, the resulting feature vector is rotated according to the expected new orientation $\varphi_t^{(i)}$ of the sample $s^{(i)}$. Since the feature vector only has a discrete number of components, we utilize a linear interpolation between the features of adjacent segments. This way, we obtain a set of N new feature vectors $\underline{\mathbf{f}}_t^{(i)}(x, y, \varphi)$ describing the expected observations of the moved samples in the new states $\underline{\mathbf{x}}_t^{(i)}$.

In the *Update phase (new observation)*, the actual panoramic view at the new robot position has to be taken into account in order to correct the sample set S_t. For this, the importance factor $w_t^{(i)}$ of each sample $s^{(i)}$ is computed. It describes the probability that the robot is located in the state $\underline{\mathbf{x}}_t^{(i)}$ of the sample. We determine the similarity $E_t^{(i)}$ between the current input feature vector $\underline{\mathbf{f}}_t^{input}$ extracted from the panoramic view at the new robot position and the expected feature vector $\underline{\mathbf{f}}_t^{(i)}$ of each sample $s^{(i)}$ simply by computing the angle between both normalized vectors applying a simple Gaussian-like *observation model*. Now $w_t^{(i)} = 1 - \alpha E_t^{(i)}$ can be determined, where α is a normalization constant that enforces $\sum_{j=1}^{N} w_t^{(j)} = 1$. The final sample set S_t for the next iteration is obtained by *re-sampling* from this weighted set. The re-sampling selects those samples with higher probability that have a high importance factor $w_t^{(i)}$. Samples with low importance factors are removed and randomly placed in the state-neighborhood of samples with high factors. After that, both phases are repeated recursively.

3 Experimental results

All experiments were carried out in the 'toom' home store Erfurt with our experimental platform PERSES, a standard B21 robot additionally equipped with an omnidirectional imaging system for vision-based navigation and human-robot interaction. The experiments were performed as off-line cross-validation tests on different sequences of images acquired in the home store. All images were labeled with the corresponding correct pose of the robot. One of the sequences is used as training data to build the graph while the other ones are used as test data (5000 pose-labeled images) to determine the localization error. Every localization experiment has a typical length of 190 movements, this corresponds to a path length of about 130 meters. Per experiment, the mean absolute localization error is determined. Every experiment was repeated 20 times, and the localization errors were averaged. It is to note that, in all cases, we studied the worst-case scenario: our robot had no prior information about its initial pose - this is a typical global localization problem. All tests can be judged as being very successful, as our localization system was able to find and continually track the position of the robot. Fig. 3 illustrates the typical course of a view-based self-localization and position

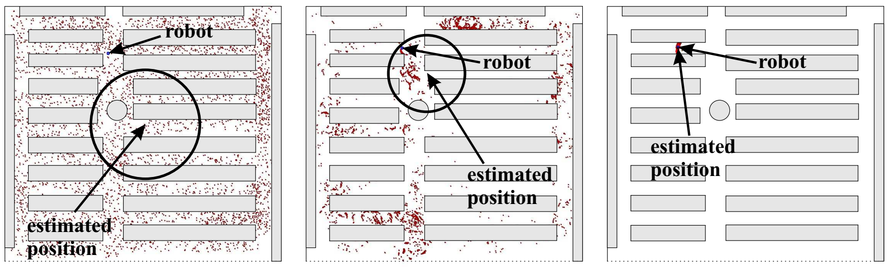

Figure 3. Self-localization and tracking experiment executed in a large section $(50 \times 45m^2)$ of the home store. The sequence depicts the temporal condensation dynamics of about 4.000 samples (initial distribution, after 3 steps, and 9 steps). In the beginning, the robot is globally uncertain, the particles are spread uniformly throughout the free space. The variance of the 10% of the samples with the highest importance factors is marked as circle. Already after 9 movements (about 5,50 m), MCL has disambiguated the robot's position - the majority of samples is now centered tightly around the correct position, the variance is drastically reduced.

tracking experiment executed in a large section of the store $(50 \times 45m^2)$. Despite the geometrical uniformity of the selected hallways and the coarse graph-structure (2007 nodes), our omniview-based MCL yields very precise localization results already after a few robot movements. For example, after 9 movements and observations, which corresponds to a travelled distance of about 5,50 meters, the difference between estimated and correct position of the robot was lower than 40 cm. The mean localization error of our test set is even smaller than 25 cm. The time required for computation of the MCL algorithm directly depends on the total number of samples. With the current on-board equipment (1500 MHz AMD Athlon), our algorithm requires about 50 ms for 4.000 samples. The time for image transformation and feature extraction takes about 25 ms per image. Therefore, our localization system enables real-time localization leaving a good amount of processing time for other navigation modules.

Dealing with occlusions: It is clear that we have to cope with occlusions in the scene, such as, for example, people walking by or objects being moved around in the environment. However, due to its wide visual field, occlusion of the entire panoramic view becomes very unlikely. For example, in Fig. 4 the two people standing as close as possible to the robot occlude no more than 10% of the visual field. To test the robustness of the localization algorithm, the test images were occluded by artificial gray-colored segments. The impact of occlusion effects was gradually controlled by the percentage of image content covered by the artificial image. Fig. 4 (bottom) depicts the results w.r.t. localization accuracy and various degrees of occlusion. For 0% occlusion, the mean position error is 25 cm and covers a range between 15 and 30 cm. The mean position error remains relatively low until 15% occlusion. Thereafter, the error vigorously increases since the image is affected by severe occlusions. However, due to the geometry of robot and vision system, it is not possible to place more than three or four people directly around the robot. Therefore, the maximum occlusion by people cannot be larger than 15-20%. Moreover, the internal particle dynamics of the MCL-algorithm realizes a kind of temporal self-stabilization of the estimation result, therefore, the influence of heavy but short occlusions can be largely neglected.

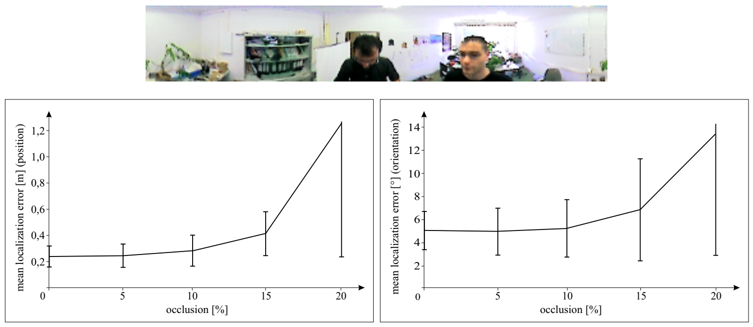

Figure 4. *(Top)* Occlusion example: two people are standing as close as possible to the robot and occlude about 10% of the visual field. *(Bottom)* Result of experiments investigating the influence of local occlusions on the position error *(left)* and and the orientation estimation *(right)*.

4 Conclusions and future work

In this paper, we have shown that particle filters in combination with a graph-based representation of the operation area by local panoramic views can be used to perform an omniview-based self-localization of a mobile robot in a challenging real-world application. Our localization system uses color omni-vision, works in real-time, and can easily be trained in new operation areas by joy-sticking. The results of the executed experiments confirm the accuracy and robustness of our omniview-based self-localization method.

Currently, theoretical and experimental studies are carried out to further improve our omniview-based MCL-system. For example, we are investigating the impact of the motion and observation models on the pose estimation and are studying the influence of a new mechanism adaptively controlling the sample rate on-the-fly on the localization accuracy. Other running experiments are dealing with the impact of appearance variations at the reference points in the learned graph, e.g. as result of a changed filling of the goods racks or modifications in the market topology. Moreover, our algorithm has to demonstrate its capabilities scaling up to the whole market area with a size of $100 \times 60m^2$ over a longer period of operation.

References

1. D. Fox, W. Burgard, F. Dellaert, S. Thrun. Monte Carlo Localization: Efficient Position Estimation for Mobile Robots. In: *Proc. AAAI-99*, 1999
2. H.-M. Gross, H.-J. Boehme. PERSES - a Vision-based Interactive Mobile Shopping Assistant. in: *Proc. IEEE Intern. Conf. on Systems, Man and Cybernetics, 2000*, pp. 80-85
3. J.-S. Gutmann, D. Fox. An Experimental Comparison of Localization Methods Continued. to appear: *Proc. IROS 2002*
4. B. Kroese, N. Vlassis, R. Bunschoten and Y. Motomura. A probabilistic model for appearance-based robot localization. *Image & Vision Computing*, 19 (6) 381-391, 2001
5. L. Paletta, S. Frintrop, and J. Hertzberg. Robust Localization Using Context in Omnidirectional Imaging. *Proc. ICRA 2001*, pp. 2072-2077, 2001.
6. S. Thrun. Learning metric-topological maps for indoor mobile robot navigation. *Artif. Intell.*, 99 (1998) 21-71