

Adaptive Reference Views for Appearance-based Localization

Alexander König Manuela Bischoff
Technische Universität Ilmenau, Fachgebiet Neuroinformatik
PF 100565
98684 Ilmenau, Germany
alexander.koenig@tu-ilmenau.de

Abstract

In this paper, we present a concept to represent dynamic environments in appearance-based maps. Today the appearance-based robot localization in static environments is not really challenging. A number of methods, as for example Monte-Carlo-Localization (MCL [2]), can solve this task pretty good. However, the assumption of an operation area without any changes does not really reflect typical circumstances. Unfortunately, the real world is not static. In our case, the experimental area is a large-scale home store, where objects are frequently moved or they change their appearance. For example, the filling of the goods shelves with different articles changes the appearance of hallways. In [3, 4] we presented an omniview-based Monte-Carlo localization approach, which used snapshots (reference views) to represent the environment. To better deal with dynamic changes, in this paper we show how to create and use adapting reference views representing more than one appearance. The basic idea of this extended approach is explained and preliminary experiments and results are presented.

1 Introduction and motivation

In [3] we introduced our long-term research project PERSES (PERSONAL SERVICE SYSTEM) which aims to develop an interactive mobile shopping assistant. The assistant should get the capability to autonomously guide its user, a customer, to desired articles within a home store, realizing a guidance function, or to follow him as a mobile service-companion while continuously observing the user and his behavior. Because of the desired intuitive interaction between customer and shopping assistant, we have to cope with a great number of demanding interaction tasks which necessarily require image processing to be solved, like e.g., the detection of users willing to interact, the learning of a highly-specific user model allowing the tracking and re-detection of the current user if lost from view, or the recognition of gesticulated user instructions. Furthermore, the robot has to be able to navigate through its operation area. Precondition for the navigation task is a robust self-localization. To develop a low-cost mobile shopping assistant we exclusively use low-cost sensor systems - sonar and camera. The sonar sensors are utilized for obstacle avoidance and odometry correction, whereas camera views are

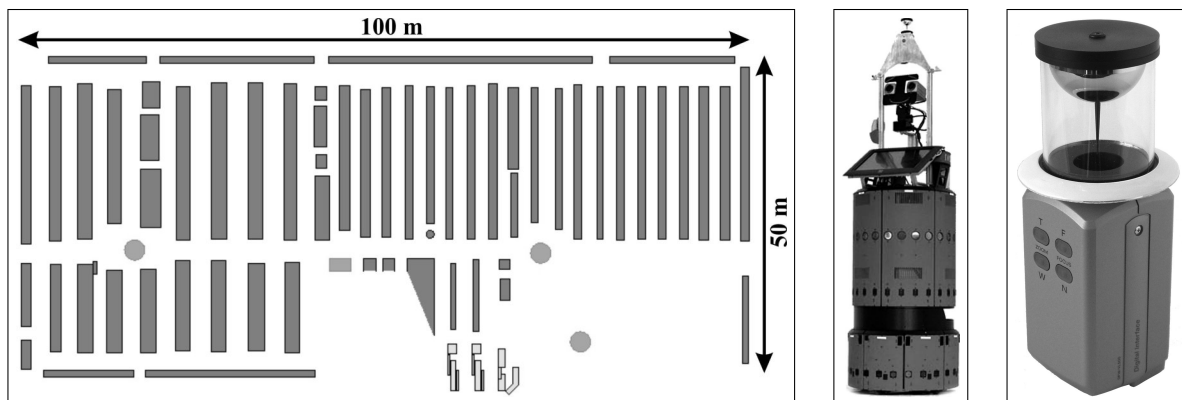


Figure 1. (Left) Location plan of our experimental area, a large-scale home store. The topology of the depot is characterized by many similar, long hallways of equal width. The goods shelves are illustrated by gray rectangles here. (Mid) Experimental platform PERSES, an extended version of a B21 robot, equipped with an omnidirectional imaging system on top. (Right) The omnidirectional vision system consists of a vertically oriented color camera and a middle-size spherical mirror.

used to interact with the user and to self-localize the robot.

Characteristics of the operation area: The topology of the operation area, a typical home store, is characterized by many similar, long hallways of equal length, width and geometrical structure (see Figure 1, left). Our localization approach is completely based on visual perception. The visual input is delivered by an omnidirectional camera, a camera type which has become very popular for vision-based robot navigation and human-robot interaction over the last years. The robot PERSES we are using as experimental platform is a standard B21 robot additionally equipped with an omnidirectional imaging system for vision-based navigation and human-robot interaction (Figure 1, mid). The omnidirectional imaging system consists of a vertically oriented color camera and a middle-size spherical mirror mounted in front of the lens. The camera is mounted on top of the mobile platform with its axis placed coincident to the platform's rotation axis. The spherical mirror yields an image of the environment around the robot, and its field of view is the widest among sensors with convex mirrors. Such a wide view is very useful for locating the robot along its route, as we are interested in. Our omnivision-based robot self-localization does work in static large-scale environments reliably and correctly (see [4]).

However, localization accuracy will decrease with the modifications and changes in the operation area are. In the real world, a lot of alterations and dynamics occur due to people walking by or new fillings of the goods shelves. These dynamic changes influence the appearance of the environment and therefore, the current view will not match (correctly) with views in the previously created map (despite the probabilistic localization scheme). Hence, a continuous adaptation of the reference map becomes necessary. The general idea of this updating is explained in this paper. The remainder of the paper is organized as follows: In section 2 we describe our overall localization approach, thereby still assuming a static environment. This is a precondition for understanding the problem of changing environments for the localization and to realize our approach to deal with these dynamic changes in environments. Section 3 introduces the basic idea of representing dynamic changes of the environment in an appropriate manner. Section 4 demonstrates the capabilities of our approach via first experiments and presents preliminary results. Finally we will give a conclusion and some outlooks.

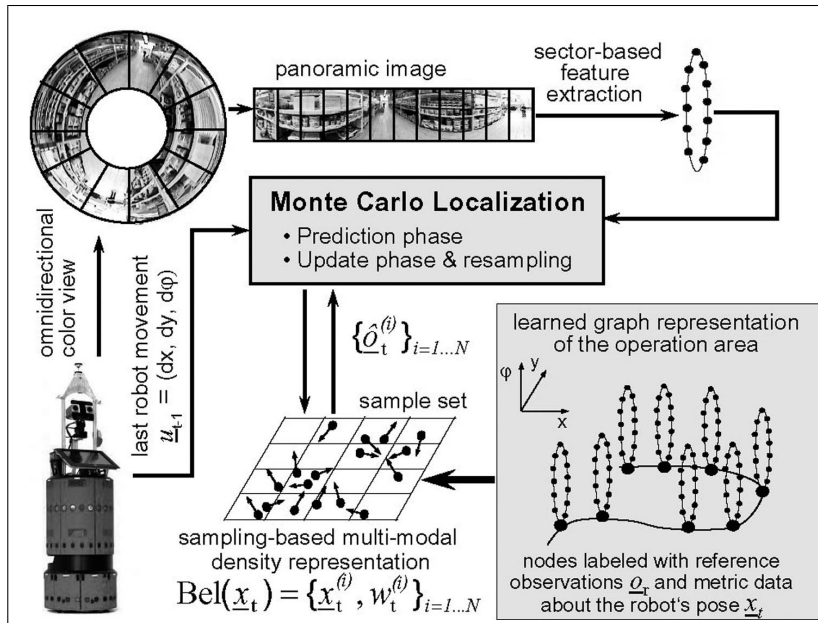


Figure 2. General idea of our omniview-based MCL. The approach is based on a graph-based representation of the operation area. The nodes of the graph are labeled with both view-based reference observations and metric information about the pose of the robot at the moment of node insertion.

2 Summary of our localization approach

Our omniview-based probabilistic self-localization approach presented in [3, 4] is a version of the Monte Carlo Localization (MCL) [2] and makes use of particle filters for approximating a multi-modal density distribution coding the robot’s belief $Bel(\underline{\mathbf{x}}_t)$ 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 $Bel(\underline{\mathbf{x}}_t)$ for being in the current state $\underline{\mathbf{x}}_t$ 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 importance weights, which sum up to one. Because the sample set constitutes a discrete approximation of the continuous density distribution, particle filters are computationally efficient since they focus the particles on those regions in state space with high likelihood, where things really matter. The general idea of our omniview-based MCL is illustrated in Figure 2. As map of the environment, our approach employs a graph representation of the environment which is learned on-the-fly while manually joy-sticking the robot through the operation area (Fig. 2, bottom right). Each node of the graph is labeled with both visual reference observations $\underline{\mathbf{o}}^r(x, y, \varphi)$ extracted from the respective panoramic view at position x, y in heading direction φ and the corresponding odometric data about the pose at the moment of the node insertion during teaching. A new node (reference points) is inserted, if either the Euclidian position distance to other reference points in a local vicinity or the Euclidian distance between the current observation $\underline{\mathbf{o}}_t$ (appearance) and the reference observations $\underline{\mathbf{o}}^r(x, y, \varphi)$ at adjacent reference points is larger than given threshold values. Henceforth, current and reference observation will be also called **current view** and **reference view**. Generally, view here describes the appearance-based visual perception of the robot, observed at a state \vec{x} . Reference views or observations respectively are estimated views for a special state \vec{x} – what will be perceived by the robot at this state \vec{x} .

2.1 Feature extraction for views

Both during map-building and self-localization, the omnidirectional image is transformed into a panoramic image that is partitioned into a fixed number n of non-overlapping sectors. The following criteria determined the selection of appropriate features to describe the omniview: a) To allow for an on-line localization, the calculation of the features should be as easy and efficient as possible. b) The feature set should include the orientation of the robot as prerequisite to estimate its heading direction. c) It should allow an easy generation of expected views for unknown poses of the robot with help of the map. Considering these criteria and the requirements of other omnivision-based localization approaches published recently, e.g. [11, 9, 5, 7], we decided to implement a very simple feature extraction method: for each segment of the panoramic image only the mean RGB-value is determined. This way, for each node in the graph just a small reference view (observation) $\underline{\mathbf{o}}^r(x, y, \varphi)$ has to be added. Thus an observation consists of n mean RGB-values. This vector represents the current appearance (the current view) of a position in the environment.

2.2 The localization algorithm

The prediction and correction update of the sample set of the MCL is achieved by a procedure often referred to as sequential importance sampling with resampling. In the prediction phase (robot motion), the sample set computed in the previous iteration (or during initialization) is moved according to the last movement of the robot $\underline{\mathbf{u}}_{t-1}$ (Fig. 2, left). Here, the motion model $p(\underline{\mathbf{x}}_t | \underline{\mathbf{x}}_{t-1}, \underline{\mathbf{u}}_{t-1})$ defines how the position of the samples changes using information $\underline{\mathbf{u}}_{t-1}$ from odometry. This way, MCL generates N new samples that approximate the expected density distribution of the robot's pose after the movement $\underline{\mathbf{u}}_{t-1}$. To determine the expected observations $\hat{\underline{\mathbf{o}}}_t^{(i)}$ (expected view in the state $\underline{\mathbf{x}}_t^{(i)}$) of the moved samples i , 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 views $\underline{\mathbf{o}}^r(x)$ of the two reference nodes closest to the respective sample position $\underline{\mathbf{x}}_t^{(i)}$ (Note, that here the expected view for sample $\underline{\mathbf{x}}_t^{(i)}$ can be incorrect or even wrong, if the environment has been changed after map building). After this, the resulting view is rotated according to the expected new orientation $\varphi_t^{(i)}$ of the sample $s^{(i)}$, utilizing a linear interpolation between the features of adjacent segments. This way, we obtain a set of N new views $\hat{\underline{\mathbf{o}}}_t^{(i)}(x, y, \varphi)$ describing the expected views of the moved samples in the new states $\underline{\mathbf{x}}_t^{(i)}$. In the update phase (new view), the actually perceived view $\underline{\mathbf{o}}_t$ at the new robot position has to be taken into account in order to correct the sample set S_t . For this, the importance weight $w_t^{(i)}$ 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. The weights are determined from the errors $E_t^{(i)}$ between the current view $\underline{\mathbf{o}}_t$ at the new robot position and the expected view $\hat{\underline{\mathbf{o}}}_t^{(i)}$ of each sample $s^{(i)}$ applying a camera-specific observation model $p(\underline{\mathbf{o}}_t | E_t)$ which returns the likelihood of an observation for a given error in a pose. Now $w_t^{(i)} = 1 - \alpha \cdot p(\underline{\mathbf{o}}_t | E_t^{(i)})$ 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 resampling from this weighted set. The resampling selects those samples with higher probability that have high importance weights. Samples with low weights are removed and randomly placed in the state-neighborhood of samples with high weights. Both phases are repeated recursively.

Obviously, comparison between actual and expected view is significant only if the environment remains relatively stable. The crucial point is to improve this comparison. The computation of the expected views with help of the map has take into account dynamic changes in the environment. Therefore, our idea is that reference views can represent more than one view (appearance) for a single position in the map. Then, for a position of a sample multiple expected views can be generated, which allow to improve the comparison. The next section describes an extended reference approach with

dynamic reference views which allow us to represent more than one view in a reference node in the map.

3 Extention of observations by dynamic views

Our aim is to find a way to respect dynamic changes of the environment in the map. Therefore, we have to change the structure of the map or of the reference view.

In a first very straightforward approach to adapting the map, one could simply replace an old reference view by a new one, provided that the robot is localized in the same state. But, in many cases, the appearance of a place in the home store is characterized by varying views, as for example, due to windows by daylight or by night or temporally placed objects in the operation area. If the reference map only contains the last view for a reference point, the comparison between current view and the expected one will deliver worse results. This is explained in the following example: First, the reference point represents \underline{o}_{t-2} (a window at daylight). Then, that view is replaced by \underline{o}_{t-1} (a window by night), because the environment was changed. \underline{o}_{t-2} and \underline{o}_{t-1} are quite different. The current view of the robot \underline{o}_t (now again the window by daylight) is compared to the reference view in the map that contains \underline{o}_{t-1} at this moment. In this case, the difference is greater than if the reference view respect both views \underline{o}_{t-2} and \underline{o}_{t-1} . To replace old information with new information isn't the right way. Therefore, we have to clarify what requirements a **dynamic reference view** – a reference view that represent multiple views – should fulfill.

- It should be able to integrate new observations for the same reference points in the reference view.
- Old observations shouldn't be removed immediately, maybe over time they could be of less importance.
- The comparison between reference views has to take into account all known appearances (saved observations) of the view. The similarity between views should be reflected by this fact.

To regard all these aspects, we look for known approaches that yield similar tasks. Three different methods were analyzed:

Bunch-Graph: The idea of the bunch-graph comes from a well-known face detection algorithm first introduced by Wiskott in 1995 [12]. Complex objects, like faces, are represented by labeled graphs. The nodes of the graph represent local features, like textures or orientations. In 1997 Wiskott [13] presented the bunch-graph for face detection. Here, the nodes represent a set of filtering results, so called *jets*. By that, each node can represent more than one appearance of a local part of an object. Unfortunately, we are not aware of a method for continuously updating the jets. That makes the bunch-graph concept unusable in our case.

FuzzyART: To solve the problem of insertion and removing of observations (jets) we tried to change the structure of the bunch-graph. The jets were replaced by the output layer (F2) of a FuzzyART network (Carpenter et al. [1]). FuzzyART allows a very fast and stable learning without the a passive forgetting. New observations are learned by the network, while old observations will not be forgotten. On the other hand this concept shows also a disadvantage: old useless observations cannot be removed or devalued over time.

Clustering and temporally fading histograms: A third idea for dynamic reference view stems from a concept of Wichert 1996 [10]. Kohonen maps and Growing Neural Gas were used to divide

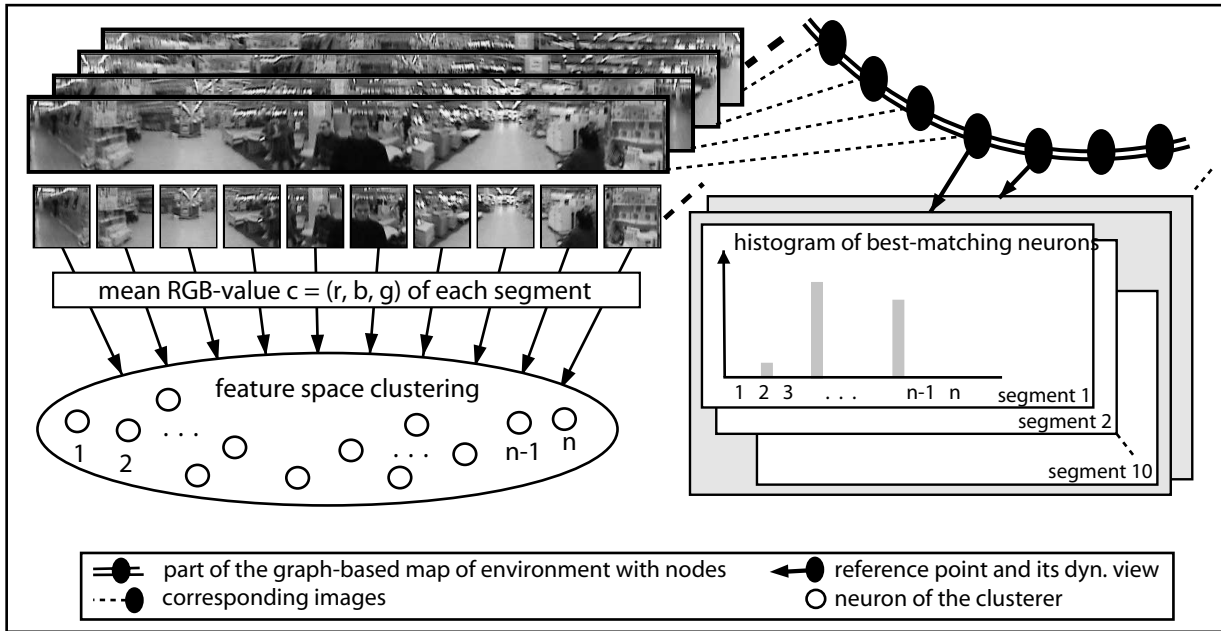


Figure 3. (Top right) Graph-based map of environment with nodes. Each node contains a dynamic reference view. The view will be created from the segmented camera image (segments). The histogram represents for each segment the frequency of occurrence of the mean RGB-values coded by the best-matching neuron of a neural clusterer. In this example, the image is partitioned into 10 segments and therefore each reference view consists of 10 histograms.

images (feature vectors of images) into patch classes and to classify them into classes. A clusterer computes a best-matching neuron (bm-neuron) for every local image patch. Now only the bm-neuron describes the local image. In our case, a reference point is described by a histogram with n bins (n = number of neurons of the clusterer) to count how often a neuron was best-matching or in other words how often a special view was percept at the reference point. The advantage is, that different views can be stored in the histogram without overwriting old ones. The histogram of a reference points describes all known appearances (and their frequency) that the robot has seen at this pose in the operation area. The histogram-based approach seems to have the greatest capability of these three concepts to deal with dynamic environments. Therefore we decided to further investigate the last idea.

3.1 Building up a dynamic view

The basic idea of our dynamic map of the environment is illustrated in Figure 3. A part of the graph (map of the environment) with nodes (reference views) is shown in the upper right corner of the figure. In section 2.1 we already explained that the omnidirectional image is divided in several segments. Now, for each segment a histogram is created. These histograms represent how often a segment-specific mean RGB-value was observed at the position of a reference node. To avoid multi-dimensional histograms, the mean RGB-values (three dimensions) were replaced by the best-matching neuron of a neural clusterer (one dimension). The clusterer divides the RGB input space and realizes a mapping onto scalar values. To get a problem-specific vector quantization, the clusterer was trained with the whole input data set (mean RGB-values of all segments and of all collected images). The images were acquired via several training tours through the operation area. The same data were used to built up the map (see section 2).

Two alternative cluster algorithms were implemented and experimentally analyzed: on the one hand the Neural Gas approach [6] and on the other hand a FuzzyART neural network [1] which was

extended by Scheidig in 2003 [8] (section 6.3). The second clusterer shows some advantages, as for example, size and overlap of the receptive fields can be controlled so that we can specify the number of neurons which influence the size of the histograms. Moreover, FuzzyART is able to learn input pattern even by one-shot learning. Therefore, our preferred clusterer is the FuzzyART. Typically, the omnidirectional image is divided into 10 segments s . For each segment s we calculate a histogram $\underline{h}(s)$ with N bins. N is also the number of neurons of the clustering network (see Fig. 3, bottom). The size of the network (value of N) depends on the vigilance parameter of the FuzzyART-network which determines the size of the clusters. In our experiments, a typical value of N was 28 (number of sub-categories represented by the clustering network).

A special case is the current view of the robot. This view, used in the probabilistic localization algorithm for comparisons between expected and actual views, does not consist of histograms. It is easy to see that the current view just contains one appearance (view). Therefore, for each segment of the current view only the bm-neuron is computed. This corresponds to an 1-of- N -coding. More details of the comparison between current and expected view, the important part of the localization algorithm (see section 2.2), are explained in section 3.3.

3.2 Initialization and update of the dynamic reference view

The initialization for a dynamic reference view is computed as follows. All histograms were filled with 0.0. The first observation for a reference point sets up the segment-specific histograms. The value of the i -th bin $h_i(s)$ of a segment s is computed as follows:

$$h_i(s) = \begin{cases} h_i(s) \cdot \gamma + d_i(s) \cdot (1.0 - \gamma) & \text{if } d_i(s) \geq h_i(s) \\ h_i(s) \cdot \beta & \text{else} \end{cases} \quad (1)$$

$d_i(s)$ is dependent on the actual mean RGB-Value \underline{c} and all neuron weights \underline{w}_i of the previously trained clusterer:

$$d_i(s) = e^{-\frac{d_i^2}{\sigma^2}} \quad d_i = \| \underline{c} - \underline{w}_i \| \quad (2)$$

In equation 2 we calculate a Gaussian activation for each bin $h_i(s)$ of the histogram based on the current input \underline{c} , the mean RGB-value of the segment s . This is necessary in order to compare current and dynamic reference views (see 3.3). The width of the Gaussian activation is controlled by the parameters σ . Thus, a mean RGB-value does not map to one bin only. In the case of initialization, $h_i(s)$ is equal to 0.0. This computation is repeated for each bin $h_i(s)$ and for each segment s of the dynamic reference view. Figure 3 shows that each segment has one or more characteristic mean RGB-values (bm-neuron respectively), and therefore, a characteristic histogram. During initialization, the parameters γ and β do not influence the histogram, because for comparison the histogram is normalized so that the maximum bin will be equal to 1.0.

The more interesting case is the adaptation of a dynamic reference view. Here, the parameters γ and β become important. They control the forgetting of old (β) and learning of new observations (γ). This is to be demonstrated by a short example: a reference point may already represent a view, and the robot perceives a new view at the same position of this reference point. Then, the previous histograms and the new histograms will be superimposed by equation 1. Now, the reference view represents more than one view (appearance). The parameter γ controls how fast new views will be learned and the parameter β how fast old views will be forgotten.

3.3 Comparison between current and dynamic reference view

The localization algorithm has to compare observations (views) to estimate the current pose of the robot (see section 2.2). For this, the robot compares its current view taken from the omnidirectional

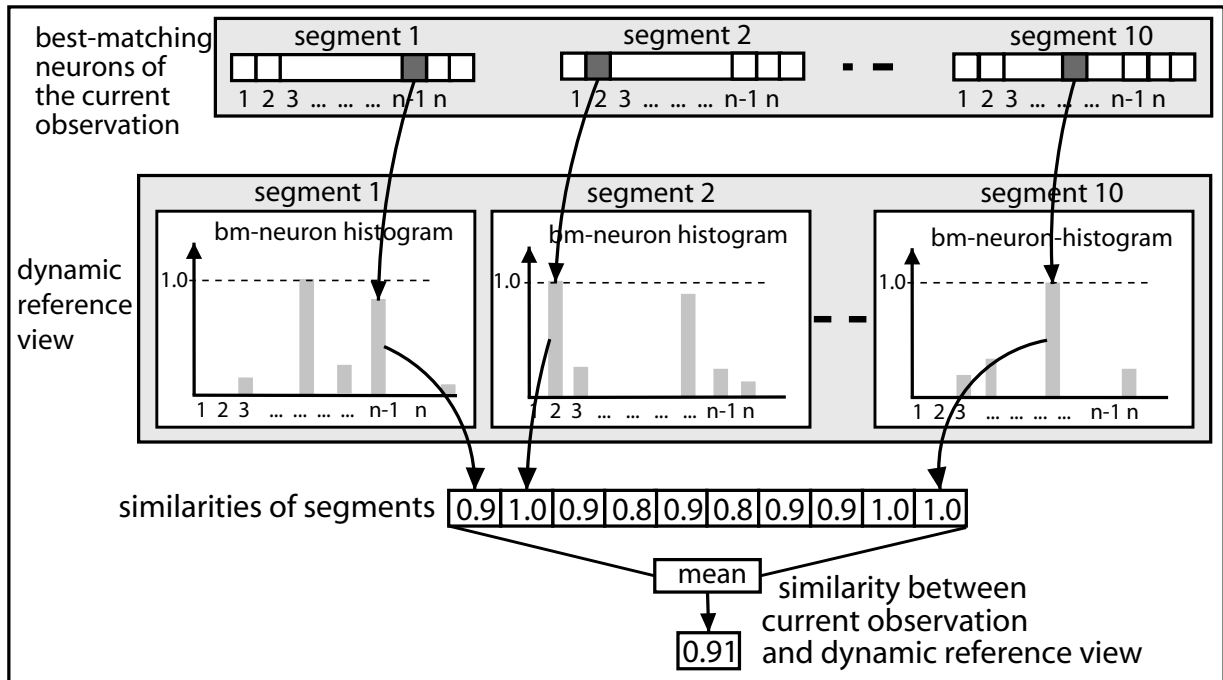


Figure 4. The similarity between current and expected reference view is separately computed for each segment. The bm -neurons i of the segments (current view) directly determine the final similarity by taking the value of the histogram bin i (expected reference view). The total likelihood of the view is calculated by the mean of the results of all segments.

camera with an expected view of a possible position taken from the map of the environment to determine how likely it is that the robot is in this position (an important part of the localization algorithm).

The comparison between the current view and our dynamic reference view is illustrated in Figure 4. The segments of the views are compared separately. At first, each histogram $\mathbf{h}(s)$ will be scaled for this comparison, so that the greatest value will be equal to 1.0. In the middle of figure 4 the segments and their scaled histograms are shown. A special view (see 3.1) is the current view of the robot that contains only one bm -neuron for one segment (Fig. 4, top). The bm -neurons i of the segments s (current view) directly determine the final similarity by taking the value of the histogram bin i (expected reference view). The mean value of all segments is coding the similarity between both views.

4 Experiments and results

To investigate our concept of dynamic views, we conducted two different experiments. The first experiment was carried out to show the capability of the reference views to integrate new observations without completely forgetting the previous observations ($\gamma = 0.5$, $\beta = 0.95$). For this, the robot was placed in a room with windows. It recorded a number of observations (50 pictures) over a period of 24 hours including the night. The lighting of the room were not used. Figure 5 (left) illustrates the result of the comparison between the current view and the continuously adapted dynamic reference view (version with forgetting $\Rightarrow \beta = 0.95$). We also calculated the similarity of the static view (the old view with one mean RGB-values per segment only, see section 2.1) with the current view, where the first image of the sequence was the reference view. It can be seen, that the course of the static reference view is strongly influenced by the dynamic changes of the environment. Our new dynamic reference view can deal with these changes much better.

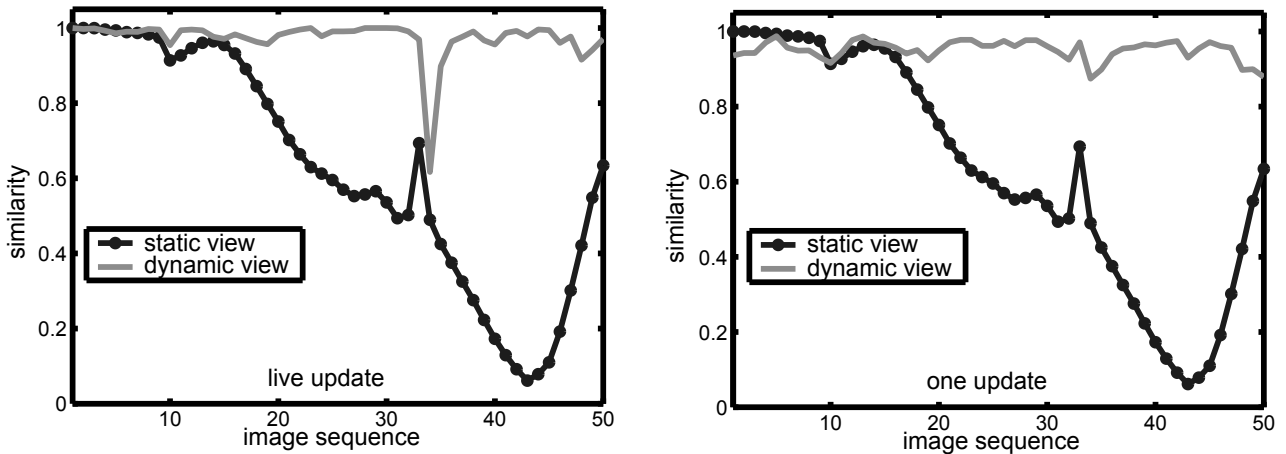


Figure 5. For the static (line with circles) we used the first image in the sequence as reference view to compute the similarities of the reference and the current view. The dynamic reference view (line without circles) was able to adapting its view with respect to the changing environment. (Left with forgetting) The dynamic view was continuously updated with the current observation based on equation 1. Thus, significantly higher similarities are obtained compared with the approach using a static view. (Right without forgetting) The dynamic view was constructed from all observation in advance, while no adaptation was carried out during operation. With this representation we obtain similar results like the dynamic view in (left).

Figure 5 (right) shows another experiment. Here, the dynamic reference view does not forget previous observations ($\beta = 1.0$, version without forgetting). The dynamic approach results high similarities reference view for almost all images. In contrast to the first experiment, where previous views were slowly forgotten (temporal lowpass filter $\beta = 0.95$), this experiment considers all views with the same weight. As expected, for both combinations of parameters we obtained better results with dynamic reference views than with the static view.

In another experiment, we simulated a very simple deterministic 1-of- n -localization. In an office environment six different positions were observed by the robot. Two kinds of observations were recorded, without and with changes of appearances. The local surroundings of the robot was occluded by a colored banner of 2 x 2 meter. The percentage of changes in the camera because of this occlusion was up to 30%. First, the robot built up a very simple map – for each position just one reference view. The static reference view uses the unchanged view only and the dynamic reference view was able to learn all views (changed and unchanged). An image (image with changes) was presented and the robot had to calculate the likelihood to be at one of the six positions. As result, a matrix of probabilities shown in Figure 6 was determined. The left matrix displays the localization results of the dynamic reference view. The cells contain the similarity between views taken at position i and position j (i, j = row (current) and column (reference)). Dark values mean high and bright values low likelihoods. In the case of $i = j$ observations from the same positions will be compared (same position not really the same appearances). Thus we see in the left figure very high values close to 1.0 at the main diagonal. Here the 1-of- n -localization works pretty good. At the right, the static view, the results are inferior to the dynamic view. Also the values of the likelihood for each cell are very similar. In contrast to the static view, the dynamic view shows a greater variance of the likelihood for the cells (see the scales in figure 6). Already in this simple experiment, some positions are hardly to distinguish by the static view and for more complex environment the localization will fail.

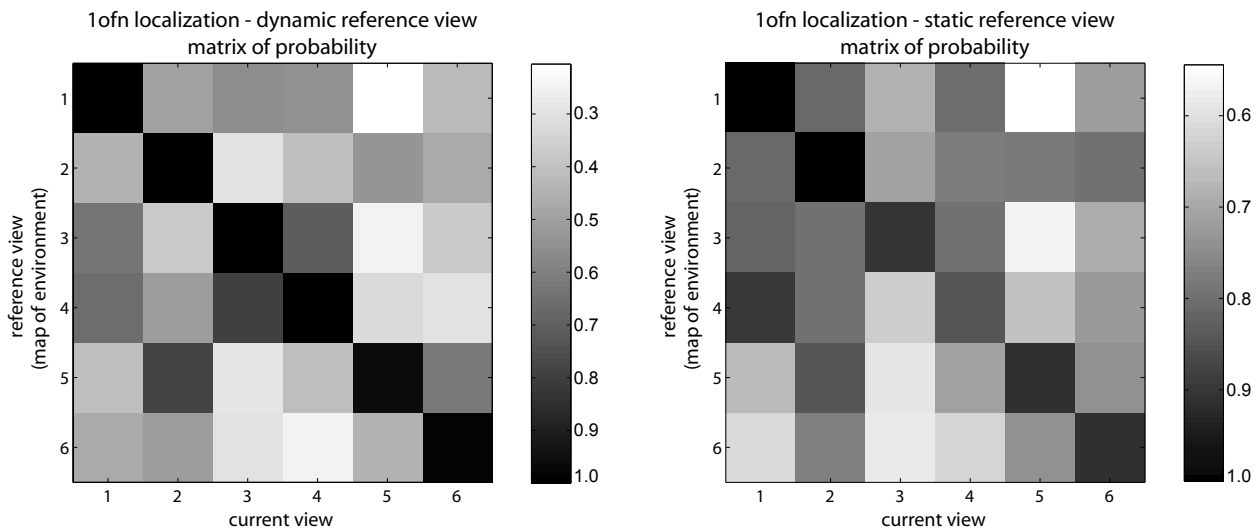


Figure 6. The results of a very simple deterministic localization experiment. (Left) Dynamic views can effectually distinguish the several positions. (Right) Localization results for static view are more ambiguous. With respect to ranges of likelihood values the position determination is clearly computed by the dynamic views as by the static view.

5 Conclusions and future work

This paper presents an early concept for online-adaptation of an appearance-based representation of the operation area for the mobile robot localization. The first results demonstrate the potential of our approach to represent changing appearances in a map. We were able to show that a dynamic reference view is able to represent the varying appearances of a whole day including the night. A simple 1-of- n -localization task with varying views for the same position was significantly improved by our approach. It can be expected, that this should also be valid for our probabilistic localization scheme [3]. As explained in the introduction, positions in the operation area can represent more than one appearances. For a reliable and correct localization, one has to consider this fact. Our adaptive view approach is a relatively simple but promising concept to build a dynamic map.

An important question still remains open: when can we update the appearance of a reference point or when can we be sure that the robot is at the correct reference point which needs to be updated? Therefore the robot has to be confident that it will update the exact reference point. A possibility approach to solve this localization problem would be to code the positions of the reference points not as deterministic, but as probabilistic values. However, this directly leads us to the problem of concurrent mapping and localization or SLAM.

References

- [1] A. Carpenter, S. Grossberg, and B. Rosen. Fuzzy art: Fast stable learning and categorization of analog patterns by an adaptive resonance system. *Neural Networks*, 4(6):759–771, 1991.
- [2] D. Fox, W. Burgard, F. Dellert, and S. Thrun. Monte carlo localization: Efficient position estimation for mobile robots. In *Proceedings of the AAAI, National Conference on Artificial Intelligence, AAAI*, Orlando, Florida, 1999.
- [3] H.-M. Gross, A. Koenig, H.-J. Boehme, and C. Schroeter. Vision-based monte carlo self-localization for a mobile service robot acting as shopping assistant in a home store. In *Proceedings of IROS'2002*,

IEEE/RSJ International Conference on Intelligent Robots and Systems, EPFL, Lausanne, Switzerland, pages 256–262, October 2002.

- [4] H.-M. Gross, A. Koenig, C. Schroeter, and H.-J. Boehme. Omnivision-based probabilistic self-localization for a mobile shopping assistant continued. In *Proceedings of IROS'2003, IEEE/RSJ International Conference on Intelligent Robots and Systems, Las Vegas*, 2003.
- [5] B. Kroese, N. Vlassis, B. R., and Y. Motomura. A probabilistic model for appearance-based robot localization. *Image and Vision Computing*, 19(6):381–391, 2001.
- [6] K. Martinetz, T. M. ; Schulten. A neural-gas network learns topologies. *Artificial Neural Networks*, pages 397–402, 1991.
- [7] L. Paletta, S. Frintrop, and J. Hertzberg. Robust localization using context in omnidirectional imaging. In *IEEE Int. Conf. on Robotics and Automation (ICRA 2001)*, pages 2072–2077, 2001.
- [8] A. Scheidig. *Sensomotorische Antizipation - ein neuer Zugang zur Verhaltensgenerierung*. PhD thesis, TU Ilmenau, 2003.
- [9] I. Ulrich and I. Nourbakhsh. Appearance-based place recognition for topological localization. In *IEEE Int. Conf. on Robotics and Automation (ICRA 2000)*, pages 1023–1029, 2000.
- [10] G. v. Wichert. Selforganizing visual perception for mobile robot navigation. In *1st Euromicro Workshop on Advanced Mobile Robots (EUROBOT), Kaiserslautern, Germany*, 1996.
- [11] N. Winters, J. Gaspar, G. Lacey, and J. Santos-Victor. Omnidirectional vision for robot navigation. In *IEEE Workshop on Omnidirectional Vision*, 2000.
- [12] L. Wiskott. *Labeled Graphs and Dynamic Link Matching for Face Recognition and Scene Analysis*. PhD thesis, Verlag Harri Deutsch, 1995.
- [13] L. Wiskott, J.-M. Fellous, N. Krueger, and C. von der Malsburg. Face recognition by elastic bunch graph matching. In *Proceedings of CAIP'97, 7th International Conference on Computer Analysis of Images and Patterns*, Christian-Albrechts-Universität zu Kiel, Germany, September 10-12 1997.