# An Adaptive Appearance-based Map for Long-Term Topological Localization of Mobile Robots

Feras Dayoub and Tom Duckett

Department of Computing and Informatics
University of Lincoln
LN6 7TS Lincoln, United Kingdom
{fdayoub,tduckett}@lincoln.ac.uk

***Abstract*— This work considers a mobile service robot which uses an appearance-based representation of its workplace as a map, where the current view and the map are used to estimate the current position in the environment. Due to the nature of real-world environments such as houses and offices, where the appearance keeps changing, the internal representation may become out of date after some time. To solve this problem the robot needs to be able to adapt its internal representation continually to the changes in the environment. This paper presents a method for creating an adaptive map for long-term appearance-based localization of a mobile robot using long-term and short-term memory concepts, with omni-directional vision as the external sensor.**

## I. INTRODUCTION

For a mobile robot to be able to work with people in their everyday environment it is essential to have the ability to localize itself using its internal representation of the environment. At the same time the robot needs to maintain this representation in response to the dynamics of the environment. Most of the work in mobile robot mapping considers only how to acquire the initial representation of the environment, but there has been very little work on how to update the map during long-term operation in changing environments.

The existing methods in mobile robot localization can generally be classified into two types: geometric localization, which aims to estimate and track the absolute position of a robot inside the map [12], and topological localization, which uses an appearance-based model of the environment [7]. In the latter approach, the map represents the environment as a graph where the nodes of this graph correspond to places in the real environment.

Topological localization has gained increasing attention in the last few years, especially the methods based on vision sensors. Recently a special type of camera, omnidirectional cameras, has become more popular. The omnidirectional camera with its $360^o$ field of view has various advantages over a standard camera. The robot can sense the whole surrounding environment in one snapshot regardless of its heading. Places can be recognized using fewer images and landmarks can be tracked over long distances.

Most of the existing work on visual localization and mapping assumes that the environment where the robot works is static (e.g., [4], [5], [15]). However, this assumption does not hold for many real environments. For example, the appearance of a room in a house is not static over time: new objects are sometimes added, existing objects like pictures or carpets may be changed or moved, and old objects may be removed. Some of these changes occur very often, such as moving chairs and cushions, etc. These transient variations need to be excluded from the long-term representation of the appearance of the environment.

In this work we use local features extracted from panoramic images to represent the appearance of a node in a topological map. Adopting concepts of short-term and long-term memory inspired by biological systems [1], our method updates the group of feature points for the reference image of a particular place. Thus the reference images are adapted to represent the information about the new appearance of the location (note that adaptation of the topology in the map is not considered in this work).

The rest of the paper is structured as follows. Section II discusses previous work on appearance-based mapping and localization. Section III describes our method for adaptive representation of the nodes in a topological map. Section IV presents the experiments and results obtained. Finally we draw conclusions and discuss future work in section V.

## II. BACKGROUND

An early approach for appearance-based mapping and localization was published in [15] where the operational area of the robot is represented as a graph. The nodes in this graph represent distinctive places and the edges represent the transitions between places. This approach consists of two stages: off-line and on-line. In the off-line stage the robot is driven through its operational area to learn a model of the environment, by taking a sequence of images in certain places and then creating a topological map from these images. In the on-line stage the robot uses the map to find its current position, i.e. the node which is most similar to the current view.

Many researches use various methods to create the map automatically without the need to label the images. Recently, Zivkovic et al. [18] formalized the topological mapping problem as an approximate solution for a graph cut problem.

Valgren et al. [16] defined the problem as incremental spectral clustering. Goedeme et al. [4] applied Dempster-Shafer probabilistic theory to topological map construction in environments with self-similarities.

During the localization stage the robot has to find the node with the most similar appearance to the current view. To solve this problem probabilistic methods such as Monte Carlo Localization [14] and Hidden Markov Models [6] can be used. These methods are based on a recursive Bayesian filter, which estimates the current position of the robot in the map given the observations. The probability of the current state $x_t$ given the sequence of observations $Z^t = \{z_1, ..., z_t\}$ up to time $t$ is

$$P(x_t|Z^t) = \frac{P(z_t|x_t)P(x_t|Z^{t-1})}{P(z_t|Z^{t-1})}, \quad (1)$$

where the sensor model $P(z_t|x_t)$ is calculated based on the similarity between the current view and the nodes in the map.

Two different approaches to measure the similarity between images have been presented in the literature: global and local methods. The global methods capture global properties of the image using approaches based on colour histograms [5], principal component analysis (PCA) [17], Fourier transform [10], etc. The local methods extract local properties from the image and produce a group of landmark features. Local feature including SIFT [8], SURF [2], MSER [9], etc., are used to find the similarity between images. The local methods have been shown to be more reliable and robust to illumination and viewpoint changes, thanks to the feature descriptors that are built using a local region around selected feature points. Each feature is described by a high-dimensional vector representation, which has high invariance to image translation, scaling and rotation, and partial invariance to illumination changes and affine projection.

Using local image feature descriptors, the similarity between two images can be measured by finding the correspondences between the features in the two images. This can be done by finding the closest feature in the feature descriptor space. However, the method can be time consuming if the number of images in the map is large and the search is done linearly. To speed up the matching process, a $K$d-Tree of the feature descriptors from all the images in the map could be used. Using a text retrieval approach, Sivic and Zisserman [13] presented a very fast retrieval system using a visual vocabulary. Nister and Stewenius [11] extended the ability of the system by using hierarchical K-means clustering to improve the performance, so that Fraundorfer et al. [3] were able to implement global localisation in real-time.

## III. The method

In the presented topological localization methods, the map could become out-of-date after some time in a changing environment. A naive solution to this problem would be simply to replace the image representation for each node in the topological map from time to time, in order to reflect the changed appearance of the corresponding location. Provided that the robot is correctly localized, this approach would enable the robot to remove out-of-date information from the map. However, it could also remove useful features due to temporary occlusions, and could lead to catastrophic results in the case of localization errors. A better solution would be to update the image representation of a node incrementally, by gradually adding information about new stable features in the environment, while removing information about features that no longer exist. In our approach, each node is represented by a group of features (using SURF in these experiments, though other features could be used). This group of features is updated over time by adding persistent new features and removing older ones that are no longer used.

The question here is how the system should choose which features to add and which features to remove from the stored image representation for a particular node? To answer this question we will adopt an information processing model (see Fig. 1) based on the multi-store model of human memory proposed by Atkinson and Shiffrin [1]. This model, which forms the basis of modern memory theories, divides human memory into three stores:

- sensory memory,
- short-term memory (STM),
- long-term memory (LTM).

The sensory memory contains information perceived by

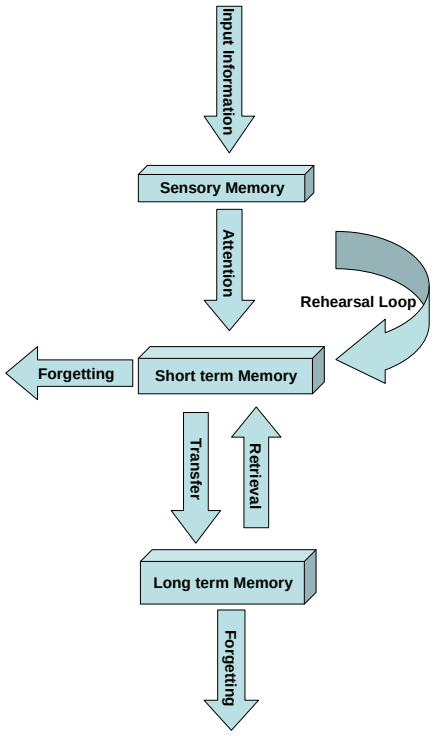


Fig. 1. The Information Processing Model.

the senses, and selective attention determines what information moves from sensory memory to short-term memory. Through the process of rehearsal, information in STM can be committed to LTM to be retained for longer periods of time. In return, the knowledge stored in LTM affects our perception of the world, and influences what information we attend to in the environment.

Applying these concepts to our approach for topological mapping, the sensory memory will contain the features extracted from the current image. Then an attentional mechanism selects which information to move to STM, which is used as an intermediate store where new observations are kept for a short time. Over this time the system uses a rehearsal mechanism to select features that are more stable for transfer to LTM. In order to limit the overall storage requirements and adapt to changes in the environment, the system also contains a recall mechanism that forgets unused feature points in LTM by removing these features from the node. LTM is used in turn by the attentional mechanism for selecting the new sensory information to update the map.

### *A. Recall, Rehearsal, Transfer*

We assume that an initial map of the whole environment has already been created by the robot, e.g. using an existing algorithm for topological mapping of static environments. (In this work we selected the places by hand.) We model the world as a set of discrete places. In our experiments, omni-directional vision is used to provide the features for localization and mapping. Each place has two memory stores: STM, which is a temporary stage, and LTM, which provides the reference views in the map used for self-localization. We assume that the robot is able to self-localize by matching features extracted from the current view to the stored reference views, though the self-localization does not need to be perfect (we measure the effects of noise and self-localization error in our experiments). The purpose of our algorithm presented here is to maintain up-to-date reference views for the nodes in the map, using recall and rehearsal concepts inspired by human memory.

To initialise the map, the image data from the robot's first tour of the environment is used. One panoramic image is selected to represent each node in the map. For each node, local features are extracted using the SURF algorithm [2], resulting in approximately 500 features per node in our experiments. These features are used directly to initialise LTM, while STM for each node is initially assigned to be empty.

**Algorithm 1** Update the reference view.

**Definitions:**
CrrNode: The reference view of the current node.
CrrView: The current view for the current node.
CrrSTM: The current STM for the current node.
STMlng: The maximum number of states in the STM.
LTMlng: The maximum number of states in the LTM.
newFP: The difference between the CrrView and CrrNode.

```
for (every visit to the node ) {
        newFP = recall( CrrNode , CurrView , LTMlng )
        rehearse( CrrSTM, newFP , STMlng )
  }
```

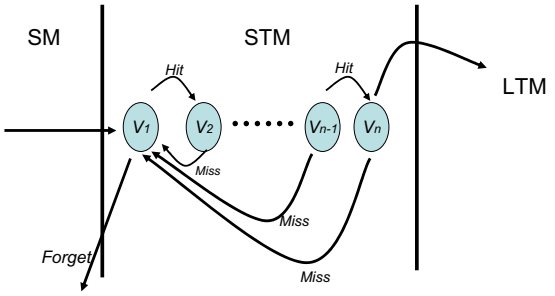


Fig. 2. The rehearsal stage in the STM.

**Algorithm 2** The rehearsal stage in the STM.

```
for (every feature in CrrSTM ){
    if (feature in newFP){
      Move the feature to the next state.
      if (feature state > STMlng){
        Move the feature to the CrrNode.
        Remove the feature from CrrSTM.
      }
      else if (feature in the first state){
            Remove the feature from CrrSTM.
      else
            Reset the feature to the first state.
      }
    }
}
for (every feature in newFP ){
    if (feature was not in CrrSTM){
      Add the feature to CrrSTM in the first state.
    }
}
```

Thereafter, every time the robot visits an existing node, the following steps are carried out. Feature points are extracted from the current view, using the SURF algorithm. Self-localization is carried out by comparing the current features to the reference features of each node (LTM) to estimate the current node. In our case, we apply global localization by place recognition, although any appropriate self-localization algorithm could be applied, e.g. Markov localization. After localization, the current features are used in the recall stage for updating the LTM of the current node. Only new features which do not match any feature in LTM are used in the rehearsal stage. Algorithm 1 describes the two main stages; (1) recall, where the difference in appearance between the reference and current views is computed, and (2) rehearsal, where this difference is used to update STM and commit persistent new features of the location to LTM.

Algorithm 2 shows the rehearsal process for a stored feature in STM, which is also represented as a finite state machine in Fig. 2. This stage represents what Atkinson and Schiffrin called rehearsal in their memory model (Fig. 1), i.e.

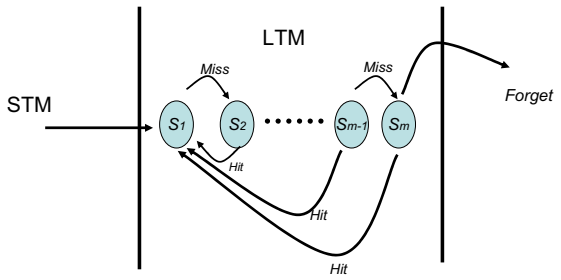


Fig. 3. The recall stage in the LTM.

**Algorithm 3** The recall stage in the LTM.

```
newFP = []
for (every feature in CrrNode){
    if (feature in CrrView){
      Rest the feature to the first state.
    else
      Move the feature to the next state.
    }
    if (feature state > LTMlng){
      Remove the feature from CrrNode.
    }
}
for (every feature in CrrView ){
    if (feature not in CrrNode){
      Add the feature to newFP.
    }
}
return newFP
```

the process of continually recalling information into the STM in order to memorise it. In order to transfer a feature point from STM to LTM the feature has to be seen frequently in that node. Features enter STM from sensory memory and must progress through several intermediate states ($V_1$ to $V_n$) before transfer to LTM. Every time the robot visits the node and finds the feature ("hit"), the state of the feature is moved closer to LTM. However if the feature is missing from the current view ("miss"), it is returned to the first state ($V_1$) or forgotten if it is already there. This policy means that spurious features should be quickly forgotten, while persistent features will be transferred to LTM.

Algorithm 3 shows the recall process for a stored feature in LTM, which is also represented as finite state machine in Fig. 3. This process first involves updating the LTM by matching the reference view to the current view. In order to remain in the LTM, a feature has to be seen occasionally in that node. In contrast to rehearsal, features enter LTM from STM and must progress through several intermediate states ($S_1$ to $S_m$) before being forgotten. Stored features which have been seen in the current view are reset to the first state ($S_1$), while the state of features which have not been seen is progressed, and a feature point that passes through all states without a "hit" is forgotten. Finally, recall returns the list of new features that were not already present in the LTM.

## IV. Experiments and results

To investigate our method of updating the reference views in a topological map, we conducted two experiments. In the first experiment we tested the system for a single node represented by a view of an office room. In the second experiment we used an image data set recorded over approximately 9 weeks from eight places in the students' restaurant of the University of Lincoln. Our experimental platform is an ActivMedia P3-AT robot equipped with a GigE progressive camera (Jai TMC-4100GE, 4.2 megapixels) with a curved mirror from 0-360.com. Using the camera with the mirror we obtain high-resolution omnidirectional images. The images in this shape have high order distortions which can affect the scale and rotation invariance of feature matching. To reduce these effects and to reduce the complexity of the required feature descriptor, the images are unwrapped to panoramic images using a simple transformation. The transformation of the output coordinates ($x_p$,$y_p$) to coordinates of the omni-directional image ($x_o$,$y_o$) can be written as

$$x_o = \cos\left(\frac{x_p}{2\pi R_O} + \textit{offset}\right) * (R_I + y_p) + center_X, \quad (2)$$

$$y_o = \sin\left(\frac{x_p}{2\pi R_O} + \textit{offset}\right) * (R_I + y_p) + center_Y, \quad (3)$$

where $R_O$, $R_I$ are radii of the outer and inner border of the omni-directional image. The parameters $center_X$ and $center_Y$ specify the circle center. The last parameter *offset* defines the origin of the panoramic image.

For local feature extraction we use the SURF algorithm. This algorithm extracts local features from the scale-space of the image based on the Hessian matrix and approximates the second order Gaussian derivatives with box filters. A fast non-maximum suppression algorithm is also used. The resulting algorithm has a good performance in the extraction process and a high accuracy. For more details, see [2].

After the extraction stage, the algorithm creates vector descriptors for the extracted features using information from the local surrounding area. This algorithm can create several types of descriptors. In our experiments we use the U-SURF descriptor (rotation invariance is removed) with descriptor length 64, taking into account that the robot is moving on a plane and that rotation invariance is not required for place recognition.

To find the similarity score between two groups of feature points, we use the number of corresponding features $M_{ij}$ between the two groups based on a nearest neighbour (NN) matching schema using the value 0.7 as a threshold between the nearest and second-nearest neighbour, following [2]. The similarity score between group $G_i$ and another group $G_j$ can be defined as

$$S_{ij} = \frac{M_{ij}}{K_i} * 100, \quad (4)$$

where $K_i$ is the number of features in $G_i$.

### A. Long-term update of a single node

To illustrate how the changing appearance of the environment affects the similarity score, we carried out an

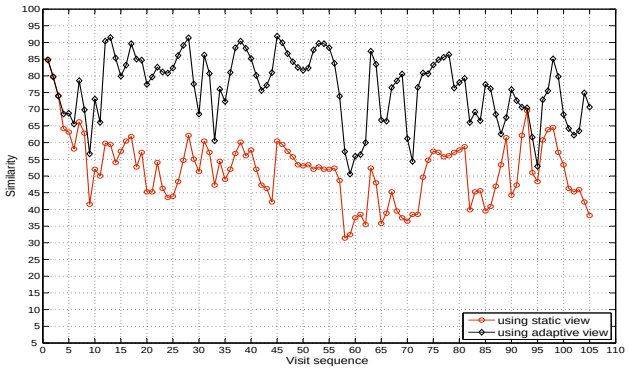


Fig. 4. The similarity score between the reference view and the current view of the node during 105 visits using the static and the adaptive reference view.

experiment where images were recorded over time at a single location. Objects in the environment were manipulated manually during this experiment. We made three types of changes. One is the temporary changes such a having a person standing in the node during the visit or moving chairs, etc. A second one is to add new objects to become part of the appearance of this node or to change the arrangement of some objects in the node. The third type is when we removed objects from the node permanently.

Fig. 4 shows how the similarity score between the reference view and the current view of the node changed during 105 visits to the node, using the static and the adaptive reference view. In this experiment the STM had 4 stages and the LTM had 5 stages. As we can see the similarity score is changing from visit to visit due to two factors. The first one is occlusion, which happens when a part of the image is blocked by a person near by, and the second factor is the changing appearance.

In the static reference view scenario, the two factors have an impact on the similarity score, which drops below 35% in some visits (e.g. visit 57). But when the adaptive reference view is used the effect of the second factor is reduced, which gives a high similarity score when the visit is occlusion free and a good similarity score in the cases when an occlusion happened.

### B. Long-term topological localization

In the second experiment we created a topological appearance map of the students' restaurant in the University of Lincoln by taking eight omni-directional images from eight different places to form the reference views for the nodes. This restaurant is used for various student activities and between these events the place is generally returned to its normal appearance by the restaurant staff but with some differences.

Over a period of approximately 9 weeks we visited the eight locations 18 times and recorded images for the places in the map. Fig. 5 shows two panoramic images for the same place recorded at different times. Using 144 images generated from these visits we tested our method for adapting

TABLE I

LONG-TERM TOPOLOGICAL LOCALIZATION TEST.

| Test | Correct global localization% Static Map | | Adaptive Map | |
|---|---|---|---|---|
| | Mean | Std | Mean | Std |
| No noise or occlusion | 95.83 | - | 98.61 | - |
| 50% occlusion, No noise | 93.42 | 1.25 | 98.41 | 0.83 |
| Added noise, No occlusion | 89.79 | 1.89 | 97.25 | 0.95 |
| 25% occlusion + noise | 88.02 | 1.89 | 96.21 | 1.73 |
| 50% occlusion + noise | 85.60 | 2.15 | 93.75 | 2.41 |

the reference views inside the map by using a Monte Carlo simulation technique. We used a global localization method based on place recognition using the similarity between the current and the reference views (winner-takes-all). Localization failures were an integral part of this experiment, i.e., in the case of incorrect place recognition the image representation for the wrong node would be updated. Monte Carlo simulation was used to simulate occlusion and added noise due to illumination changes, etc. in the current view. 100 trials were used for every test to evaluate the localization performance. We carried out five different tests using the restaurant dataset with 4 stages in the LTM and 2 stages in the STM. Results from these tests are illustrated in Table I.

In the first one we tested the localization performance for the 144 images without any simulated occlusion or added noise. In the second test we simulated occlusion by removing 50% of randomly chosen extracted features from the current view before each of the 144 localization attempts. In the third experiment we tested the localization performance with noise in the matching schema by adding Gaussian noise ($\mu$=0,$\sigma$=0.1) to the distance threshold between the nearest and second nearest neighbour. By adding this noise some of the true matches will be missed and some of the false matches will be counted. In the fourth experiment we combined the two factors: 25% of randomly chosen extracted features were removed from the current view before the localization stage then the Gaussian noise was also added into the matching schema. In the last one, 50% of randomly chosen extracted features were removed then the Gaussian noise was added.

The observed performance differences between static and adaptive mapping were tested using Student's t-test and shown to be statistically significant ($p < 0.01$). The results show that the adaptive map yields better localization performance due to its better representation of the real appearance of the environment. In this experiment, each node contained an average of $395.3 \pm 64.5$ features in LTM and $416.4 \pm 50.0$ features in STM, meaning that the approach should scale well to large environments.

## V. CONCLUSIONS AND FUTURE WORKS

This paper introduced a complimentary component for topological localization methods that use features extracted from images to represent the appearance of the nodes in the map. It updates the reference views of the nodes and tracks the changing appearance of the environment, while

Fig. 5. Two panoramic views from the same place at different times.

the robot is working over long periods of time. To achieve this we adopted short-term and long-term memory concepts to adapt the reference views of the nodes in response to the dynamics of the environment. To test our method, we conducted two experiments. The first one was in an office room where we manually changed the appearance of the place. For the second experiment we used data recorded from a real dynamic environment (a students' restaurant) over 9 weeks. In both experiments the method gave improved results over static mapping.

In this work, the number of the stages in LTM and STM were determined empirically based on the recorded sensor data. As a future work, the number of the stages could be learned depending on the dynamics of the real environment. The attention mechanism could be improved by adding real-time tracking of features in the scene to filter out spurious features due to noise or temporary occlusion. The adaptive capability of the map could be further extended to the topological level, by making the robot able to add or remove nodes and links from the map.

## REFERENCES


[1] R.C. Atkinson and R.M. Shiffrin. Human memory: A proposed system and its control processes. *In K.W. Spence & J.T. Spence (Eds.), The Psychology of Learning and Motivation*, 2:89–195, 1968.

[2] H. Bay, T. Tuytelaars, and L. Van Gool. SURF: Speeded up robust features. In *Proc. European Conference on Computer Vision (ECCV)*, 2006.

[3] C. Fraundorfer, F. Engels and D. Nister. Topological mapping, localization and navigation using image collections. In *Proc. IEEE International Conference on Intelligent Robots and Systems (IROS)*, 2007.

[4] T. Goedemé, M. Nuttin, T. Tuytelaars, and L. Van Gool. Omnidirectional Vision Based Topological Navigation. *International Journal of Computer Vision*, 74(3):219–236, 2007.

[5] H.M. Gross, A. Koenig, C. Schroeter, and H.J. Boehme. Omnivision-based probabilistic self-localization for a mobile shopping assistant continued. In *Proc. IEEE International Conference on Intelligent Robots and Systems (IROS)*, 2003.

[6] J. Kosecka and F. Li. Vision based topological Markov localization. In *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, 2004.

[7] B. Kuipers and Y. T. Byun. A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations. *Toward Learning Robots. MIT Press, Cambridge, Massachusetts*, page 4763, 1993.

[8] D.G. Lowe. Object recognition from local scale-invariant features. In *Proc. IEEE International Conference on Computer Vision (ICCV)*, 1999.

[9] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide-baseline stereo from maximally stable extremal regions. *Image and Vision Computing*, 22(10):761–767, 2004.

[10] E. Menegatti, M. Zoccarato, E. Pagello, and H. Ishiguro. Image-based Monte Carlo localisation with omnidirectional images. *Robotics and Autonomous Systems*, 48(1):17–30, 2004.

[11] D. Nister and H. Stewenius. Scalable recognition with a vocabulary tree. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2006.

[12] S. Se, D. Lowe, and J. Little. Mobile Robot Localization and Mapping with Uncertainty using Scale-Invariant Visual Landmarks. *The International Journal of Robotics Research*, 21(8):735, 2002.

[13] J. Sivic and A. Zisserman. Video Google: a text retrieval approach to object matching in videos. In *Proc. IEEE International Conference on Computer Vision (ICCV)*, 2003.

[14] H. Tamimi, H. Andreasson, A. Treptow, T. Duckett, and A. Zell. Localization of mobile robots with omnidirectional vision using particle filter and iterative sift. In *Proc. European Conference on Mobile Robots (ECMR)*, 2005.

[15] I. Ulrich and I. Nourbakhsh. Appearance-based place recognition for topological localization. In *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, 2000.

[16] C. Valgren, A. Lilienthal, and T. Duckett. Incremental Topological Mapping Using Omnidirectional Vision. In *Proc. IEEE International Conference on Intelligent Robots and Systems (IROS)*, 2006.

[17] N. Vlassis, B. Terwijn, and B. Krose. Auxiliary particle filter robot localization from high-dimensional sensor observations. In *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, 2002.

[18] Z. Zivkovic, O. Booij, and B. Kröse. From images to rooms. *Robotics and Autonomous Systems*, 55(5):411–418, 2007.