International Journal of Computer Vision 74(3), 219–236, 2007

DOI: 10.1007/s11263-006-0025-9

Omnidirectional Vision Based Topological Navigation

TOON GOEDEMÉ
ESAT - PSI - VISICS, University of Leuven, Belgium
toon.goedeme@esat.kuleuven.ac.be

MARNIX NUTTIN
PMA, University of Leuven, Belgium

TINNE TUYTELAARS
ESAT - PSI - VISICS, University of Leuven, Belgium

LUC VAN GOOL
ESAT - PSI - VISICS, University of Leuven, Belgium;
BIWI, ETH Zürich, Switzerland

Received October 12, 2005; Accepted December 11, 2006

First online version published in January, 2007

Abstract. In this work we present a novel system for autonomous mobile robot navigation. With only an omnidirectional camera as sensor, this system is able to build automatically and robustly accurate topologically organised environment maps of a complex, natural environment. It can localise itself using such a map at each moment, including both at startup (kidnapped robot) or using knowledge of former localisations. The topological nature of the map is similar to the intuitive maps humans use, is memory-efficient and enables fast and simple path planning towards a specified goal. We developed a real-time visual servoing technique to steer the system along the computed path.

A key technology making this all possible is the novel *fast wide baseline feature matching*, which yields an efficient description of the scene, with a focus on man-made environments.

Keywords: omnidirectional vision, topological maps, wide baseline matching, visual servoing

1. Introduction

1.1. Application

This paper describes a total navigation solution for mobile robots. It enables a mobile robot to efficiently localise itself and navigate in a large man-made environment, which can be indoor, outdoor or a combination of both. For instance, the inside of a house, an entire university campus or even a small city lie in the possibilities.

Traditionally, other sensors than cameras are used for robot navigation, like GPS and laser scanners. Because GPS (and Galileo also) needs a direct line of sight to the satellites (Pérer-Fontán et al., 2004), it cannot be used indoors or in narrow city centre streets, i.e. the very conditions we foresee in our application. Time-of-flight laser scanners are widely applicable, but are expensive and voluminous, even when the scanning field is restricted to a horizontal plane. The latter only yields a poor world representation, with the risk of not detecting essential obstacles such as table tops.

That is why we aim at a *vision-only* solution to navigation. Vision is, in comparison with these other sensors, much more informative. Moreover, cameras are quite compact and increasingly cheap. We observe also that many biological species, in particular migratory birds, use mainly their visual sensors for navigation. We chose to use an omnidirectional camera as visual sensor,

because of its wide field of view and thus rich information content of the images acquired with. For the time being, we added a range sensing device for obstacle detection, but this is to be replaced by an omnidirectional vision range estimator under development (Millnert et al., 2006).

Our method works with *natural* environments. That means that the environment does not have to be modified for navigation in any way. Indeed, adding artificial markers to every room in a house or to an entire city doesn't seem feasible nor desirable.

In contrast to classical navigation methods, we chose a *topological* representation of the environment, rather than a metrical one, because of its resemblance to the intuitive system humans use for navigation, its flexibility, wide usability, memory-efficiency and ease for map building and path planning.

The targeted application of this research is the visual guidance of electric wheelchairs for severely disabled people. In particular, the target group are people not able to give detailed steering commands to navigate around in their homes and local city neighbourhoods. If it is possible for them to perform complicated navigational tasks by only giving simple commands, their autonomy can be greatly enhanced. For most of them such an increase of mobility and independence from other people is very welcome.

Our test platform and camera are shown in Fig. 1. The price of such a robotic wheelchair is a serious issue. With our method, the only additional hardware required is a laptop (or an equivalent embedded processor), a webcam, a mirror and (for the time being) some ultrasound sensors. Because of the increased independence of the users the cost of personal helpers is reduced, making the robotic wheelchair even more economically feasible.

Figure 1. Left: the robotic wheelchair platform. Right: the omnidirectional camera, composed by a colour camera and an hyperbolic mirror.

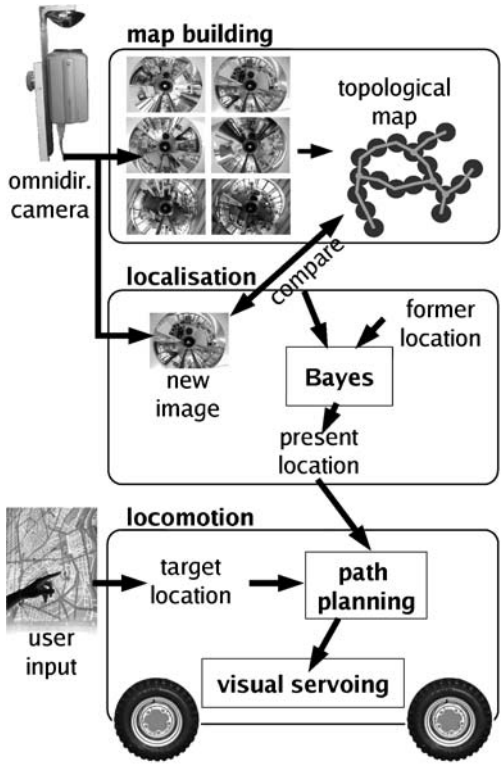

Figure 2. Overview of the navigation method.

1.2. Method Overview

An overview of the navigation method presented is given in Fig. 2. The system can be subdivided in three parts: map building, localisation and locomotion.

The *map building* stage has to be gone through only once, to train the system in a new environment. The mobile system is lead through all parts of the environment, while it takes images at a constant rate (in our set-up one per second). Later, this large set of omnidirectional images is automatically analysed and converted into a topological map of the environment, which is stored in the system's memory and will be used when the system is actually in use.

The next stage is *localisation*. When the system is powered up somewhere in the environment, it takes a new image with its camera. This image is rapidly compared with all the images in the environment map, and an hypothesis is formed about the present location of the mobile robot. This hypothesis is refined using Bayes' rule as soon as the robot starts to move and new images come in.

The final stage is *locomotion*. When the present location of the robot is known and a goal position is communicated by the user to the robot, a path can be planned towards that goal using the map. The planned route is specified as a sequence of map images, serving as a reference for what the robot should subsequently see if on course. This path is executed by means of a visual

servoing algorithm: each time a visual homing procedure is executed towards the location where the next path image is taken.

The main contributions of this paper are:

1. A fast wide baseline matching technique, which allows efficient, online comparison of images,
2. A method to construct a topological map, which is robust to self-similarities in the environment thanks to the use of Demster-Shafer evidence collection,
3. A visual servoing algorithm which is rubust to occlusions and tracking losses,
4. The integration of all these components in an operational system.

The remainder of this paper is organised as follows. The next section gives an overview of the related work. In Section 3, our core image analysis and matching technique is explained: fast wide baseline matching. The sections thereafter describe the different stages of our approach. Section 4 discusses the map building process, Section 5 explains the localisation method, Section 6 describes the path planning, and Section 7 details the visual servoing algorithm. We end with an overview of experimental results (Section 8) and a conclusion (Section 9).

2. Related Work

2.1. Image Comparison

A good image comparison method is of utmost importance in a vision-based navigation approach. Global methods compute a measure using all the pixels of the entire image. Although these methods are fast, they cannot cope with e.g. occlusions and severe viewpoint changes. On the other hand, techniques that work at a local scale, extracting and recognising *local features*, can be made robust to these effects. The traditional disadvantage of these local techniques is time complexity. In our approach, we combine novel global and local approaches resulting in fast and accurate image comparison.

2.1.1. Global Techniques. Many researchers use global image comparison techniques. Straightforward *global methods* like histogram-based matching, used by Ulrich and Nourbakhsh (2000) don't seem distinctive enough for our application. Stricker et al. (2001) proposed a method based on the Fourier-Mellin transform to compare images. Unfortunately, the baseline can not be large which restricts that method to tracking. Another popular technique is the use of an eigenspace decomposition of the training images (Jogan and Leonardis, 1999), which yields a compact database. However, these methods proved not useful in general situations because they are not robust enough against occlusions and illumination changes. That is why Jogan and Leonardis (1999) and Bischof et al. (2001) developed a PCA-based image comparison that is robust against partial occlusions, respectively varying illumination.

2.1.2. Local Techniques. A solution to be able to cope with partial occlusions is comparing local regions in the images. The big question is how to detect these local features, also known as *visual landmarks*.

A simple solution to do this is by adding artificial markers to strategically chosen places in the world. To make these features easily detectable with a normal camera, they are given special (individual) photometric appearances (for instance coloured patterns (Okuma et al., 2000), LEDs (Aliaga, 2001) or even 2D barcodes (Rekimoto et al., 2000)). Using such artificial markers is perfectly possible for some applications, but often difficult. Navigation through an entire city or inside someone's house are examples of cases where pasting these markers all over the place is hardly feasible and in no case desirable.

That is why, in this project we use *natural landmarks*, extracted from the scene itself, without modifications. Moreover, the extraction of these landmarks must be automatic and robust against changes in viewpoint and illumination to ensure the detection of these landmarks under as many circumstances as possible.

Many researchers proposed algorithms for natural landmark detection. Mostly, local regions are defined around interest points in the images. The characterisation of these local regions with descriptor vectors enables the regions to be compared across images. Differences between approaches lie in the way in which interest points, local image regions, and descriptor vectors are extracted. An early example is the work of Schmid et al. (1997), where geometric invariance was still under image rotations only. Scaling was handled by using circular regions of several sizes. Lowe (1999) extended these ideas to real scale-invariance. More general affine invariance has been achieved in the work of Tuytelaars et al. (1999), Tuytelaars and Van Gool (2000), Matas et al. (2002), and Mikolajczyk and Schmid (2002).

Although these methods are capable to find high quality correspondences, most of them are too slow to use in a real-time mobile robot algorithm. That is why we propose a much faster alternative, as explained in Section 3.

2.2. Map Structure

Many researchers proposed different ways to represent the environment perceived by vision sensors. We can order all possible map organisations by metrical detail: from dense 3D over sparse 3D to topological maps. We believe

that the outer topological end of this spectrum offers the top opportunities.

2.2.1. Dense 3D Maps. One approach is building dense 3D models out of the incoming visual data (Pollefeys et al., 2004; Nistér et al., 2004). Such approach has some disadvantages. It is computationally and memory demanding, and can not cope with planar and ill-textured parts of the environment such as walls. Nevertheless, these structures are omnipresent in our application, and collisions need to be avoided.

2.2.2. Sparse 3D Maps. One way to reduce the computational burden is to make abstraction of the visual data. Instead of modelling a dense 3D model containing billions of voxels, a sparse 3D model is built containing only special features, i.e. *visual landmarks*.

Examples of researchers solving the navigation problem with sparse 3D maps of natural landmarks are Se et al. (2001) and Davison (2003). They position natural features in a metrical frame, which is as big as the entire mapped environment. Although less than the dense 3D variant, these methods are still computationally demanding for large environments since their complexity is quadratic in the number of features in the model. Also, for larger models the metric error accumulates, so that feature positions are drifting away.

2.2.3. Topological maps. As a matter of fact, the need for explicit 3D maps in navigation is questionable. One step further in the abstraction of environment information is the introduction of topological maps. The psychological experiments of van Veen et al. (1998) show that people rely more on a topological map than a metrical one for their navigation. In these topological maps, locally places are described as a configuration of natural landmarks. These places form the nodes of the graph-like map, and are interconnected by traversable paths. Other researchers (Vale and Isabel Ribeiro, 2003; Ulrich and Nourbakhsh, 2000; Košecká and Yang, 2004) also chose for topological maps, mainly because they scale better to real-world applications than metrical, deterministic representations, given the complexity of unstructured environments. Other advantages are the ease of path planning in such a map and the absence of drift.

2.3. Topological Map Building

Vale and Isabel Ribeiro (2003) developed a clustering-based method for automatic building of a topological environment map out of a set of images. Unfortunately, his method is only suited for image comparison techniques which are a metric function (which doesn't hold for the similarity measure we use), and does not give correct results if *self-similarities* are present in the environment, i.e. places that are different but look similar.

Very popular are various *probabilistic* approaches of the topological map building problem. Ranganathan et al. (2005) for instance use Bayesian inference to find the topological structure that explains best a set of panoramic observations, while Shatkay and Kaelbling (1997) fit hidden Markov models to the data. If the state transition model of this HMM is extended with robot action data, the latter can be modeled using a partially observable Markov decision process or POMDP, as in Koenig and Simmons (1996) and Tapus and Siegwart (2005). Zivkovic et al. (2005) solve the map building problem using graph cuts.

In contrast to these global topology fitting approaches, an alternative way is detecting *loop closings*. During a ride through the environment, sensor data is recorded. Because it is known that the driven path is traversable, an initial topological representation consists of one long edge between start and end node. Now, extra links are created where a certain place is revisited, i.e. an equivalent sensor reading occurs twice in the sequence. This is called a loop closing. A correct topological map results if all loop closing links are added.

Also in loop closing, probabilistic methods are introduced to cope with the uncertainty of link hypotheses and avoid links at self-similarities. Chen and Wang (2005), for instance, use Bayesian inference. Beevers and Huang (2005) recently introduced Dempster-Shafer probability theory into loop closing, which has the advantage that ignorance can be modelled and no prior knowledge is needed. Their approach is promising, but limited to simple sensors and environments. In this paper, we present a new framework for loop closing using rich visual sensors in natural complex environments, which is also based on Dempster-Shafer.

2.4. Visual Servoing

As explained in Section 6, the execution of a path using such a topological environment map boils down to a series of visual servoing operations between places defined by images.

Cartwright and Collett (1987) proposed the so-called bearing-only 'snapshot' model, inspired by the visual homing behaviour of insects such as bees and ants. Their proposed algorithm consists of the construction of a home vector, computed as the average of landmark displacement vectors. Franz et al. (1998) analysed the computational foundations of this method and derived its error and convergence properties. They conclude that every visual homing method based solely on bearing angles of landmarks like this one, inevitably depends on basic assumptions such as equal landmark distances, isotropic landmark distribution or the availability of an external

compass reference. Unfortunately, because none of these assumptions generally hold in our targeted application we propose an alternative approach.

If both image dimensions are taken into account, not limiting the available information to the bearing angle, the most obvious choice is working via epipolar geometry estimation (e.g. Tuytelaars et al., 1999; Basri et al., 1998). Unfortunately, for perspective cameras this problem is in many cases ill conditioned, although Svoboda et al. (1998) proved that motion estimation with omnidirectional images is much better conditioned. That is why we chose a method based on omnidirectional epipolar geometry. Other work in this field is the research of Mariottini et al. (2005), who split the homing procedure in a rotation phase and a translation phase, but this approach can not be used in our application because of the non-smooth robot motion it produces.

3. Fast Wide Baseline Matching

The novel technique we use for image comparison is *fast wide baseline matching*. This key technique enables extraction of natural landmarks and image comparison for our map building, localisation and visual servoing algorithms.

We use a combination of two different kinds of wide baseline features, namely a rotation reduced and colour enhanced form of Lowe's *SIFT* features (Lowe, 1999), and the *invariant column segments* we developed (Goedemé et al., 2004). These techniques extract local regions in each image, and describe these regions with a vector of measures which are invariant to image deformations and illumination changes. Across different images, similar regions can be found by comparing these descriptors. This makes it possible to find correspondences between images taken from very different positions, or under different lighting conditions. The crux of the matter is that the extraction of these regions can be done beforehand on each image separately, rather than during the matching. Database images can be processed off-line, so that the images themselves do not have to be available at the time of matching with another image.

3.1. Camera Motion Constraint

The camera we use is a catadioptric system, consisting of an upward looking camera with a hyperboloidal mirror mounted above it. The result is a field of view of 360° in horizontal direction and more than 180° in vertical direction. The disadvantage is that these images contain severe distortions, as seen for instance in Fig. 5.

We presume the robot to move on one horizontal plane. The optical axis of the camera is oriented vertically. In other words, allowed movements consist of translations in the plane and rotation around a vertical axis, see also Fig. 3.

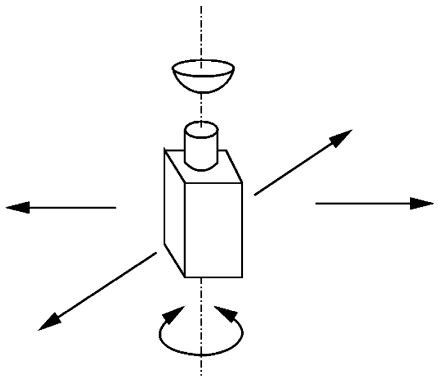

Figure 3. Illustration of the allowed movements of the camera.

3.2. Rotation Reduced and Colour Enhanced SIFT

David Lowe presented the *Scale Invariant Feature Transform* (Lowe, 1999), which finds interest points around local extrema in a scale-space of difference-of-Gaussian (DoG) images. The latter tend to correspond to blobs which contrast with their background. A dominant gradient orientation and scale factor define an image patch around each interest point so that a local image descriptor can be computed as a histogram of normalised gradient orientations. SIFT features are invariant to rotation and scaling, and robust to other transformations.

A reduced form of these SIFT features for use on mobile robots is proposed by Ledwich and Williams (2004). They used the fact that rotational invariance is not needed for a camera with a motion constraint as in Fig. 3. Elimination of the rotational normalisation and rotational part of the descriptor yields a somewhat less complex feature extraction and more robust feature matching performance.

Because the original SIFT algorithm works on greyscale images, some mismatches occur at similar objects in different colours. That is why we propose an outlier filtering stage using a colour descriptor of the feature patch based on global colour moments, introduced by Mindru et al. (1999), which is invariant to illumination changes (scale factor and offset in each band). We chose three colour descriptors: C_{RB}, C_{RG} and C_{GB}, with

$$C_{PQ} = \frac{\int P Q \, \mathrm{d}\Omega \int \mathrm{d}\Omega}{\int P \, \mathrm{d}\Omega \int Q \, \mathrm{d}\Omega}, \tag{1}$$

where $P, Q \in \{R, G, B\}$, i.e. the red, green, and blue colour bands, centralised around their means. After matching, the correspondences with Euclidean distance between the colour description vectors above a fixed threshold are discarded.

We tested these algorithms on the image pair in Fig. 5. With the original SIFT algorithm, the first 13 matches are correct. Using our rotation reduced and colour enhanced algorithm, we see that up to 25 correct matches are found without including erroneous ones.

3.3. Invariant Column Segments

In addition to the rotation reduced and color enhanced SIFT features, we developed wide baseline features which are specially suited for mobile robot navigation. There we exploited the special camera motion (Section 3.1) and the fact that man-made environments contain many vertical structures. Examples are walls, doors, and furniture. These don't have to be planar, so cylindrical elements like pillars do comply too. Vertical lines in the world always project to radial lines in the omnidirectional image for the constrained camera motions.

Here, these new wide baseline features are described for the use on omnidirectional images. More details and how they can be used on perspective camera images are described in Goedemé et al. (2004).

The extraction process of the wide baseline features starts as illustrated in Fig. 4. We stress that every step is invariant to changes in viewpoint and illumination. Along every line through the centre of the image (left), we look for points having a local maximum gradient value (centre). Every consecutive pair of gradient maxima along the line defines the begin and end of a new *invariant column segment* (right).

We characterise the extracted column segments with a descriptor that holds information about colour and intensity properties of the segment. This 10-element vector includes:

- Three *colour invariants*. To include colour information in the descriptor vector, we compute the colour invariants, based on generalised colour moments Eq. (1), over the column segment. To include information about the close neighbourhood of the segment, the line segment is expanded on both sides with a constant fraction of the segment length (in our experiments 0.2). Figure 4 (right) shows this.
- Seven *intensity invariants*. To characterise the intensity profile along the column segment, the best features to use are those obtained through the Karhunen-Lòeve transform (PCA). But because all the data is not known beforehand this is not practical. As is well known (Jain, 1989), the Fourier coefficients can sometimes offer a close approximation of the KL coefficients. In our method, because it is computationally less intensive and gives real output values, we choose to use the seven first coefficients of the *discrete cosine transform* (DCT), instead of Fourier. The DCT computations in our algorithm are executed fast using the 8-point 1D method described in Loeffler et al. (1989).

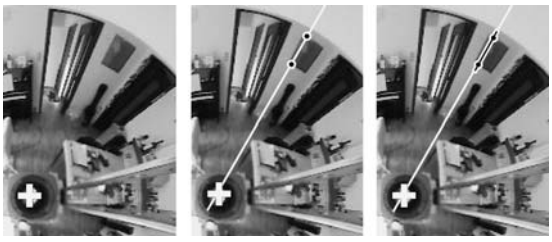

Figure 4. Illustration of the invariant column segment extraction algorithm: (left) part of the original image, the white cross identifies the projection centre, (centre) local maxima of the gradient for one radius, (right) one pair of maxima defines a column segment.

In many cases there are horizontally constant elements in the scene. This leads to many very resembling column segments next to each other. To avoid matching over and over again very similar line segments, we first do a clustering of the line segments in each image. As a clustering measure we use the Mahalanobis distance of the descriptor vectors, extended with the horizontal distance between the line segments. In each cluster a prototype segment is chosen for use in the matching procedure.

3.4. Matching

These two kinds of local wide baseline features are very suited to quickly find correspondences between two widely separated images. A correspondence pair is established if for a feature the other one is the closest to it in the feature space, and vice versa. Also, this match must be at least a fixed ratio better than the second best match. To be able to cope with different ranges of the elements of the descriptor vectors, distances are computed using the Mahalanobis measure (where we assume the elements to be independent):

$$d_{ij} = \sqrt{\sum_k \frac{(x_{ik} - x_{jk})^2}{\sigma_k^2}} \tag{2}$$

To speed up the matching, a Kd-tree of the reference image data is built. We used the on-line available package ANN (Approximate Nearest Neighbour) by Arya et al. (1998), for this.

Figure 5 shows the matching results on a pair of omnidirectional images. As seen in these examples, the SIFT features and the column segments are complementary, which pleads for the combined use of the two. The computing time required to extract features in two 320×240 images and find correspondences between them is about 800 ms for the enhanced SIFT features and only 300 ms

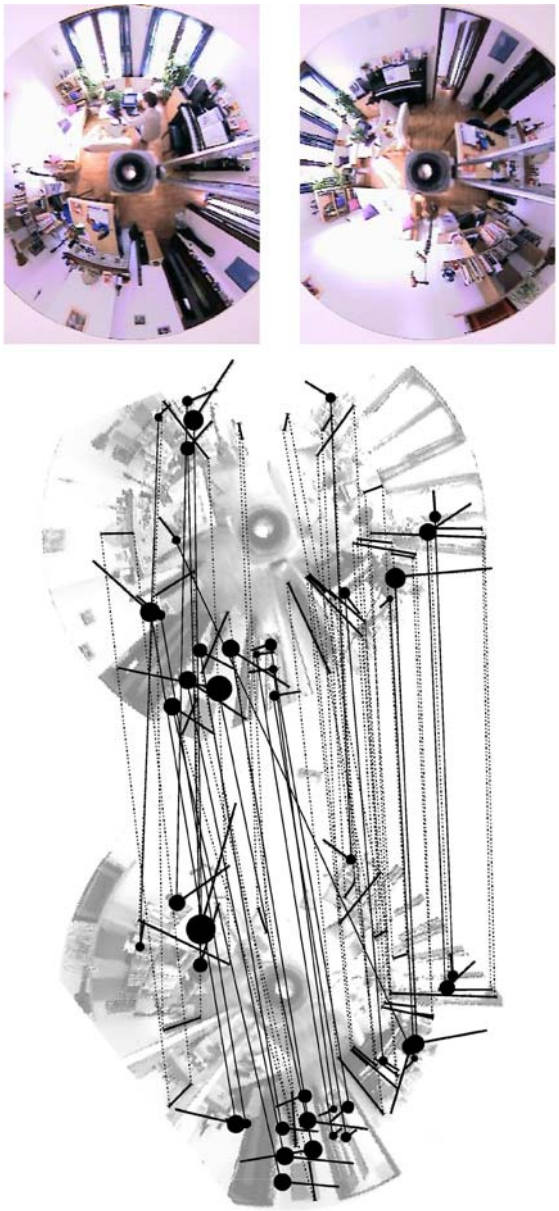

Figure 5. Top: a pair of omnidirectional images. Bottom: corresponding column segments (radial lines, matches indicated with dotted line) and SIFT features (circles with tail, matches with continuous line). The images are rotated for optimal visibility of the matches.

for the vertical column segments (on a 800 MHz laptop). Typically 30 to 50 correspondences are found.

For the description of a feature only the descriptors are used in the end, and not the underlying pixel data. As a result, the memory requirements for storing the reference images of entire environments can be kept limited.

4. Map Building

The navigation approach proposed is able to automatically construct a topological world representation out of a sequence of training images. During a training tour through the entire environment, omnidirectional images are taken at regular time intervals. The order of the training images is known. Section 4.1 describes the map structure targeted. In Section 4.2, the image comparison technique based on fast wide baseline features is described which is used by the actual map building algorithm, presented in Section 4.4. The mathematical formalism used in this algorithm, Dempster-Shafer probability theory, is briefly introduced in Section 4.3.

4.1. Topological Maps

To be of use in the following parts of the navigation method, the topological map must describe all 'places' in the environment and the possible connections between these places. The topology of the world, being the maze of streets in a city or the structure of a house, must be reflected in the world model. The question remains what exactly is meant with such a *place* and how to delimit it. In a building, a place can be defined as a separate room. But that is not such a good definition either. What to do with long corridors, and outdoors with city streets?

That is why we define a place with regard to the needs of the localisation and locomotion algorithms. To be able to get a sufficiently detailed localisation output, the sampling of places must be dense enough. For the locomotion algorithm, which performs visual homing between two places at a time, the distance between these places must be not too big to ensure errorless motion. On the other hand, a compact topological map with fewer places requires less memory and enables faster localisation and path planning.

We discuss the image comparison method used in the map building algorithm before deciding on this place definition, as this comparison will lie at its basis.

4.2. Image Comparison Measure

The main goal of this section is to determine for each arbitrary pair of images a certain similarity measure, which tells how visually similar the two images are. Our image comparison approach consists of two levels, a global and a local comparison of the images. We first compare two images with a coarse but fast global technique. After that, a relatively slower comparison with more precision based on local features only has to be carried out on the survivors of the first stage.

4.2.1. Global Colour Similarity Measure. To achieve a fast global image similarity measure between two images, we compute the same moments we used for the local features (Eq. (1)) over the entire image. These moments are invariant to illumination changes, i.e. offset and scal-

ing in each colour band. The Euclidean distance between two sets of these image colour descriptors gives a visual dissimilarity measure between two images.

With this dissimilarity measure, we can clearly see for instance the difference of images taken in different rooms. Because images taken in the same room but at different positions have approximately the same colour scheme, a second dissimilarity measure based on local features is needed to distinguish them.

4.2.2. Local Measure Based on Matches. First, we search for feature matches between the two images, using the techniques described in Section 3. The dissimilarity measure is taken to be inversely proportional to the *number of matches*, relative to the average number of features found in the images. Also the difference in relative configuration of the matches is taken into account. Therefore, we first compute a global angular alignment of the images by computing the average angle difference of the matches. The dissimilarity measure D_m is now also made proportional to the average angle difference of the features after this global alignment:

$$D_m = \frac{1}{N} \cdot \frac{n_1 + n_2}{2} \cdot \frac{\sum |\theta_i|}{N} \tag{3}$$

$$= \frac{(n_1 + n_2) \sum |\theta_i|}{2N^2}, \tag{4}$$

where N corresponds to the number of matches found, n_i the number of extracted features in image i, θ_i the angle difference for one match after global alignment.

4.2.3. Combined Dissimilarity Measure. We combine these two measures: only those pairs of images who have a colour dissimilarity under a predefined threshold are candidates for computing a matching dissimilarity.

This combined *visual* distance between two images is related to the physical distance between the corresponding viewpoints, but is certainly not a linear measure for it. As a matter of fact, the disparity and appearance difference of features is also related to the distance of the corresponding natural landmark to the cameras. Therefore, in large spaces (halls, market squares), a certain visual distance will be corresponding to a much larger physical distance, compared to the same visual distance between two images in a small space.

With this visual distance, the place definition problem in Section 4.1 can be addressed on the basis of a constant visual distance between places instead of a constant physical distance.

4.3. Introduction to Dempster-Shafer Theory

The proposed topological map building algorithm relies on Dempster-Shafer theory (Dempster, 1967; Shafer, 1976) to collect evidence for each loop closing hypothesis. Therefore, a brief overview of the central concepts of Dempster-Shafer theory is presented in this section.

Dempster-Shafer theory offers an alternative to traditional probabilistic theory for the mathematical representation of uncertainty. The significant innovation of this framework is that it makes a distinction between multiple types of uncertainty. Unlike traditional probability theory, Dempster-Shafer defines two types of uncertainty:

- *Aleatory Uncertainty*—the type of uncertainty which results from the fact that a system can behave in random ways (a.k.a. stochastic or objective uncertainty)
- *Epistemic Uncertainty*—the type of uncertainty which results from the lack of knowledge about a system (a.k.a. subjective uncertainty or ignorance)

This makes it a powerful technique to combine several sources of evidence to try to prove a certain hypothesis, where each of these sources can have a different amount of knowledge (*ignorance*) about the hypothesis. That is why Dempster-Shafer is typically used for sensor fusion.

For a certain problem, the set of mutually exclusive possibilities, called the *frame of discernment*, is denoted by Θ. For instance, for a single hypothesis H about an event this becomes $\Theta = \{H, \neg H\}$. For this set, traditional probability theory will define two probabilities $P(H)$ and $P(\neg H)$, with $P(H) + P(\neg H) = 1$. Dempster-Shafer's analogous quantities are called *basic probability assignments* or *masses*, which are defined on the *power set* of Θ: $2^\Theta = \{A \mid A \subseteq \Theta\}$. The *mass* $m : 2^\Theta \to [0, 1]$ is a function meeting the following conditions:

$$m(\emptyset) = 0 \quad \sum_{A \in 2^\Theta} m(A) = 1. \tag{5}$$

For a single hypothesis H, the power set becomes $2^\Theta = \{\emptyset, \{H\}, \{\neg H\}, \{H, \neg H\}\}$. A certain sensor or other information source can assign masses to each of the elements of 2^Θ. Because some sensors do not have knowledge about the event (e.g. it is out of the sensor's field-of-view), they can assign a certain fraction of their total mass to $m(\{H, \neg H\})$. This mass, called the *ignorance*, can be interpreted as the probability mass assigned to the outcome 'H OR $\neg H$', i.e. when the sensor does not know about the event, or is—to a certain degree—uncertain about the outcome. This means also that using this technique no prior probability function is needed, *no knowledge* can be expressed as total ignorance.

Sets of masses about the same power set, coming from different information sources can be combined together using *Dempster's rule of combination*:

$$m_1 \oplus m_2(C) = \frac{\sum_{A \cap B = C} m_1(A) m_2(B)}{1 - \sum_{A \cap B = \emptyset} m_1(A) m_2(B)} \tag{6}$$

This combination rule is useful to combine evidence coming from different sources into one set of masses. Because these masses can not be interpreted as classical probabilities, no conclusions about the hypothesis can be drawn from them directly. That is why two additional notions are defined, *support* and *plausibility*. They are computed as:

$$Spt(A) = \sum_{B \subseteq A} m(B) \qquad Pls(A) = \sum_{A \cap B \neq \emptyset} m(B) \quad (7)$$

These values define a confidence interval for the real probability of an outcome: $P(A) \in [Spt(A), Pls(A)]$. Indeed, due to the vagueness implied in having non-zero ignorance, the exact probability cannot be computed. But, decisions can be made based on the lower and upper bounds of this confidence interval.

4.4. Map Building Algorithm

We apply this mathematical theory on the topological map building problem posed. Out of a series of omnidirectional images, acquired at constant rate during a tour through the environment. Firstly, these images are clustered into *places*. Then, loop closing hypotheses are formulated between visually similar places of which evidence is collected using Dempster-Shafer theory. Once decisions are made about these hypotheses, the correct topology of the world is known.

This technique makes it possible to cope with *self-similar* environments. Places that look alike but are different will more likely get their link hypothesis rejected.

4.4.1. Image Clustering. The dots in the sketch figure of 6 denote places where images are taken. Because they were taken at constant time intervals and the robot did not drive at a constant speed, they are not evenly spread. We perform agglomerative clustering with complete linkage based on the combined visual distance (see Section 4.2.3) on all the images, yielding the ellipse shaped clusters in Fig. 6. The black line shows the exploration path as driven by the robot.

4.4.2. Hypothesis Formulation. As can be seen in the lower part of Fig. 6, not all image groups nicely cover one distinct place. This is due to *self-similarities*, or distinct places in the environment that are different but look alike and thus yield a small visual distance between them.

For each of the clusters, we can define one or more subclusters. Images within one cluster which are linked by exploration path connections are grouped together. For each of these *subclusters* a prototype image is chosen as the *medoid*[1] based on the visual distance, denoted as a star in the figure.

For each pair of these subclusters within the same cluster, we define a *loop closing hypothesis* H, which states

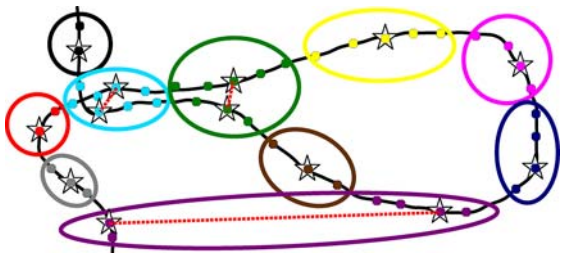

Figure 6. Example for the image clustering and hypothesis formulation algorithms. Dots are image positions, black is exploration path, clusters are visualised with ellipses, prototypes of (sub)clusters with a star. Hypotheses are denoted by a dotted red line.

that if $H = true$, the two subclusters describe the same physical place and must be merged together. We will use Dempster-Shafer theory to collect evidence about each of these hypotheses.

4.4.3. Dempster-Shafer Evidence Collection. For each of the hypotheses defined in the previous step, a decision must be made if it was correct or wrong. Figure 7 illustrates four possibilities for one hypothesis. We observe that a hypothesis has more chance to be true if there are more hypotheses in the neighbourhood, like in case *a* and *b*. If no neighbouring hypotheses are present (*c*, *d*), no more evidence can be found and no decision can be made based on this data.

We conclude that for a certain hypothesis, a neighbouring hypothesis adds evidence to it. It is clear that, the further away this neighbour is from the hypothesis, the less certain the given evidence is. We chose to model this subjective uncertainty by means of the ignorance notion in Dempster-Shafer theory. That is why we define an *ignorance function* ξ containing the distance between two hypotheses H_a and H_b:

$$\xi(H_a, H_b) = \begin{cases} 1 - \sin\left(\dfrac{d_H(H_a, H_b)\pi}{2d_{th}}\right) & (d_H \leq d_{th}) \\ 0 & (d_H > d_{th}) \end{cases} \quad (8)$$

where d_{th} is a distance threshold and $d_H(H_a, H_b)$ is the sum of the distances between the two pairs of prototypes of both hypotheses, measured in number of exploration images.

To gather *aleatory* evidence, we look at the visual similarity of both subcluster prototypes, normalised by the

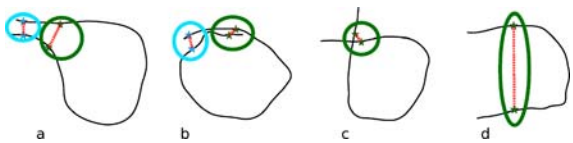

Figure 7. Four topological possibilities for one hypothesis.

standard deviation of the intra-subcluster visual similarities. The visual similarity s_V is the inverse of the visual distance, defined in Eq. (3).

Each neighbouring hypothesis H_b yields the following set of Dempster-Shafer masses, to be combined with the masses of the hypothesis H_a itself:

$$\begin{aligned} m(\{\emptyset\}) &= 0 \\ m(\{H_a\}) &= s_V(H_b)\xi(H_a, H_b) \\ m(\{\neg H_a\}) &= (1 - s_V(H_b))\xi(H_a, H_b) \\ m(\{H_a, \neg H_a\}) &= 1 - \xi(H_a, H_b) \end{aligned} \tag{9}$$

Hypothesis masses are initialised with the visual similarity of its subcluster prototypes and an initial ignorance value (0.25 in our experiments), which models its influenceability by neighbours.

4.5. Hypothesis Decision

After combination of each hypothesis's mass set with the evidence given by neighbouring hypotheses (up to a maximum distance d_{th}), a decision must be made if this hypothesis was correct and thus if the subclusters must be united into one place or not.

Unfortunately, as stated above, only positive evidence can be collected, because we can not gather more information about totally isolated hypotheses (like c and d in Fig. 7). This is not too bad, because of different reasons. Firstly, the chance for correct, but isolated hypothesis (case c) is low in typical cases. Also, adding erroneous loop closings (c and d) will yield an incorrect topological map, whereas leaving them out will keep the map useful for navigation, but a bit less complete. Of course, new data about these places can be aqcuired later, during navigation.

It is important to remind oneself that the computed Dempster-Shafer masses can not directly be interpreted as probabilities. That is why we use Eq. (7) to compute the support and plausibility of each hypothesis after evidence collection. Because these values define a confidence interval for the real probability, a hypothesis can be accepted if the lower bound (the support) is greater than a threshold.

After this decision, a final topological map can be built. Subclusters connected with accepted hypotheses are merged into one place, and a new medoid is computed as prototype of it. For hypotheses that are not accepted, two distinct places should be constructed.

5. Localisation

When the system has learnt a topological map of an environment, this map can be used for a variety of navigational tasks, firstly *localisation*. For each arbitrary new position in the known environment, the system can find out *where* it is. The output of this localisation algorithm is a *location*, which is—opposed to other methods like GPS—not expressed as a metric coordinate, but as one of the topological places defined earlier in the formerly explained map building stage.

The training set doesn't need to cover every imaginable position in the environment, in contrast to Kröse et al. (2003). A relatively sparse coverage is sufficient to localise every possible position. That is because the image comparison method we developed is based on wide baseline techniques and hence can recognise scene items from substantially different viewpoints.

Actually, two localisation modes exist. When starting up the system, there is no *a priori* information on the location. Every location is equally probable. This is called *global localisation*, alias the *kidnapped robot problem*. Traditionally, this is known to be a hard problem in robot localisation. In contrast, if there is knowledge about a former localisation not too long ago, the locations in the proximity of that former location have a higher probability than others further away. This is called *location updating*.

We propose a system that is able to cope with both localisation modes. A probabilistic approach is taken. Instead of making a hard decision about the location, a probability value is given to each location at each time instant. The Bayesian approach we follow is explained in the next subsection.

5.1. Bayesian Filtering

Define $x \in X$ a place of the topological map. Z is the collection of all omnidirectional images z, so that $z(x)$ corresponds to the training observation at place x. At a certain time instant t, the system acquires a new image z_t. The goal of the localisation algorithm is to reveal the place x_t where this image was taken.

We define the *Belief function* $Bel(x, t)$ as the probability of being at place x at time t, given all previous observations. So,

$$Bel(x, t) = P(x_t \mid z_t, z_{t-1}, \ldots, z_0), \tag{10}$$

for all $x \in X$. In the *kidnapped robot* case, there is no knowledge about previous observations hence $Bel(x, t_0)$ is initialised equal for all x.

Using Bayes' rule, we find:

$$Bel(x, t) = \frac{P(z_t \mid x_t, z_{t-1}, \ldots, z_0) P(x_t \mid z_{t-1}, \ldots, z_0)}{P(z_t \mid z_{t-1}, \ldots, z_0)}. \tag{11}$$

Because the denominator of this fraction is not dependent on x, we replace it by the normalising constant

η. If we know the current location of the system, we assume that future locations do not depend on past locations. This property is called the *Markov Assumption*: $P(z_t \mid x_t, z_{t-1}, \ldots, z_0) = P(z_t|x_t)$. Using it, together with the probabilistic sum rule, Eq. (11) yields:

$$Bel(x,t) = \eta\ P(z_t|x_t) \sum_{x_{t-1} \in X} [P(x_t|x_{t-1}) Bel(x, t-1)] \tag{12}$$

This allows us to calculate the belief recursively based on two variables: the *next state density* or *motion model* $P(x_t \mid x_{t-1})$ and the *sensor model* $P(z_t \mid x_t)$.

5.2. Motion Model

The motion model $P(x_t \mid x_{t-1})$ explicits the probability of a transition from one place x_{t-1} to another x_t. Is seems logical to assume that a transition in the time instant $(t-1) \rightarrow t$ between places that are far from each other is less probable than between places close to each other. We model this effect with a Gaussian:

$$P(x_t \mid x_{t-1}) = \frac{1}{\beta_x} e^{-dist(x_{t-1}, x_t)/{\sigma_x}^2}. \tag{13}$$

In this equation, the function $dist(x_1, x_2)$ corresponds to a measurement of the distance between the two places. We approximate it as the minimum number of place transitions needed to go from x_1 to x_2 on the topological map, computed with the Dijkstra algorithm (Dijkstra, 1959). In Eq. (13), β_x is a normalisation constant, and ${\sigma_x}^2$ is the variance of the distances, measured on the map data. Once the topological map is known, the complete motion model can be computed off-line for usage during localisation.

5.3. Sensor Model

The entity $P(z_t \mid x_t)$, called the sensor model, is the probability of acquiring a certain observation z_t if the location x_t is known. This is related to the visual dissimilarity of that observation z_t and the training observation $z(x_t)$ at location x_t. The probability of acquiring an image at a certain place that differs much from the training image taken at that place has a low probability. We model this sensor model also by a Gaussian:

$$P(z_t \mid x_t) = \frac{1}{\beta_z} e^{-diss(z_t, z(x_t))/\sigma_z}. \tag{14}$$

This time, the function $diss(z_1, z_2)$ refers to the visual dissimilarity explained in Section 4.2.3. σ_z is the standard deviation of it, measured on the data (or for large maps a local subset of the data) and β_z is a normalisation constant.

Unlike the motion model, the sensor model cannot be computed beforehand. It depends on the newly incoming query image data. Every location update step the visual dissimilarities of the query image with many database images must be computed. This validates our efforts to make the computation of the visual dissimilarity measure as fast as possible.

6. Path Planning

With the method of the previous section, at each time instant the most probable location of the robot can be found, from which a path to a goal can be determined. How the user of the system, for instance the wheelchair patient, gives the instruction to go towards a certain goal is highly dependent on the situation. For every disabled person, for instance, an individual interface must be designed adapted to his/her possibilities.

We assume a certain goal is expressed as a certain place of the topological map, e.g. as a voice command *≪Kitchen!≫*. From the present pose, computed by the localisation algorithm, a path can be easily found towards it using Dijkstra's algorithm (Dijkstra, 1959). This path is expressed as a series of topological places which are traversed.

7. Visual Servoing

The algorithm described in this section makes the robot move along a path, computed by the previous section. Such a path is given as a sparse set of prototype images of places. The physical distance between two consecutive path images is variable (1 to 5 metres in our tests), but the visual distance is constant, such that there are enough local feature matches as needed by this algorithm.

It is easy to see that following such a sparse visual path boils down to a succession of *visual homing* operations. First, the robot is driven towards the place where the first image on the path is taken. When arrived, it is driven towards the next path image, and so on. Because a smooth path is desired for the application, the motion must be continuous without stops at path image positions.

We tackle this problem by estimating locally the spatial structure of the wide baseline features using epipolar geometry. Hence, at this point we bring in some 3D information. This may seem at odds with our topological approach, but the depth maps are very sparse and only calculated locally so that errors are kept local, don't suffer from error build-up, and are efficient to compute.

Figure 8 offers an overview of the proposed method. Each of the *visual homing* operations is performed in two phases, an initialisation phase (Section 7.1) and an iterated motion (Section 7.2) phase.

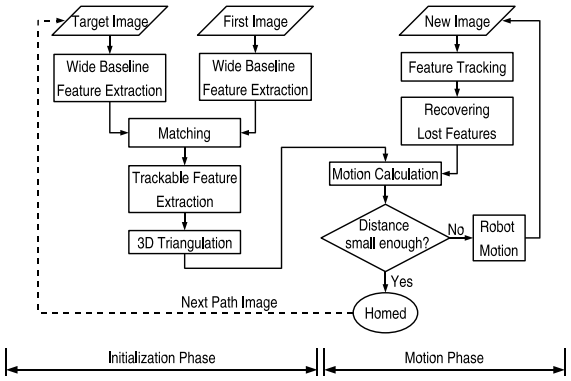

Figure 8. Flowchart of the proposed algorithm for visual servoing along a path.

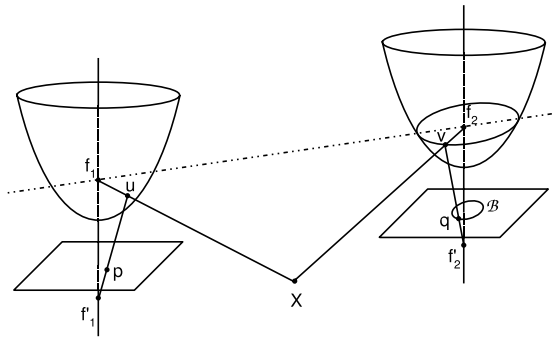

Figure 9. Projection model for a pair of omnidirectional images.

7.1. *Initialisation Phase*

From each position within the reach of the next path image (the *target* image), a visual homing procedure can be started. Our approach first establishes wide baseline local feature correspondences between the present and the target image, as described in Section 3. That information is used to compute the epipolar geometry, which enables us to construct a local map containing the feature world positions, and to compute the initial homing vector.

7.1.1. Epipolar Geometry Estimation. Our calibrated single-viewpoint omnidirectional camera is composed of a hyperbolic mirror and a perspective camera. As imaging model, we use the model proposed by Svoboda et al. (1998) (which is less general, but less complicated than the one by Geyer and Daniilidis (2003)). This enables the computation of the epipolar geometry based on 8 point correspondences. In Svoboda, Svoboda describes a way to robustly estimate the *essential matrix* E, when there are outliers in the correspondence set. The essential matrix is the equivalent of the fundamental matrix in the case of known internal camera calibration, $F = K^{-T} E K^{-1}$.

Svoboda's so-called *generate-and-select* algorithm to estimate E is based on repeatedly solving an overdetermined system built from the correspondences that have a low *'outlierness'* and evaluating the *quality measure* of the resulting essential matrix. Because our tests with this method did not yield satisfactory results, we implemented an alternative method based on the well-known Random Sample Consensus (RANSAC (Fischler and Bolles, 1981)) paradigm.

The set-up is sketched in Fig. 9. One visual feature with world coordinates $\mathbf{X}$ is projected via point $\mathbf{u}$ on the first mirror to point $\mathbf{p}$ in the image plane of the first camera. In the second camera, the mirror point is called $\mathbf{v}$ and the image plane point $\mathbf{q}$. For each of the correspondences, the mirror points $\mathbf{u}$ and $\mathbf{v}$ can be computed as

$$\mathbf{u} = \mathcal{F}(K^{-1}\mathbf{p})K^{-1}\mathbf{p} + \mathbf{t}_C, \tag{15}$$

with $\mathbf{t}_C = [0, 0, -2e]^T$ and

$$\mathcal{F}(\mathbf{x}) = \frac{b^2(ex_1 + a\|\mathbf{x}\|)}{b^2x_1^2 - a^2x_2^2 - a^2x_3^2}. \tag{16}$$

In these equations K is the internal calibration matrix of the camera, and a, b and e are the parameters of the hyperbolic mirror, with $e = \sqrt{a^2 + b^2}$.

If E is the *essential matrix*, for all correspondences $\mathbf{v}^T E\mathbf{u} = 0$. This yields for each correspondence pair one linear equation in the coefficients of $E = [e_{ij}]$.

For each random sample of 8 correspondences, an E matrix can be calculated. This is repeatedly done and for each E matrix candidate the inliers are counted. A correspondence is regarded an inlier if the second image point $\mathbf{q}$ lies within a predefined distance from the epipolar *ellipse*, defined by the first image point $\mathbf{q}$. This epipolar ellipse $\mathcal{B}$ with equation $\mathbf{x}^T B\mathbf{x} = 0$ is computed with $B =$

$$\begin{bmatrix} -4t^2a^2e^2 + r^2b^4 & rsb^4 & rtb^2(-2e^2 + b^2) \\ rsb^4 & -4t^2a^2e^2 + s^2b^4 & stb^2(-2e^2 + b^2) \\ rtb^2(-2e^2 + b^2) & stb^2(-2e^2 + b^2) & t^2b^4 \end{bmatrix} \tag{17}$$

and $[r, s, t] = E\mathbf{u} = E(\mathcal{F}(K^{-1}\mathbf{p})K^{-1}\mathbf{p} + \mathbf{t}_C)$. Fortunately, this ellipse becomes a circle when the motion is in one plane, so that the distance from a point to this shape is easy to compute.

From the one essential matrix E with the maximal number of inliers the motion between the cameras can be computed using the SVD based method proposed by Hartley (1992). If more than one E-matrix is found with the same maximum number of inliers, the one is chosen with the best (i.e. smallest) quality measure $q_E = \sigma_1 - \sigma_2$, where σ_i is the ith singular value of the matrix E.

Out of this relative camera motion, a first estimate of the homing vector is derived. During the motion phase this homing vector is refined.

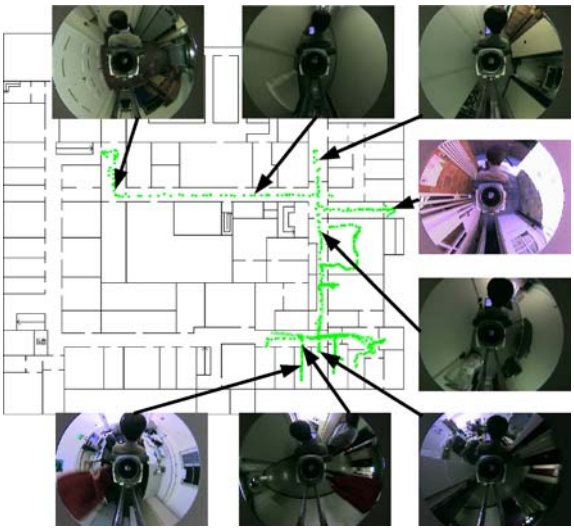

Figure 10. A map of the test environment with image positions (• for a database image and × for a test image) and some of the images.

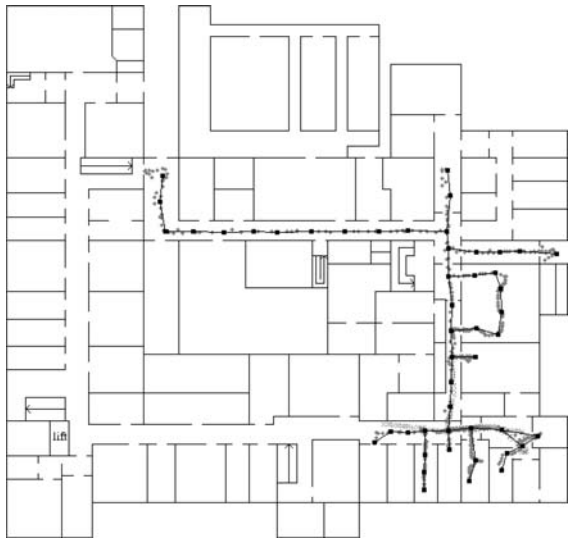

Figure 12. The resulting topological map: locations of the place prototypes with interconnections.

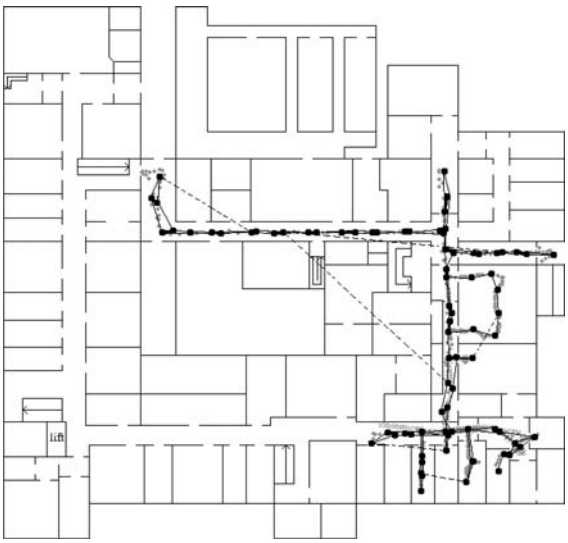

Figure 11. Topological loop closing: accepted hypotheses are shown in thick black lines, rejected in dashed thin black lines.

7.1.2. Local Feature Map Estimation. In order to start up the succession of tracking iterations, an estimate of the local map must be made. In our approach the local feature map contains the 3D world positions of the visual features, centred at the starting position of the visual homing operation. These 3D positions are easily computed by triangulation.

We only use two images, the first and the target image, for this triangulation. This has two reasons. Firstly, these two have the widest baseline and therefore triangulation is best conditioned. Our wide baseline matches between these two images are also more plentiful and less influenced by noise than the tracked features.

7.2. Motion Phase

Then, the robot is put into motion in the direction of the homing vector and an image sequence is recorded. We rely on lower-level collision detection, obstacle avoidance and trajectory planning algorithms to drive safely (Demeester et al., 2003; Nuttin et al., 2003). In each new incoming image the visual features are tracked. Robustness to tracking errors (caused by e.g. occlusions) is achieved by reprojecting lost features from their 3D positions back into the image. These tracking results enable the calculation of the present location and from that the homing vector towards which the robot is steered.

When the (relative) distance to the target is small enough, the entire homing procedure is repeated with the next image on the sparse visual path as target. If the path ends, the robot is stopped at a position close to the position where the last path image was taken. This yields a smooth trajectory along a sparsely defined visual path.

7.2.1. Feature Tracking. The corresponding features found between the first image and the target image in the previous step, also have to be found in the incoming images during driving. This can be done very reliably performing every time wide baseline matching with the first or target image, or both. Although our methods are relatively fast this is still too time-consuming for a driving robot.

Because the incoming images are part of a smooth continuous sequence, a better solution is *tracking*. In the

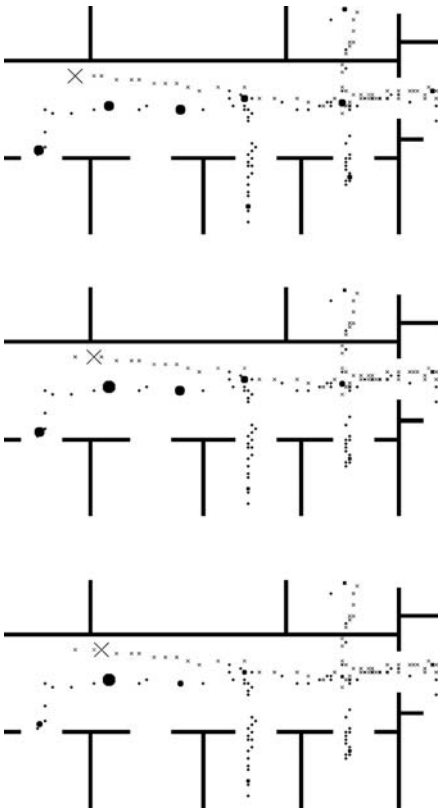

Figure 13. Three belief update cycles in a typical localisation experiment. The black × denotes the location of the new image. Place prototypes with a higher belief value are visualised as larger black circles.

image sequence, visual features move only a little from one image to the next, which enables to find the new feature position in a small search space.

A widely used tracker is the KLT tracker of Kanade Lucas, Shi and Tomasi (1994). KLT starts by identifying interest points (corners), which then are tracked in a series of images. The basic principle of KLT is that the definition of corners to be tracked is exactly the one that guarantees optimal tracking. A point is selected if the matrix

$$\begin{bmatrix} g_x^2 & g_x g_y \\ g_x g_y & g_y^2 \end{bmatrix}, \tag{18}$$

containing the partial derivatives g_x and g_y of the image intensity function over an $N \times N$ neighbourhood, has large eigenvalues. Tracking is then based on a Newton-Raphson style minimisation procedure using a purely translational model. This algorithm works surprisingly fast: we were able to track 100 feature points at 10 frames per second in 320 × 240 images on a 1 GHz laptop.

Because the well trackable points are not necessarily coinciding with the anchor points of the wide baseline features to be tracked, the best trackable point in a small window around such an anchor point is selected. In the assumption of local planarity we can always find back the corresponding point in the target image via the relative reference system offered by the wide baseline feature.

7.2.2. Recovering Lost Features. The main advantage of working with this calibrated system is that we can recover features that were lost during tracking. This avoids the problem of losing all features by the end of the homing manoeuvre, a weakness of our previous approach (Goedemé et al., 2005). This feature recovery technique is inspired by the work of Davison (2003), but is faster because we do not work with probability ellipses.

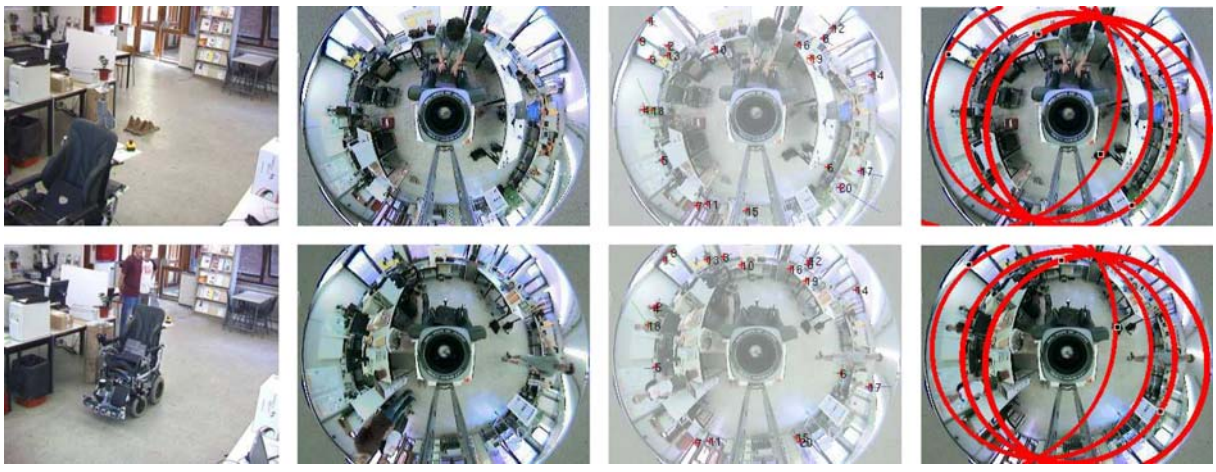

Figure 14. Results of the initialisation phase. Top row: target, bottom row: start. From left to right, the robot position, omnidirectional image, visual correspondences and epipolar geometry are shown.

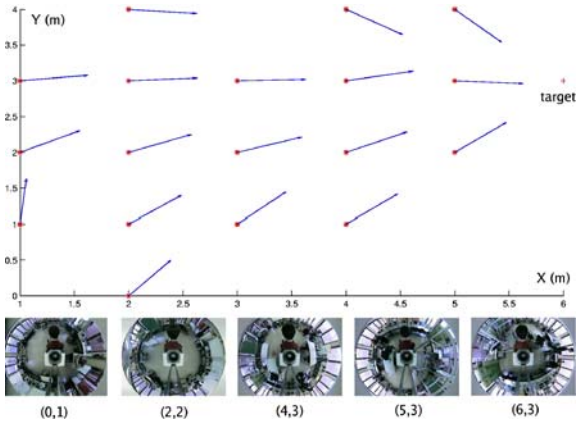

Figure 15. Homing vectors from 1-meter-grid positions and some of the images.

In the initialisation phase, all features are described by a local intensity histogram, so that they can be recognised after being lost during tracking. Each time a feature is successfully tracked, this histogram is updated.

When tracking, some features are lost due to invisibility because of e.g. occlusion. Because our local map contains the 3D positions of each feature, and the last robot position in that map is known, we can reproject the 3D feature in the image. Svoboda shows that the world point $\mathbf{X}_C$ (i.e. the point $\mathbf{X}$ expressed in the camera reference frame) is projected on point $\mathbf{p}$ in the image:

$$\mathbf{p} = \frac{K}{2e}(\lambda \mathbf{X}_C - \mathbf{t}_C), \tag{19}$$

wherein λ is the largest solution of

$$\lambda = \frac{b^2(-e)\mathbf{X}_{C3} \pm a\|\mathbf{X}_C\|}{b^2\mathbf{X}_{C_3}^2 - a^2\mathbf{X}_{C_1}^2 - a^2\mathbf{X}_{C_2}^2}. \tag{20}$$

Based on the histogram descriptor, all trackable features in a window around the reprojected point $\mathbf{p}$ are compared to the original feature. When the histogram distance is under a fixed threshold, the feature is found back and tracked further in the next steps.

7.2.3. Motion Computation. When in a new image the feature positions are computed by tracking or backprojection, the camera position (and thus the robot position) in the general coordinate system can be found based on these measurements.

It is shown that the position of a camera can be computed when for three points the 3D positions and the image coordinates are known. This problem is know as the *three point perspective pose estimation problem*. An overview of the proposed algorithms to solve it is given by Haralick et al. (1994). We chose the method of Grunert, and adapted it for our omnidirectional case.

Also in this part of the algorithm we use RANSAC to obtain a robust estimation of the camera position. Repeatedly the inliers belonging to the motion computed on a three-point sample are counted, and the motion with the greatest number of inliers is kept.

7.2.4. Robot Motion. In Section 7.1.1, it is explained how the position and orientation of the target can be extracted from the computed epipolar geometry. Together with the present pose results of the last subsection, a homing vector can easily be computed. This command is communicated to the locomotion subsystem. When the homing is towards the last image in a path, also the relative distance and the target orientation w.r.t. the present orientation is given, so that the locomotion subsystem can steer the robot to stop at the desired position. This is needed for e.g. docking at a table.

8. Experiments

8.1. Test Platform

We have implemented the proposed algorithm, using our modified electric wheelchair 'Sharioto'. A picture of it is shown in the left of Fig. 1. It is a standard electric wheelchair that has been equipped with an omnidirectional vision sensor (consisting of a Sony firewire colour camera and a Neovision hyperbolic mirror, right in Fig. 1). The image processing is performed on a 1 GHz laptop. As additional sensors for obstacle detection, 16 ultrasound sensors and a Lidar are added. A second laptop with a 840 MHz processor reads these sensors, receives visual homing vector commands, computes the necessary manoeuvres, and drives the motors via a CAN-bus. More information on this platform can also be found in Nuttin et al. (2001) and Demeester et al. (2003).

8.2. Map Building

The wheelchair was guided around in a large environment, while taking images. The environment was a large part of our office floor, containing both indoor and outdoor locations. This experiment yielded a database of 545 colour images with a resolution of 320×240 pixels. The total distance travelled was approximately 450 m. During a second run 123 images were recorded to test the localisation. A map and some of these images are shown in Fig. 10.

After place clustering with a fixed place size threshold (in our experiments 0.5), this resulted in a set of 53 clusters. Using the Dempster-Shafer based evidence collection, 6 of 41 link hypotheses were rejected, as shown in Fig. 11. Figure 12 shows the resulting 59 place prototypes along with the accepted interconnections.

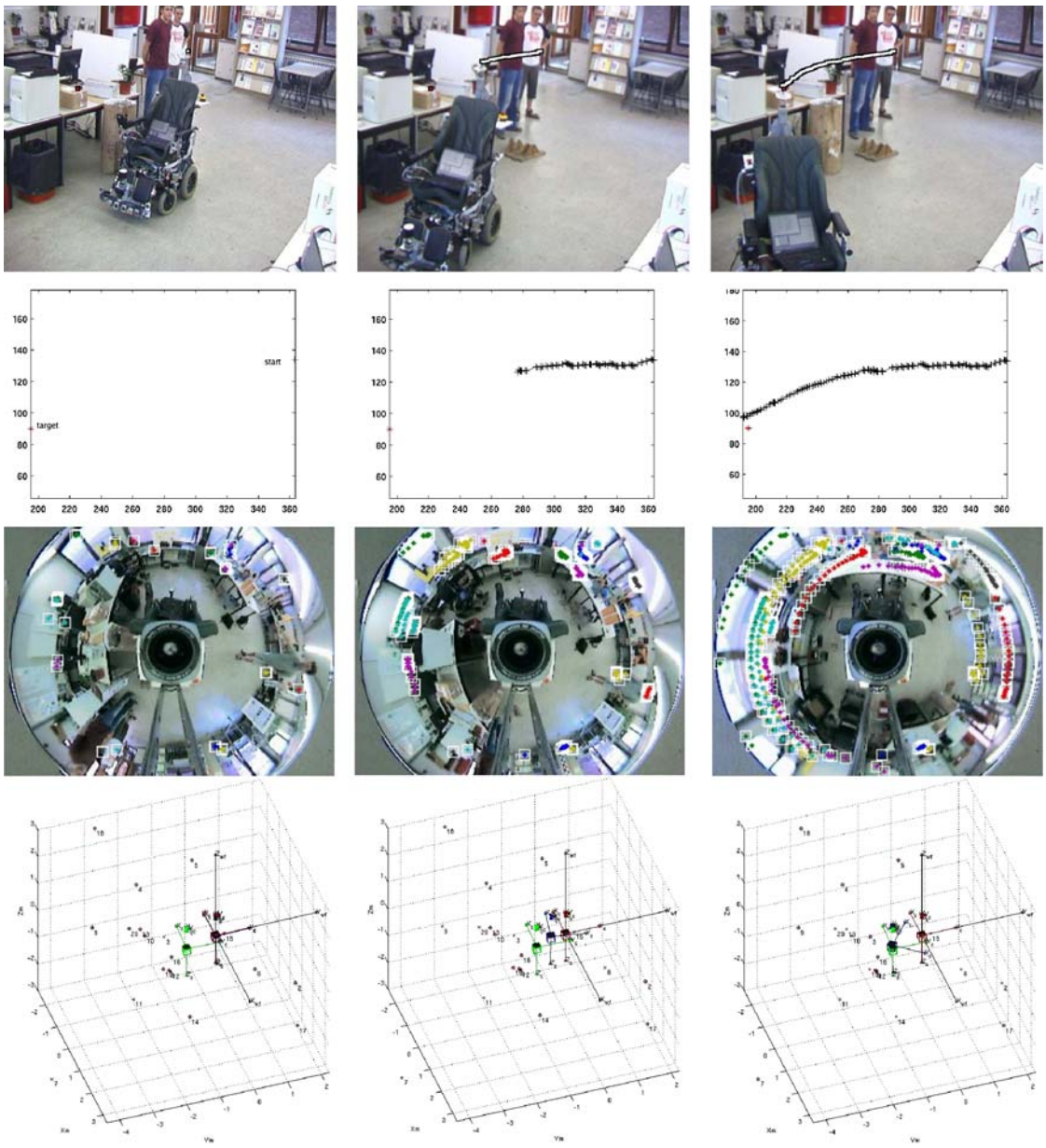

Figure 16. Three snapshots during the motion phase: in the beginning (left), half (centre) and at the end (right) of the homing motion. The first row shows the external camera image with tracked robot position. The second row shows the computed world robot positions [cm]. The third row shows the colour-coded feature tracks. The bottom row shows the sparse 3D feature map (encircled features are not lost).

Instead of keeping all the images in memory, the database is now reduced to only the descriptor sets of each prototype image. In our experiment, the memory needed for the database was reduced from 275 MB to 1.68 MB.

8.3. Localisation

From this map, the motion model is computed offline as explained in Section 5.2. Now, for the separate test set, the accuracy of the localisation algorithm is tested. A typical experiment is illustrated in Fig. 13.

In total, for 78% of the trials the maximum of the belief function was located at the closest place at the first iteration, after the second and third belief update this percentage raised to 89% and 97%.

8.4. Visual Servoing

8.4.1. Initialisation Phase. During the initialisation phase of one visual homing step, correspondences between the present and target image are found and the

epipolar geometry is computed. This is shown in Fig. 14.

To test the correctness of the initial homing vector, we took images with the robot positioned at a grid with a cell size of 1 meter. The resulting homing vectors towards one of these images (taken at (6, 3)) are shown in Fig. 15. This proves the fact that even if the images are situated more than 6 metres apart the algorithm works thanks to the use of *wide baseline* correspondences.

8.4.2. Motion Phase. We present a typical experiment in Fig. 16. During the motion, the top of the camera system was tracked in a video sequence from a fixed camera. This video sequence, along with the homography computed from some images taken with the robot at reference positions, permits calculation of metrical robot position ground truth data.

Repeated similar experiments showed an average homing accuracy of 11 cm, with a standard deviation of 5 cm, after a homing distance of around 3 m.

8.4.3. Timing Results. The algorithm runs surprisingly fast on the rather slow hardware we used: the initialisation for a new target lasts only 958 ms, while afterwards every 387 ms a new homing vector is computed. For a wheelchair driving at a cautious speed, it is possible to keep on driving while initialising a new target. This resulted in a smooth trajectory without stops or sudden velocity changes.

9. Conclusion

This paper describes and demonstrates a novel approach for a service robot to navigate autonomously in a large, natural complex environment. The only sensor is an omnidirectional colour camera. As environment representation, a topological map is chosen. This is more flexible and less memory demanding than metric 3D maps. Moreover, it does not show error build-up and enables fast path planning. As natural landmarks, we use two kinds of fast wide baseline features which we developed and adapted for this task. Because these features can be recognised even if the viewpoint is substantially different, a limited number of images suffice to describe a large environment.

Experiments show that our system is able to build autonomously a map of a natural environment it drives through. The localisation ability, with and without knowledge of previous locations, is demonstrated. With this map, a path towards each desired location can be computed efficiently. Experiments with a robotic wheelchair show the feasibility of executing such a path as a succession of visual servoing steps.

Acknowledgments

This work is partially supported by the Inter-University Attraction Poles, Office of the Prime Minister (IUAP-AMS), the Flemish Institute for the Advancement of Science in Industry (IWT), and the Fund for Scientific Research Flanders (FWO-Vlaanderen, Belgium).

Note

1. The medoid of a cluster is computed analogous to the centroid, but using the median instead of the average.

References

1. Aliaga, D. 2001. Accurate catadioptric calibration for real-time pose estimation in room-size environments. In *Proc. IEEE Int. Conf. Computer Vision (ICCV)*, Vancouver, pp. 127–134.
2. Arya, S., Mount, D., Netanyahu, N., Silverman, R., and Wu, A. 1998. An optimal algorithm for approximate nearest neighbor searching. *Journal of the ACM*, 45:891–923.
3. Basri, R., Rivlin, E., and Shimshoni, I. 1998. Visual homing: Surfing on the epipoles. In *IEEE International Conference on Computer Vision ICCV'98*, Bombay, pp. 863–869.
4. Beevers, K. and Huang, W. 2005. *Loop Closing in Topological Maps*. ICRA.
5. Bischof, H., Wildenauer, H., and Leonardis, A. 2001. Illumination insensitive eigenspaces. In *Proc. ICCV01*, IEEE Computer Society, pp. 233–238.
6. Cartwright, B. and Collett, T. 1987. Landmark maps for Honeybees. *Biol. Cybernetics*, 57:85–93.
7. Chen, C. and Wang, H. 2005. Appearance-based topological Bayesian inference for loop-closing detection in cross-country environment. IROS, Edmonton, pp. 322–327.
8. Davison, A. 2003. Real-time simultaneous localisation and mapping with a single camera. *Intl. Conf. on Computer Vision*, Nice.
9. Dempster, A.P. 1967. Upper and lower probabilities induced by a multivalued mapping. *The Annals of Statistics*, (28):325–339.
10. Demeester, E., Nuttin, M., Vanhooydonck, D., and Van Brussel, H. 2003. Fine motion planning for shared wheelchair control: Requirements and preliminary experiments. *International Conference on Advanced Robotics*, Coimbra, pp. 1278–1283.
11. Dijkstra, E.W. 1959. A note on two problems in connection with graphs. *Numerische Mathematik*, 1:269–271.
12. Fischler, M. and Bolles, R. 1981. Random sample consensus: A paradigm for model fitting with applications to image analysis. *Comm. of the ACM*, 24:381–395.
13. Franz, M., Schölkopf, B., Mallot, H., and Bülthoff, H. 1998. Where did I take that snapshot? Scene-based homing by image matching. *Biological Cybernetics*, 79:191–202.
14. Geyer, C. and Daniilidis, K. 2003. Mirrors in motion: Epipolar geometry and motion estimation. *Proc. ICCV*, Nice, p. 766.
15. Goedemé, T., Tuytelaars, T., and Van Gool, L. 2004. Fast wide baseline matching with constrained camera position. *Computer Vision and Pattern Recognition*. Washington, DC, pp. 24–29.
16. Goedemé, T., Tuytelaars, T., Vanacker, G., Nuttin, M., and Van Gool, L. 2005. Feature based omnidirectional sparse visual path following. *IEEE/RSJ International Conference on Intelligent Robots and Systems*, IROS 2005, Edmonton.
17. Haralick, R., Lee, C., Ottenberg, K., and Nölle, M. 1994. Review and analysis of solutions of the three point perspective pose estimation problem. *International Journal of Computer Vision*, 13(3):331–356.

18. Hartley, R. 1992. Estimation of relative camera positions for uncalibrated cameras. In *2nd European Conference on Computer Vision*, Springer-Verlag, LNCS 588, pp. 579–587.
19. Jain, A. 1989. *Fundamentals of Digital Image Processing*. Prentice Hall.
20. Jogan, M. and Leonardis, A. 1999. Panoramic eigenimages for spatial localisation, In *Proceedings 8th International Conference on Computer Analysis of Images and Patterns(1689)*, F. Solina, and A. Leonardis, (Eds.), Ljubljana, pp. 558–567.
21. Jogan, M. and Leonardis, A. 2000. Robust localization using panoramic view-based recogniton, In *Proceedings 15th International Conference on Pattern Recognition ICPR'00*, Barcelona, Spain, pp. 136–139.
22. Koenig, S. and Simmons, R. 1996. *Unsupervised Learning of Probabilistic Models for Robot Navigation*, ICRA.
23. Košecká, J. and Yang, X. 2004. Global localization and relative pose estimation based on scale-invariant features. *ICPR*, (4):319–322.
24. Kröse, B., Porta, J., van Breemen, A., Crucq, K., Nuttin, M., and Demeester, E. 2003. *Lino, the User-Interface Robot, First European Symposium on Ambient Intelligence (EUSAI 2003)*, Veldhoven, The Netherlands, pp. 264–274.
25. Ledwich, L. and Williams, S. 2004. Reduced SIFT features for image retrieval and indoor localisation. *Australasian Conf. on Robotics and Automation ACRA*, Canberra.
26. Loeffler, C., Ligtenberg, A., and Moschytz, G. 1989. *Practical Fast 1-D DCT Algoritmh with 11 Multiplications*, ICASSP, pp. 988–991.
27. Lowe, D. 1999. Object recognition from local scale-invariant features. In *International Conference on Computer Vision*, pp. 1150–1157.
28. Matas, J., Chum, O., Urban, M., and Pajdla, T. 2002. Robust wide baseline stereo from maximally stable extremal regions. *British Machine Vision Conference*, Cardiff, Wales, pp. 384–396.
29. Mariottini, G., Alunno, E., Piazzi, J., and Prattichizzo, D. 2005. Epipole-Based Visual Servoing with Central Catadioptric Camera. IEEE ICRA05, Barcelona.
30. Mikolajczyk, K. and Schmid, C. 2002. An affine invariant interest point detector. *ECCV*, 1:128–142.
31. Millnert, O., Goedemé, T., Tuytelaars, T., Van Gool, L., Huntemann, A., and Nuttin, M. 2006. Range determination for mobile robots using one omnidirectional camera. In *International Conference on Informatics in Control, Automation and Robotics, ICINCO 2006*, Setubal.
32. Mindru, F., Moons, T., and Van Gool, L. 1999. Recognizing color patters irrespective of viewpoint and illumination. *Computer Vision and Pattern Recognition*, 1:368–373.
33. Nistér, D., Naroditsky, O., and Bergen, J. 2004. *Visual Odometry, Conference on Computer Vision and Pattern Recognition*. Washington, DC.
34. Nuttin, M., Demeester, E., Vanhooydonck, D., and Van Brussel, H. 2001. Shared autonomy for wheelchair control: Attempts to assess the user's autonomy. *Autonome Mobile Systeme*, pp. 127–133.
35. Nuttin, M., Vanhooydonck, D., Demeester, E., Van Brussel, H., Buijsse, K., Desimpelaere, L., Ramon, P., and Verschelden, T. 2003. *A Robotic Assistant for Ambient Intelligent Meeting Rooms, First European Symposium on Ambient Intelligence (EUSAI 2003)*, Veldhoven, The Netherlands, pp. 304–317.
36. Okuma, T., Sakaue, K., Takemura, H., and Yokoya, N. 2000. Real-time camera parameter estimation from images for a Mixed Reality system. In *Proc. ICPR*, Barcelona, Spain.
37. Pérer-Fontán, F., Sanmartin, B., Steingaß, A., Lehner, A., Kubista, E., Martin, M.J., and Arbesser-Rastburg, B. 2004. Measurements and modelling of the satellite-to-indoor channel for Galileo. *European Navigation Conference GNSS*, Rotterdam.
38. Pollefeys, M., Van Gool, L., Vergauwen, M., Verbiest, F., Cornelis, K., Tops, J., and Koch, R. 2004. Visual modeling with a hand-held camera. *International Journal of Computer Vision*, 59(3):207–232.
39. Ranganathan, A., Menegatti, E., and Dellaert, F. 2005. Bayesian inference in the space of topological maps. *IEEE Transactions on Robotics*
40. Rekimoto, J. and Ayatsuka, Y. 2000. CyberCode: Designing augmented reality environments with visual tags. *Designing Augmented Reality Environments* (*DARE 2000*).
41. Schmid, C., Mohr, R., and Bauckhage, C. 1997. Local grey-value invariants for image retrieval. *International Journal on Pattern Analysis an Machine Intelligence*, 19(5):872–877.
42. Se, S., Lowe, D., and Little, J. 2001. Local and global localization for mobile robots using visual landmarks. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '01)*, Hawaii, USA.
43. Shafer, G. 1976. *A Mathematical Theory of Evidence*. Princeton University Press.
44. Shatkay, H. and Kaelbling, L.P. 1997. Learning topological maps with weak local odometric information. *IJCAI*, (2):920–929.
45. Shi, J. and Tomasi, C. 1994. Good features to track. *Computer Vision and Pattern Recognition*, Seattle, pp. 593–600.
46. Stricker, D., Daehne, P., Seibert, F., Christou, I.T., Almeida, L., Carlucci, R., and Ioannidis, N.I. 2001. Design and development issues for archeoguide: An augmented reality based cultural heritage on-site guide. In *International Conference on Augmented*, Virtual Environments and Three-Dimensional Imaging, Mykonos.
47. Svoboda, T., Pajdla, T., and Hlaváč V. 1998. Motion estimation using panoramic cameras. In *Conf. on Intelligent Vehicles*, Stuttgart, pp. 335–340.
48. Svoboda, T. 2000. Central Panoramic Cameras, Design, Geometry, Egomotion, PhD Thesis, Czech Technical University.
49. Tapus, A. and Siegwart, R. 2005. *Incremental Robot Mapping with Fingerprints of Places*, IROS, Edmonton.
50. Tuytelaars, T., Van Gool, L., D'haene, L., and Koch, R. 1999. Matching of affinely invariant regions for visual servoing. *International Conferemce on Robotics and Automation*, pp. 1601–1606.
51. Tuytelaars, T. and Van Gool, L. 2000. Wide baseline stereo based on local, affinely invariant regions. *British Machine Vision Conference*, Bristol, UK, pp. 412–422.
52. Ulrich, I. and Nourbakhsh, I. 2000. Appearance-based place recognition for topological localisation. In *IEEE International Conference on Robotics and Automatisation*, San Francisco, pp. 1023–1029.
53. van Veen, H., Distler, H.K., Braun, S.J., and Bülthoff, H.H. 1998. Navigating through a virtual city: Using virtual reality technology to study human action and perception. *Future Generation Computer Systems*, 14:231–242.
54. Vale, A. and Isabel Ribeiro, M. 2003. *Environment Mapping as a Topological Representation*. ICAR, Coimbra, Portugal.
55. Zivkovic, Z., Bakker, B., and Kröse, B. 2005. *Hierarchical Map Building Using Visual Landmarks and Geometric Constraints*, IROS, Edmonton, pp. 7–12.