A Robust Descriptor for Tracking Vertical Lines in Omnidirectional Images and Its Use in Mobile Robotics

Davide Scaramuzza, Roland Siegwart and Agostino Martinelli

The International Journal of Robotics Research 2009; 28; 149
DOI: 10.1177/0278364908099858

A Robust Descriptor for Tracking Vertical Lines in Omnidirectional Images and Its Use in Mobile Robotics

Davide Scaramuzza
Roland Siegwart
Swiss Federal Institute of Technology (ETH),
Tannenstrasse 3,
8092 Zurich,
Switzerland
davide.scaramuzza@ieee.org, rsiegwart@ieee.org

Agostino Martinelli
INRIA Rhône-Alpes,
655 avenue de l'Europe
Montbonnot,
38334 Saint Ismier Cedex,
France
agostino.martinelli@inrialpes.fr

Abstract

In this paper we introduce a robust descriptor for matching vertical lines among two or more images from an omnidirectional camera. Furthermore, in order to make such a descriptor usable in the framework of indoor mobile robotics, this paper introduces a new simple strategy to extrinsically self-calibrate the omnidirectional sensor with the odometry reference system. In the first part of this paper we describe how to build the feature descriptor. We show that the descriptor is very distinctive and is invariant to rotation and slight changes in illumination. The robustness of the descriptor is validated through real experiments on a wheeled robot. The second part of the paper is devoted to the extrinsic self-calibration of the camera with the odometry reference system. We show that by implementing an extended Kalman filter that fuses the information from the visual features with the odometry, it is possible to extrinsically and automatically calibrate the camera while the robot is moving. In particular, it is theoretically shown that only one feature suffices to perform the calibration. Experimental results validate the theoretical contributions.

KEY WORDS—omnidirectional camera, visual tracking, feature descriptor, extrinsic camera calibration.

The International Journal of Robotics Research
Vol. 28, No. 2, February 2009, pp. 149–171
DOI: 10.1177/0278364908099858
©The Author(s), 2009. Reprints and permissions:
http://www.sagepub.co.uk/journalsPermissions.nav
Figures 1–2, 4–8, 10, 13–26 appears in color online: http://ijr.sagepub.com

1. Introduction

1.1. Motivation and Contribution

One of the biggest challenges in mobile robotics is designing autonomous vehicles able to perform high-level tasks despite the quality/cost of the sensors. Vision sensors and encoder sensors are in general cheap and suitable for indoor navigation. In particular, regarding vision, an omnidirectional camera is very effective owing to the panoramic view it provides from a single image. In this paper, we introduce a robust descriptor for matching vertical lines among two or more images from an omnidirectional camera. Furthermore, in order to make such a descriptor usable in combination with encoder data, we also introduce a new simple strategy for extrinsically self-calibrating the omnidirectional sensor with the odometry reference system.

The main contributions of this paper are therefore as follows: (1) we introduce a new descriptor for matching vertical lines among two or more images from an omnidirectional camera; and (2) we introduce a simple strategy to extrinsically calibrate an omnidirectional camera with the odometry system.

1.2. Previous Work

One of the most important problems in vision-based robot navigation systems is the search for correspondences in images taken from different viewpoints. In the last few decades, the feature correspondence problem has been investigated largely for standard perspective cameras. Furthermore, several works have provided robust solutions for wide-baseline stereo matching, structure from motion, ego-motion estimation and robot navigation (Lindeberg 1998; Baumberg 2000; Mikolajczyk and Schmid 2001; Matas et al. 2002; Mikolajczyk and Schmid 2002; Kadir et al. 2004; Lowe 2004; Tuytelaars and Van Gool 2004). Some of these works normalize the region around each detected feature using a local affine transformation, which attempts to compensate for the distortion introduced by the perspective projection. However, such methods cannot be applied directly to images taken by omnidirectional imaging devices because of the non-linear distortions introduced by their large field of view.

In order to apply those methods, one needs first to generate a perspective view out of the omnidirectional image, provided that the imaging model is known and that the omnidirectional camera possesses a single effective viewpoint (Nayar 1997). An application of this approach was given by Mauthner et al. (2006). There, the authors generate perspective views from each region of interest of the omnidirectional image. This image unwrapping removes the distortions of the omnidirectional imaging device and enables the use of state-of-the-art wide-baseline algorithms designed for perspective cameras. Nevertheless, other researchers have attempted to apply to omnidirectional images standard feature detectors and matching techniques which have been traditionally employed for perspective images. Micusik and Pajdla (2006), for instance, checked the candidate correspondences between two views using the RANSAC algorithm.

Finally, other works have been developed, which extract one-dimensional features from new images called Epipolar plane images, under the assumption that the camera is moving on a flat surface (Briggs et al. 2006). These images are generated by converting each omnidirectional picture into a one-dimensional circular image, which is obtained by averaging the scan lines of a cylindrical panorama. Then, one-dimensional features are extracted directly from such kinds of images.

In this paper, we deal with real-world vertical features because they are predominant in structured environments. In our experiments, we used a wheeled robot equipped with a catadioptric omnidirectional camera with the mirror axis perpendicular to the plane of motion (Figure 1). If the environment is flat, this implies that all world vertical lines are mapped to radial lines on the camera image plane.

The use of vertical line tracking is not new in the robotics community. Since the beginning of machine vision, roboticians have been using vertical lines or other sorts of image measure for autonomous robot localization or place recognition. Several methods dealing with automatic line matching have been proposed for standard perspective cameras and can be divided into two categories: those that match individual line segments; and those that match groups of line segments. Individual line segments are generally matched on their geometric attributes (e.g. orientation, length, extent of overlap) (Medioni and Nevatia 1985; Ayache 1990; Zhang 1994). Crowley and Stelmazyk (1990), Deriche and Faugeras (1990) and Huttenlocher et al. (1993) use a nearest line strategy which is better suited to image tracking where the images and extracted segments are similar. Matching groups of line segments has the advantage that more geometric information is available for disambiguation. A number of methods have been developed around the idea of graph-matching (Ayache and Faugeras 1987; Horaud and Skordas 1989; Gros 1995; Venkateswar and Chellappa 1995). The graph captures relationships such as "left of", "right of", cycles, "collinear with", etc., as well as topological connectedness. Although such methods can cope with more significant camera motion, they often have a high complexity and again they are sensitive to error in the segmentation process.

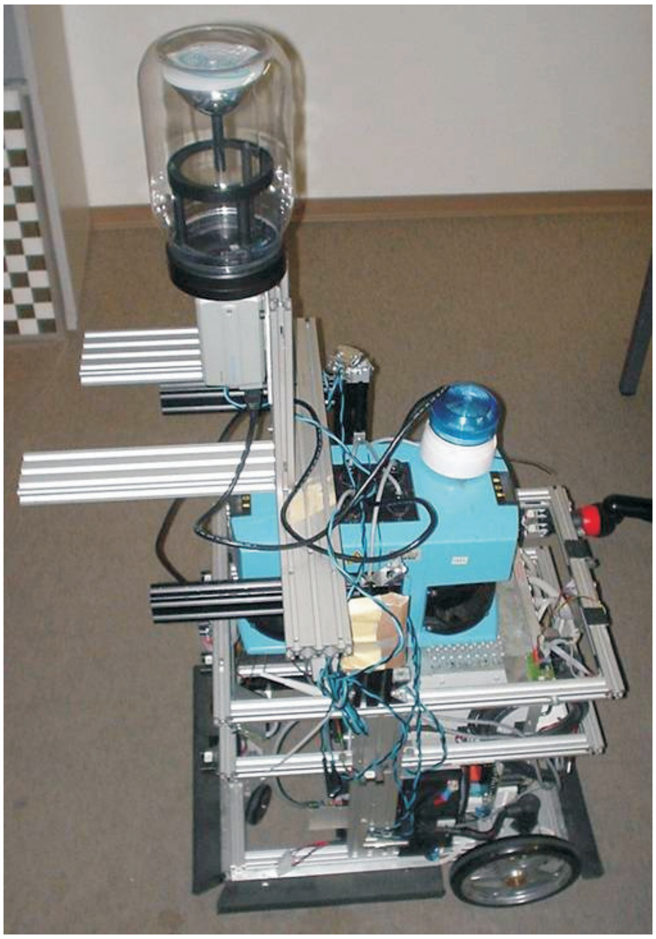
Fig. 1. The robot used in our experiments equipped with encoder sensors, omnidirectional camera and two laser range finders.

In addition to these methods, other approaches to individual line matching exist, which use some similarity measure commonly used in template matching and image registration (e.g. sum of squared differences (SSD), simple or normalized cross-correlation (NCC), image histograms; see Gonzalez and Woods (2002)). An interesting approach was proposed by Baillard et al. (1999). In addition to using the topological information of the line, the authors also used the photometric neighborhood of the line for disambiguation. Epipolar geometry was then used to provide a point-to-point correspondence on putatively matched line segments over two images and the similarity of the line's neighborhoods was then assessed by cross-correlation at the corresponding points.

A novel approach, using the intensity profile along the line segment, was proposed by Tell and Carlsson (2000). Although the application of the method was to wide baseline point matching, the authors used the intensity profile between two distinct points (i.e. a line segment) to build a distinctive descriptor. The descriptor is based on affine invariant Fourier coefficients that are directly computed from the intensity profile. Another approach designed for wide baseline point matching on affine invariant regions was proposed by Goedeme et al. (2004) and its application on robot localization with omnidirectional imagery was demonstrated by Sagues et al. (2006).

The methods cited above were defined for perspective images but the same concepts have been also used by roboticians in omnidirectional images under certain circumstances. The use of omnidirectional vision even facilitated the task because of the 360° field of view (Yagi and Yachida 1991; Brassart et al. 2000; Prasser et al. 2004). However, to match vertical lines among different frames only mutual and topological relations have been used (e.g. neighborhood or ordering constraints) sometimes along with the similarity measures cited above (e.g. SSD, NCC).

Finally, another important issue to be addressed when using a vision sensor in mobile robotics is the extrinsic calibration of the camera with respect to the robot odometry, i.e. the estimation of the parameters characterizing the transformation between the two references attached on the robot (robot origin) and on the vision sensor. When a given task is performed by fusing vision and odometry data, the performance will depend on this calibration. The problem of sensor-to-sensor calibration in robotics has received significant attention recently and a number of approaches have been developed (e.g. for IMU–camera (Mirzaei and Roumeliotis 2007), robot body–camera (Hesch et al. 2008) or laser scanner–camera (Zhang and Pless 2004; Scaramuzza et al. 2007b) calibration). However, very little attention has been devoted to determining the odometry–camera transformation. This is necessary in order to correctly fuse visual information and dead-reckoning in robot localization and mapping. In this paper, we focus on auto-calibration, that is, without user intervention.

1.3. Outline

In this paper we propose two contributions. In the first part of the paper, we describe how we build our robust descriptor for vertical lines. We show that the descriptor is very distinctive and is invariant to rotation and slight changes of illumination.

In the second part of the paper, we introduce a strategy based on the extended Kalman filter (EKF) to perform automatically the estimation of the extrinsic parameters of the omnidirectional camera during the robot motion. The strategy is theoretically validated through an observability analysis which takes into account the system non-linearities. In particular, it is theoretically shown that only one feature suffices to perform the calibration. This paper extends our two previous works (Martinelli et al. 2006; Scaramuzza et al. 2007a).

The present document is organized as follows. First, we describe our procedure to extract vertical lines (Section 2) and build the feature descriptor (Section 3). In Section 4 we provide our matching rules while in Sections 5 and 6 we characterize the performance of the descriptor. In Section 7, we describe the calibration problem, while in Section 8 we provide the equations to build the EKF. In Section 9, we present experimental results which validate both the theoretical contributions.

2. Vertical Line Extraction

Our platform consists of a wheeled robot equipped with a catadioptric omnidirectional camera (see Figure 1). The main advantage of this type of camera is that it provides a 360° field of view in the azimuth plane. In our arrangement, we set the camera–mirror system perpendicular to the floor where the robot moves. This setting guarantees that all vertical lines are mapped to radial lines on the camera image plane (Figure 2). In this section, we provide details of our procedure to extract prominent vertical lines. Our procedure consists of five steps.

The first step towards vertical line extraction is the detection of the image center (i.e. the point where all radial lines intersect). As the circular external boundary of the mirror is visible in the image, we used a circle detector to determine the coordinates of the center. Note that because the diameter of the external boundary is known and does not change dramatically during the motion, the detection of the center can be performed very efficiently and with high accuracy on every frame (this guarantees to cope also with the vibrations of the platform). The circle detection algorithm works in the following way: first, the radius of the circle has to be computed from a static image. Then, for each frame we use a circular mask (the same radius of the circle to be detected) which is convolved with the binary edge image. The output of this convolution is an accumulator matrix where the position of the maximum coincides with the position of the circle center.

The second step is the computation of the image gradients. We compute the two components $\mathbf{I_x}$, $\mathbf{I_y}$ of the image gradient by convolving the input image $\mathbf{I}$ with the two 3×3 Sobel masks. From $\mathbf{I_x}$, $\mathbf{I_y}$, we can calculate the magnitude $\mathbf{M}$ and the orientation $\mathbf{\Phi}$ of the gradients as

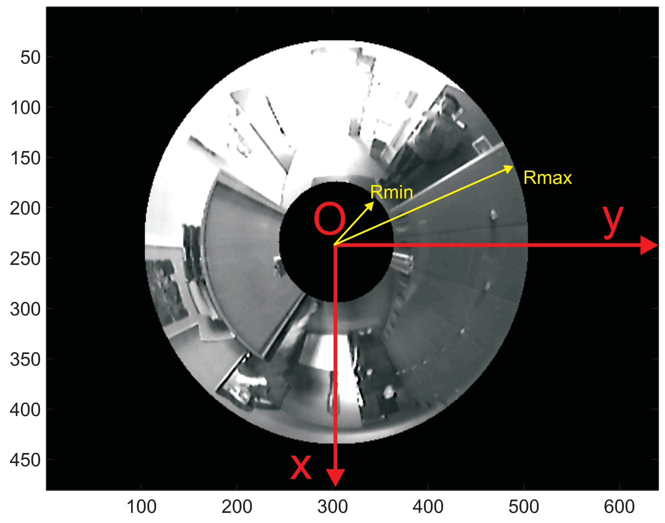

Fig. 2. An image taken by our omnidirectional camera. We used a KAIDAN-360-One-VR hyperbolic mirror and a SONY CCD camera the resolution of 640 × 480 pixels. The camera used in shown in Figure 1.

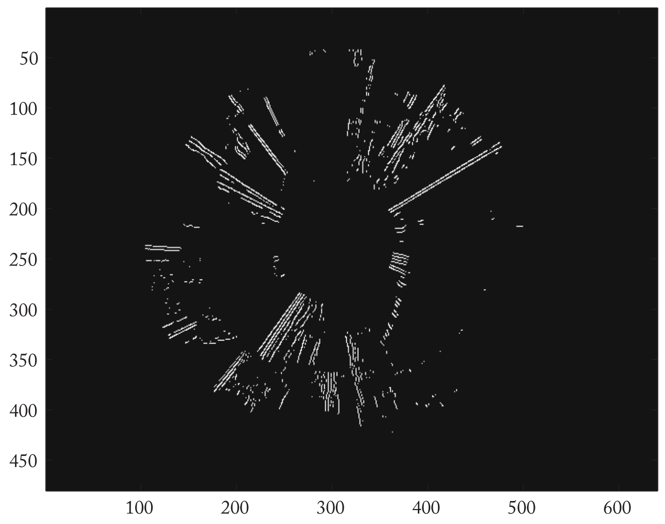

Fig. 3. Edge image of Figure 2.

$$\mathbf{M} = \sqrt{\mathbf{I_x}^2 + \mathbf{I_y}^2}, \quad \mathbf{\Phi} = \mathrm{atan}(\mathbf{I_y}/\mathbf{I_x}). \tag{1}$$

Then, we perform a thresholding on $\mathbf{M}$, $\mathbf{\Phi}$ by retaining those vectors whose orientation looks towards (or away from) the image center up to $\pm 5°$. This $10°$ tolerance allows us to handle the effects of floor irregularities on the appearance of vertical lines. After this thresholding, we apply edge thinning and we obtain the binary edge map depicted in Figure 3.

The third step consists of detecting the most reliable vertical lines. To this end, we divide the omnidirectional image into

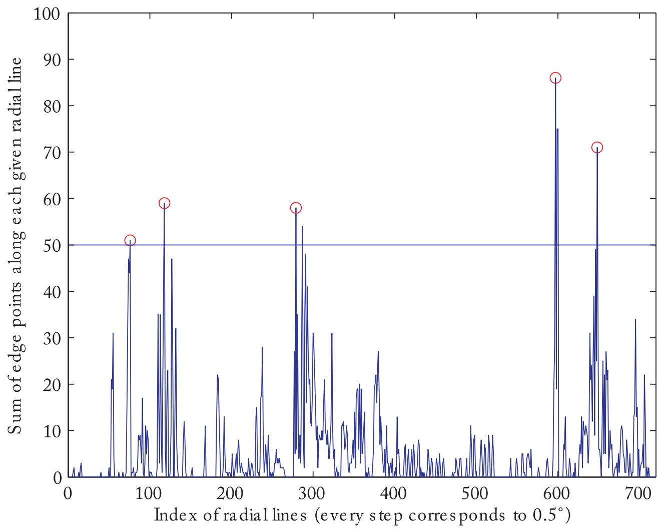

Fig. 4. Number of binary pixels voting for a given orientation angle.

720 predefined uniform sectors, which give us an angular resolution of $0.5°$. By summing up all binary pixels that vote for the same sector, we obtain the histogram shown in Figure 4. Then, we apply non-maxima suppression to identify all local peaks.

The final step is histogram thresholding. As observed in Figure 3, there are many potential vertical lines in structured environments. In order to keep the most reliable and stable lines, we put a threshold on the number of pixels of the observed line. As observed in Figure 4, we set our threshold equal to 50% of the maximum allowed line length, i.e. $R_{\max} - R_{\min}$. An example of vertical lines extracted using this threshold is shown in Figure 5.

3. Building the Descriptor

In Section 4, we describe our method for matching vertical lines between consecutive frames while the robot is moving. To make the feature correspondence robust to false positives, each vertical line is given a descriptor which is very distinctive. Furthermore, this descriptor is invariant to rotation and slight changes of illumination. In this way, finding the correspondent of a vertical line can be done by looking for the line with the closest descriptor. In the following sections, we describe how we built our descriptor.

3.1. Rotation Invariance

Given a radial line, we divide the space around it into three equal non-overlapping circular areas such that the radius r_a of

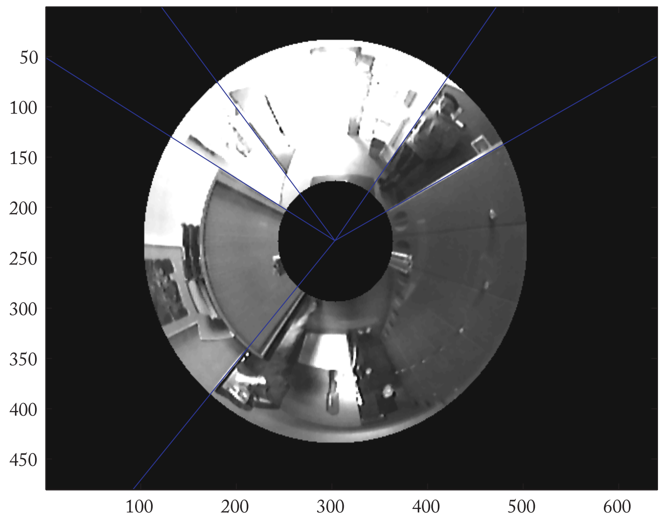

Fig. 5. Extraction of the most reliable vertical features from an omnidirectional image.

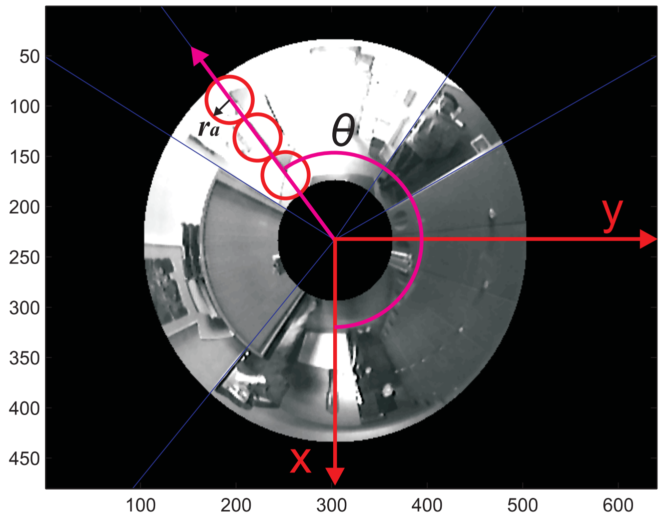

Fig. 6. Extraction of the circular areas. To achieve rotation invariance, the gradient orientation Φ of all points is redefined relative to the angle of the radial line θ.

each area is equal to $(R_{\max} - R_{\min})/6$ (see Figure 6). Then, we smooth each area with a Gaussian window with $\sigma_G = r_a/3$ and compute the image gradients (magnitude $\mathbf{M}$ and orientation Φ) within each of these areas. Concerning rotation invariance, this is achieved by redefining the gradient orientation Φ of all points relatively to the angle of the radial line θ (see Figure 6).

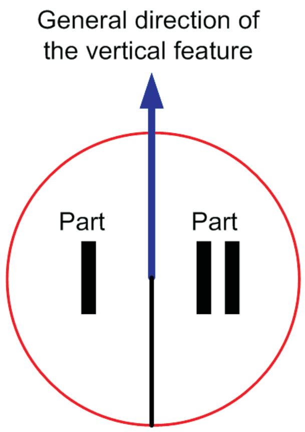

Fig. 7. The two sections of a circular area.

3.2. Orientation Histograms

To make the descriptor robust to false matches, we split each circular area into two parts and consider each one individually (Figure 7). In this way, we preserve the information about what we have on the left- and right-hand sides of the feature.

For each side of each circular area, we compute the gradient orientation histogram (Figure 8). The whole orientation space (from $-\pi$ to π) is divided into N_b equally spaced bins. In order to decide how much of a certain gradient magnitude m belongs to the adjacent inferior bin b and how much to the adjacent superior bin, each magnitude m is weighted by the factor $(1 - w)$, where

$$w = N_b \frac{\varphi - b}{2\pi}, \tag{2}$$

with φ being the observed gradient orientation in radians. Thus, $m(1 - w)$ will vote for the adjacent inferior bin, while mw will vote for the adjacent superior bin.

According to what we mentioned so far, each bin contains the sum of the weighted gradient magnitudes which belong to the correspondent orientation interval. We observed that this weighted sum made the orientation histogram more robust to image noise. Finally, observe that the orientation histogram is already rotation invariant because the gradient orientation has been redefined relative to the angle of the radial line (Section 3.1).

To recap, in the end we have three pairs of orientation histograms:

$$\begin{aligned} \mathbf{H}_1 &= \left[\mathbf{H}_{1,\mathrm{L}}, \mathbf{H}_{1,\mathrm{R}}\right], \\ \mathbf{H}_2 &= \left[\mathbf{H}_{2,\mathrm{L}}, \mathbf{H}_{2,\mathrm{R}}\right], \\ \mathbf{H}_3 &= \left[\mathbf{H}_{3,\mathrm{L}}, \mathbf{H}_{3,\mathrm{R}}\right], \end{aligned} \tag{3}$$

where the subscripts L, R identify respectively the left and right section of each circular area.

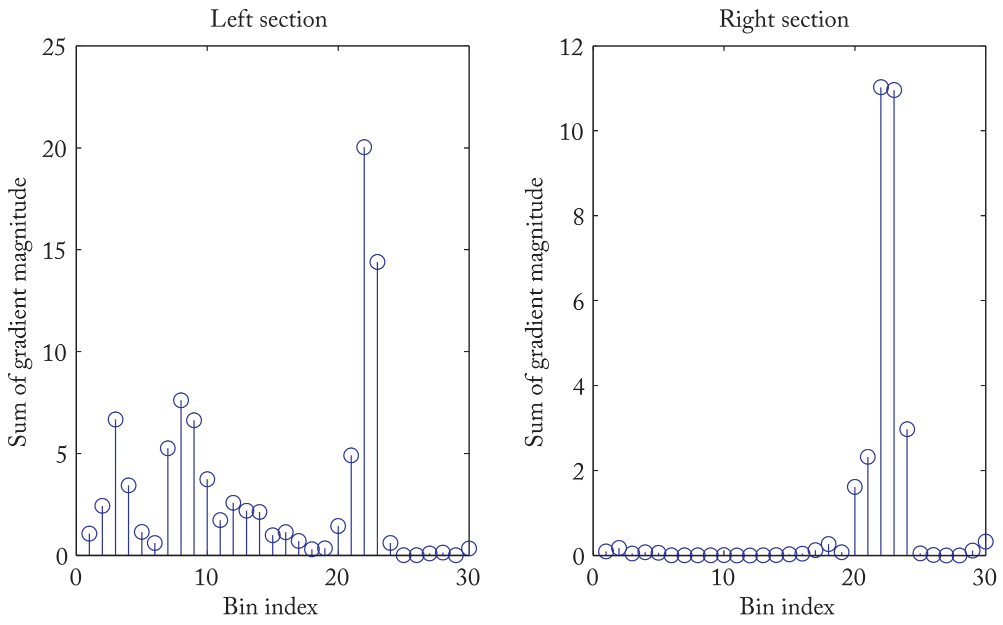

Fig. 8. An example of gradient orientation histograms for the left- and right-hand sides of a circular area.

3.3. Building the Feature Descriptor

From the computed orientation histograms, we build the final feature descriptor by stacking all three histogram pairs as follows:

$$\mathbf{H} = [\mathbf{H}_1, \mathbf{H}_2, \mathbf{H}_3]\,. \tag{4}$$

To have slight illumination invariance, we pre-normalize each histogram $\mathbf{H_i}$ to have unit length. This choice relies on the hypothesis that the image intensity changes linearly with illumination. However, non-linear illumination changes can also occur due to camera saturation or due to illumination changes that affect three-dimensional surfaces with different orientations by different amounts. These effects can cause a large change in relative magnitude for some gradients, but are less likely to affect the gradient orientations. Therefore, we reduce the influence of large gradient magnitudes by clipping the values in each unit histogram vector so that each bin is no larger than 0.1, and then renormalizing to unit length. This means that matching the magnitudes for large gradients is no longer as important, and that the distribution of orientations has greater emphasis. The value 0.1 was determined experimentally and is justified in Section 6.

To recap, our descriptor is an N-element vector containing the gradient orientation histograms of the circular areas. In our setup, we extract three circular areas from each vertical feature and use 32 bins for each histogram; thus the length of the descriptor is

$$N = 3 \text{ areas} \times 2 \text{ parts} \times 32 \text{ bins} = 192. \tag{5}$$

Observe that all feature descriptors are the same length.

4. Feature Matching

As every vertical feature has its own descriptor, its correspondent in consecutive images can be searched among the features with the closest descriptor. To this end, we need to define a dissimilarity measure (i.e. distance) between two descriptors.

In the literature, several measures have been proposed for the dissimilarity between two histograms $\mathbf{H} = \{h_i\}$ and $\mathbf{K} = \{k_i\}$. These measures can be divided into two categories. The *bin-by-bin* dissimilarity measures only compare the contents of the corresponding histogram bins, that is, they compare h_i and k_i for all i, but not h_i and k_i for $i \neq j$. The *cross-bin* measures also contain terms that compare non-corresponding bins. Among the *bin-by-bin* dissimilarity measures are the Minkoski-form distance, the Jeffrey divergence, the χ^2 statistics and the Bhattacharya distance. Among the *cross-bin* measures, one of the most widely used is the quadratic-form distance. An exhaustive review of all of these methods is given by Raman et al. (2005) and Rubner et al. (2000, 2001).

In our work, we tried the dissimilarity measures mentioned above but the best results were obtained using the L_2 distance (i.e. Euclidean distance) that is a particular case of the Minkoski-form distance. Therefore, in our experiments we used the Euclidean distance as a measure of the dissimilarity between descriptors, which is defined as

$$d(\mathbf{H}, \mathbf{K}) = \sqrt{\sum_{i=1}^{N} |h_i - k_i|^2}. \tag{6}$$

By the definition of distance, the correspondent of a feature, in the observed image, is expected to be that, in the consecutive image, with the minimum distance. However, if a feature is no longer present in the next image, there will be a closest feature anyway. For this reason, we defined three tests to decide whether a feature correspondent exists and which one the correspondent is. Before describing these tests, let us introduce some definitions.

Let $\{\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_{N_A}\}$ and $\{\mathbf{B}_1, \mathbf{B}_2, \ldots, \mathbf{B}_{N_B}\}$ be two sets of feature descriptors extracted at time t_A and t_B, respectively, where N_A, N_B are the number of features in the first and second image. Then, let

$$D_i = \{d(\mathbf{A}_i, \mathbf{B}_j),\ j = 1, 2, \ldots, N_B\} \tag{7}$$

be the set of all distances between a given $\mathbf{A}_i$ and all $\mathbf{B}_j$ ($j = 1, 2, \ldots, N_B$). Finally, let $minD_i = \min_i (D_i)$ be the minimum of the distances between given $\mathbf{A_i}$ and all $\mathbf{B_j}$ and $SminD_i$ the distance to the second closest descriptor.

4.1. Test 1

The first test checks that the distance from the closest descriptor is smaller than a given threshold, that is,

$$minD_i \leq F_1. \tag{8}$$

By this criterion, we actually set a bound on the maximum acceptable distance to the closest descriptor.

4.2. Test 2

The second test checks that the distance from the closest descriptor is smaller than the mean of the distances from all other descriptors, that is,

$$minD_i \leq F_2 \cdot \langle D_i \rangle, \tag{9}$$

where $\langle D_i \rangle$ is the mean value of D_i and F_2 clearly ranges from zero to one. This criterion comes from experimental results.

4.3. Test 3

Finally, the third test checks that the distance from the closest descriptor is smaller than the distance from the second closest descriptor $SminD_i$:

$$minD_i \leq F_3 \cdot SminD_i, \tag{10}$$

where F_3 clearly ranges from zero to one. As in the previous test, the third test raises from the observation that, if the correct correspondence exists, then there must be a big gap between the closest and the second closest descriptor.

Table 1. The Distances between the Descriptor A_1 at Time t_A and All Descriptors $\mathbf{B_j}$, $j = 1, 2, \ldots, N_B$ at Time t_B.

B_1	B_2	B_3	B_4	B_5
0.57	0.72	0.74	0.78	0.83

Table 2. The Parameters used by Our Algorithm with their Empirical Values.

$F_1 = 1.05$	$F_2 = 0.75$	$F_3 = 0.8$

In Table 1, we show an example of real comparison among the distances between descriptor $\mathbf{A}_1$ at time t_A and all descriptors $\mathbf{B}_j$ at time t_B. Observe that descriptor $\mathbf{B}_1$ is not the correct correspondent of $\mathbf{A}_1$. In fact, it passes tests 1 and 3 but not test 2.

Factors F_1, F_2, F_3 were determined experimentally. The values used in our experiments are shown in Table 2. The motivation for this choice of values is given in Section 6.

5. Comparison with other Image Similarity Measures

A good method to evaluate the distinctiveness of the descriptors in the observed image is to compute a similarity matrix $\mathbf{S}$ where each element $\mathbf{S}(i, j)$ contains the distance between the ith and jth descriptor. That is,

$$\mathbf{S}(i, j) = d(\mathbf{H}_i, \mathbf{H}_j), \tag{11}$$

where $\mathbf{H}_i$ is the descriptor of the ith radial line and distance d is defined as in (6). Observe that to build this matrix we compute the descriptor of the radial line for every $\theta \in [0°, 360°]$. We used a θ increment of 1° and thus $i = 1, 2, \ldots, 360$. Furthermore, note that $\mathbf{S}$ is symmetric and that $\mathbf{S}(i, j) = 0$ for $i = j$. The similarity matrix computed for the image of Figure 6 is shown in Figure 9.

In this section, we want to compare our descriptor with two other image similarity measures that are well known in image registration and are also commonly used for matching individual lines. These are SSD and zero-mean normalized cross-correlation (ZNCC) (their definitions are given by Gonzalez and Woods (2002)). When using SSD and ZNCC for comparing two patterns, the pattern descriptor can be seen as the pattern intensity. In our case, we take as a pattern the rectangular region around the observed radial line as shown in Figure 10. As we did to build the similarity matrix for our descriptors, we compare a given pattern $\mathbf{P}_i$ with pattern $\mathbf{P}_j$ using either SSD or ZNCC and build the respective similarity matrices, that is,

$$\mathbf{S}_{\mathrm{SSD}}(i, j) = SSD(\mathbf{P}_i, \mathbf{P}_j), \tag{12}$$

$$\mathbf{S}_{\mathrm{ZNCC}}(i, j) = ZNCC(\mathbf{P}_i, \mathbf{P}_j), \tag{13}$$

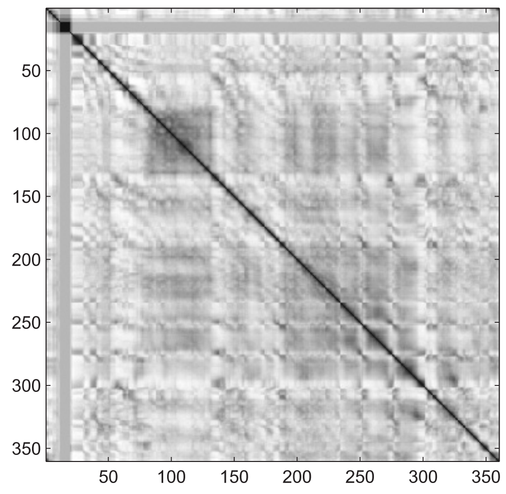

Fig. 9. Similarity matrix for descriptors.

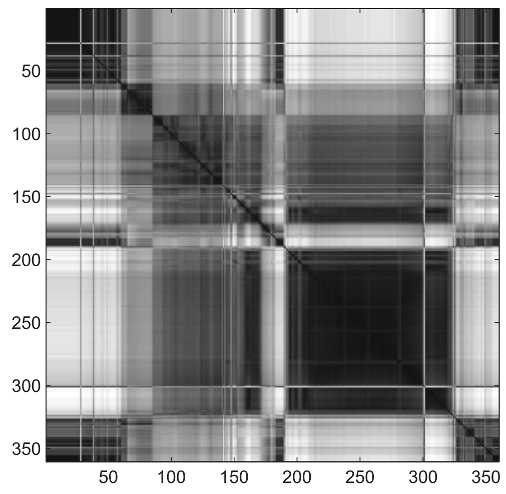

Fig. 11. Similarity matrix for SSD.

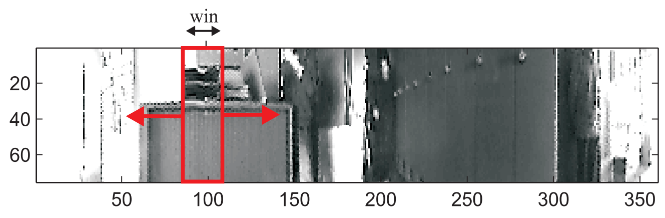

Fig. 10. This is the same image as Figure 6 after unwrapping into a cylindrical panorama. The rectangular region used to compute SSD and ZNCC is also shown.

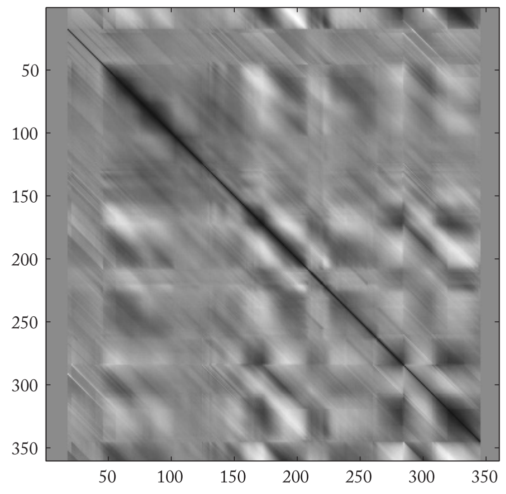

Fig. 12. Similarity matrix for ZNCC.

The two similarity matrices for the image in Figure 6 are shown in Figures 11 and 12. Concerning the size *win* of the patterns for computing SSD and ZNCC, we chose $win = 2r_a$. Observe that this choice is reasonable as $2r_a$ is also the size (diameter) of the three circular areas used to build our descriptor. Furthermore observe that, for SSD, the maximum similarity between two patterns occurs when $SSD = 0$. Conversely, for ZNNC, maximum similarity (correlation) occurs when $ZNCC = 1$; however, observe that Figure 12 has been inverted to enable comparison with Figures 9 and 11 (this means that black indicates maximum similarity and white the minimum similarity).

To interpret the similarity matrix, consider points along the diagonal axis in Figure 9. Each point is perfectly similar to itself, so all of the points on the diagonal are dark. Starting from a given point on the diagonal, one can compare how its correspondent descriptor relates to its neighbors forwards and backwards by tracing horizontally or vertically on the matrix. To compare a given descriptor $\mathbf{H}_i$ with descriptor $\mathbf{H}_{i+n}$, simply start at point (i, i) on the matrix and trace horizontally to the right to $(i, i + n)$.

In the similarity matrix for SSD, one can see large blocks of dark which indicate that there are repeating patterns in the image or that the patterns are poorly textured. Rectangular blocks of dark that occur off the diagonal axis indicate reoccurring patterns. This can be better understood by observing Figure 10. As observed, there are poorly textured objects and repeating structure.

Similar comments can be made regarding the similarity matrix for ZNCC. However, observe that the behavior of ZNCC is better than SSD: first, the size of the blocks along or off the diagonal axis is smaller; then, points on the diagonal are much darker than points off the diagonal.

Regarding the similarity matrix of our descriptor the diagonal axis is well demarcated, in fact points on the diagonal are much darker than those off the diagonal; the contrast with the regions off the diagonal is higher than ZNCC. Finally, observe

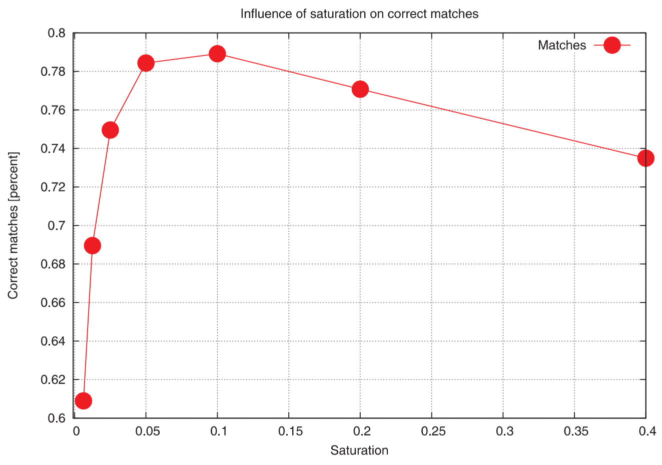

Fig. 13. Influence of saturation on correct matches.

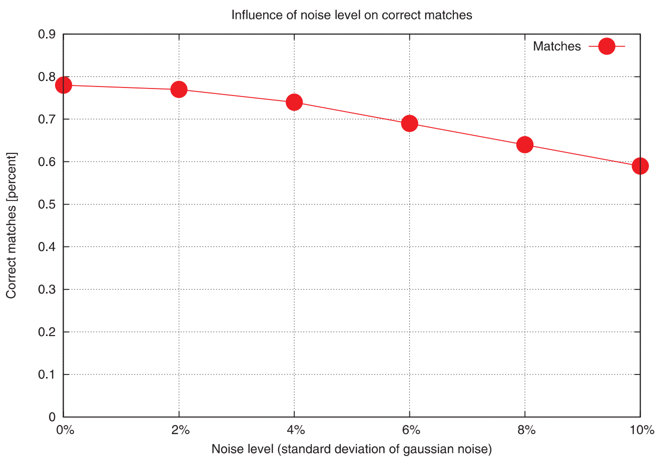

Fig. 14. Influence of noise level (%) on correct matches. The correct matches are found using only the nearest descriptor in the database.

that blocks along or off the diagonal axis are much smaller or lighter than SSD and ZNCC; this indicates that even on poorly textured surfaces our descriptor is distinctive enough. This is mainly due to use of the gradient information and to the fact of having split the region around the line in three areas instead of taking the entire region as whole.

6. Performance Evaluation

In this section, we characterize the performance of our descriptor on a large image dataset by taking into account the sensitiveness to different parameters, which are image saturation, image noise, number of histogram bins and the use of overlapping circular areas. Furthermore, we also motivate the choice of the values of F_1, F_2 and F_3 shown in Table 2.

6.0.1. Ground Truth

To generate the ground truth for testing our descriptor, we used a database of 850 omnidirectional pictures that is a subset of the video sequence (1,852 images) used in Section 9.1. First, we extracted verticals lines from each image. Then we manually labeled all of the corresponding features with the same ID. The images were taken from the hallway of our department. Figure 21 shows three sample images from our dataset. The images show that the illumination conditions vary strongly. Owing to large windows, a mixture of natural and artificial lighting produces difficult lighting conditions such as highlights and specularities.

In the following sections, we characterize the performance of our descriptor. We would like to remark that the features of each image were matched against all of the other images of the dataset where the same features appeared. Furthermore, the images of our dataset were taken such that each vertical line could be continuously observed for at least 2 meters of translational motion. This means that the features were matched also among images with strong baseline (up to 2 meters).

6.0.2. Image Saturation

As we mentioned in Section 3.3, we limit the values of the histogram vectors to reduce the influence of image saturation. The percentage of correct matches for different threshold values is shown in Figure 13. The results show the percentage of features that find a correct match to the single closest neighbor in the entire database. As the graph shows, the maximum percentage of correct matches is reached when using a threshold value equal to 0.1. In the remainder of this paper, we always use this value.

6.0.3. Image Noise

The percentage of correct matches for different amounts of Gaussian image noise (from 0% to 10%) is shown in Figure 14. Again, the results show the percentage of correct matches found using the single nearest neighbor in the entire database. As this graph shows, the descriptor is resistant even to large amounts of pixel noise.

6.0.4. Histogram Bins and Circular Areas

There are two parameters that can be used to vary the complexity of our descriptor: the number of orientation bins (N_b) in the histograms and the number of circular areas. Although in the

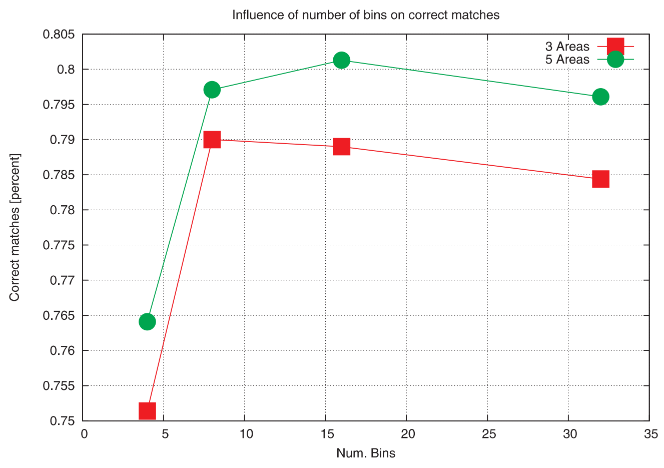

Fig. 15. Influence of number of bins on correct matches.

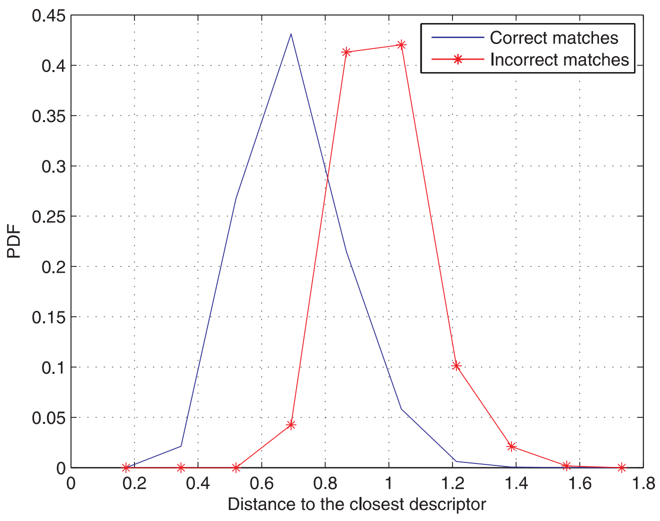

Fig. 16. The PDF that a match is correct according to the first rule.

explanation of the descriptor we used three non-overlapping circular areas, we evaluated the effect of using five overlapping areas with 50% overlap between two circles. The results are shown in Figure 15. As the graph shows, there is a slight improvement in using five overlapping areas (the amelioration is only 1%). Also, the performance is quite similar using 8, 16 or 32 orientation bins. Following this considerations, the best choice would seem to use three areas and eight histogram bins in order to reduce the dimension of the descriptor. However, we chose to use 32 orientation bins. For 32 bins, in fact, we had the biggest separation between the probability density functions (PDFs) of correct and incorrect matches shown in Figures 16, 17 and 18. Finally observe that we considered powers of two due to computational efficiency. The final computation time of the entire process (feature extraction, descriptor computation and matching) took less than 20 ms on a dual-core laptop computer.

6.0.5. Matching Rules

Figure 16 shows the PDF for correct and incorrect matches in terms of the distance to the closest neighbor of each keypoint. In our implementation of the first rule, we chose $F_1 = 1.05$. As observed in the graph, by this choice we reject all matches in which the distance to the closest neighbor is greater than 1.05, which eliminates 50% of the false matches while discarding less than 5% of correct matches.

Similarly, Figure 17 shows the PDFs in the terms of the ratio of closest to average-closest neighbor of each keypoint. In our implementation of the second rule, we chose $F_2 = 0.75$. As observed in the graph, by this choice we reject all matches where the ratio between the closest neighbor distance and the mean of all other distances is greater than 0.75, which eliminates 45% of the false matches while discarding less than 8% of correct matches.

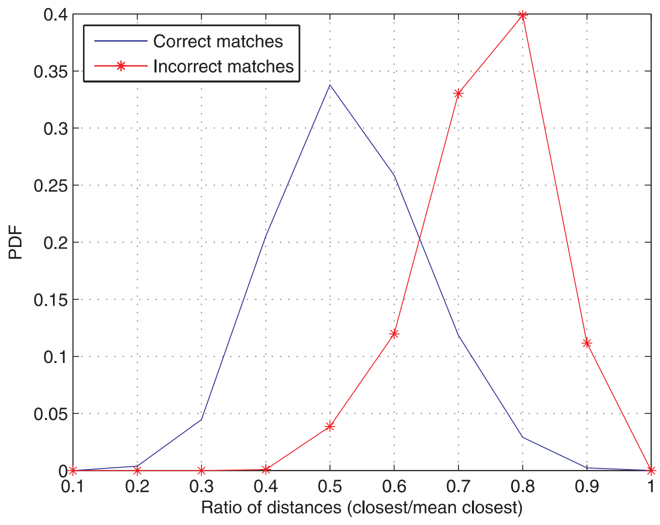

Fig. 17. The PDF that a match is correct according to the second rule.

Finally, Figure 18 shows the PDFs in terms of the ratio of closest to second-closest neighbor of each keypoint. In our implementation of the third rule, we chose $F_3 = 0.8$; in this way we reject all matches in which the distance ratio is greater than 0.8, which eliminates 92% of the false matches while discarding less than 10% of correct matches.

7. Camera–Robot Self-calibration: The Problem

Accurate extrinsic calibration of a camera with the odometry system of a mobile robot is a very important step towards

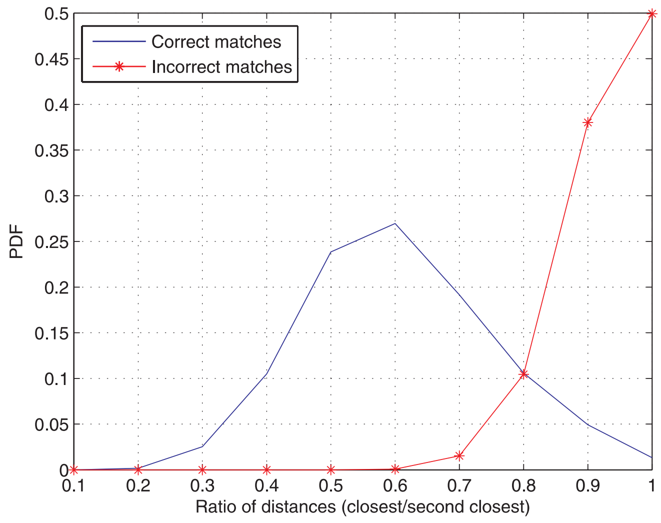

Fig. 18. The PDF that a match is correct according to the third rule.

precise robot localization. This stage is usually poorly documented and is commonly carried out by manually measuring the position of the camera with respect to the robot frame. In this section, we describe a new method that uses an EKF to extrinsically and automatically calibrate the camera while the robot is moving. The approach is similar to that presented by Martinelli et al. (2006) where just a single landmark (a source of light) was tracked during the motion to perform calibration. In this section, we extend the method of Martinelli et al. (2006) by providing the EKF equations to cope with multiple features. The features in use are vertical features which are extracted and tracked as described in the previous sections.

In order to simplify the problem, we make the following assumptions: we assume that the robot is moving in a flat environment and that it is equipped with an omnidirectional camera whose z-axis is parallel to the z-axis of the robot, that is, the mirror axis is perpendicular to the floor. According to this, the three-dimensional camera–odometry calibration problem becomes a two-dimensional problem.

Our first goal is the estimation of the three parameters ϕ, ρ, ψ which characterize the rigid transformation between the two references frames attached to the robot and to the camera (see Figure 19). The second goal is to perform calibration automatically and while the robot is moving. The available data are the robot wheels displacements $\delta\rho_R$ and $\delta\rho_L$ (see later) delivered by the encoder sensors and the bearing angle observations β of several features in the camera reference frame (Figure 19).

As we consider the case of a mobile robot moving in a two-dimensional environment, its configuration is described through the state $\mathbf{X}_R = [x_R, y_R, \theta_R]^T$ containing its position and orientation (as indicated in Figure 19). Furthermore, we consider the case of a robot equipped with a differential drive system. The robot configuration $\mathbf{X}_R$ can then be estimated by integrating the encoder data. In particular, we have

$$\begin{aligned} x_{R_{i+1}} &= x_{R_i} + \delta\rho_i \cos\left(\theta_{R_i} + \frac{\delta\theta_i}{2}\right), \\ y_{R_{i+1}} &= y_{R_i} + \delta\rho_i \sin\left(\theta_{R_i} + \frac{\delta\theta_i}{2}\right), \\ \theta_{R_{i+1}} &= \theta_{R_i} + \delta\theta_i, \end{aligned} \quad (14)$$

where quantities $\delta\rho$ and $\delta\theta$ are related to the displacements $\delta\rho_R$ and $\delta\rho_L$ (of the right and left wheel,respectively) directly provided by the encoders through

$$\delta\rho = \frac{\delta\rho_R + \delta\rho_L}{2}, \quad \delta\theta = \frac{\delta\rho_R - \delta\rho_L}{e}, \quad (15)$$

where e is the distance between the wheels.

For a particular bearing angle observation β, we obtain the following analytical expression (see Figure 19):

$$\beta = \pi - \psi - \theta_R - \phi + \alpha, \quad (16)$$

with

$$\alpha = \tan^{-1}\left(\frac{y_R + \rho\sin(\theta_R + \phi)}{x_R + \rho\cos(\theta_R + \phi)}\right). \quad (17)$$

8. EKF-based Calibration

An intuitive procedure to determine parameters ϕ, ρ, ψ is to use the data from the encoders to estimate the robot configuration (provided that the initial robot configuration is known). Then, by measuring the bearing angle β at several different robot configurations (at least three), it is possible to obtain parameters ϕ, ρ, ψ by solving a non-linear system in three unknowns. However, the drawback of this method is that, when the robot configuration is estimated by using only the encoder data, the error integrates over the path. This means that this procedure can be applied only for short paths and therefore the achievable accuracy on the estimation of ϕ, ρ, ψ is limited. Furthermore, the initial robot configuration has to be known with high accuracy.

One way to overcome these problems is to integrate the encoder data with the bearing angle measurements to estimate the robot configuration. This can be done by introducing an augmented state $\mathbf{X}_a$ containing the robot configuration and the calibration parameters ϕ, ρ, ψ:

$$\mathbf{X}_a = [x_R, y_R, \theta_R, \phi, \rho, \psi]^T. \quad (18)$$

An EKF can be adopted to estimate the state $\mathbf{X}_a$. The inputs $\mathbf{u}$ of the dynamics of this state are provided directly by the encoder data and the observations $\mathbf{z}$ are the bearing angles provided by the vision sensor. However, as was pointed out by

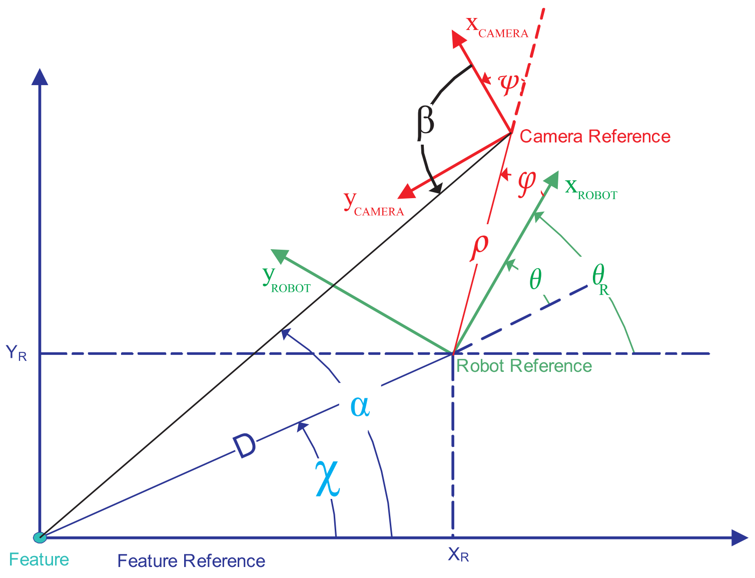

Fig. 19. The two reference frames attached to the robot and to the camera. The five parameters estimated by the EKF $(D, \theta, \phi, \rho, \psi)$ are also indicated.

Martinelli et al. (2006), by considering the system state $\mathbf{X}_a$ as defined in (18) the system is not observable, that is, it does not contain all of the necessary information to perform the estimation with an error which is bounded. Conversely, Martinelli et al. (2006) proved that the system becomes observable if, instead of considering $\mathbf{X}_a$, we introduce a new state $\mathbf{X}$ defined as follows:

$$\mathbf{X} = [D, \theta, \phi, \rho, \psi]^{\mathrm{T}}, \tag{19}$$

with $D = \sqrt{x_{\mathrm{R}}^2 + y_{\mathrm{R}}^2}$ and

$$\theta = \theta_{\mathrm{R}} - \tan^{-1}\left(\frac{y_{\mathrm{R}}}{x_{\mathrm{R}}}\right)$$

(see Figure 19). Note also that D is the distance from the observed feature.

Observe that, without loss of generality, we can use $\mathbf{X}$ instead of $\mathbf{X}_a$. In fact, $\mathbf{X}_a$ contains the whole robot configuration whose estimation is not our goal, indeed we just want to estimate parameters ϕ, ρ, ψ.

8.1. Observability Properties with Respect to the Robot Trajectory

In control theory, a system is defined as observable when it is possible to reconstruct its initial state by knowing, in a given time interval, the control inputs and the outputs (Isidori 1995). The observability property has a very practical meaning. When a system is observable it contains all of the necessary information to perform the estimation with an error which is bounded (Isidori 1995).

In this section, we investigate the observability properties for the state X with respect to the robot trajectories. Our analysis takes into account the system non-linearities. Indeed, the observability analysis changes dramatically from linear to non-linear systems (Isidori 1995). First of all, in the non-linear case, the observability is a local property. For this reason, in a non-linear case the concept of the *local distinguishability property* was introduced by Hermann and Krener (1977). The same authors introduced also a criteria, *the observability rank condition*, to verify whether a system has this property. This criteria plays a very important role since in many cases a non-linear system, whose associated linearized system is not observable, has however the local distinguishability property. Note that it is the distinguishability property which implies that the system contains the necessary information to have a bounded estimation error (actually, provided that the locality is large enough with respect to the sensor accuracy).

The dynamics of our system are described by the following equations:

$$\dot{D} = v\cos\theta,$$
$$\dot{\theta} = \omega - \frac{v}{D}\sin\theta,$$
$$\dot{\phi} = \dot{\rho} = \dot{\psi} = 0. \tag{20}$$

Our system is affine in the input variables, i.e. the previous equations can be written in the following compact form:

$$\dot{X} = f(X, u) = \sum_{k=1}^{M} f_k(X)u_k, \tag{21}$$

where M is the number of the input controls (which are independent). In our case $M = 2$ and the controls are $u_1 = v$, $u_2 = \omega$ and

$$\begin{aligned} f_1 &= \left[\cos\theta, -\frac{\sin\theta}{D}, 0, 0, 0\right]^{\mathrm{T}}, \\ f_2 &= [0, 1, 0, 0, 0]^{\mathrm{T}}. \end{aligned} \tag{22}$$

The observation is defined by the equation

$$\beta_i = \tan^{-1}\left(\frac{-\rho_i \sin(\theta_i + \phi_i)}{-D_i - \rho_i \cos(\theta_i + \phi_i)}\right) - \theta_i - \phi_i - \psi_i. \tag{23}$$

We now want to recall some concepts of the theory of Hermann and Krener (1977). We adopt the following notation. We indicate the Kth-order Lie derivative of a field Λ along the vector fields $v_{i_1}, v_{i_2}, \ldots, v_{i_K}$ with $L^K_{v_{i_1}, v_{i_2}, \ldots, v_{i_K}} \Lambda$. Note that the Lie derivative is not commutative. In particular, in $L^K_{v_{i_1}, v_{i_2}, \ldots, v_{i_K}} \Lambda$ it is assumed to differentiate along v_{i_1} first and along v_{i_K} at the end. Let us indicate with Ω the space spanned by all of the Lie derivatives $L^K_{f_{i_1}, f_{i_2}, \ldots, f_{i_K}} h(X)|_{t=0}$ ($i_1, \ldots, i_K = 1, 2, \ldots, M$ and the functions f_{i_j} are defined in (22)). Furthermore, we denote with $d\Omega$ the space spanned by the gradients of the elements of Ω.

In this notation, the observability rank condition can be expressed in the following way: *the dimension of the observable sub-system at a given X_0 is equal to the dimension of $d\Omega$.*

Martinelli et al. (2006) showed that the state X satisfying the dynamics in (20) is observable when the observation is that given in (23). Here we want to investigate the observability properties depending on the robot trajectory. In particular, we consider separately the case of straight motion and pure rotations about the robot origin. Furthermore, we consider separately the case when the observed feature is far from the robot and the case when it is close. Before considering the mentioned cases separately we observe that β depends on X through the ratio $\lambda \equiv D/\rho$ and the sum $\gamma \equiv \theta + \phi$:

$$\beta = \operatorname{atan}\left(\frac{\sin\gamma}{\lambda + \cos\gamma}\right) - \gamma - \psi. \tag{24}$$

The case of far feature and close feature corresponds respectively to have $\lambda \gg 1$ and $\lambda \sim 1$. In the first case the expression of β can be approximated with

$$\beta \cong -\theta - \phi - \psi. \tag{25}$$

8.1.1. Pure Rotations and Far Feature

This motion is obtained by setting $u_1 = 0$. Hence, the Lie derivatives must be calculated only along the vector f_2. The observation is that given in (25). It is easy to realize that the dimension of $d\Omega$ is 1 (the first-order Lie derivative is equal to -1, i.e. is a constant). This result is intuitive: the observation in (25) does not provide any information on D and ρ and it is not able to distinguish among θ, ϕ and ψ. Furthermore, the pure rotation does not provide any additional information. Hence, it is only possible to observe the sum $\theta + \phi + \psi$.

8.1.2. Straight Motion and Far Feature

This motion is obtained by setting $u_2 = 0$. Hence, the Lie derivatives must be calculated only along the vector f_1. We note that f_1 depends only on θ and D. Furthermore, $\beta = -\theta - \eta$ (having defined $\eta \equiv \phi + \psi$). Hence, the best we can hope is that the following quantities are observable: D, θ, η. In Appendix A we show that this is the case.

8.1.3. Pure Rotation and Close Feature

Let us consider the state $X_\lambda \equiv [\lambda, \gamma, \phi, \rho, \psi]^{\mathrm{T}}$. When $\dot{X} = \omega f_2$ we have $\dot{X}_\lambda = \omega f_2$. On the other hand, f_2 is independent of X_λ. Furthermore, the expression in (24) depends only on λ, γ and ψ. Hence, the best we can hope is that the quantities λ, γ and ψ are observable. In Appendix A we show that this is the case.

8.1.4. Straight Motion and Close Feature

This is the hardest case. *A priori*, it is not possible to exclude that the entire state X is observable. However, a direct computation of the dimension of $d\Omega$ (see Appendix A) shows that this dimension is smaller than 5, meaning that X is not observable.

We conclude this section with the following important remark. It is possible to estimate the parameters ϕ, ρ and ψ by combining a straight motion far from the feature with pure rotations close to the feature. Indeed, by performing the first trajectory it is possible to observe the sum $\phi + \psi$, D and θ. Once the robot starts to rotate, D does not change. Furthermore, the sum $\phi + \psi$ is time independent. On the other hand, with the pure rotation λ, γ and ψ are observable. Therefore, from the values of λ and D it is possible to determine ρ, and from ψ and the sum $\phi + \psi$ it is possible to determine ϕ.

8.2. The Filter Equations

By using D, θ and Equation (14), we obtain the following dynamics for the state $\mathbf{X}$:

$$\begin{aligned} D_{i+1} &= D_i + \delta\rho_i \cos\theta_i, \\ \theta_{i+1} &= \theta_i + \delta\theta_i - \frac{\delta\rho_i}{D_i} \sin\theta_i, \end{aligned}$$

$$\begin{aligned}\phi_{i+1} &= \phi_i,\\ \rho_{i+1} &= \rho_i,\\ \psi_{i+1} &= \psi_i, \end{aligned} \qquad (26)$$

where, from now on, subscript i will be used to indicate the time.

Similarly, the bearing angle observations β_i (16) can be read as

$$\beta_i = \tan^{-1}\left(\frac{-\rho_i \sin(\theta_i + \phi_i)}{-D_i - \rho_i \cos(\theta_i + \phi_i)}\right) - \theta_i - \phi_i - \psi_i. \quad (27)$$

Observe that so far we have taken into account only the observation of a single feature. As we want to cope with multiple features, we need to extend the definition of $\mathbf{X}$ (19) as follows:

$$\mathbf{X} = [D^1, \theta^1, D^2, \theta^2, \ldots, D^Z, \theta^Z, \phi, \rho, \psi]^{\mathrm{T}}, \qquad (28)$$

where the superscript identifies the observed feature and Z is the number of features.

Before implementing the EKF, we need to compute the dynamics function f and the observation function h, both depending on the state $\mathbf{X}$. From (26) and using (28), the dynamics f of the system can be written as

$$\mathbf{X}_{i+1} = f(\mathbf{X}_i, \mathbf{u}_i) = \begin{bmatrix} D_i^1 + \delta\rho_i \cos\theta_i^1 \\ \theta_i^1 + \delta\theta_i - \frac{\delta\rho_i}{D_i^1}\sin\theta_i^1 \\ D_i^2 + \delta\rho_i \cos\theta_i^2 \\ \theta_i^2 + \delta\theta_i - \frac{\delta\rho_i}{D_i^2}\sin\theta_i^2 \\ \vdots \\ D_i^Z + \delta\rho_i \cos\theta_i^Z \\ \theta_i^Z + \delta\theta_i - \frac{\delta\rho_i}{D_i^Z}\sin\theta_i^Z \\ \phi_i \\ \rho_i \\ \psi_i \end{bmatrix}, \qquad (29)$$

with $\mathbf{u} = [\delta\rho_{\mathrm{R}}, \delta\rho_{\mathrm{L}}]^{\mathrm{T}}$. Regarding the observation function h, from (27) we have

$$h(\mathbf{X}_i) = \begin{bmatrix} \beta_i^1 \\ \beta_i^2 \\ \vdots \\ \beta_i^Z \end{bmatrix} = \begin{bmatrix} \tan^{-1}\left(\frac{-\rho_i \sin(\theta_i^1+\phi_i)}{-D_i^1-\rho_i\cos(\theta_i^1+\phi_i)}\right) - \theta_i^1 - \phi_i - \psi_i \\ \tan^{-1}\left(\frac{-\rho_i \sin(\theta_i^2+\phi_i)}{-D_i^2-\rho_i\cos(\theta_i^2+\phi_i)}\right) - \theta_i^2 - \phi_i - \psi_i \\ \vdots \\ \tan^{-1}\left(\frac{-\rho_i \sin(\theta_i^Z+\phi_i)}{-D_i^Z-\rho_i\cos(\theta_i^Z+\phi_i)}\right) - \theta_i^Z - \phi_i - \psi_i \end{bmatrix}. \quad (30)$$

The previous equations, along with a statistical error model of the odometry (we used that by Chong and Kleeman (1997)), allow us to implement an EKF to estimate $\mathbf{X}$. In order to implement the standard equations of the EKF, we need to compute the Jacobians $\mathbf{F}_x$ and $\mathbf{F}_u$ of the dynamics (29) with respect to the state $\mathbf{X}$ and with respect to encoder readings ($\delta\rho_{\mathrm{R}}$ and $\delta\rho_{\mathrm{L}}$). Furthermore, we need to compute the Jacobian $\mathbf{H}$ of the observation function (30) with respect to $\mathbf{X}$. These matrices are required to implement the EKF (Bar-Shalom and Fortmann 1988) and are given in Appendix A.

Finally, to make the method robust with respect to the system non-linearities, we inflate the covariance matrix characterizing the non-systematic odometry error. As in the Chong–Kleeman model (Chong and Kleeman 1997), it is assumed that the true distance traveled by each wheel during a given time step is a Gaussian random variable. In particular, it is assumed that its mean value is that returned by the wheel encoder and the variance increases linearly with the absolute value of the traveled distance:

$$\delta\rho_{\mathrm{R/L}} = N(\delta\rho^{\mathrm{e}}_{\mathrm{R/L}}, K\delta\rho^{\mathrm{e}}_{\mathrm{R/L}}), \qquad (31)$$

where the subscript R/L denotes the right and left wheel, respectively, the superscript e indicates the value returned by the encoder and K is a parameter characterizing the non-systematic error whose value is needed to implement the filter. In order to inflate this error, we adopt a value for K increased by a factor of five with respect to that estimated by previous experiments (Martinelli et al. 2007).

9. Experimental Results

In our experiments, we adopted a mobile robot with a differential drive system endowed with encoder sensors on the wheels. Furthermore, we equipped the robot with an omnidirectional camera consisting of a KAIDAN 360° One VR hyperbolic mirror and a SONY CCD camera the resolution of

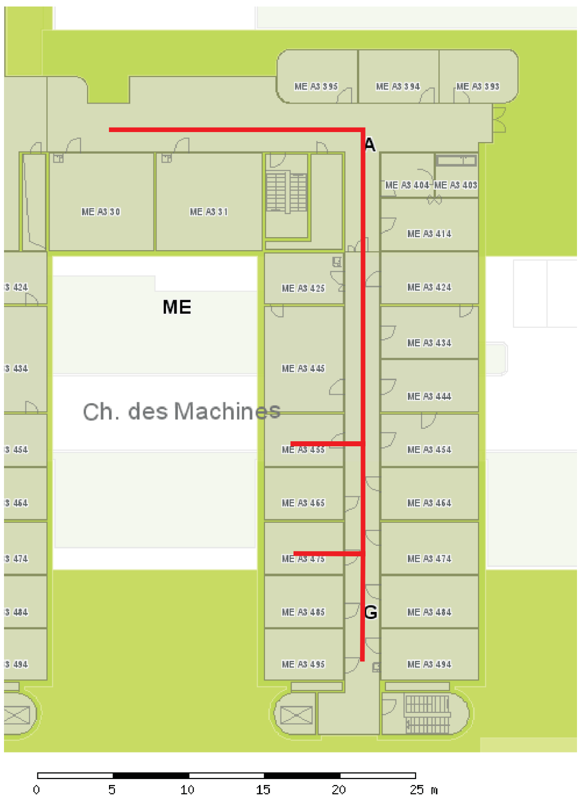

Fig. 20. Floorplan of the institute showing the robot path.

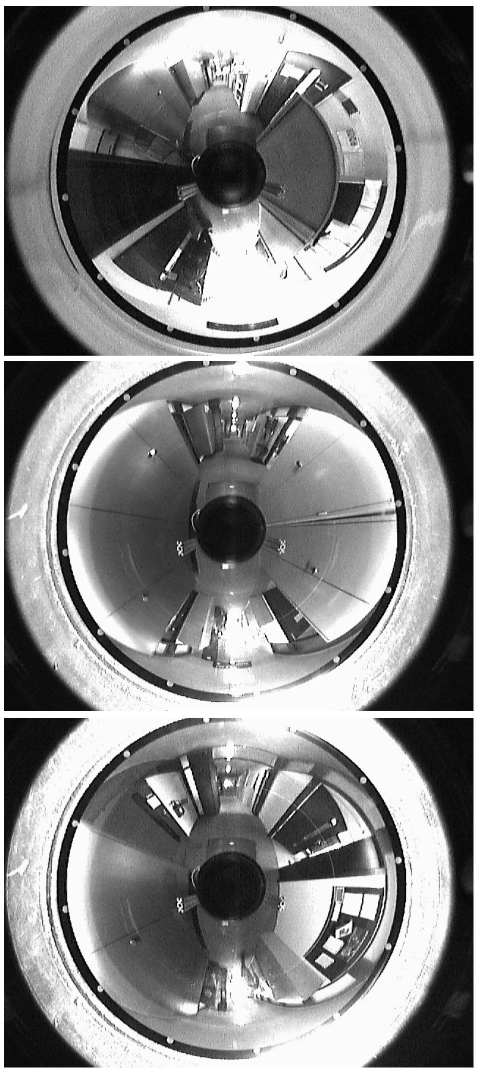

Fig. 21. Omnidirectional images taken at different locations.

640×480 pixels. Our platform is depicted in Figure 1. Finally, observe that the entire algorithm ran in real-time. In particular, all processes (image capture, feature extraction, description computation, feature matching) could be computed in less than 20 ms on a dual-core laptop computer.

9.1. Results on Feature Tracking by using the Proposed Descriptor

In this section, we show the performance of our feature extraction and matching method by capturing pictures from our robot in a real indoor environment. Furthermore, we show that the parameters of the descriptor generalize also outside of the chosen dataset used for "learning" in Section 6.

The robot was moving at about 0.15 m s^{-1} and was acquiring frames at 3 Hz, meaning that during straight paths the traveled distance between two consecutive frames was 5 cm. The robot was moved in the hallway of our institute along the path shown in Figure 20. A total of 1,852 frames were extracted during the whole path. Figure 21 shows three sample images from the dataset. The images show that the illuminations conditions vary strongly.

The result of feature tracking is shown only for the first 150 frames in Figure 22. The video sequence from where this graph was generated can be found in the multimedia extension of this paper (see Appendix B). In the video, every vertical line is labeled with the corresponding number and color with which it appears in Figure 22. The graph shown in Figure 22 was obtained using only the three matching rules described in Sections 4.1, 4.2 and 4.3. No other constraints, such as mutual and topological relations, have been used. This plot refers to a short path of the whole trajectory while the robot was moving in a straight line (between frames 0 and 46), then rotating 180° (between frames 46 and 106) and then moving straight again. As observed, most of the features are correctly tracked over the time. Indeed, most of the lines appear smooth and homogeneous. The lines are used to connect features that belong to the same track. When a new feature is detected, this feature is given a label with progressive numbering and a new line (i.e.

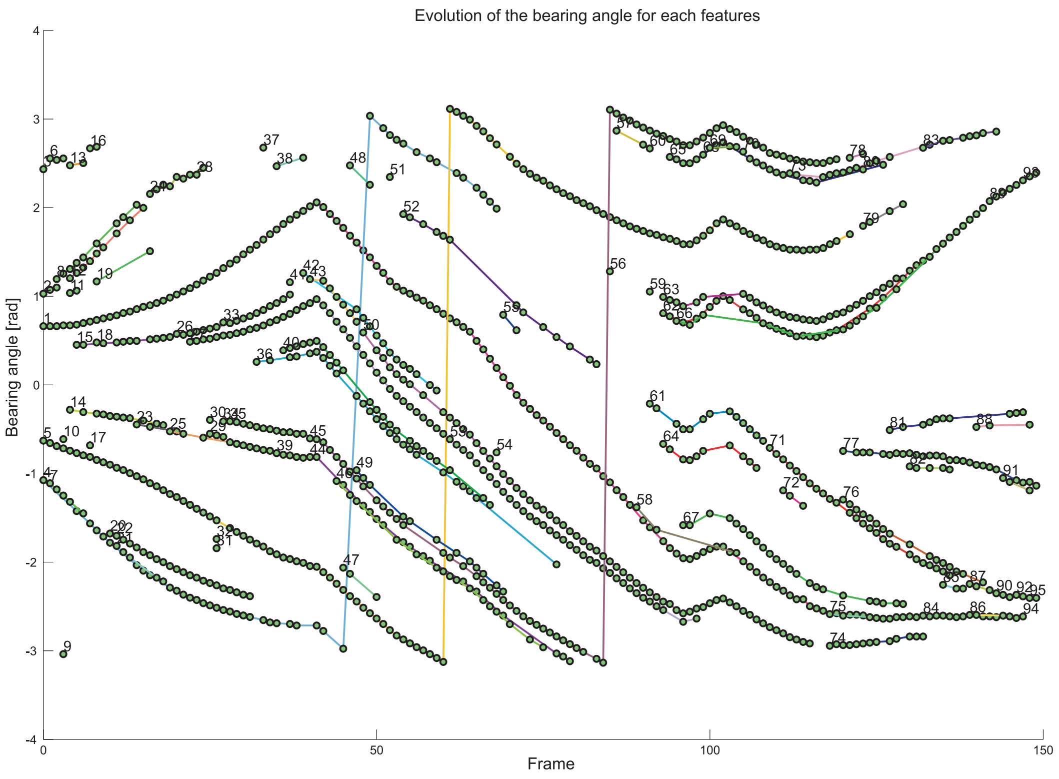

Fig. 22. Feature tracking during the motion of the robot. The y-axis is the angle of sight of each feature and the x-axis is the frame number. Each circle represents a feature detected in the observed frame. Lines represent tracked features. Numbers appear only when a new feature is detected. This plot corresponds to the video contained in the multimedia extension of this paper (see Appendix B).

Table 3. Recognition rate.

Frame interval	Number of matches	Rate of correct matches (%)	Rate of false matches (%)	Rate of false new entries (%)
0–200	735	97.48	0.53	1.98
200–400	972	98.58	0.20	1.22
400–600	823	98.68	0.35	0.96
600–800	857	97.83	0.80	1.37
800–1,000	685	98.13	0.57	1.29
1,000–1,200	740	98.40	0.26	1.33
1,200–1,400	906	98.26	0.43	1.30
1,400–1,600	784	97.75	0.62	1.62
1,600–1,852	771	98.34	0.76	1.89

track) starts from it. In this graph, there are three false matches that occur at the points where two tracks intersect (e.g. at the intersection between tracks 1 and 58, between tracks 84 and 86 and between tracks 65 and 69). Observe that the three huge jumps in the graph are not false matches; they are only due to the angle transition from $-\pi$ to π.

Observe that our method was able to match features even when their correspondents were not found in the previous

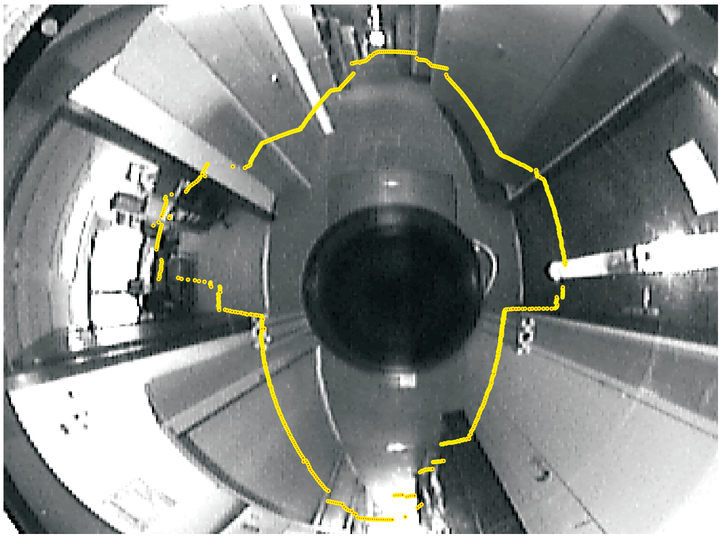

Fig. 23. Laser points reprojected onto the omnidirectional image before calibration. The edges in the omnidirectional image do not correctly intersect the corners of the laser scan.

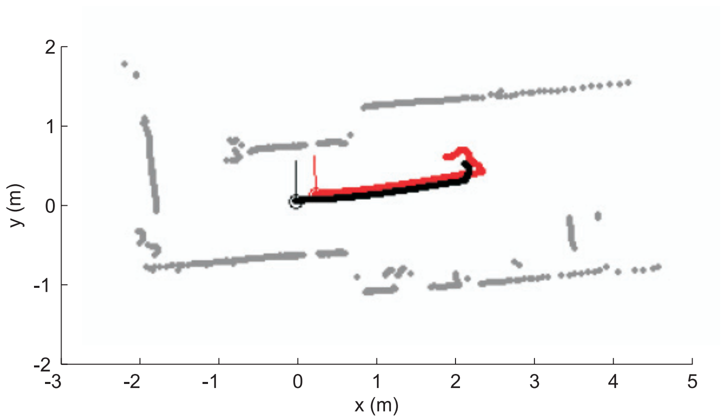

Fig. 24. The path performed by the robot during self-calibration, i.e. straight path followed by a rotation.

frames. This can be seen by observing that sometimes circles are missing on the tracks (look, for instance, at track 52). When a correspondence is not found in the previous frame, our tracking algorithm starts looking into all previous frames (actually up to 20 frames back) and stops when a correspondence is found.

By examining the graph, one can see that some tracks are suddenly given different numbers. For instance, observe that feature 1, which is the first detected feature and starts at frame 0, is tracked correctly until frame 120 and is then labeled as feature 75. This is because at this frame no correspondence was found and the feature was then labeled as a new entry (but, in fact, is a false new entry). Another example is feature 15 that is then labeled as feature 18 and 26. By a careful visual inspection, one can find only a few other examples of false new entries. Indeed, tracks that at a first glance seem to be given different numbers, belong in fact to other features that are very close to the observed feature.

After visually inspecting every single frame of the whole video sequence (composed of 1,852 frames), we found 35 false matches and 101 false new entries. The detection rate over the entire dataset is shown in Table 3 at intervals of 200 frames. Comparing these errors with the 7,408 corresponding pairs detected by the algorithm over the whole video sequence, we had 1.8% of mismatches. Furthermore, we found that false matches occurred every time the camera was facing objects

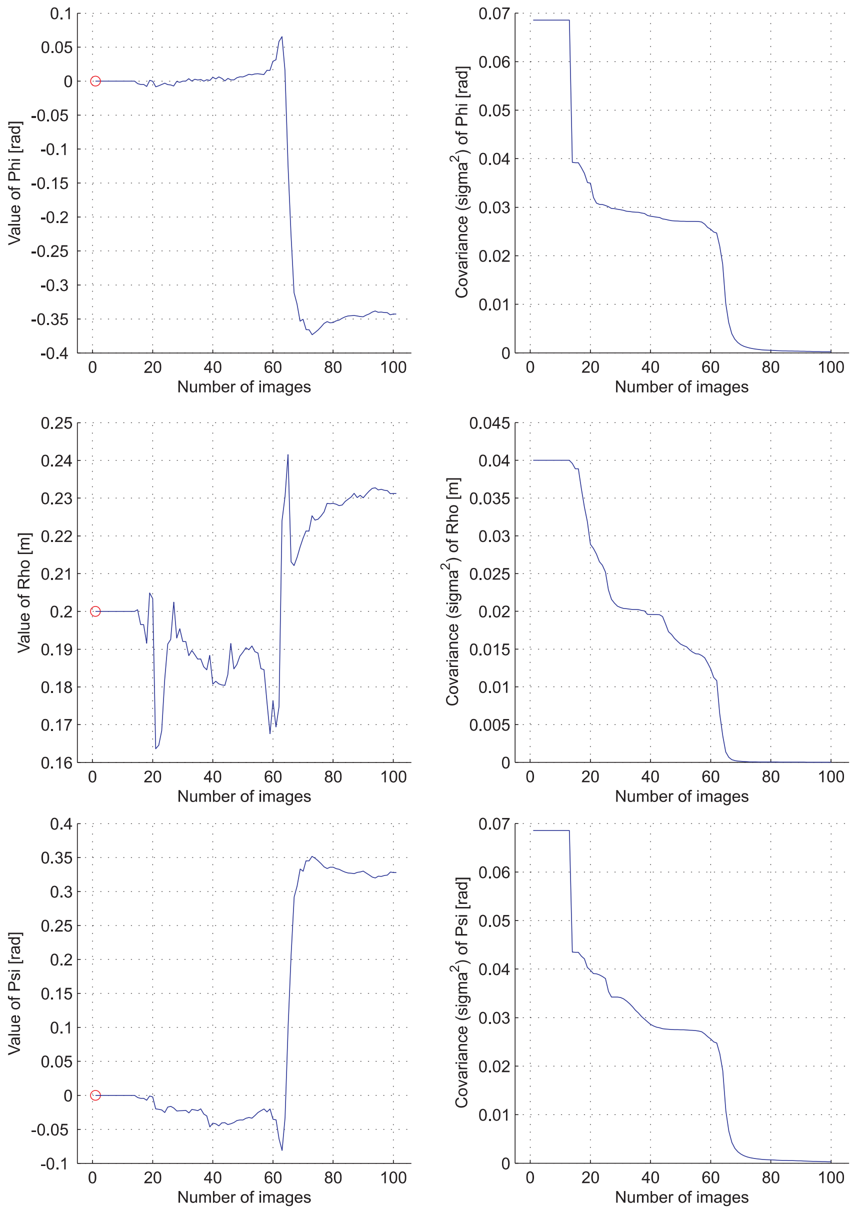

Fig. 25. The values of ϕ, ρ, ψ, $\sigma_\phi{}^2$, $\sigma_\rho{}^2$, $\sigma_\psi{}^2$ as a function of the frame number. The distance traveled between two frames is about 3.8 cm.

with a repetitive texture (as in Figure 10 or in the second image of Figure 21). Thus, ambiguity was caused by the presence of vertical elements which repeat almost identically in the same image. On the other hand, a few false new entries occurred when the displacement of the robot between two successive images was too large. However, observe that when a feature matches with no other feature in previous frames, it is better to believe this feature to be new than commit a false matching.

As we already mentioned above, the results reported in this section were obtained using only the three matching rules described in Sections 4.1, 4.2 and 4.3. Obviously, the performance of tracking could be further improved by adding other constraints such as mutual and topological relations among features.

9.2. Calibration Results

In our experiments, we adopted the same mobile robot and omnidirectional camera described in Section 9.1. Furthermore,

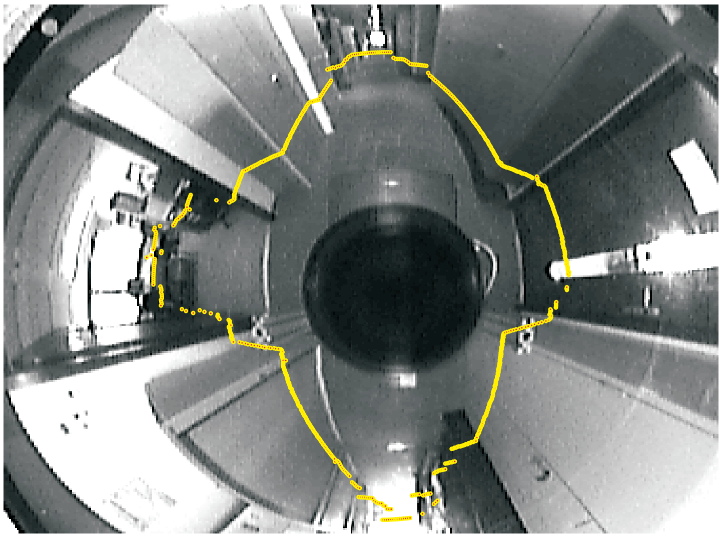

Fig. 26. Laser points reprojected onto the omnidirectional image after calibration. The edges in the omnidirectional image appropriately intersect the corners of the laser scan.

two laser range finders (model SICK LMS 200) were also installed on the robot. Observe that these laser scanners are used in our experiments just for comparison and are considered already calibrated with the odometry system according to the specifications provided by the manufacturer.

For our experiments, we positioned the omnidirectional camera on our robot as in Figure 1 and we measured its position relative to the robot manually. We measured the following values: $\phi \simeq 0$ rad, $\rho \simeq 0.2$ m and $\psi \simeq 0$ rad. Figure 23 shows the laser points reprojected onto the omnidirectional image using the above values. As observed, because the relative pose of the camera and the robot references is not measured accurately, the edges in the omnidirectional image do not correctly intersect the corners of the laser scan. However, we used these rough values to initialize our EKF.

The trajectory chosen for the experiments consisted of a straight path, approximately 2.3 m long, and a 180° rotation about the center of the wheels. The trajectory is depicted in Figure 24. For this experiments, about 10 vertical lines were tracked.

The values of ϕ, ρ, ψ estimated during the motion are plotted as a function of the frame number in Figure 25. The covariances σ_ϕ, σ_ρ, σ_ψ are also plotted. Observe that after about 60 frames (corresponding to about 2.3 m of navigation) the parameters start suddenly to converge to a stable value. The resulting estimated parameters are $\phi = -0.34$ rad, $\rho = 0.23$ m and $\psi = 0.33$ rad. The sudden jump starting at frame 60 actually occurs when the robot starts to rotate.

As demonstrated in Section 8.1, when the robot accomplishes a straight trajectory far from the feature, it is possible to observe the sum $\phi + \psi$, D and θ. Once the robot starts to rotate, D does not change. Furthermore, the sum $\phi + \psi$ is time independent. On the other hand, with the pure rotation λ, γ and ψ are observable. Therefore, as the robot starts to rotate the value of ρ is determined from the values of λ and D. Furthermore, both ϕ and ψ are determined. Note that during the jump the sum $\phi + \psi$ is constant. As it was already pointed out by Martinelli et al. (2006), the convergence is very fast when the robot performs trajectories alternating short straight paths and pure rotations about the robot origin.

Furthermore, extensive simulations by Martinelli et al. (2006), show that even when the estimation process starts by first moving the robot along a straight path (i.e. during this initial phase the overall state is unobservable) the EKF is always able to recover, at the end, the true values of the parameters. Several experiments and many simulations showed consistency among the results.

Figure 26 shows the laser points reprojected onto the omnidirectional after calibration. As observed, the calibration parameters are well estimated. Indeed, the edges in the omnidirectional image appropriately intersect the corners of the laser scan.

10. Conclusion

In this paper, we have presented a robust method for matching vertical lines among omnidirectional images. Furthermore, in order to make such a method usable in the framework of indoor mobile robotics, we introduced a new simple strategy to extrinsically self-calibrating the omnidirectional sensor with the odometry reference system.

Concerning the first part, the basic idea to achieve robust feature matching consists of creating a descriptor which is very distinctive. Furthermore, this descriptor is invariant to rotation and slight changes of illumination. The performance of the descriptor was validated through a deep analysis and an experiment of feature tracking was also carried out. The performance of tracking was very good as many features were correctly detected and tracked over a long time. Furthermore, because the results were obtained using only the three matching rules described in Section 4, we expect that the performance would be notably improved by adding other constraints such as mutual and topological relations among features.

Concerning the second part, we adopted the visual tracking method to implement our strategy of camera–robot self-calibration. The novelty of the method is the use of an EKF that automatically estimates the calibration parameters while the robot is moving. The present strategy had already been proposed in our previous work (Martinelli et al. 2006). In our previous work we provided the equations and performed several experiments on both simulated and real data by tracking only a single feature. In that work, we also showed that by choosing suitable trajectories (alternating straight path with pure rotations), it is possible to estimate the calibration parameters with high accuracy by moving the robot along very short paths (a few meters). In this paper, we extended our previous work to cope with multiple features and showed that by tracking multiple features the convergence is faster than using a single feature. Furthermore, the calibration parameters start to converge when the robot undergoes a pure rotation after moving along a straight path. Although experiments have been conducted using an omnidirectional camera, more generally the proposed method can be adopted to calibrate any robot bearing sensor.

The two contributions introduced in this paper allow the use of an omnidirectional camera in the framework of mobile robotics, in particular in combination with odometry data.

Acknowledgments

The research leading to these results has received funding from the European Community's Sixth Framework Programme (FP6/2003-2006) under grant agreement no. FP6-2006-IST-6-045350 (robots@home) and from European project BACS (Bayesian Approach to Cognitive Systems).

Appendix A

A.1. Straight Motion and Far Feature

We want to prove that the state $X_{\mathrm{sf}} \equiv [D, \theta, \eta]^{\mathrm{T}}$ satisfying the dynamics $\dot{X}_{\mathrm{sf}} = vf$ with $f = [\cos\theta, -\sin\theta/D, 0]^{\mathrm{T}}$ is observable when the observation is $\beta = -\theta - \eta$.

It is sufficient to show that the gradients $dL^0\beta$, $dL^1_f\beta$, $dL^2_{ff}\beta$ are independent.

We have

$$L^0\beta = \beta = -\theta - \eta,$$

$$dL^0\beta = [0, -1, -1],$$

$$L^1_f\beta = \frac{\sin\theta}{D},$$

$$dL^1_f\beta = \left[-\frac{\sin\theta}{D^2}, \frac{\cos\theta}{D}, 0\right],$$

$$L^2_{ff}\beta = -\frac{\sin 2\theta}{D^2},$$

$$dL^2_{ff}\beta = \left[2\frac{\sin 2\theta}{D^3}, -2\frac{\cos 2\theta}{D^2}, 0\right].$$

The determinant of the matrix containing these gradients is

$$\det\begin{bmatrix} dL^0\beta \\ dL^1_f\beta \\ dL^2_{ff}\beta \end{bmatrix} = 2\frac{\sin\theta}{D^4},$$

which is different from zero when $\theta \neq n\pi$

A.2. Pure Rotation and Close Feature

We want to prove that the state $X_{\mathrm{rc}} \equiv [\lambda, \gamma, \psi]^{\mathrm{T}}$ satisfying the dynamics $\dot{X}_{\mathrm{rc}} = \omega f$ with $f = [0, 1, 0]^{\mathrm{T}}$ is observable when the observation is

$$\beta = \operatorname{atan}\left(\frac{sin\gamma}{\lambda + \cos\gamma}\right) - \gamma - \psi.$$

It is sufficient to show that the gradients $dL^0\beta$, $dL^1_f\beta$, $dL^2_{ff}\beta$ are independent. By directly computing these gradients we finally obtain

$$\det\begin{bmatrix} dL^0\beta \\ dL^1_f\beta \\ dL^2_{ff}\beta \end{bmatrix} = -\frac{\lambda(\lambda^2 - 1)}{(\lambda^2 + 2\lambda\cos\gamma + 1)^3}.$$

By assuming $\lambda > 1$ (i.e. $D > \rho$) these gradients are independent and the observability holds.

A.3. Straight Motion and Close Feature

The state X satisfying the dynamics: $\dot{X} = vf_1$ with f_1 defined in (22) is not observable when the observation is

$$\beta = \operatorname{atan}\left(\frac{\sin\gamma}{\lambda + \cos\gamma}\right) - \gamma - \psi.$$

To prove this we compute the determinant

$$\det \begin{bmatrix} dL^0\beta \\ dL^1_{f_1}\beta \\ dL^2_{f_1 f_1}\beta \\ dL^3_{f_1 f_1 f_1}\beta \\ dL^4_{f_1 f_1 f_1 f_1}\beta \end{bmatrix}.$$

This computation was performed by using the symbolic tool of Matlab and the result obtained is zero.

A.4. Jacobians

The Jacobians $\mathbf{F}_x$ and $\mathbf{F}_u$ of the dynamics are

$$F_x = \begin{bmatrix} A^1 & 0 & \cdots & 0 & 0 & 0 & 0 \\ 0 & A^2 & \cdots & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \cdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & A^Z & 0 & 0 & 0 \\ 0 & 0 & \cdots & 0 & 1 & 0 & 0 \\ 0 & 0 & \cdots & 0 & 0 & 1 & 0 \\ 0 & 0 & \cdots & 0 & 0 & 0 & 1 \end{bmatrix},$$

and

$$F_u = \begin{bmatrix} B^1 \\ B^2 \\ \vdots \\ B^Z \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

with

$$\begin{aligned} A^i &= \begin{bmatrix} 1 & -\delta\rho\sin\theta^i \\ \dfrac{\delta\rho}{D^{i\,2}}\sin\theta^i & 1 - \dfrac{\delta\rho}{D^i}\cos\theta^i \end{bmatrix}, \\ B^i &= \begin{bmatrix} \dfrac{\cos\theta^i}{2} & \dfrac{\cos\theta^i}{2} \\ \dfrac{1}{e} - \dfrac{\sin\theta^i}{2D^i} & -\dfrac{1}{e} - \dfrac{\sin\theta}{2D^i} \end{bmatrix}. \end{aligned} \tag{32}$$

The Jacobian $\mathbf{H}$ of the observations is

$$\mathbf{H} = \begin{bmatrix} H1^1 & H2^1 & 0 & 0 & \cdots & 0 & 0 & H3^1 & H4^1 & -1 \\ 0 & 0 & H1^2 & H2^2 & \cdots & 0 & 0 & H3^2 & H4^2 & -1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & H1^Z & H2^Z & H3^Z & H4^Z & -1 \end{bmatrix}$$

with

$$\begin{aligned} H1^i &= \frac{-\rho\sin(\theta^i + \phi)}{D^{i\,2} + 2\rho D^i\cos(\theta^i + \phi) + \rho^2}, \\ H2^i &= \frac{-D^i\rho\cos(\theta^i + \phi) - D^{i\,2}}{D^{i\,2} + 2\rho D^i\cos(\theta^i + \phi) + \rho^2}, \\ H3^i &= \frac{-D^i\rho\cos(\theta^i + \phi) - D^{i\,2}}{D^{i\,2} + 2\rho D^i\cos(\theta^i + \phi) + \rho^2}, \\ H4^i &= \frac{D^i\sin(\theta^i + \phi)}{D^{i\,2} + 2\rho D^i\cos(\theta^i + \phi) + \rho^2}. \end{aligned}$$

Appendix B: Index to Multimedia Extensions

The multimedia extension page is found at http://www.ijrr.org

Table of Multimedia Extensions

Extension	Type	Description
1	Video	Feature tracking

References

Ayache, N. (1990) *Stereovision and Sensor Fusion*. Cambridge, MA, The MIT Press.
Ayache, N. and Faugeras, O. (1987) Building a consistent 3D representation of a mobile robot environment by combining

multiple stereo views. *Proceedings of IJCAI*, pp. 808–810.

Baillard, C., Schmid, C., Zisserman, A. and Fitzgibbon, A. (1999) Automatic line matching and 3D reconstruction of buildings from multiple views. *Proceedings of the SPRS Conference on Automatic Extraction of GIS Objects from Digital Imagery*, (*International Archives of Photogrammetry and Remote Sensing*, Vol. 32, Part 3-2W5). International Society for Photogrammetry and Remote Sensing, pp. 69–80.

Bar-Shalom, Y. and Fortmann, T. E. (1988) *Tracking and Data Association* (*Mathematics in Science and Engineering*, Vol. 179). New York, Academic Press.

Baumberg, A. (2000) Reliable feature matching across widely separated views. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Hilton Head, SC, pp. 774–781.

Brassart, E., Delahoche, L., Cauchois, C., Drocourt, C., Pegard, C. and Mouaddib, E. M. (2000) Experimental results got with the omnidirectional vision sensor: SYCLOP. *Proceedings of the International Workshop on Omnidirectional Vision (OMNIVIS 2000)*.

Briggs, A., Li, Y., Scharstein, D. and Wilder, M. (2006) Robot navigation using 1D panoramic images. *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 2006)*, Orlando, FL, May 2006.

Chong, K. S. and Kleeman, L. (1997) Accurate odometry and error modelling for a mobile robot. *Proceedings of the International Conference on Robotics and Automation*, pp. 2783–2788.

Crowley, J. and Stelmazyk, P. (1990) Measurement and integration of 3D structures by tracking edge lines. *Proceedings of ECCV*, pp. 269–280.

Deriche, R. and Faugeras, O. (1990) Tracking line segments. *Proceedings of ECCV*, pp. 259–267.

Goedeme, T., Tuytelaars, T. and Van Gool, L. (2004) Fast wide baseline matching with constrained camera position. *Proceedings of the Conference on Computer Vision and Pattern Recognition*, Washington, DC, pp. 24–29.

Gonzalez, R. and Woods, R. (2002) *Digital Image Processing*, 2nd edition. Reading, MA, Addison-Wesley/Englewood Cliffs, NJ, Prentice-Hall.

Gros, P. (1995) Matching and clustering: two steps towards object modelling in computer vision. *The International Journal of Robotics Research*, **14**(6): 633–642.

Hermann, R. and Krener, A. J. (1977) Nonlinear controllability and observability. *IEEE Transactions on Automatic Control*, **22**(5): 728–740.

Hesch, J. A., Mourikis, A. I. and Roumeliotis, S. I. (2008) Determining the camera to robot-body transformation from planar mirror reflections. *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2008)*, Nice, France, September 2008.

Horaud, R. and Skordas, T. (1989) Stereo correspondence through feature grouping and maximal cliques. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **11**(11): 1168–1180.

Huttenlocher, D. P., Klanderman, G. A. and Rucklidge, W. J. (1993) Comparing images using the Hausdorff distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **15**(9): 850–863.

Isidori, A. (1995) *Nonlinear Control Systems*, 3rd edition. Berlin, Springer.

Kadir, T., Zisserman, A. and Brady, M. (2004) An affine invariant salient region detector. *Proceedings of the 7th European Conference on Computer Vision*, Prague, Czech Republic, Vol. I, pp. 228–241.

Lindeberg, T. (1998) Feature detection with automatic scale selection. *International Journal of Computer Vision*, **30**(2): 79–116.

Lowe, D. G. (2004) Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, **60**(2): 91–110.

Martinelli, A., Scaramuzza, D. and Siegwart, R. (2006) Automatic self-calibration of a vision system during robot motion. *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 2006)*.

Martinelli, A., Tomatis, N. and Siegwart, R. (2007) Simultaneous localization and odometry self calibration for mobile robot. *Autonomous Robots*, **22**: 75–85.

Matas, J., Chum, O., Urban, M. and Pajdla, T. (2002) Robust wide baseline stereo from maximally stable extremal regions. *Proceedings of the 13th British Machine Vision Conference*, Cardiff, UK, pp. 384–393.

Mauthner, T., Fraundorfer, F. and Bischof, H. (2006) Region matching for omnidirectional images using virtual camera planes. *Proceedings of the Computer Vision Winter Workshop 2006*, Telc, Czech Republic.

Medioni, G. and Nevatia, R. (1985) Segment-based stereo matching. *Computer Vision, Graphics and Image Processing*, **31**: 2–18.

Micusik, B. and Pajdla, T. (2006) Structure from motion with wide circular field of view cameras. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **28**(7): 1135–1149.

Mikolajczyk, K. and Schmid, C. (2001) Indexing based on scale invariant interest points. *Proceedings of the 8th International Conference on Computer Vision*, Vancouver, Canada, pp. 525–531.

Mikolajczyk, K. and Schmid, C. (2002) An affine invariant interest point detector. *Proceedings of 7th European Conference on Computer Vision*, Copenhagen, Denmark.

Mirzaei, F. M. and Roumeliotis, S. I. (2007) A Kalman filter-based algorithm for IMU-camera callibration. *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2007)*, San Diego, CA, October 2007, pp. 2427–2434.

Nayar, S. K. (1997) Catadioptric omnidirectional camera. *CVPR '97: Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition (CVPR '97)*. Washington, DC, IEEE Computer Society, p. 482.

Prasser, D., Wyeth, G. and Milford, M. J. (2004) Experiments in outdoor operation of RatSLAM. *Proceedings of the Australian Conference on Robotics and Automation*, Canberra, Australia.

Rahman, M. M., Bhattacharya, P. and Desai, B. C. (2005) Similarity searching in image retrieval with statistical distance measures and supervised learning. *Pattern Recognition and Data Mining* (*Lecture Notes in Computer Science*, Vol. 3686). Berlin, Springer, pp. 315–324.

Rubner, Y., Tomasi, C. and Guibas, L. J. (2000) The earth mover's distance as a metric for image retrieval. *International Journal of Computer Vision*, **40**(2): 99–121.

Rubner, Y. et al. (2001) Empirical evaluation of dissimilarity measures for color and texture. *Computer Vision and Image Understanding*, **84**(1): 25–43.

Sagues, C., Murillo, A. C., Guerrero, J. J., Goedeme, T., Tuytelaars, T. and Van Gool, L. (2006) Hierarchical localization by matching vertical lines in omnidirectional images. *Robotics and Autonomous Systems*, **55**(5): 372–382.

Scaramuzza, D., Criblez, N., Martinelli, A. and Siegwart, R. (2007a) Robust feature extraction and matching for omnidirectional images. *Proceedings of the 6th International Conference on Field and Service Robotics (FSR 2007)*, Chamonix, France, July 2007.

Scaramuzza, D., Harati, A. and Siegwart, R. (2007b) Extrinsic self calibration of a camera and a 3d laser range finder from natural scenes. *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2007)*, San Diego, CA, October 2007.

Tell, D. and Carlsson, S. (2000) Wide baseline point matching using affine invariants computed from intensity profiles. *Proceedings of the European Conference on Computer Vision*, Dublin, Ireland, pp. 814–828.

Tuytelaars, T. and Van Gool, L. (2004) Matching widely separated views based on affine invariant regions. *International Journal of Computer Vision*, **1**(59): 61–85.

Venkateswar, V. and Chellappa, R. (1995) Hierarchical stereo and motion correspondence using feature groupings. *International Journal of Computer Vision*, **15**(3): 245–269.

Yagi, Y. and Yachida, M. (1991) Real-time generation of environmental map and obstacle avoidance using omnidirectional image sensor with conic mirror. *Proceedings of CVPR'91*, pp. 160–165.

Zhang, Z. (1994) Token tracking in a cluttered scene. *Image and Vision Computing*, **12**(2): 110–120.

Zhang, Q. and Pless, R. (2004) Constraints for heterogeneous sensor auto-calibration. *Proceedings of the IEEE Workshop on Realtime 3D Sensors and Their Use*, pp. 38–43.