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

Davide Scaramuzza[1], Agostino Martinelli[2], Roland Siegwart[1]

[1] Swiss Federal Institute of Technology (ETH), Zurich, Switzerland
[2] INRIA Rhone-Alpes, Grenoble, France

***Abstract*—This paper introduces 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. The first part of this paper describes how to build the feature descriptor. We show that the descriptor is very distinctive and is invariant to rotation and slight changes of 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 of 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.**

***Index Terms*—omnidirectional camera, visual tracking, feature descriptor, extrinsic camera calibration.**

I. Introduction

A. Motivation and Contribution

One of the challenges in mobile robotics is designing autonomous vehicles able to perform high level tasks despite of the quality/cost of the sensors. Vision sensors and encoder sensors are in general cheap and suitable for indoor navigation. In particular, regarding vision, omnidirectional camera is very effective due to the panoramic view 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 to extrinsically self-calibrating the omnidirectional sensor with the odometry reference system.

The contributions of the paper are therefore the following two: 1) Introduction of a new descriptor for matching vertical lines among two or more images from an omnidirectional camera; 2) Introduction of a simple strategy to extrinsically calibrate an omnidirectional camera with the odometry system.

B. 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 decades, the feature correspondence problem has been largely investigated 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 (see [Matas02], [Kadir04], [Mikolajczyk01], [Lowe04], [Mikolajczyk98], [Mikolajczyk02], [Baumberg00], [Tuytelaars04]). 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 directly applied 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 (see [Nayar97]). An application of this approach can be found in [Mauthner06]. 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. In [Micusik06], for instance, the authors check the candidate correspondences between two views using 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 (see [Briggs06]). These images are generated by converting each omnidirectional picture into a 1D circular

image, which is obtained by averaging the scan lines of a cylindrical panorama. Then, 1D 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 (Fig. 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 works 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) (see [Medioni85], [Ayache90], [Zhang94]). Some such as [Crowley90], [Deriche90], [Huttenlocher93] 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 (see [Ayache87], [Horaud89], [Gros95], [Venkateswar95]). 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.

Besides 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 [Gonzalez02])).
An interesting approach was proposed in [Baillard99]. Besides 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 lines neighborhoods was then assessed by cross-correlation at the corresponding points.

A novel approach, using the intensity profile along the line segment, was proposed in [Tell00]. 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 also proposed in [Goedeme04] and is application on robot localization with omnidirectional imaging was demonstrated in [Sagues06].

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 (see [Yagi91], [Brassart00], [Prasser04]). 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 some of 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 respectively 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 recently received significant attention and a number of approaches have been developed (e.g., for IMU-camera [Mirzaei07], robot-body camera [Hesch08], or laser scanner-camera [Zhang04], [Scaramuzza07b]). 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.

C. Outline

This paper proposes 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 nonlinearities. In particular, it is theoretically shown that only one feature suffices to perform the calibration.
This paper extends our two previous works [Scaramuzza07] and [Martinelli06].

The present document is organized as follows. First, we describe our procedure to extract vertical lines (Section II) and build the feature descriptor (Section III). In Section IV we provide our matching rules while in Sections V and VI we

characterize the performance of the descriptor. In Section VII, we describe the calibration problem, while in Section VIII we provide the equations to build the EKF. In Section IX, we will present experimental results which validate both the theoretical contributions.

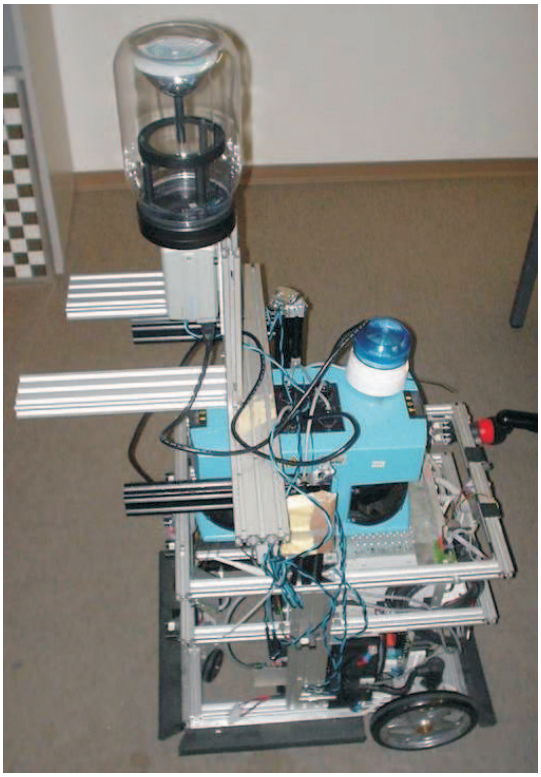

Fig. 1. The robot used in our experimets equipped with encoder sensors, omnidirectional camera, and two laser range finders

II. Vertical Line Extraction

Our platform consists of a wheeled robot equipped with a catadioptric omnidirectional camera (see Fig. 1). The main advantage of such kind 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 (Fig. 2) In this section, we detail our procedure to extract prominent vertical lines. Our procedure consists of five steps.

The first step toward vertical line extraction is the detection of the image center (i.e. the point where all radial lines intersect in). 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 done 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

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

Then, we do 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 Fig. 3.

The third step consists in detecting the most reliable vertical lines. To this end, we divide the omnidirectional image into 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 Fig. 4. Then, we apply non-maxima suppression to identify all local peaks.

The final step is histogram thresholding. As observed in Fig. 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 pixel of the observed line. As observed in Fig. 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 thresold is shown if Fig. 5.

III. Building the Descriptor

In Section IV, we will 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 next subsections, we describe how we built our descriptor.

A. 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 each area is equal to $(R_{max} - R_{min})/6$ (see Fig. 6). Then, we smooth each area with a Gaussian window with

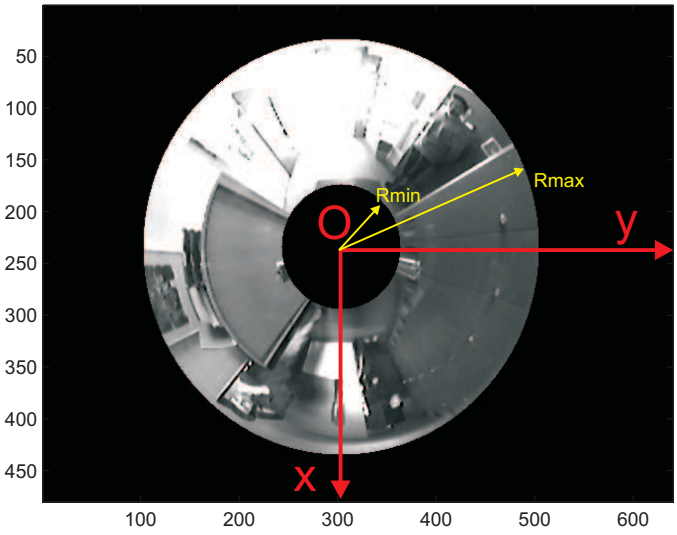

Fig. 2. An image taken by our omnidirectional camera. We used a KAIDAN-360-One-VR hyperbolic mirror and a SONY CCD camera the resolution of 640480 pixels. The camera used in shown in Fig. 1.

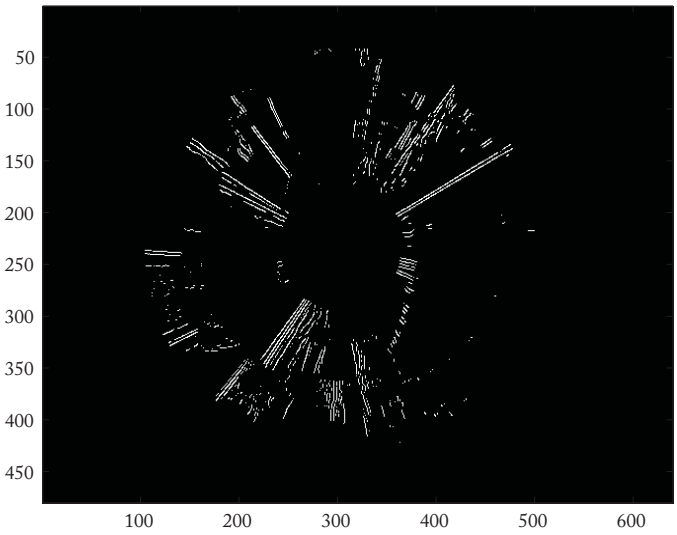

Fig. 3. Edge image of Fig. 2.

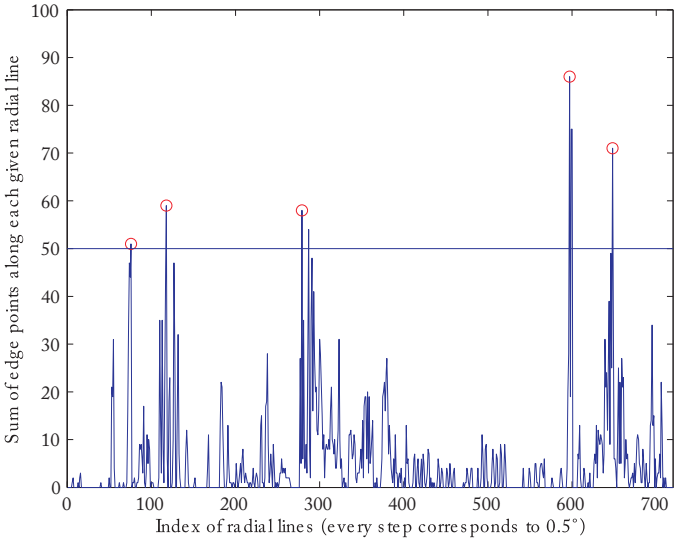

Fig. 4. Number of binary pixels voting for a given orientation angle.

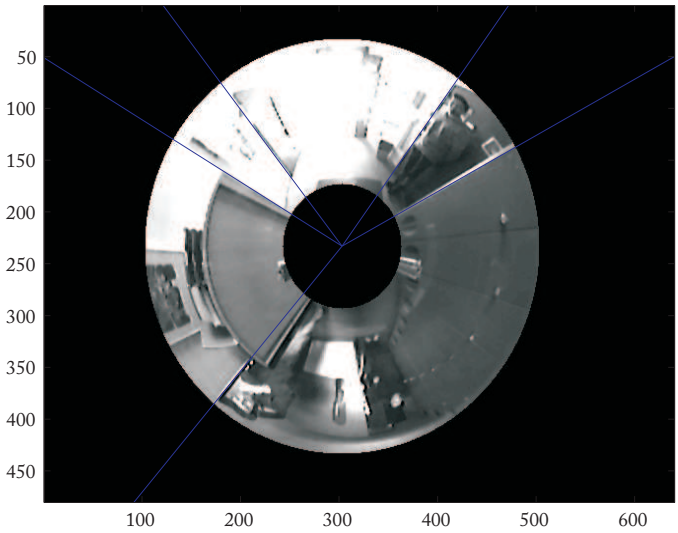

Fig. 5. Extraction of the most reliable vertical features from an omnidirectional image.

$\sigma_G = r_a/3$ and compute the image gradients (magnitude $\mathbf{M}$ and orientation $\mathbf{\Phi}$) within each of these areas.
Concerning rotation invariance, this is achieved by redefining the gradient orientation $\mathbf{\Phi}$ of all points relatively to the radial line's angle θ (see Fig. 6).

B. Orientation Histograms

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

For each side of each circular area, we compute the gradient orientation histogram (Fig. 8). The whole orientation space (from -π 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 relatively to the radial line's angle (Section III-A).

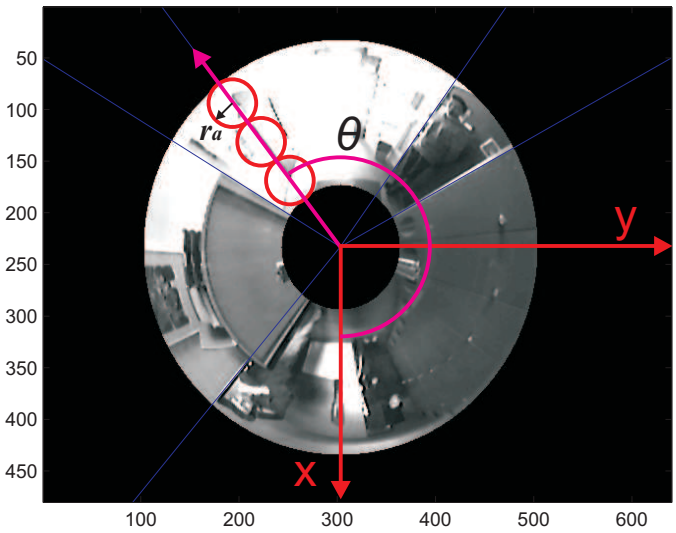

Fig. 6. Extraction of the circular areas. To achieve rotation invariance, the gradient orientation $\mathbf{\Phi}$ of all points is redefined relatively to the radial line's angle θ.

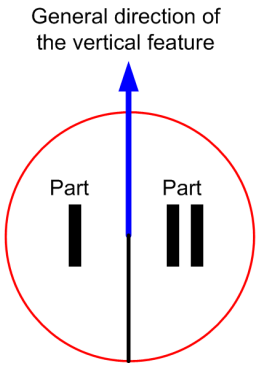

Fig. 7. The two sections of a circular area.

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

$$\begin{aligned}\mathbf{H_1} &= [\mathbf{H_{1,L}}, \mathbf{H_{1,R}}] \\ \mathbf{H_2} &= [\mathbf{H_{2,L}}, \mathbf{H_{2,R}}] \\ \mathbf{H_3} &= [\mathbf{H_{3,L}}, \mathbf{H_{3,R}}]\end{aligned} \tag{3}$$

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

C. 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 3D surfaces with different orientations by different amounts. These effects can cause a large change

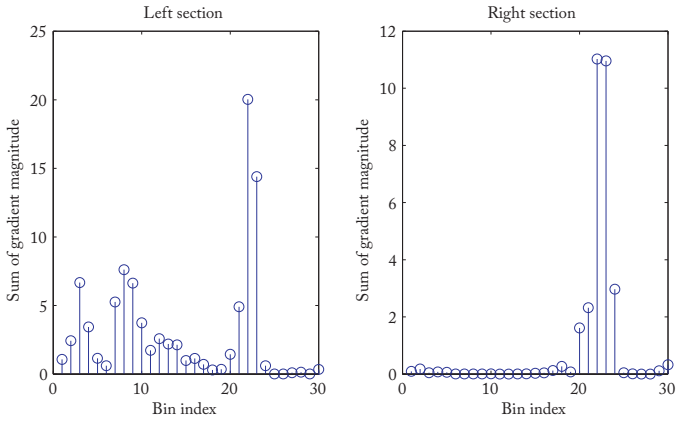

Fig. 8. An example of gradient orientation histograms for the left and right sides of a circular area.

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 will be justified in Section VI.

To recap, our descriptor is an N-element vector containing the gradient orientation histograms of the circular areas. In our setup, we extract 3 circular areas from each vertical feature and use 32 bins for each histogram; thus the length of the descriptor is

$$N = 3areas \cdot 2parts \cdot 32bins = 192 \tag{5}$$

Observe that all feature descriptors are the same length.

IV. 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 contents of 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, fall the Minkoski-form distance, the Jeffrey divergence, the χ^2 statistics, and the Bhattacharya distance. Among the *cross-bin* measures, one of the most used is the Quadratic-form distance. An exhaustive review of all these methods can be found in [Bhattacharya05], [Rubner00], [Rubner01].

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 definition of distance, the correspondent of a feature, in the observed image, is expected to be the one, 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 $\mathbf{B_j}$ $(j = 1, 2, \cdots, N_B)$.
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.

A. 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.

B. 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:

$$minD_i \leq F_2 \cdot < D_i > \tag{9}$$

where $< D_i >$ is the mean value of D_i and F_2 clearly ranges from 0 to 1. This criterion comes out of experimental results.

TABLE I
THE DISTANCES BETWEEN THE DESCRIPTOR $\mathbf{A_1}$ AT TIME t_A AND ALL DESCRIPTORS $\mathbf{B_j}$, $j = 1, 2, .., N_B$ AT TIME t_B

B1	B2	B3	B4	B5
0.57	0.72	0.74	0.78	0.83

TABLE II
THE PARAMETERS USED BY OUR ALGORITHM WITH THEIR EMPIRICAL VALUES

F_1 = 1.05	F_2 = 0.75	F_3 = 0.8

C. 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 0 to 1. 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.

In Table I, 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 test 1 and 3 but not 2.

Factors F_1, F_2, F_3 were determined experimentally. The values used in our experiments are shown in Table II. The choice of these values will be motivated in Section VI.

V. 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 radial line's descriptor 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

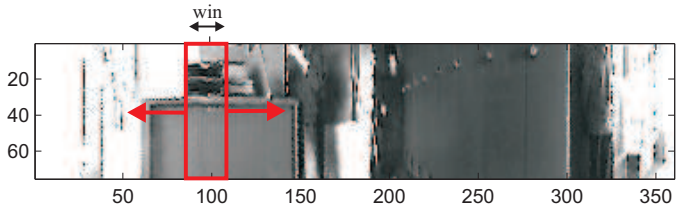

Fig. 9. This is the same image of Fig. 6 after unwrapping into a cylindrical panorama. The rectangular region used to compute SSD and ZNCC is also shown.

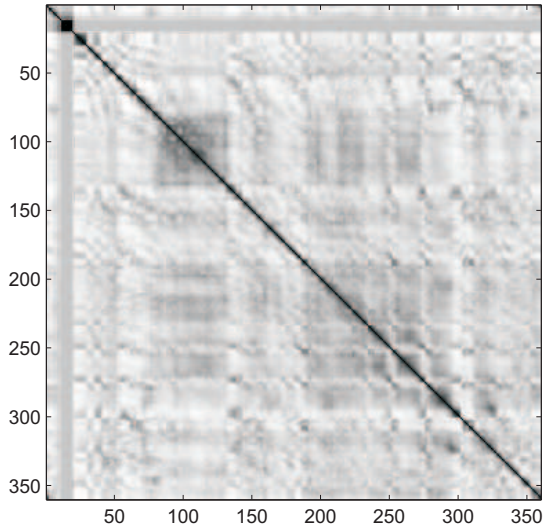

Fig. 10. Similarity matrix for descriptors.

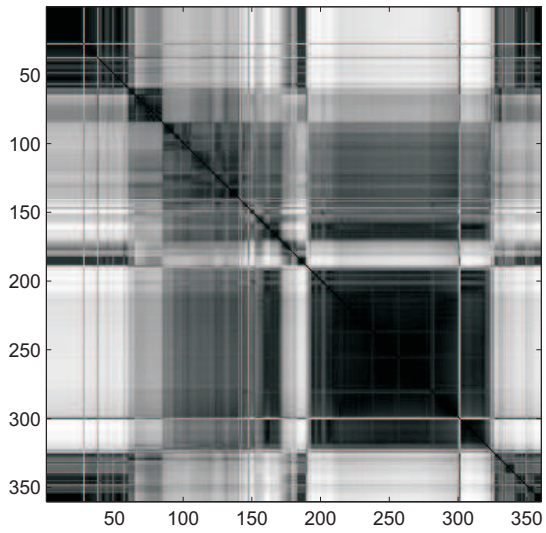

Fig. 11. Similarity matrix for SSD.

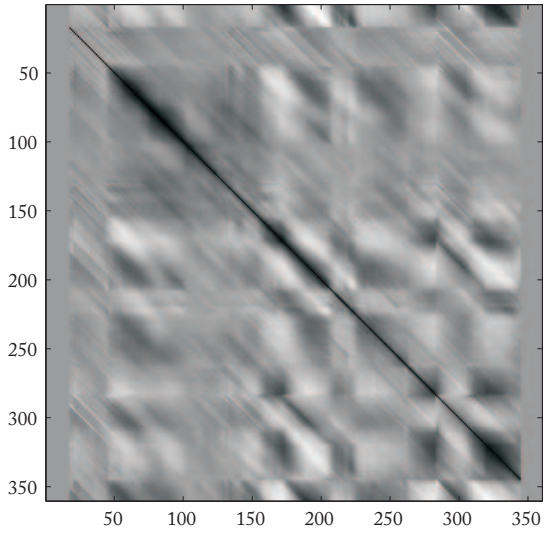

Fig. 12. Similarity matrix for ZNCC.

Fig. 6 is shown in Fig. 10.

In this section, we want to compare our descriptor with other two image similarity measures that are well known in image registration and are also commonly used for matching individual lines. These are Sum of Squared Differences (SSD) and Zero mean Normalized Cross-Correlation (ZNCC) (their definitions can be found in [Gonzalez02]). 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 Fig. 9. As we did to build the similarity matrix for our descriptors, we compare given pattern $\mathbf{P}_i$ with pattern $\mathbf{P}_j$ using either SSD or ZNCC and build the respective similarity matrices, that is:

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

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

The two similarity matrices for the image in Fig. 6 are shown in Fig. 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, maximum similarity between two patterns occurs when SSD=0. Conversely, for ZNNC, maximum similarity (correlation) occurs when ZNCC=1; however, observe that Fig. 12 has been inverted to enable comparison with figures 10 and 11 (this means that black indicates maximum similarity and white minimum similarity).

To interpret the similarity matrix, consider points along the diagonal axis in Fig. 10. Each point is perfectly similar to itself, so all 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 forward and backward by tracing horizontally or vertically on the matrix. To compare 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 Fig. 9. As observed, there are poorly textured objects and repeating structure.

Similar comments can be done 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 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.

VI. 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 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 II.

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 (1852 images) used in Section IX-A. First, we extracted verticals lines from each image. Then we manually labeled all 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. Due to big windows, a mixture of natural and artificial lighting produces difficult lighting conditions like highlights and specularities.

In the following subsections, we characterize the performance of our descriptor. We would like to remark that the features of each image were matched against all 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).

2) Image saturation: As we mentioned in Section III-C, 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 Fig. 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 will always use this value.

3) Image noise: The percentage of correct matches for different amounts of gaussian image noise (from 0% to 10%) is shown in Fig. 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.

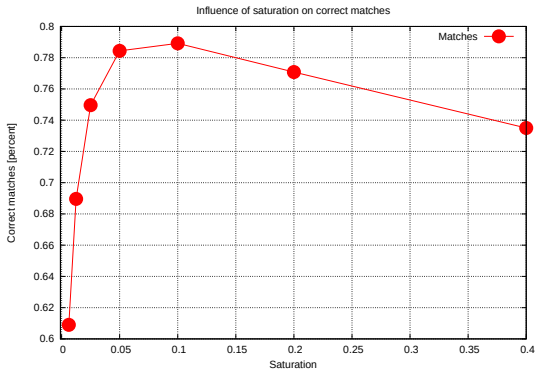

Fig. 13. Influence of saturation on correct matches.

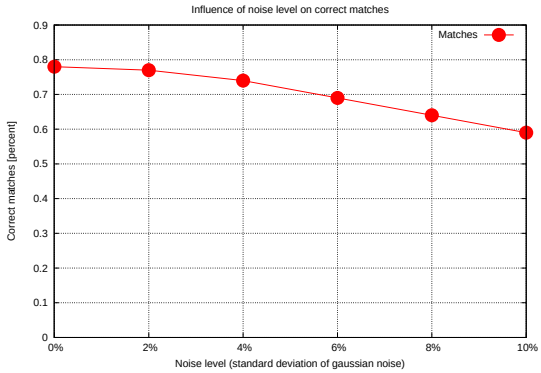

Fig. 14. Influence of noise level ($\%$) on correct matches. The correct matches are found using only the nearest descriptor in the database.

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 explanation of the descriptor we used 3 non overlapping circular areas, we evaluated the effect of using 5 overlapping areas with 50% overlap between two circles. The results are shown in Fig. 15. As the graph shows, there is a slight improvement in using 5 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 3 areas and 8 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 of correct and incorrect matches shown in figures 16, 17, and 18. Finally observe that we considered powers of 2 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.

5) Matching rules: Figure 16 shows the Probability Density Function (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

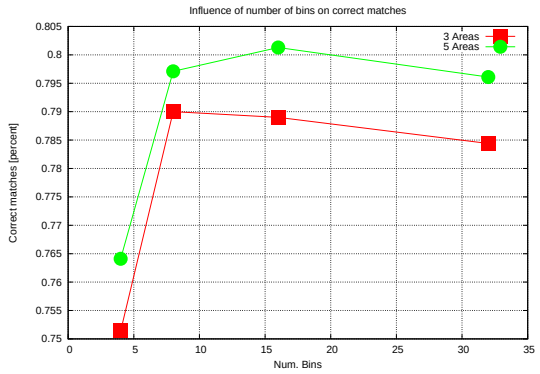

Fig. 15. Influence of number of bins on correct matches.

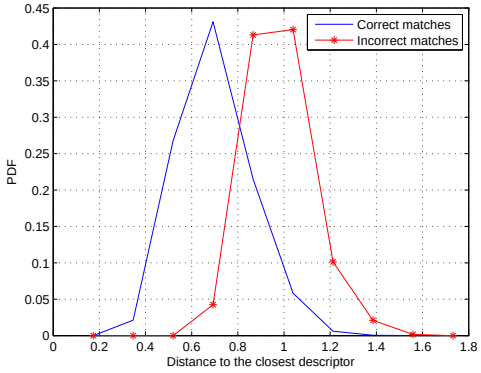

Fig. 16. The probability density function that a match is correct according to the first rule.

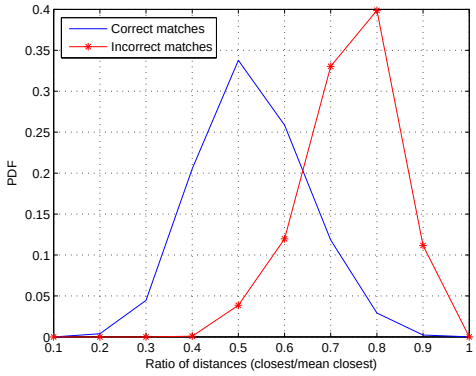

Fig. 17. The probability density function that a match is correct according to the second rule.

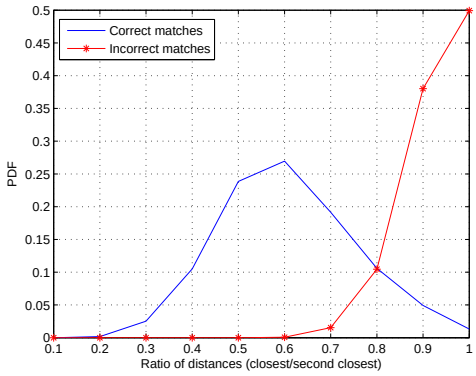

Fig. 18. The probability density function that a match is correct according to the third rule.

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, Fig. 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.

Finally, Fig. 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.

VII. 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 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 we presented in [Martinelli06] where just a single landmark (we used a source of light) was tracked during the motion to perform calibration. In this section, we extend the method in [Martinelli06] 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 do 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 respectively to the robot and to the camera (see Fig. 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 (Fig. 19).

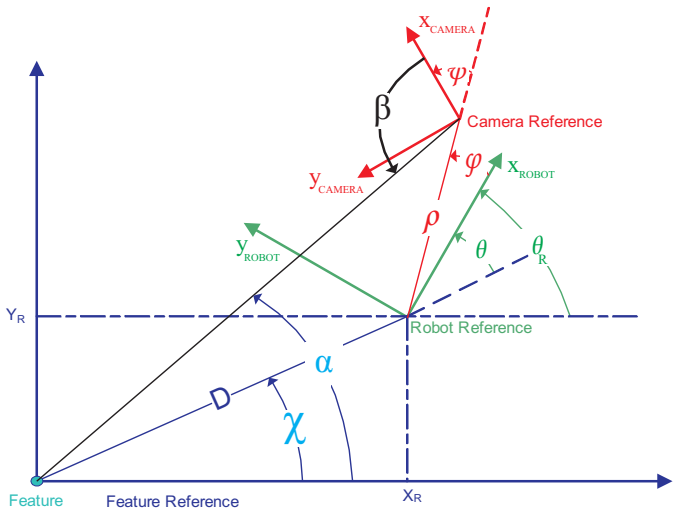

Fig. 19. The two reference frames respectively attached to the robot and to the camera. The five parameters estimated by the EKF $(D, \theta, \phi, \rho, \psi)$ are also indicated.

As we consider the case of a mobile robot moving in a $2D$ 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 Fig. 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:

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

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

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

where e is the distance between the wheels.

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

$$\beta = \pi - \psi - \theta_R - \phi + \alpha \tag{16}$$

with

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

VIII. 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:

$$\mathbf{X_a} = [x_R, y_R, \theta_R, \phi, \rho, \psi]^T \tag{18}$$

An EKF can be adopted to estimate the state $\mathbf{X_a}$. The inputs $\mathbf{u}$ of the dynamics of this state are directly provided by the encoder data and the observations $\mathbf{z}$ are the bearing angles provided by the vision sensor. However, as it was pointed out in [Martinelli06], by considering the system state $\mathbf{X_a}$ as defined in (18) the system is not observable, that is, it does not contain all the necessary information to perform the estimation with an error which is bounded. Conversely, in [Martinelli06] it was proved that the system becomes observable if, instead of considering $\mathbf{X_a}$, we introduce a new state $\mathbf{X}$ defined as follows:

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

with $D = \sqrt{x_R^2 + y_R^2}$ and $\theta = \theta_R - \tan^{-1}\left(\frac{y_R}{x_R}\right)$ (see Fig. 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.

A. Observability Properties with Respect to the Robot Trajectory

In control theory, a system is defined observable when it is possible to reconstruct its initial state by knowing, in a given time interval, the control inputs and the outputs (see

[Isidori95]). The observability property has a very practical meaning. When a system is observable it contains all the necessary information to perform the estimation with an error which is bounded (see [Isidori95]).

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 nonlinear systems (see [Isidori95]). First of all, in the nonlinear case, the observability is a local property. For this reason, in a nonlinear case the concept of the *local distinguishability property* was introduced by Hermann and Krener (see [Hermann77]). The same authors introduced also a criteria, *the observability rank condition*, to verify if a system has this property. This criteria plays a very important role since in many cases a nonlinear 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 is described through the following equations:

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

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

$$f_1 = \left[\cos\theta, -\frac{\sin\theta}{D}, 0, 0, 0\right]^T \quad f_2 = [0,1,0,0,0]^T \tag{22}$$

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 remind some concepts in the theory by Hermann and Krener in [Hermann77]. We will adopt the following notation. We indicate the K^{th} order Lie derivative of a field Λ along the vector fields $v_{i_1}, v_{i_2}, ..., v_{i_K}$ with $L^K_{v_{i_1},v_{i_2},...,v_{i_K}}\Lambda$. Note that the Lie derivative is not commutative. In particular, in $L^K_{v_{i_1},v_{i_2},...,v_{i_K}}\Lambda$ 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 the Lie derivatives $L^K_{f_{i_1},f_{i_2},...,f_{i_K}}h(X)|_{t=0}$ $(i_1,...,i_K = 1,2,...,M$ 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 $d\Omega$.*

In [Martinelli06] it was shown that the state X satisfying the dynamics in (20) is observable when the observation is the one given in (23). Here we want to investigate the observability properties depending on the robot trajectory. In particular, we will consider separately the case of straight motion and pure rotations about the robot origin. Furthermore, we will 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 $\lambda \equiv \frac{D}{\rho}$ and the sum $\gamma \equiv \theta + \phi$:

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

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

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

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 the one 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 $\theta + \phi + \psi$.

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 the Appendix we show that this is the case.

3) Pure Rotation and close feature: Let us consider the state $X_\lambda \equiv [\lambda, \gamma, \phi, \rho, \psi]^T$. 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 are observable the quantities: λ, γ and ψ. In the Appendix we show that this is the case.

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 the Appendix) 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.

B. The Filter Equations

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

$$\begin{cases} D_{i+1} &= D_i + \delta\rho_i \cos\theta_i \\ \theta_{i+1} &= \theta_i + \delta\theta_i - \frac{\delta\rho_i}{D_i}\sin\theta_i \\ \phi_{i+1} &= \phi_i \\ \rho_{i+1} &= \rho_i \\ \psi_{i+1} &= \psi_i \end{cases} \tag{26}$$

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:

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

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

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

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:

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

with $\mathbf{u} = [\delta\rho_R, \delta\rho_L]^T$.

Regarding the observation function h, from (27) we have:

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

The previous equations, along with a statistical error model of the odometry (we used the one by Chong and Kleeman [Chong97]), 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 ($\delta\rho_R$ and $\delta\rho_L$). 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 (see [Bar-Shalom88]) and are given in the Appendix.

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 (see [Chong97]), 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 the one returned by the wheel encoder and the variance increases linearly with the absolute value of the traveled distance.

$$\delta\rho_{R/L} = N(\delta\rho^e_{R/L}, K\delta\rho^e_{R/L}) \tag{31}$$

where the index R and L stand respectively for the right and left wheel, the apex 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 5 with respect to the one estimated by previous experiments (see [Martinelli07]).

IX. Experimental Results

In our experiments, we adopted a mobile robot with a differential drive system endowed of 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 640×480 pixels. A picture of our platform is depicted in Fig. 1. Finally, observe that the entire algorithm ran in real-time. In particular, the all process (image capture, feature extraction, description computation, feature matching) could be computed in less than 20 ms on a dual-core laptop computer.

A. 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 VI.

The robot was moving at about $0.15\ m/s$ 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 Fig. 20. 1852 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 Fig. 22. The video sequence from where this graph was generated can be found in the multimedia extension of this paper (Appendix A). In the video, every vertical line is labeled with the corresponding number and color with which it appears in Fig. 22. The graph shown in Fig. 22 was obtained using only the three matching rules described in Sections IV-A, IV-B, IV-C. No other constraint, like mutual and topological relations, has been used. This plot refers to a short path of the whole trajectory while the robot was moving straight (between frame no. 0 and 46), then doing a $180°$ rotation (between frame no. 46 and 106), and 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. 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 no. 1 and 58,

Fig. 20. Floorplan of the institute showing the robot path (in red).

between track no. 84 and 86, and between track no. 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 frames. This can be seen by observing that sometimes circles are missing on the tracks (look for instance at track no. 52). When a correspondence is not found in the previous frame, our tracking algorithm starts looking into all previous frames (actually up to twenty 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 no. 1 - that is the first detected feature and starts at frame no. 0 - is correctly tracked until frame no. 120 and is then labeled as feature no. 75. This is because at this frame no correspondence was found and then the feature was labeled as a new entry (but in fact is a false new entry). Another example is feature no. 15 that is then labeled as no. 18 and no. 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 one.

After visually inspecting every single frame of the whole video sequence (composed of 1852 frames), we found 35 false matches and 101 false new entries. The detection rate over the entire dataset is shown in Table III at intervals of 200 frames.

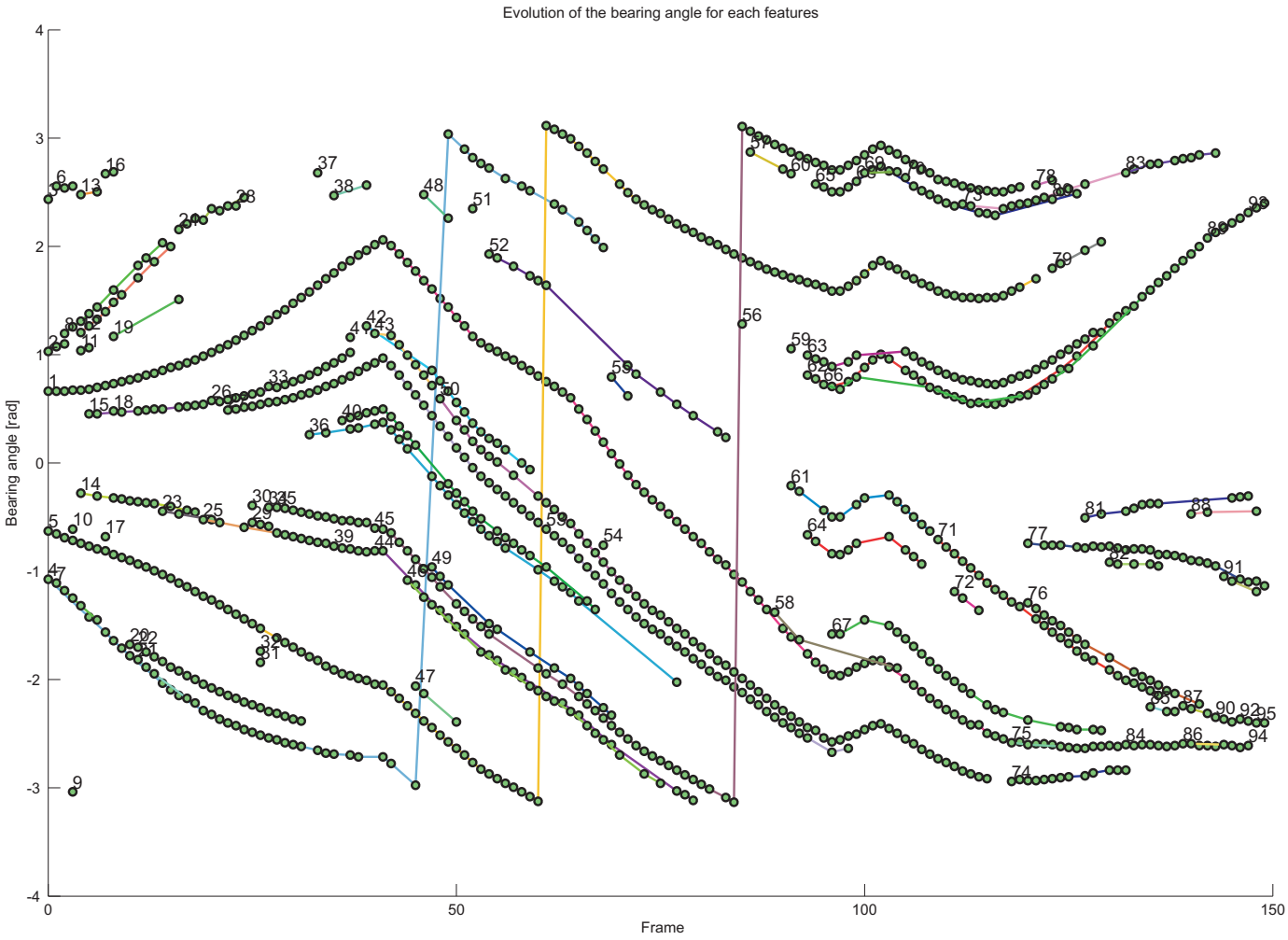

Fig. 22. Feature tracking during the motion of the robot. In y-axis is the angle of sight of each feature and in the x-axis the frame number. Each circle represents a feature detected in the observed frame. Lines represent tracked features. Numbers appear only when a new feature is detected. This plot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

TABLE III
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
800-1000	685	98.13	0.57	1.29
1000-1200	740	98.40	0.26	1.33
1200-1400	906	98.26	0.43	1.30
1400-1600	784	97.75	0.62	1.62
1600-1852	771	98.34	0.76	1.89

Comparing these errors to the 7408 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 with repetitive texture (like in Fig 9 or in the second image of Fig. 21). Thus, ambiguity was caused by the presence of vertical elements which repeat almost identical 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 IV-A, IV-B, IV-C. Obviously, the performance of tracking could be further improved by adding other constraints like mutual and topological relations among features.

B. Calibration Results

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

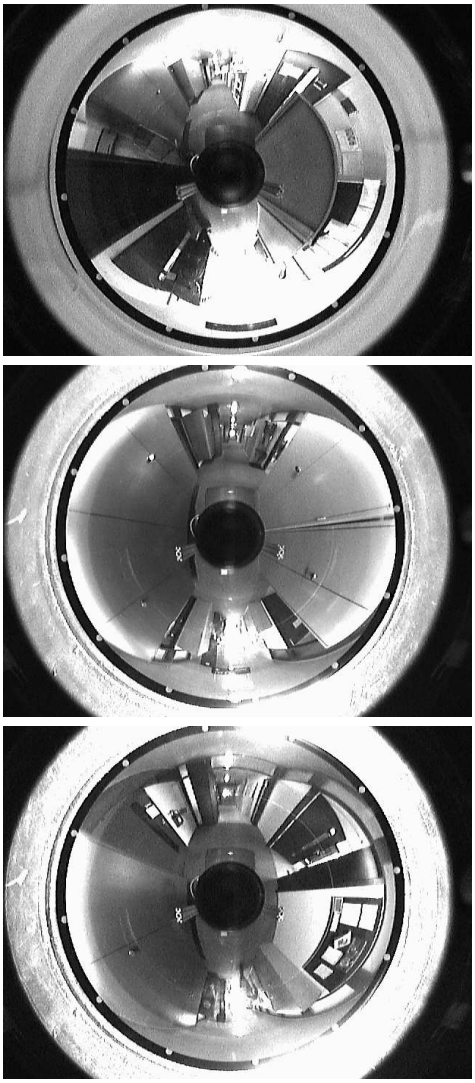

Fig. 21. Omnidirectional images taken at different locations.

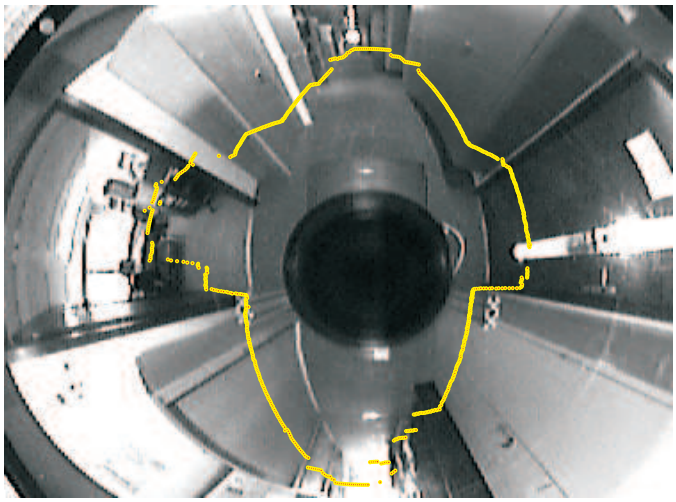

Fig. 23. Laser points reprojected onto the omnidirectional image before calibration. The edges in the omnidirectional image do not correctly intersect the corners of the laser scan.

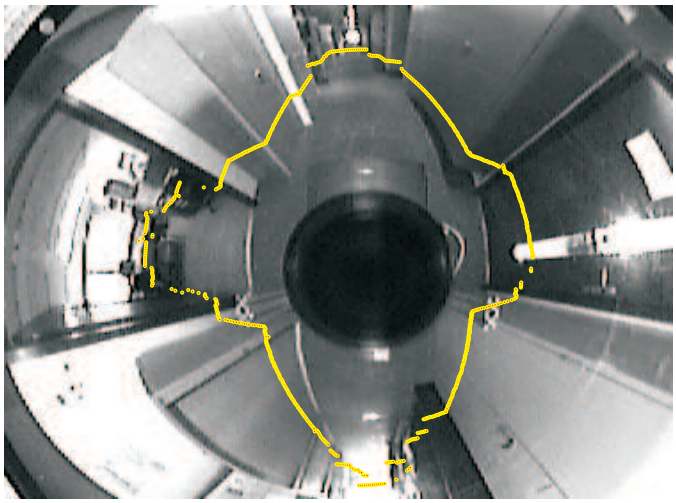

Fig. 24. Laser points reprojected onto the omnidirectional image after calibration. The edges in the omnidirectional image appropriately intersect the corners of the laser scan.

For our experiments, we positioned the omnidirectional camera on our robot as in Fig. 1 and we measured manually its position relative to the robot. We measured the following values: $\phi \simeq 0\ rad$, $\rho \simeq 0.2\ m$, $\psi \simeq 0\ rad$. 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 accurately measured, 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 fig. 25. For this experiments, about ten vertical lines were tracked.

The values of ϕ, ρ, ψ estimated during the motion are plotted as a function of the frame number in Fig. 26. 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 $\phi = -0.34rad$, $\rho = 0.23m$ and $\psi = 0.33rad$. The sudden jump starting at frame no. 60 actually occurs when the robot starts to rotate.

As demonstrated in section VIII-A, 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 in [Martinelli06], the convergence is very fast when the robot performs trajectories alternating short straight paths and pure rotations about the robot origin.

Furthermore, extensive simulations in [martinelli06], show that even when the estimation process starts by firstly moving

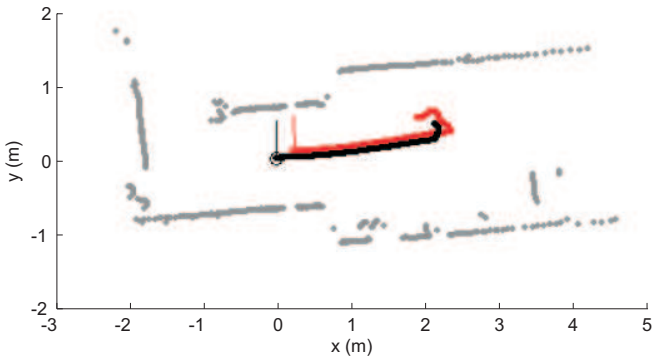

Fig. 25. The path performed by the robot during self-calibration, i.e. straight path followed by a rotation.

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.

Fig. 24 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.

X. Conclusion

In this paper, we 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 in 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 long time. Furthermore, because the results were obtained using only the three matching rules described in Section IV, we expect that the performance would be notably improved by adding other constraints like 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 extended Kalman filter that automatically estimates the calibration parameters while the robot is moving. The present strategy had been already proposed in our previous work (see [Martinelli06]). In [Martinelli06], 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 (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 straight path. Although experiments have been conducted using an omnidirectional camera, more in general the proposed method can be adopted to calibrate any robot bearing sensor.

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

Acknowledgment

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).

References

[Ayache87] Ayache, N. and Faugeras, O., Building a consistent 3D representation of a mobile robot environment by combining multiple stereo views. In: Proc. IJCAI, pp. 808810, 1987.

[Ayache90] Ayache, N., 1990. Stereovision and Sensor Fusion. MITPress, 1990.

[Baillard99] Baillard, C., Schmid, C., Zisserman, A., and Fitzgibbon, A., Automatic line matching and 3D reconstruction of buildings from multiple views, SPRS Conference on Automatic Extraction of GIS Objects from Digital Imagery, IAPRS Vol.32, Part 3-2W5, pp. 69-80, 1999.

[Bar-Shalom88] Y. Bar-Shalom and T.E. Fortmann, Tracking and data association, mathematics in science and engineering, vol. 179, Academic Press, 1988.

[Baumberg00] A. Baumberg. Reliable feature matching across widely separated views. In Proc. IEEE Conference on Computer Vision and Pattern Recognition, Hilton Head, South Carolina, pages 774-781, 2000.

[Bhattacharya05] M.M. Rahman, P. Bhattacharya, and B.C. Desai, Similarity Searching in Image Retrieval with Statistical Distance Measures and Supervised Learning, in Pattern Recognition and Data Mining, pp. 315-324, 2005.

[Brassart00] Brassart, E., Delahoche, L., Cauchois, C., Drocourt, C., Pegard, C., Mouaddib, E.M., Experimental Results got with the Omnidirectional Vision Sensor: SYCLOP, International workshop on omnidirectional vision (OMNIVIS 2000), 2000.

[Briggs06] A. Briggs, Y. Li, D. Scharstein, M. Wilder. Robot Navigation Using 1D Panoramic Images. In Proc. of IEEE Intl. Conference on Robotics and Automation (ICRA 2006), Orlando, FL, May 2006.

[Chong97] Chong, K.S. and Kleeman, L., Accurate Odometry and Error Modelling for a Mobile Robot, in International Conference on Robotics and Automation, pp. 2783-2788, 1997.

[Crowley90] Crowley, J. and Stelmazyk, P., Measurement and integration of 3d structures by tracking edge lines. In: Proc. ECCV, pp. 269280, 1990.

[Deriche90] Deriche, R. and Faugeras, O., Tracking line segments. In: Proc. ECCV, pp. 259267, 1990.

[Goedeme04] T. Goedeme, T. Tuytelaars, and L. Van Gool, Fast Wide Baseline Matching with Constrained Camera Position, In the poceedings of the Conference on Computer Vision and Pattern Recognition, Washington, DC, pp. 24-29, 2004.

[Gonzalez02] R. Gonzalez, R. Woods, Digital Image Processing, Addison Wesley, Prentice Hall, ed. 2, ISBN: 0201180758, 2002.

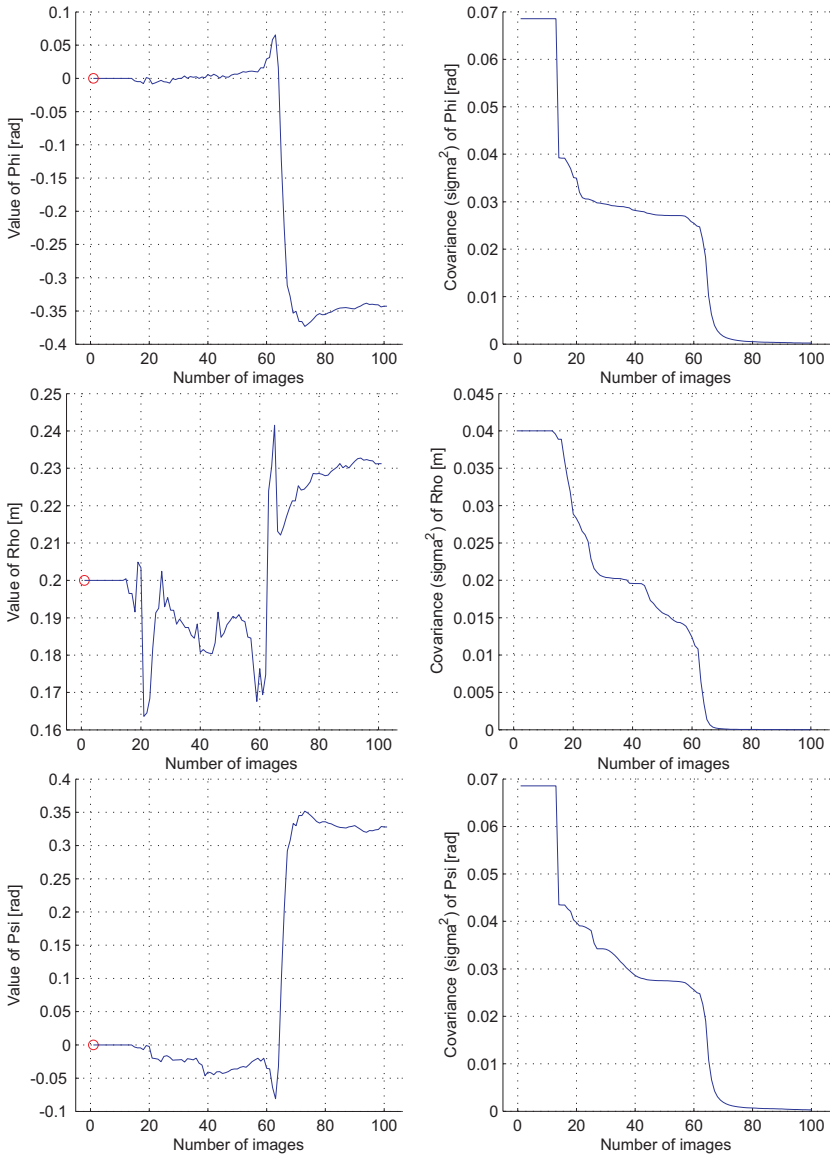

Fig. 26. ϕ, ρ, ψ, ${\sigma_\phi}^2$, ${\sigma_\rho}^2$, ${\sigma_\psi}^2$ as a function of the frame number. The distance traveled between two frames is about 3.8 cm.

[Gros95] Gros, P., Matching and clustering: Two steps towards object modelling in computer vision. Intl. J. of Robotics Research 14(6), pp. 633642, 1995.

[Hermann77] Hermann R. and Krener A.J., Nonlinear Controllability and Observability, IEEE Transaction On Automatic Control, AC-22(5): 728-740, 1977.

[Hesch08] J. A. Hesch, A. I. Mourikis, S. I. Roumeliotis, Determining the Camera to Robot-body Transformation from Planar Mirror Reflections, in Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2008), Nice, France, September, 2008.

[Horaud89] Horaud, R. and Skordas, T., Stereo correspondence through feature grouping and maximal cliques. IEEE TPAMI 11(11), pp. 11681180, 1989.

[Huttenlocher93] Huttenlocher, D. P., Klanderman, G. A. and Rucklidge, W. J., Comparing images using the Hausdorff distance. IEEE T-PAMI, 1993.

[Isidori95] Isidori A., Nonlinear Control Systems, 3rd ed., Springer Verlag, 1995.

[Kadir04] T. Kadir, A. Zisserman, and M. Brady. An affine invariant salient region detector. In Proc. 7th European Conference on Computer Vision, Prague, Czech Republic, pages Vol I: 228-241, 2004.

[Lowe04] D.G. Lowe. Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision, 60(2):91-110, November 2004.

[Martinelli06] Martinelli, A.,Scaramuzza, D. and Siegwart, R.,Automatic Self-Calibration of a Vision System during Robot Motion, in IEEE International Conference on Robotics and Automation (ICRA 2006), 2006.

[Martinelli07] A. Martinelli, N. Tomatis and R. Siegwart, Simultaneous Localization and Odometry self Calibration for Mobile Robot, in Autonomous Robots , vol 22, pp.75-85, 2007.

[Matas02] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide baseline stereo from maximally stable extremal regions. In Proc. 13th British Machine Vision Conference,Cardiff, UK, pages 384-393, 2002.

[Mauthner06] T. Mauthner, F. Fraundorfer, H. Bischof. Region matching for omnidirectional images using virtual camera planes. In Proc. of Computer Vision Winter Workshop 2006, Telc, Czech Republic, 2006.

[Medioni85] Medioni, G. and Nevatia, R., Segment-based stereo matching. Computer Vision, Graphics and Image Processing 31, pp. 218, 1985.

[Mikolajczyk98] T. Lindeberg. Feature detection with automatic scale selection. International Journal of Computer Vision, 30(2):79-116, 1998.

[Mikolajczyk01] K. Mikolajczyk and C. Schmid. Indexing based on scale invariant interest points. In Proceedingsof the 8th International Conference on Computer Vision, Vancouver, Canada, pages 525-531, 2001.

[Mikolajczyk02] K. Mikolajczyk and C. Schmid. An affine invariant interest point detector. In Proc. 7th European Conference on Computer Vision, Copenhagen, Denmark, page I: 128 ff., 2002.

[Micusik06] B.Micusik, T.Pajdla. Structure from Motion with Wide Circular

Field of View Cameras. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 28, no. 7, pp. 1135-1149, Jul, 2006.

[Mirzaei07] F. M. Mirzaei and S. I. Roumeliotis, A Kalman filter-based algorithm for IMU-camera callibration, in Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2007), San Diego, pp. 24272434, October, 2007.

[Nayar97] S.K. Nayar. Catadioptric omnidirectional camera. In CVPR '97: Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition (CVPR '97), page 482, Washington, DC, USA, 1997. IEEE Computer Society.

[Prasser04] D. Prasser, G. Wyeth, M. J. Milford (2004b), Experiments in Outdoor Operation of RatSLAM, Australian Conference on Robotics and Automation, Canberra Australia, 2004.

[Rubner00] Y. Rubner, C. Tomasi, and L. J. Guibas, .The earth mover's distance as a metric for image retrieval,. International Journal of Computer Vision, vol. 40, no. 2, pp. 99-121, 2000.

[Rubner01] Y. Rubner et al, .Empirical evaluation of dissimilarity measures for color and texture,. Computer Vision and Image Understanding, vol. 84, no. 1, pp. 25-43, 2001.

[Sagues06] C. Sagues, A.C. Murillo, J.J. Guerrero, T. Goedeme, T. Tuytelaars, and L. Van Gool, Hierarchical localization by matching vertical lines in omnidirectional images, in special issue of RAS Robotics and Autonomous Systems journal, Elsevier, 2006.

[Scaramuzza07] Scaramuzza, D., Criblez, N., Martinelli, A. and Siegwart, R., Robust Feature Extraction and Matching for Omnidirectional Images, Proceedings at the 6th International Conference on Field and Service Robotics (FSR 2007), Chamonix, France, July 2007.

[Scaramuzza07b] Scaramuzza, D., Harati, A., and Siegwart, R., 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, USA, October 2007.

[Tell00] D. Tell and S. Carlsson, Wide baseline point matching using affine invariants computed from intensity profiles, In Proceedings of the European Conference Computer Vision, pp. 814828, Dublin, Ireland, 2000.

[Tuytelaars04] T. Tuytelaars and L. Van Gool. Matching widely separated views based on affine invariant regions. International Journal of Computer Vision, 1(59):61-85, 2004.

[Venkateswar95] Venkateswar, V. and Chellappa, R., Hierarchical stereo and motion correspondence using feature groupings. IJCV pp. 245269, 1995.

[Yagi91] Yagi, Y., and Yachida, M., Real-Time Generation of Environmental Map and Obstacle Avoidance Using Omnidirectional Image Sensor with Conic Mirror, CVPR'91, pp. 160-165, 1991.

[Zhang94] Zhang, Z., Token tracking in a cluttered scene. Image and Vision Computing 12(2), pp. 110120, 1994.

[Zhang04] Q. Zhang and R. Pless. Constraints for heterogeneous sensor auto-calibration. In IEEE Workshop on Realtime 3D Sensors and Their Use, pages 38-43, 2004.

APPENDIX A

INDEX TO MULTIMEDIA EXTENSIONS

The multimedia extensions to this article are at: http://www.ijrr.org. The video corresponds to the plot reported in Fig. 22.

TABLE IV
MULTIMEDIA EXTENSION

Extension	Type	Description
1	Video	Feature tracking

APPENDIX B

STRAIGHT MOTION AND FAR FEATURE

We want to prove that the state $X_{sf} \equiv [D, \theta, \eta]^T$ satisfying the dynamics: $\dot{X}_{sf} = vf$ with $f = [\cos\theta, -\frac{sin\theta}{D}, 0]^T$ 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:

$$\begin{aligned} &L^0\beta = \beta = -\theta - \eta \\ &dL^0\beta = [0, -1, -1] \\ &L^1_f\beta = \tfrac{sin\theta}{D} \\ &dL^1_f\beta = [-\tfrac{sin\theta}{D^2}, \tfrac{\cos\theta}{D}, 0] \\ &L^2_{ff}\beta = -\tfrac{sin2\theta}{D^2} \\ &dL^2_{ff}\beta = [2\tfrac{sin2\theta}{D^3}, -2\tfrac{\cos 2\theta}{D^2}, 0] \end{aligned}$$

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 $\theta \neq n\pi$

PURE ROTATION AND CLOSE FEATURE

We want to prove that the state $X_{rc} \equiv [\lambda, \gamma, \psi]^T$ satisfying the dynamics: $\dot{X}_{rc} = \omega f$ with $f = [0, 1, 0]^T$ is observable when the observation is $\beta = atan\left(\frac{sin\gamma}{\lambda + \cos\gamma}\right) - \gamma - \psi$.

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.

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

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

$$A^i = \begin{bmatrix} 1 & -\delta\rho\ sin\theta^i \\ \frac{\delta\rho}{D^{i2}}\ sin\theta^i & 1 - \frac{\delta\rho}{D^i}\cos\theta^i \end{bmatrix},$$

$$B^i = \begin{bmatrix} \frac{\cos\theta^i}{2} & \frac{\cos\theta^i}{2} \\ \frac{1}{e} - \frac{sin\theta^i}{2D^i} & -\frac{1}{e} - \frac{sin\theta}{2D^i} \end{bmatrix}$$

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

$$H1^i = \frac{-\rho sin(\theta^i + \phi)}{{D^i}^2 + 2\rho D^i \cos(\theta^i + \phi) + \rho^2},$$

$$H2^i = \frac{-D^i\rho\ \cos(\theta^i + \phi) - {D^i}^2}{{D^i}^2 + 2\rho D^i \cos(\theta^i + \phi) + \rho^2},$$

$$H3^i = \frac{-D^i\rho\ \cos(\theta^i + \phi) - {D^i}^2}{{D^i}^2 + 2\rho D^i \cos(\theta^i + \phi) + \rho^2},$$

$$H4^i = \frac{D^i sin(\theta^i + \phi)}{{D^i}^2 + 2\rho D^i \cos(\theta^i + \phi) + \rho^2}\ .$$