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

Davide Scaramuzza, Agostino Martinelli, Roland Y. Siegwart

A Robust Descriptor for Tracking Vertical Lines in Omnidirectional Images and its Use in Robot Self-Calibration

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

[1] Swiss Federal Institute of Technology Zurich (ETHZ), Switzerland
[2] INRIA Rhone-Alpes, France
Contact author, Davide Scaramuzza, Email: davide.scaramuzza@ieee.org

***Abstract*—This paper presents a robust descriptor for matching vertical lines among omnidirectional images and a method for automatically calibrating an omnidirectional camera with the robot reference system. The first part of this paper describes how to build the feature descriptor. We show that the descriptor is unique and distinctive for each feature 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 robot 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.**

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

I. Introduction

A. 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 [1]–[9]). 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 [10]. An application of this approach can be found in [11]. 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 [14], 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 [15]. 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 an 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) [24]–[26]. Some such as [27]–[29] 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 [30]–[33]. 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 [21]).
An interesting approach was proposed in [22]. Besides using the topological information of the line, the authors also used the photometric neighbourhood 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 neighbourhoods was then assessed by cross-correlation at the corresponding points.

A novel approach, using the intensity profile along the line segment, was proposed in [23]. 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.

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 [18]–[20]). 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).

B. Outline

The contributions of this paper are two. In the first part of the paper, we describe how we built our robust descriptor for vertical lines. We show that the descriptor is unique and very distinctive for each feature and is invariant to rotation and slight changes of illumination. The robustness of the descriptor is validated through real experiments using our robot.
In the second part of the paper, we show an application of our visual tracking to the problem of robot-camera self-calibration. In particular, we describe how to fuse the visual information with the robot odometry to extrinsically and automatically calibrate the camera while the robot is moving.
This paper extends our two previous works [16], [38].

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). Then, we provide our matching rules (Section IV) and present experimental results (Section VI). In Section VII, we describe the calibration problem and provide the equations to build the Extended Kalman Filter (EKF) (Section VIII). In Section IX, we will present the calibration results with our mobile robot.

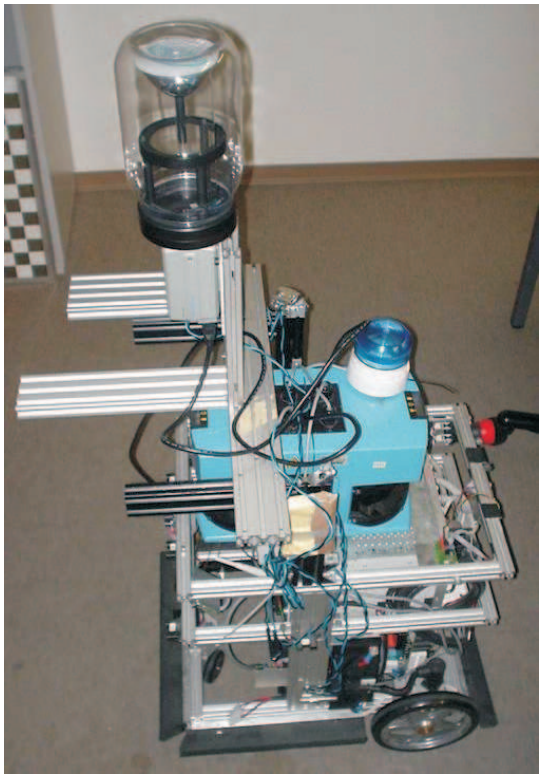

Fig. 1. The robot used in our experimets equipped with encoder sensors, omnidirectional camera, and two laser range finders

II. VERTICAL LINE EXTRACTION

Our platform consists of a wheeled robot equipped with an omnidirectional camera looking upwards (see Fig. 1). The main advantage of such kind of camera is that it provides a 360° field of view of the scene, which gives a very rich and sparse information. 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 towards 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 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 Sobel masks. From $\mathbf{I_x}$, $\mathbf{I_y}$, we can calculate the magnitude $\mathbf{M}$ and the phase $\mathbf{\Phi}$ of the gradients as

$$\mathbf{M} = \sqrt{\mathbf{I_x}^2 + \mathbf{I_y}^2}, \quad \mathbf{\Phi} = atan2(\mathbf{I_y}, \mathbf{I_x}). \tag{1}$$

Then, we do a thresholding on $\mathbf{M}$, $\mathbf{\Phi}$ by retaining those vectors whose orientation looks towards the image center up to $\pm 5^\circ$. 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 line length. As observed in Fig. 4), we set our threshold equal to 50% of the maximum allowed line length, i.e. $R_{max} - R_{min}$. Obviously, this choice is purely arbitrary and a different criterion could be used depending on the purpose (for instance, one can decide to have always a constant number of lines per each frame). Another criterion is to have this threshold adaptive. This can be done by computing the mean $\bar{l}$ and the standard deviation σ_l of all line lengths in the observed image and keeping only those lines whose length l satisfies $l \geq \bar{l} + 3\sigma_l$.

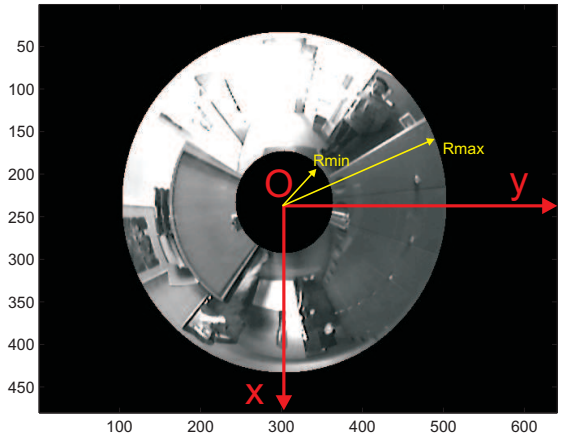

Fig. 2. An image taken by our omnidirectional camera.

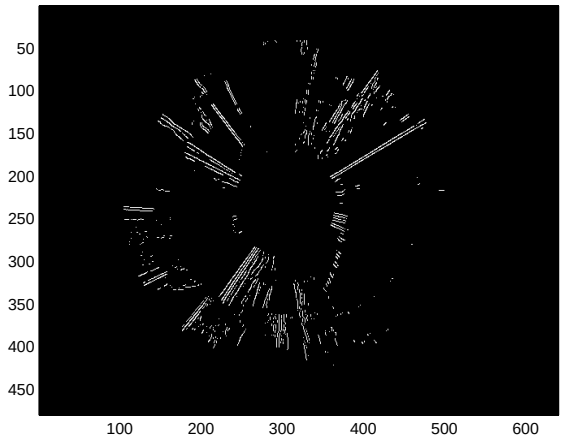

Fig. 3. Edge image of Fig. 2.

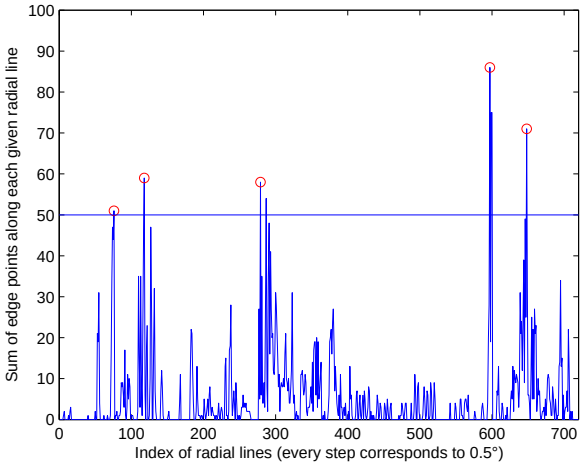

Fig. 4. Number of binary pixels voting for a given orientation angle.

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 unique and distinctive for each feature. 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

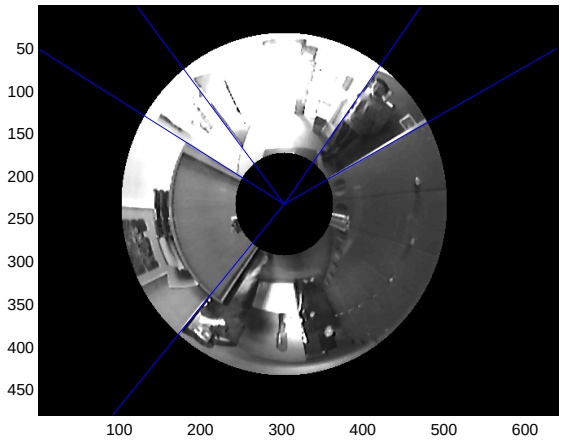

Fig. 5. Extraction of the most reliable vertical features from an omnidirectional image.

of each area is equal to $(R_{max} - R_{min})/6$ (see Fig. 7). Then, we smooth each area with a Gaussian window (Fig. 6) with $\sigma_G = r_a/3$ and compute the image gradients (magnitude $\mathbf{M}$ and phase $\mathbf{\Phi}$) within each of these areas.
Concerning rotation invariance, this is achieved by redefining the gradient phase $\mathbf{\Phi}$ of all points relatively to the radial line's angle θ (see Fig. 7).

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. 8). 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. 9). 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 phase 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 phase has been redefined relatively to the radial line's angle (Section III-A).

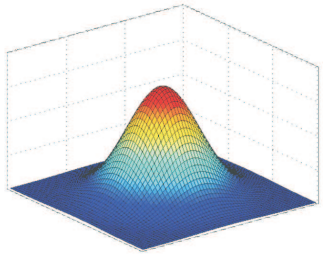

Fig. 6. Gaussian smoothing filter

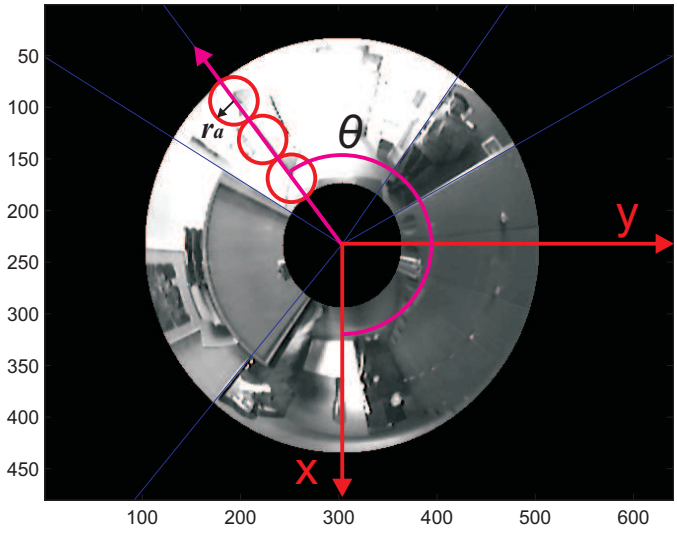

Fig. 7. Extraction of the circular areas. To achieve rotation invariance, the gradient phase $\mathbf{\Phi}$ of all points is redefined relatively to the radial line's angle θ.

To resume, 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.

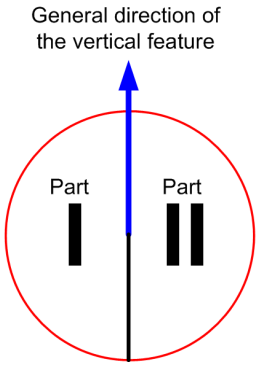

Fig. 8. The two sections of a circular area.

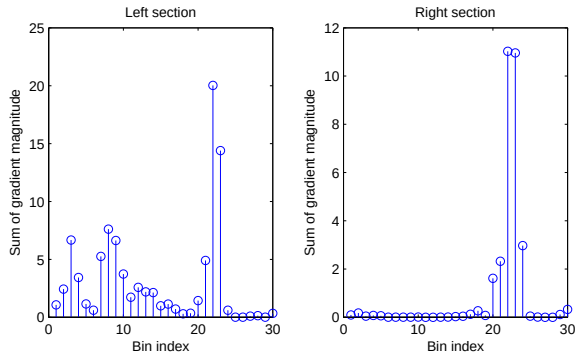

Fig. 9. An example of gradient orientation histograms for the left and right sides of a circular area.

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 area. This choice relies on the hypothesis that the image intensity changes linearly with illumination. Although this is not true in nature, this approximation proved to work properly.

To resume, 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 30 bins for each histogram; thus the length of the descriptor is

$$N = 3areas \cdot 2parts \cdot 30bins = 180 \tag{5}$$

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 [34]–[36].

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 $\mathbf{B_j}$.

A. First test

The first test checks that the distance from the closest descriptor is smaller than a given threshold. As the threshold depends on the length of the descriptor, we set

$$minD_i = F_1 \cdot N \tag{8}$$

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

B. Second test

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

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

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. 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 the correct correspondent of $\mathbf{A_1}$. Also note that its distance is smaller than the mean of all other distances.

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	B6	B7
2.38	5.42	4.55	5.79	5.66	6.17	5.43

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

F1 = 0.0075	F2 = 0.55	F3 = 0.85

C. Third test

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

$$minD_i = F_3 \cdot SecondSmallestDistance, \qquad (10)$$

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.

Factors F_1, F_2, F_3 are to be determined experimentally. The empirical values used in our experiments are shown in Table II.

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,

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

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 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. 7 is shown in Fig. 11.

In this section, we want to compare our descriptor with other two image similarity measures that are very used in image registration but are also commonly used for matching individual lines, that is, Sum of Squared Differences (SSD) and Normalized Cross-Correlation (NCC) (their definitions can be found in [21]). When using SSD and NCC 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. 10. 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 NCC and build the respective similarity matrices, that is:

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

$$\mathbf{S}_{NCC}(i, j) = NNC(\mathbf{P}_i, \mathbf{P}_j), \qquad (13)$$

The two similarity matrices for the image in Fig. 7 is shown in Fig. 12 and 13. Concerning the size win of the patterns for computing SSD and NCC, 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 NNC, maximum similarity (correlation) occurs when NCC=1 (this explains why the diagonal axis in Fig. 13 is white instead of black.

To interpret the similarity matrix, consider points along the diagonal axis in Fig. 11. Each point is perfectly similar to itself, so all the points on the diagonal are dark. Starting from a given point on the diagonal, you 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, you 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. 10. As you can see, there are poorly textured objects and repeating structure.

Similar comments can be done regarding the similarity matrix for NCC, but we have to invert word "dark" with "light", due to the inverse definition of NCC. However, observe that the behavior of NCC is much better than SSD: first, the size of the blocks along or off the diagonal axis is smaller; then, points on the diagonal are much lighter than points off the diagonal, meaning that NNC captures the distinctiveness of the pattern better than SSD.

When compared with SSD and NCC, the similarity matrix of our descriptor outperforms SSD and NCC. This can be seen by observing that 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 even higher than NCC. Finally, observe that blocks along or off the diagonal axis are much smaller or lighter than SSD and NCC; this indicates that even on poorly textured surfaces our descriptor is more distinctive than SSD and NCC. The distinctiveness of the descriptor is due to the use of gradient orientation histograms. In the concept, our method is similar to SIFT [37], which also uses gradient histograms to build distinctive descriptors of image keypoints.

VI. 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

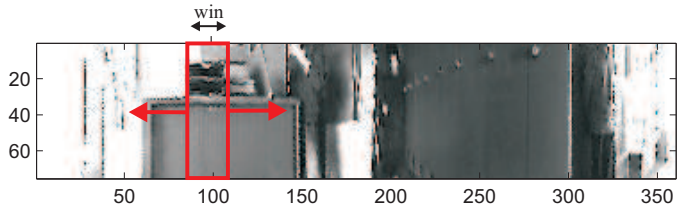

Fig. 10. This is the same image of Fig. 7 after unwrapping into a cylindrical panorama. The rectangular region used to compute SSD and NCC is also shown.

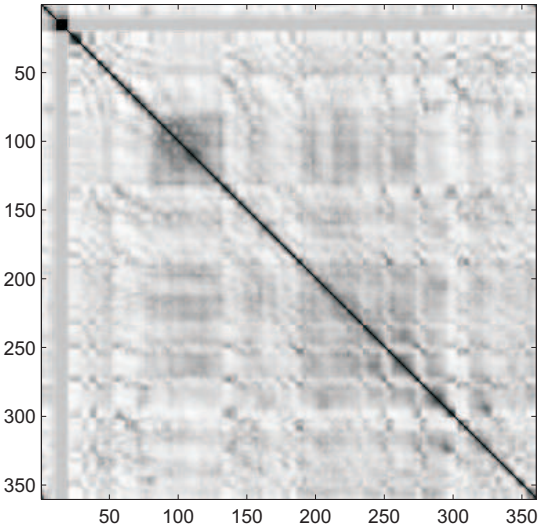

Fig. 11. Similarity matrix for descriptors.

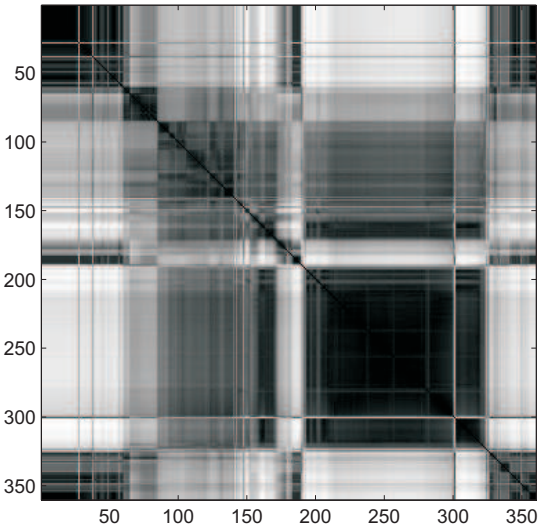

Fig. 12. Similarity matrix for SSD.

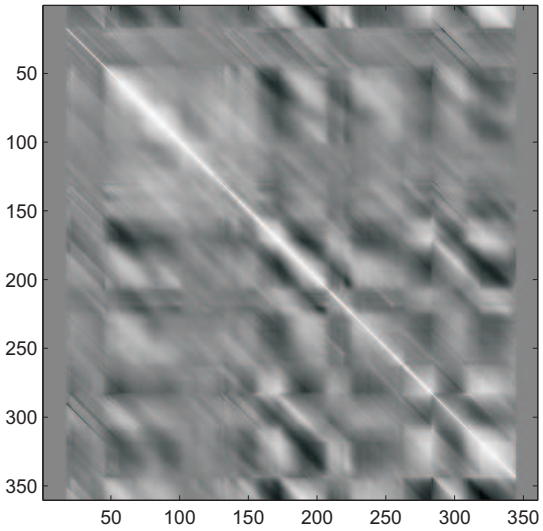

Fig. 13. Similarity matrix for NCC.

hyperbolic mirror and a SONY CCD camera the resolution of 640×480 pixels. A picture of the all settings is depicted in Fig. 1.

In this section, we present some experimental results obtained moving our robot in a real indoor environment. We show the performance of our feature tracker during the motion of the robot. In these experiments, the robot was moving at about $0.15\ m/s$ and was acquiring frames at $3\ Hz$, meaning that the traveled distance between two consecutive frames was $5\ cm$.

We guided our robot through an office-like environment for about 40 meters. The results of feature tracking are shown in Fig. 15. Observe that these results were obtained using only the three matching rules described in Sections IV-A, IV-B, IV-C. No other criterion as mutual and topological relations has been used. The plot refers to a short path of the whole trajectory while the robot was coping with an L-shaped trajectory. As the reader may observe, there many features that are detected and 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 starts from it. A few false matches are also present at the time where two lines intersect. Observe, that the three huge jumps in the plot are not false matches; they are only due to the angle transition from $-\pi$ to π.

Observe that our algorithm is able to match features even when their correspondents are not found in previous frames. This can be seen by observing that sometimes circles are missing on the tracks (look for instance at track no. 52). When correspondence is not found in the previous frame, we start looking into all previous frames (actually up to twenty frames back) and stop when the correspondence is found.

If you examine the graph, you can see that some tracks are suddenly given different numbers. For instance, observe that feature no. 1 - that is the fist 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, you 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, we found 8 false matches and 26 false new entries. Comparing these errors to the 2758 corresponding pairs detected by the algorithm over the whole video sequence, we had 1.23% of mismatches. Furthermore, we found that false matches occurred every time the camera was facing objects with repetitive texture (as in Fig 10). 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 whenever 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 rather 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.

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 [16] 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 [16] 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. 14).
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. 14).

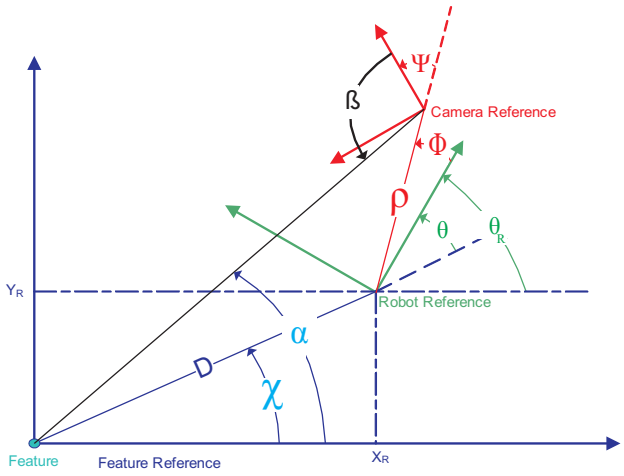

Fig. 14. The two reference frames respectively attached to the robot and to the camera. The five parameters estimated by the EKF $(D, \theta, \phi, \rho, \psi)$ are also indicated.

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. 14). 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:

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

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:

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

where e is the distance between the wheels.

For a particular bearing angle observation β, we obtain the following analytical expression (see Fig. 14):

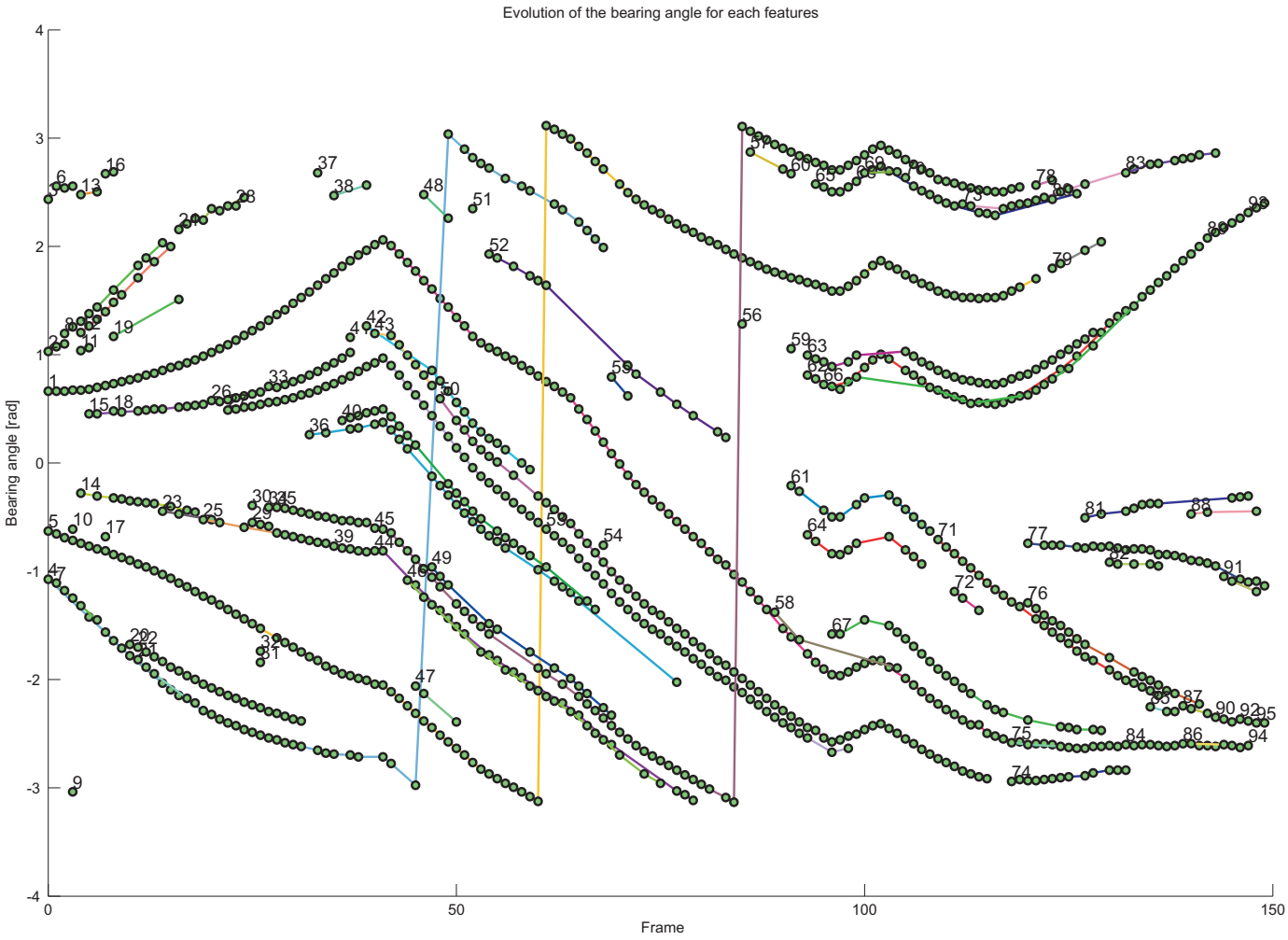

Fig. 15. Feature tracking during the motion of the robot. In y-axis is the angle of sight of each feature and in the x-axis the frame number. Each circle represents a feature detected in the observed frame. Lines represent tracked features. Numbers appear only when a new feature is detected.

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

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 [16], by considering the system state $\mathbf{X_a}$ as defined in (18), the system is not observable, that is, it does not contain whole the necessary information to perform the estimation with an error which is bounded. Conversely, in [16] 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. 14). 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.

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

$$\begin{cases} D_{i+1} &= D_i + \delta\rho_i cos\theta_i \\ \theta_{i+1} &= \theta_i + \delta\theta_i - \frac{\delta\rho_i}{D_i} sin\theta_i \\ \phi_{i+1} &= \phi_i \\ \rho_{i+1} &= \rho_i \\ \psi_{i+1} &= \psi_i \end{cases} \tag{20}$$

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:

$$\beta_i = \tan^{-1}\left(\frac{-\rho_i sin(\theta_i + \phi_i)}{-D_i - \rho_i\ cos(\theta_i + \phi_i)}\right) - \theta_i - \phi_i - \psi_i \tag{21}$$

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:

$$\mathbf{X} = [D^1, \theta^1, D^2, \theta^2, \ldots, D^Z, \theta^Z, \phi, \rho, \psi]^T, \tag{22}$$

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

Before implementing the EKF, we need to compute the the dynamics function f and the observation function h, both depending on the state $\mathbf{X}$. From (20) and using (22), the dynamics f of the system can be written as:

$$\mathbf{X}_{i+1} = f(\mathbf{X}_i, \mathbf{u}_i) = \begin{bmatrix} D_i^1 + \delta\rho_i cos\theta_i^1 \\ \theta_i^1 + \delta\theta_i - \frac{\delta\rho_i}{D_i^1} sin\theta_i^1 \\ D_i^2 + \delta\rho_i cos\theta_i^2 \\ \theta_i^2 + \delta\theta_i - \frac{\delta\rho_i}{D_i^2} sin\theta_i^2 \\ \vdots \\ D_i^Z + \delta\rho_i cos\theta_i^Z \\ \theta_i^Z + \delta\theta_i - \frac{\delta\rho_i}{D_i^Z} sin\theta_i^Z \\ \phi_i \\ \rho_i \\ \psi_i \end{bmatrix} \tag{23}$$

with $\mathbf{u} = [\delta\rho_R, \delta\rho_L]^T$.
Regarding the observation function h, from (21) we have:

$$h(\mathbf{X}_i) = \begin{bmatrix} \beta_i^1 \\ \beta_i^2 \\ \vdots \\ \beta_i^Z \end{bmatrix} =$$

$$= \begin{bmatrix} \tan^{-1}\left(\frac{-\rho_i sin(\theta_i^1 + \phi_i)}{-D_i^1 - \rho_i\ cos(\theta_i^1 + \phi_i)}\right) - \theta_i^1 - \phi_i - \psi_i \\ \tan^{-1}\left(\frac{-\rho_i sin(\theta_i^2 + \phi_i)}{-D_i^2 - \rho_i\ cos(\theta_i^2 + \phi_i)}\right) - \theta_i^2 - \phi_i - \psi_i \\ \vdots \\ \tan^{-1}\left(\frac{-\rho_i sin(\theta_i^Z + \phi_i)}{-D_i^Z - \rho_i\ cos(\theta_i^Z + \phi_i)}\right) - \theta_i^Z - \phi_i - \psi_i \end{bmatrix} \tag{24}$$

The previous equations, along with a statistical error model of the odometry (we used the one by Chong and Kleeman [17]), 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 (23) with respect to the state $\mathbf{X}$ and with respect to encoder readings ($\delta\rho_R$ and $\delta\rho_L$). Finally, we need to compute the Jacobian $\mathbf{H}$ of the observation function (24) with respect to $\mathbf{X}$. These matrices are required to implement the EKF [39] and are given in the Appendix.

IX. Calibration Results

In our experiments, we adopted the same mobile robot and omnidirectional camera described in Section VI. 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. A picture of the all settings is depicted in Fig. 1.

For our experiments, we positioned an omnidirectional camera on our robot as in Fig. 1 and we measured manually its position relative to the robot. We measured the following values: $\phi \simeq 0\ rad$, $\rho \simeq 0.2\ m$, $\psi \simeq 0\ rad$. The scenario is shown in Fig. 17. In this figure, the features used for the calibration are highlighted. A 2D scan of the test environment is also shown; as we mentioned above, the laser scan is used only as a ground truth. The robot reference system (in black) and the camera reference system (in red) are also indicated. The rays departing from the camera origin show the bearing angle of the features.

In Fig. 17, the position of the camera relative to the robot before calibration is shown. Note that, since the calibration was initially done by hand, the rays of bearing do not properly intersect the expected corners of the 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. 16.

The values of ϕ, ρ, ψ estimated during the motion are plotted as a function of the frame number in Fig. 19. The covariances σ_ϕ, σ_ρ, σ_ψ are also plotted. Observe that after about 60 frames (corresponding to about $1.5\ m$ 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 it was already pointed out in [16], the convergence is very fast when the robot performs trajectories alternating short straight paths and pure rotations.

In Fig. 18, the scenario after calibration is shown and can be compared with that before calibration (Fig. 17). Observe, that the calibration parameters are well estimated. Indeed, the rays of bearing intersect very well the expected corners of the laser scan.

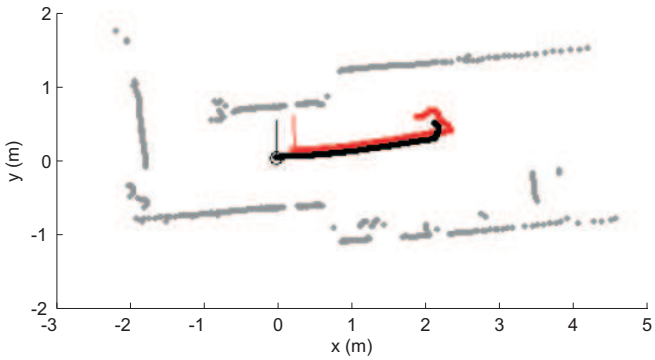

Fig. 16. The path performed by the robot during self-calibration, i.e. straight path followed by a rotation.

X. Conclusion

In this paper, we presented a robust method for matching vertical lines among omnidirectional images and a method for automatically calibrating an omnidirectional camera with the robot system.

Concerning the first part, the basic idea to achieve robust feature matching consists in creating a descriptor which is unique and distinctive for each feature. The distinctiveness of the descriptor in comparison to other image similarity metrics was also pointed out. Furthermore, this descriptor is invariant to rotation and slight changes of illumination. To evaluate the performance of our approach, we performed real experiments where we evaluated the quality of matching. 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 [16]. In [16], 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.

XI. 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

[1] 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.

[2] 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.

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

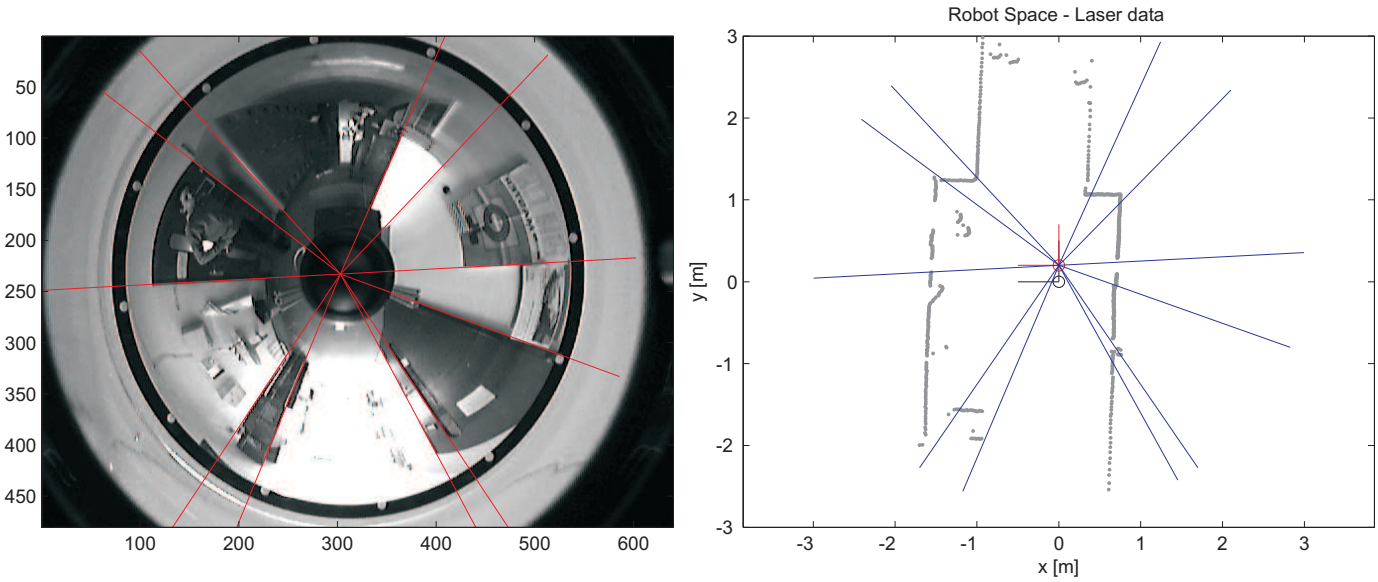

Fig. 17. Scenario before calibration: (left) vertical features; (right) rays of bearing of the correspondent features. The rays corresponding to the vertical features do not accurately intersect the corners in the laser scan.

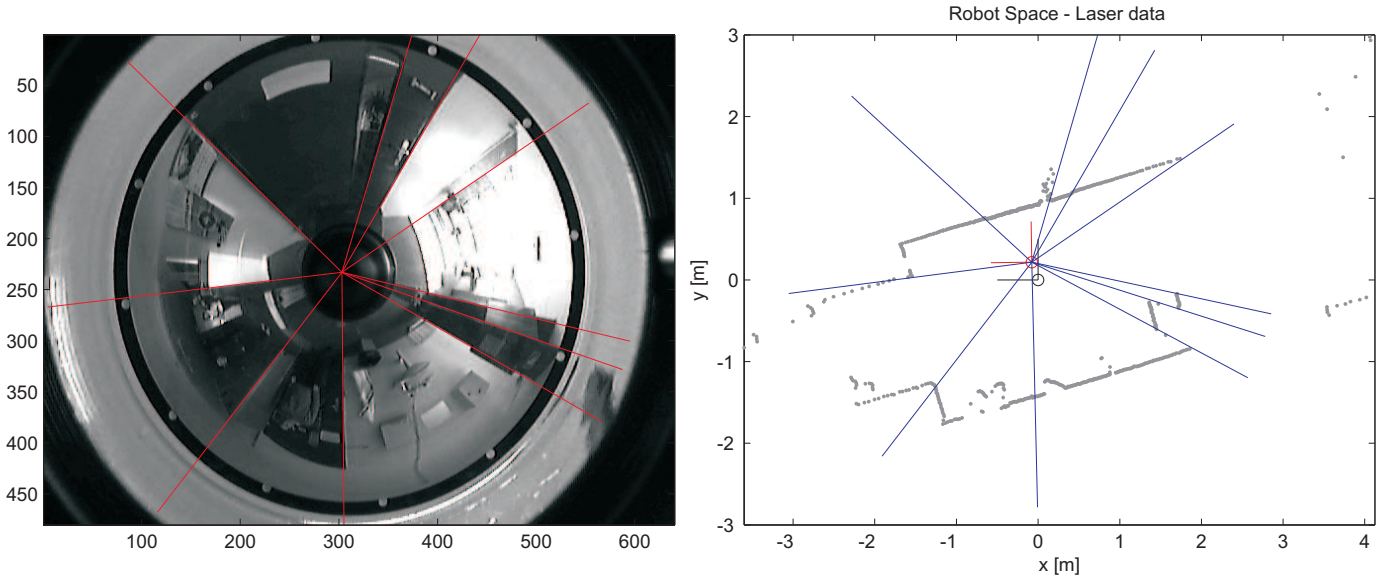

Fig. 18. Scenario after calibration: (left) vertical features; (right) rays of bearing. The rays corresponding to the vertical features accurately intersect the corners in the laser scan.

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

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

[6] 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.

[7] 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.

[8] 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.

[9] 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.

[10] Shree 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.

[11] 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.

[12] D. Scaramuzza, A. Martinelli, R. Siegwart. A Flexible Technique for Accurate Omnidirectional Camera Calibration and Structure from Motion. In Proc. of the IEEE International Conference on Vision Systems, New York, New York, 2006.

[13] D. Scaramuzza, A. Martinelli, R. Siegwart. A Toolbox for Easily Calibrating Omnidirectional Cameras. In Proc. of the IEEE International Conference on Intelligent Systems, IROS06, Beijing, China, 2006.

[14] 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.

[15] 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.

[16] 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.
[17] 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.
[18] 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.
[19] 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.
[20] D. Prasser, G. Wyeth, M. J. Milford (2004b), Experiments in Outdoor Operation of RatSLAM, Australian Conference on Robotics and Automation, Canberra Australia, 2004.
[21] R. Gonzalez, R. Woods, Digital Image Processing, Addison Wesley, Prentice Hall, ed. 2, ISBN: 0201180758, 2002.
[22] 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.
[23] 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.
[24] Medioni, G. and Nevatia, R., 1985. Segment-based stereo matching. Computer Vision, Graphics and Image Processing 31, pp. 218.
[25] Ayache, N., 1990. Stereovision and Sensor Fusion. MITPress.
[26] Zhang, Z., 1994. Token tracking in a cluttered scene. Image and Vision Computing 12(2), pp. 110120.
[27] Crowley, J. and Stelmazyk, P., 1990. Measurement and integration of 3d structures by tracking edge lines. In: Proc. ECCV, pp. 269280.
[28] Deriche, R. and Faugeras, O., 1990. Tracking line segments. In: Proc. ECCV, pp. 259267.
[29] Huttenlocher, D. P., Klanderman, G. A. and Rucklidge, W. J., 1993. Comparing images using the Hausdorff distance. IEEE T-PAMI.
[30] Ayache, N. and Faugeras, O., 1987. Building a consistent 3D representation of a mobile robot environment by combining multiple stereo views. In: Proc. IJCAI, pp. 808810.
[31] Horaud, R. and Skordas, T., 1989. Stereo correspondence through feature grouping and maximal cliques. IEEE TPAMI 11(11), pp. 11681180.
[32] Gros, P., 1995. Matching and clustering: Two steps towards object modelling in computer vision. Intl. J. of Robotics Research 14(6), pp. 633642.
[33] Venkateswar, V. and Chellappa, R., 1995. Hierarchical stereo and motion correspondence using feature groupings. IJCV pp. 245269.
[34] 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.
[35] 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.
[36] 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.
[37] D.G. Lowe, Distinctive image features from scale-invariant keypoints, in International Journal of Computer Vision, vol. 20, pp. 91-110, 2003.
[38] 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.
[39] Y. Bar-Shalom and T.E. Fortmann, Tracking and data association, mathematics in science and engineering, vol. 179, Academic Press, 1988.

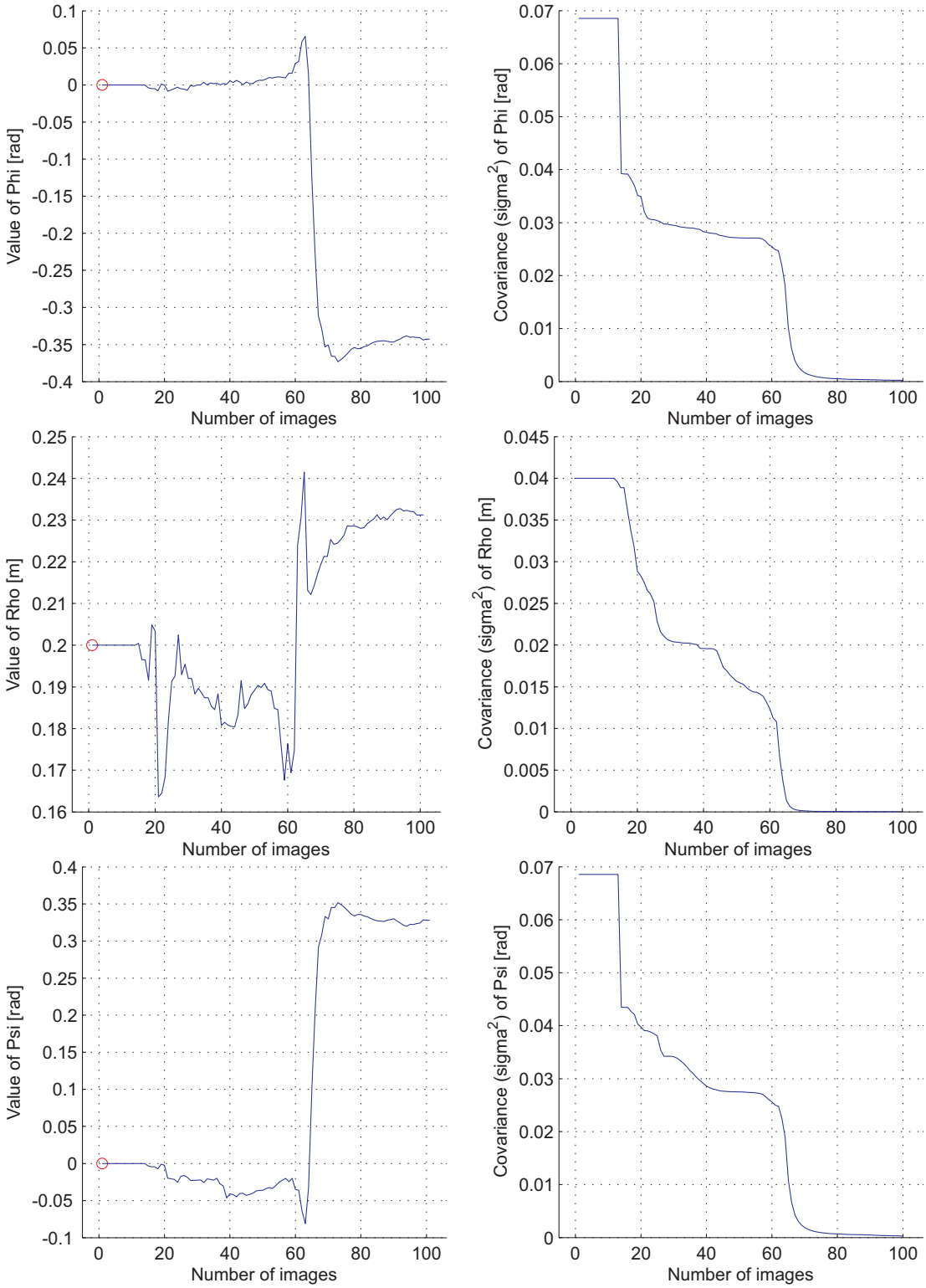

Fig. 19. ϕ, ρ, ψ, ${\sigma_\phi}^2$, ${\sigma_\rho}^2$, ${\sigma_\psi}^2$ as a function of the frame number. The distance traveled between two frames is about 2.5 cm.

XII. APPENDIX

The Jacobians $\mathbf{F_x}$ and $\mathbf{F_u}$ of the dynamics are:

$$F_x = \begin{bmatrix} A^1 & 0 & \cdots & 0 & 0 & 0 & 0 \\ 0 & A^2 & \cdots & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \cdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & A^Z & 0 & 0 & 0 \\ 0 & 0 & \cdots & 0 & 1 & 0 & 0 \\ 0 & 0 & \cdots & 0 & 0 & 1 & 0 \\ 0 & 0 & \cdots & 0 & 0 & 0 & 1 \end{bmatrix}, \tag{25}$$

and

$$F_u = \begin{bmatrix} B^1 \\ B^2 \\ \vdots \\ B^Z \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{26}$$

with

$$A^i = \begin{bmatrix} 1 & -\delta\rho\ sin\theta^i \\ \frac{\delta\rho}{D^{i2}}\ sin\theta^i & 1 - \frac{\delta\rho}{D^i} cos\theta^i \end{bmatrix}, \tag{27}$$

$$B^i = \begin{bmatrix} \frac{cos\theta^i}{2} & \frac{cos\theta^i}{2} \\ \frac{1}{e} - \frac{sin\theta^i}{2D^i} & -\frac{1}{e} - \frac{sin\theta}{2D^i} \end{bmatrix} \tag{28}$$

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

$$\mathbf{H} = \begin{bmatrix} H1^1 & H2^1 & 0 & 0 & \cdots & 0 & 0 & H3^1 & H4^1 & -1 \\ 0 & 0 & H1^2 & H2^2 & \cdots & 0 & 0 & H3^2 & H4^2 & -1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & H1^Z & H2^Z & H3^Z & H4^Z & -1 \end{bmatrix} \tag{29}$$

with

$$H1^i = \frac{-\rho sin(\theta^i + \phi)}{{D^i}^2 + 2\rho D^i cos(\theta^i + \phi) + \rho^2}, \tag{30}$$

$$H2^i = \frac{-D^i\rho\ cos(\theta^i + \phi) - {D^i}^2}{{D^i}^2 + 2\rho D^i cos(\theta^i + \phi) + \rho^2}, \tag{31}$$

$$H3^i = \frac{-D^i\rho\ cos(\theta^i + \phi) - {D^i}^2}{{D^i}^2 + 2\rho D^i cos(\theta^i + \phi) + \rho^2}, \tag{32}$$

$$H4^i = \frac{D^i sin(\theta^i + \phi)}{{D^i}^2 + 2\rho D^i cos(\theta^i + \phi) + \rho^2}\ . \tag{33}$$