Robust Feature Extraction and Matching for Omnidirectional Images

Davide Scaramuzza[1], Nicolas Criblez[2], Agostino Martinelli[3], and Roland Siegwart[4]

[1] Swiss Federal Institute of Technology Zurich davide.scaramuzza@ieee.org
[2] Swiss Federal Institute of Technology Lausanne nicolas.criblez@gmail.com
[3] INRIA agostino.martinelli@ieee.org
[4] Swiss Federal Institute of Technology Zurich r.siegwart@ieee.org

Summary. This paper presents a new and robust method for extracting and matching visual vertical features between images taken by an omnidirectional camera. Matching robustness is achieved by creating a descriptor which is unique and distinctive for each feature. Furthermore, the proposed descriptor is invariant to rotation. The robustness of the approach is validated through real experiments with a wheeled robot equipped with an omnidirectional camera. We show that vertical lines are very well extracted and tracked during the robot motion. At the end, we also present an application of our algorithm to the robot simultaneous localization and mapping in an unknown environment.

1 Introduction

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, some works have provided robust solutions for wide-baseline stereo matching, structure from motion, ego-motion estimation, and robot navigation (see [1], [2], [3], [4], [5], [6], [7], [8], and [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 [15], 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 [16]. 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, the features we want to track from omnidirectional images are real world vertical features, which are predominant in structured environments. In our experiments, we used a wheeled robot equipped with an omnidirectional camera, which had the camera axis perpendicular to the direction of motion of the robot. Because of this settings and assuming the environment to be flat, all world vertical lines project into radial lines on the image plane.
The novelty of this paper consists of a robust method to match vertical lines between images taken by an omnidirectional camera during the motion of the robot. Matching robustness is achieved by creating a descriptor which is unique and distinctive for each feature. Furthermore, the proposed descriptor is invariant to rotation. This descriptor is based on the image gradients. The robustness of the approach was validated through real experiments by using a robot equipped with an omnidirectional camera. In this paper, we show that vertical lines are very well extracted and tracked during the robot motion. At the end, we also present an application of our algorithm to the robot Simultaneous Localization And Mapping (SLAM) in an unknown environment.
This paper is organized as follows. In section 2, we will discuss the vertical line extraction. In section 3, we will illustrate how to mark a feature by means of a descriptor, and, in section 4, we will describe the feature matching process. Finally, in section 5, we will present our experimental results and the application of our algorithm to SLAM.

2 Vertical Line Extraction

Our platform consists of a wheeled robot equipped with an omnidirectional camera looking upwards. 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.
Figure 1 shows a sample picture taken by our omnidirectional camera. As the circular external boundary of the mirror is visible in the image, we use a circle

detector to determine the location of the center.
In our arrangement, we set the camera-mirror system perpendicular to the floor where the robot moves. This guarantees that all vertical lines of the environment converge towards the image center. To extract the vertical lines, we first compute the image gradients (e.g. we used a Sobel filter), and then we keep only those gradients whose orientation looks towards the image center up to $\pm 5^\circ$. This 10° tolerance allows us to deal with the effects of the floor irregularities in the projections of the 3D vertical lines. After this filtering, we apply non-maxima suppression and we end up with a binary edge map (Fig. 2).
The next step will be identifying 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. 3. As observed in Fig. 2, there are many potential vertical lines in a structured environment. To keep the most reliable and robust features, we choose only those lines whose length covers at least half of the angle of view (Fig. 3). Finally, we use non-maxima suppression to avoid the features to be too close (Fig. 4).

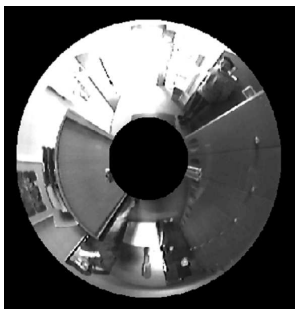

Fig. 1. An image taken by our omnidirectional camera.

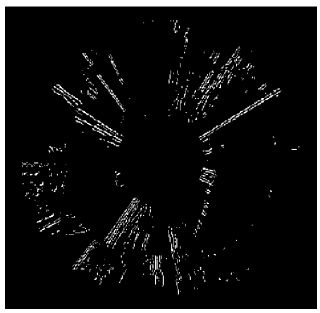

Fig. 2. The final binary edge map after non-maxima suppression.

3 Building the Descriptor

In section 4, we will describe our method for matching vertical lines between consecutive frames while the robot is moving in an unknown environment. 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. In this way, finding the correspondent of a vertical line can be done by looking for the line with the

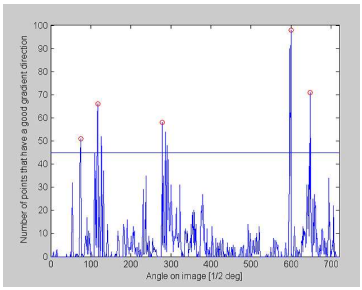

Fig. 3. The number of binary pixels voting for a given orientation angle. The orientation axis ranges from 0 to 720 half degrees.

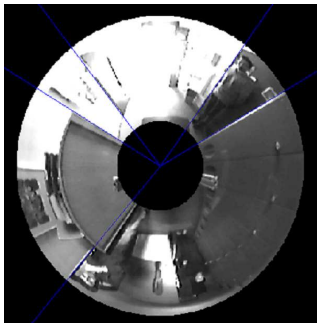

Fig. 4. Extraction of the most reliable vertical features from an omnidirectional image.

closest descriptor. In the next subsections, we will describe how to build this descriptor.

3.1 Rotation Invariance

To build the descriptor, we extract a predefined number of circular areas (namely 3 areas) in fixed position along a given radial line (Fig. 5). The centers of these circular areas are equally spaced and the radius is chosen such that the circles touch without overlapping. Then, each area is smoothed by a Gaussian window of variance σ and the image gradients (magnitude and phase) are computed within each of these areas. The rotation invariance is achieved redefining the gradient phase relatively to the radial line's angle. (Fig. 5).

3.2 Orientation Histograms

To make the descriptor robust to false matches, we split each circular area into two parts and consider each one individually (Fig. 6). 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 an orientation histogram (Fig. 7) of all gradient vectors. 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

$$w = N_b \cdot \frac{o - b}{2\pi} \tag{1}$$

with o being the observed 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 has been 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. The reader observe that the orientation histogram is already rotation invariant because the gradient angles have been referred to the radial line's angle.

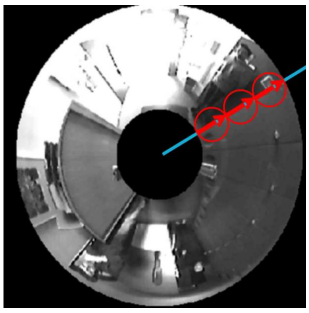

Fig. 5. Extraction of the circular areas. To have rotation invariance, the gradient phase is referred to the orientation of the vertical line.

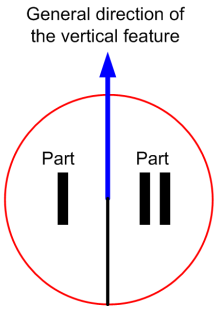

Fig. 6. Two parts of a circular area.

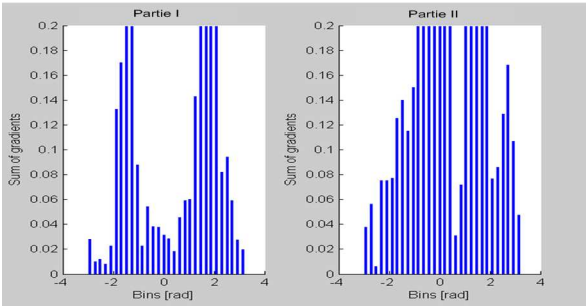

Fig. 7. An example of gradient orientation histograms for the left and right sides of a circular area.

3.3 Building the Feature descriptor

The computed orientation histograms help to build the feature descriptor. Indeed, the descriptor is an N-element vector containing all histogram values of the circular areas. For instance, by extracting 3 circular areas for each

vertical feature and choosing 30 bins for each histogram, the length of the feature descriptor will be

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

Finally, it is important to know that all feature descriptors have the same length N. To achieve slight illumination invariance, every descriptor is normalized to 1. 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.

4 Feature Matching

As every vertical feature has its own descriptor, the correspondent of a vertical line in the consecutive images can be searched among the features with the closest descriptor. As a distance measure between two vector descriptors $\mathbf{A}$ and $\mathbf{B}$, we use the Euclidean distance:

$$d(\mathbf{A}, \mathbf{B}) = \sqrt{\sum_{k=1}^{N} (\mathbf{A}(k) - \mathbf{B}(k))^2} \tag{3}$$

As a consequence, the correspondent of a feature, in the current 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 define three tests to decide whether a feature correspondent exists and which the correspondent is. Before describing the three criterions, let us introduce some definitions.
Say $\{\mathbf{A_1}, \mathbf{A_2}, ..., \mathbf{A_{N_A}}\}$ and $\{\mathbf{B_1}, \mathbf{B_2}, ..., \mathbf{B_{N_B}}\}$ two sets of feature descriptors extracted at time t and $t-1$ respectively (where N_A and N_B are the number of features in the first and second image). Then, say

$$D_i = \{d(\mathbf{A_i}, \mathbf{B_j}), j = 1, 2, ..., N_B)\} \tag{4}$$

the set of all distances between a given $\mathbf{A_i}$ and all $\mathbf{B_j}$ ($j = 1, 2, , N_B$). Finally, say $minD_i = min(D_i)$ the minimum of the distances between $\mathbf{A_i}$ and all $\mathbf{B_j}$.

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

$$minD_i = F_1 \cdot N \tag{5}$$

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

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:

$$minD_i = F_2 \cdot < D_i > \tag{6}$$

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 1, we show a real comparison among the distances between the descriptor $\mathbf{A_1}$ at time t and all the descriptors at time $t-1$. There, the descriptor $\mathbf{B_1}$ is the correct correspondent of $\mathbf{A_1}$. The reader might also observe that its distance is smaller than the mean of all other distances.

Table 1. The distances between the descriptor $\mathbf{A_1}$ at time t and all descriptors $\mathbf{B_j}$, $j = 1, 2, .., N_B$ at time $t-1$

B1	B2	B3	B4	B5	B6	B7
2.38	5.42	4.55	5.79	5.66	6.17	5.43

Third Test

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

$$minD_i = F_3 \cdot SecondSmallestDistance, \tag{7}$$

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 jump between the closest and the second closest descriptor. Factors F_1, F_2, and F_3 are to be determined experimentally. The empirical values we used for these parameters are shown in Table 2.

Table 2. The parameters used by our algorithm with their empirical values

F1 = 0.0075	F2 = 0.55	F3 = 0.85

5 Experimental Results

In this section, we present some experimental results obtained by moving our robot in a real indoor environment. In the first subsection, we show the performance of our feature tracker during the motion of the robot, while in the second one, we present the results of our feature tracker applied to SLAM. In these experiments, the robot was moving at about 0.15 m/s. The image size was $640x480$ $pixels$ and the frame rate was 3 Hz.

5.1 Performance of the Feature Tracking

In this experiment, we guided our robot through an office-like environment for about 70 meters. The results of the feature tracking are shown in Fig. 8. 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, many features are detected and tracked over the time. Indeed, all lines of the plot appear to be smooth and homogeneous. Furthermore, the lines do not intersect, meaning that there was no false matching. We can also notice that the algorithm was able to match vertical elements even when the correspondent features were not observed in the previous image (e.g. observe the large gap between the dots pointed to by the arrow in Fig. 8). Indeed, when the correspondence is not found in the last frame, our algorithm starts looking into all previous frames (actually up to the twentieth frame), and stops when the correspondence is found. By zooming into the plot of Fig. 8, we found that some lines are given different numbers. For instance, feature number 24 is labeled as 31 after some frames. And the same happens with features 42 and 49. When this happens, it means that the algorithm found no correspondence for the current feature, and thus, the feature is labeled as a new entry, but in fact this is a false new entry. After having visually checked every single frame of the video sequence, we found 6 false matches and 22 false new entries. Comparing these errors to the 2631 corresponding pairs detected by the algorithm over the whole video sequence, we had 1.06% of mismatches. Furthermore, we found that the false matches occurred every time the camera was facing objects with repetitive texture. Thus, the ambiguity was caused by the presence of vertical elements which were almost identical. On the other hand, a few of false new entries occurred whenever the displacement of the robot between two successive images was big. However, the reader should observe that when a feature matches with no other previous feature of the last frames, it is better to believe this feature to be new rather than commit a false matching.

5.2 Application to SLAM

We applied our feature tracker to two important problems in autonomous navigation, that is, sensor self-calibration and SLAM. Regarding the former, the results can be found in [12], [13], and [14]. In this section, we show only the results we obtained for SLAM. We implemented the standard EKF based SLAM. In particular, the EKF estimates the vector:

$$X = [x_r, y_r, \theta_r, X_1, Y_1, ..., X_{N_O}, Y_{N_O}]^T \tag{8}$$

where $[x_r, y_r, \theta_r]$ is the robot configuration, X_i, Y_i are the Cartesian coordinates of the i-feature in the map, and N_O is the number of observed features. Our mobile robot is equipped with wheel encoders and with the same omnidirectional camera adopted in the experiments described in section 5.1. The

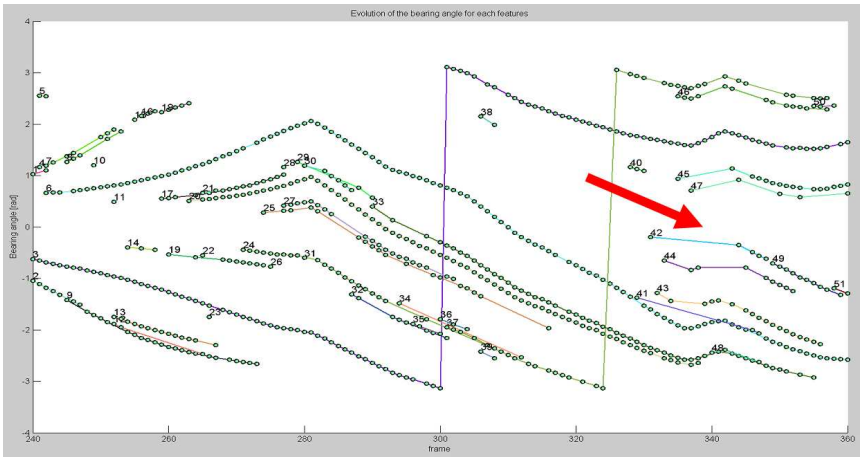

Fig. 8. Feature tracking during the robot motion. In y-axis, we have the angle of sight of each feature, and in the x-axis, the frame number. Each dot represents a feature detected in the current frame. The lines represent the tracked features. The number reported on some dot appears only when a new feature is detected.

bearing observations provided by the omnidirectional camera consist of the vector z whose components are:

$$\begin{aligned} z_1 &= arctan(\frac{Y_1 - y_r}{X_1 - x_r}) \\ &\vdots \\ z_{N_O} &= arctan(\frac{Y_{N_O} - y_r}{X_{N_O} - x_r}) \end{aligned} \tag{9}$$

To initialize a new feature in the map, consecutive bearing observations as in [14], which refer to the same feature, are integrated with the odometry. Then, the estimation is improved by integrating the information coming from all the bearing observations through the EKF. The result is shown in Fig. 9 where both the robot trajectory and the position of the features are shown.

6 Conclusion

In this paper we introduced and discussed a new and robust method to extract and match vertical lines between images taken by an omnidirectional camera. The basic idea to achieve robust feature matching consists of creating a descriptor which is unique and distinctive for each feature. Furthermore, this descriptor is invariant to rotation. To evaluate the performance of our approach, we performed real experiments where we evaluated the quality of the

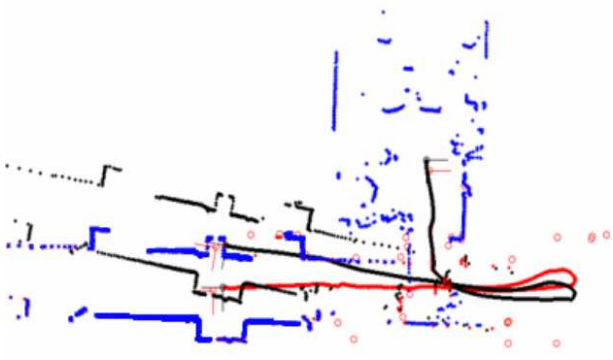

Fig. 9. The results obtained by implementing a simple EKF-SLAM, which uses the proposed feature tracker. The black line is the trajectory estimated by using the odometry alone. The red line is the trajectory estimated by the EKF using both the odometry and vertical lines. The blue points represent the map ground truth provided by a laser range finder. The red circles are the detected verticals features.

matching. We conclude that the proposed approach is very robust and precise. Finally, we adopted the proposed method to implement an EKF based SLAM.

7 Acknowledgment

This work was conducted within the EU Integrated Projects COGNIRON (*The Cognitive Robot Companion*) and BACS (*Bayesian Approach to Cognitive Systems*). It was funded by the European Commission Division FP6-IST Future and Emerging Technologies under the contracts FP6-IST-002020 and FP6-IST-027140 respectively.

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.
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. A. Martinelli, D. Scaramuzza, R. Siegwart, Automatic Self-Calibration of a Vision System during Robot Motion, in Proc. of the IEEE International Conference on Robotics and Automation, ICRA06, Orlando, Florida, USA, 2006.
15. 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.
16. 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.