Precise and Robust Line Detection for Highly Distorted and Noisy Images

Dominik Wolters(✉) and Reinhard Koch

Department of Computer Science, Kiel University, Kiel, Germany
dwol@informatik.uni-kiel.de

Abstract. This article presents a method to detect lines in fisheye and distorted perspective images. The detection is performed with subpixel accuracy. By detecting lines in the original images without warping the image with a reverse distortion, the detection accuracy can be noticeably improved. The combination of the edge detection and the line detection to a single step provides a more robust and more reliable detection of larger line segments.

1 Introduction

The purpose of this research is the development of a line detector which can handle highly distorted images, in particular fisheye images, without undistorting these images. This should lead to greater accuracy since the detection takes place in the original image and no warping is necessary. In addition, the entire image area can be used. The focus is on the detection of line segments that are suitable for use in line-based structure from motion (SfM).

To achieve a high accuracy in the reconstruction, the lines should be detected with subpixel accuracy. Furthermore, a complete, fast and robust detection of the entire line without interruptions is important, even if the background changes or the surfaces are heavily textured. In the reconstruction short line segments often lead to inaccurate triangulation results and multiple edges. In addition, the matching of many small line segments is time-consuming and error-prone.

1.1 Related Work

Line Detection. Accurate detection of lines is relevant for many areas of image processing. Lines are an important low-level feature, especially in man-made environments. They are used e.g. to describe the objects in images, pose estimation [19], structure from motion [16,18] and stereo matching [3] but also for object detection [10] or camera calibration [17].

The aim of the line detection is to find edge pixels belonging to the same straight line segments. Edges are contours or borders of different image regions. A classical approach to edge detection is the Canny edge detector [8].

Simple approaches to detect lines are RANSAC [13] or the Hough transform [2]. There are many improved and optimized variants which use the Hough

B. Rosenhahn and B. Andres (Eds.): GCPR 2016, LNCS 9796, pp. 3–13, 2016.
DOI: 10.1007/978-3-319-45886-1_1

transform, e.g. [12]. However, these approaches usually require a binary edge map and detect infinitely long lines, that need to be split into line segments. To obtain good results, the parameters have to be adjusted for each scene.

First attempts at line segment detection by grouping pixel based on the gradient magnitude and gradient direction were described by Burns et al. [7]. The line segment detector presented by Gioi et al. [14] enables detection of lines in images without tuning parameters. The EDLine detector [1] presented by Akinlar and Topal uses the Edge Drawing algorithm instead of a classical edge detector, e.g. Canny edge detector, to find edges in the image. The detector is capable of real time and also requires no parameter tuning.

The condition for the application of these algorithms is the presence of perspective input images without distortion.

Some works deal with the detection of lines in fisheye images. Examples are e.g. [4,5]. They detect great circles on the equivalence sphere. Other approaches detect arcs in fisheye images [6,20].

Fisheye Camera Model. In this work the fisheye camera model from OpenCV is used for the calibration of the fisheye cameras. It is based on the generic camera model proposed by [15]. The model, however, can easily be replaced by any other distortion model.

Contribution. In this paper, we propose a line detector, which is capable of processing both distorted perspective images and fisheye images, without requiring them to be undistorted. The detection is done with subpixel accuracy. We combine the edge and line detection to a single detection step. The advantage of this approach is that the line is already used as a model for the detection. In this way a reliable and robust detection is achieved, which makes it possible to detect continuous line segments, that usually were fragmented in current approaches.

2 Methods

2.1 Line Detection on Fisheye Images

Approach. The standard methods used for detecting lines in images expect undistorted images as input. Therefore, the conventional approach to detect lines in distorted perspective or fisheye images is to correct those distortions by warping the image with a reverse distortion and use this software-corrected images as input (Fig. 1).

To use a software-corrected image instead of the original image for the line detection has many disadvantages and should be avoided. The reason for this is on the one hand the performance, because the image has to be warped, and on the other hand, the accuracy because the original image is transformed e.g using a bilinear interpolation. Without adjusting the focal length, the software-corrected fisheye image either gets very large or large parts of the images are cut. An adjustment of the focal length means that the image center, which usually

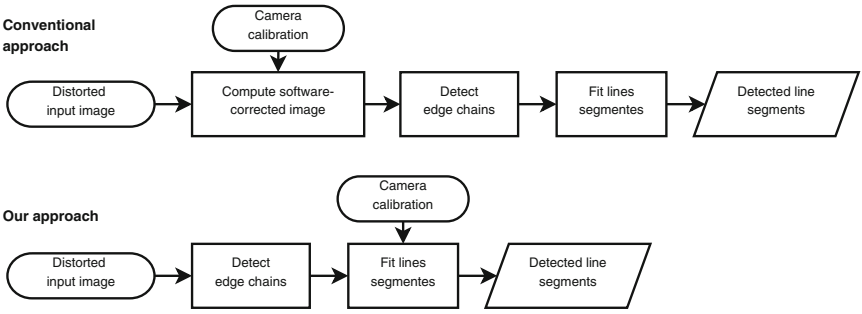


Fig. 1. Application of line detection for fisheye images

contains the relevant image content, is scaled down, so that information is lost here. Another aspect is that errors in the calibration directly affect the detection accuracy of the lines, if the image is warped with a inaccurate reverse distortion. If the intrinsic camera parameters are corrected in a later processing step, for example, by a bundle adjustment, the complete detection including the software correction of all images and the calculation of the descriptors must be performed again.

For these reasons we do not use a software-corrected image but use the original image as input to the line detection (Fig. 1).

Algorithms. Our approach is based on the EDLines algorithm [1]. Its advantages over other line-segment detectors are that it provides robust and accurate results and is also very fast. Unlike classical edge detectors, the Edge Drawing step produces contiguous and exactly one pixel wide, well localized edges.

The EDLines algorithm consists of three sub-steps: First, edge detection is done with Edge Drawing (ED) algorithm. This generates continuous chains of pixels. Line segments are then extracted from the generated pixel chains. The line candidate are finally validated by the Helmholtz principle [11].

In order to perform the approach to fisheye images, some adjustments are needed. The Edge Drawing step detects edges that do not necessarily correspond to straight lines so it can also be used for fisheye images. But for the line fitting step undistorted pixel coordinates are needed.

We undistort the edge coordinates for the line-fit instead of fitting circle arcs. We do this because in our case the calibrations of the cameras are already known and so any distortions can be considered. The approach can be used both for perspective images as well as fisheye images and it is ensured that only straight line segments are detected.

Adjustments to the Line Detection Process. The edge detection with Edge Drawing consists of several steps. First, the image is smoothed with a Gaussian filter to reduce noise (Fig. 2(a)). Subsequently, the gradient magnitude

(a) smoothed image (b) gradient map (c) anchor points (d) edge chains

Fig. 2. Process flow of the edge detection with edge drawing

(Fig. 2(b)), and the gradient direction are calculated for each pixel. In the gradient image anchor points are extracted (Fig. 2(c)), which are pixels with high probability of being edge elements. Specifically, these are the peaks in the gradient image. The anchor points are connected with smart routing (Fig. 2(d)). The gradient magnitude and gradient direction are used in this step. This produces a set of edge segments, which are connected chains of edge pixels.

The edge segments obtained are the input to the line extraction step. Here, the edge segments are split into one or more line segments. The basic idea is to run along the chain of edge pixel and fit line segments using a least-squares line fitting method. If the deviation of the pixels of the line is below a threshold value, the line is extended. If the threshold is exceeded, the line segment is terminated and the remaining pixels of the chain are then processed to extract further line segments. We choose the parameters according to the recommendations for the EDLines algorithm [1].

To enable line detection for fisheye images, in our approach, the pixel coordinates are undistorted during the line extraction step (Listing 1.1). In order to increase the robustness and prevent the premature breaking off of line segments, we allow some outlier pixels during the line fitting step in addition. Outliers are pixels that are connected to the chain, but do not fit to the line. Up to 3 pixels are skipped if thereafter again are pixels in the chain, which can be added to the line.

Listing 1.1. Pseudocode for the adjusted line extraction step

```
ExtractLineFisheye:
  undistort edge pixels
  while number of edge pixels greater than or equal n
    fit line to first n pixels
    if residual of fit greater than threshold
      remove first edge pixel
    else
      set first n edge pixels as line pixels
      while number of edge pixels greater than 0 and max. of iterations not reached
        fit line to line pixels
        set number of outliers to 0
        while number of edge pixels greater than 0
          if distance between next edge pixel and line greater than threshold
            increment number of outliers
            if number of outliers greater than outlier threshold
              break
          else
            add pixel to line
          remove first edge pixel
      store line
```

2.2 Line Detection with Subpixel Accuracy

So far, the detection of the line pixel is only done with pixel accuracy. The line equation is, however, set as the best-fit line through all discrete pixel positions so that the start and the end of the line can be specified with subpixel positions.

In the detection step only the pixels with the maximum gradient in the local environment are considered. If the local environment of the line is taken into account, a higher detection accuracy could be achieved. Possible approaches include an analysis of the distribution of the gradient magnitude perpendicular to the line or a subpixel localization of each line pixel before the line fit. We use the second approach, since in this case the rotation can be corrected and no warping of the line environment is required.

For each pixel of a line, the local environment (4-neighborhood or 8-neighborhood) is considered. The subpixel position is the average of the positions of the pixels in the neighborhood weighted by the gradient magnitude of the pixels.

2.3 Optimized Line Detection

Problems with High-Resolution Images. In high-resolution images, the lines often split into several individual segments. These short line segments have numerous disadvantages, especially for use in SfM: A triangulation of short line segments is more susceptible to noise, so that the position of the triangulated lines is inaccurate. A real line is described by a plurality of segments in the image, which are triangulated individually by line-based structure from motion applications. This leads to redundant lines in the reconstruction.

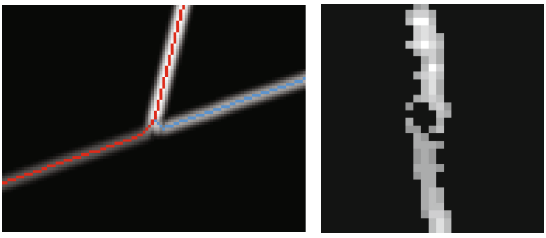
The Causes of Fragmented Lines. The fragmentation of the lines is caused by the Edge Drawing step. Here, contiguous pixel chains are formed starting from anchor points. The edge always follows the strongest gradient magnitude. By doing so, the edge chain kinks in the case of irregularities along the line and the pixel chains tear off. The reasons for such irregularities can be e.g. a junction of two lines, structured surfaces such as wood, but also noise. In the subsequent line-fitting step only short pixel chains are available. The result is that only short line segments are extracted.

Figure 3(a) shows the gradient map of a junction of two lines. The intersecting edge has a stronger gradient magnitude, therefore the pixel chain (red) kinks at the junction and does not form a continuous line.

Through gaps in the gradient (Fig. 3(b)), the pixel chains tear off and the lines fragment into several pieces.

An additional problem occurs with closed contours. The detection of edges starts from the anchor point in one direction and as long as pixels can be added, and then in the opposite direction. For closed contours the starting point is also the end point. If the start point is in the middle of a line, this line is unnecessarily divided into two parts.

Our goal to use lines for the reconstruction requires a reliable detection of lines. The lines should be detected without interruptions, even with a change of



(a) Junction of two lines (b) Gap in gradient

Fig. 3. Gradient map of challenging locations for the line detection (Color figure online)

the background or along heavily textured surfaces, which have a strong influence on the gradient magnitude and the gradient direction along the edge.

Our Optimized Detection Approach. To solve these problems, we developed a method that combines the detection of edge pixels and lines to one step (Fig. 4). This makes it possible that only the relevant edges, i.e. lines, are detected. At the same time the approach prevents the premature chipping of the pixel chains and thus the fragmentation of the lines.

In the first step, the gradient magnitude and the gradient direction per pixel are calculated and the anchor points are determined as in the Edge Drawing method. Starting from the anchor points, chains of edge pixels are determined. Once a minimum length (e.g. 15 pixels) is achieved, the fitting of an initial line is attempted.

If an initial line is found, the next step is the extension of the line (Listing 1.2). For this purpose, the linear equation of the initial line segment is determined. The linear equation is used to predict the search directions. The extension of the line is performed simultaneously at both ends of the line segment. The pixel with the strongest gradient is selected from the neighboring pixels in the search direction. Subsequently, the linear equation is updated. If all neighboring pixels are below the threshold for an edge or if an already detected edge is reached, up to 3 pixels can be skipped in line direction, to bridge gaps or intersections in the gradient map. Thus, the line is detected continuously. The advantage of this approach is that the line is already used as a model for the detection.

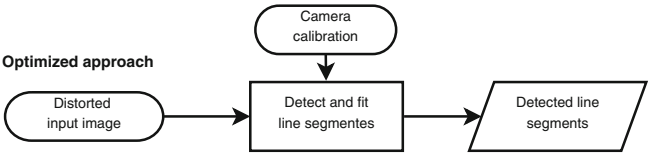


Fig. 4. Optimized line detection approach

Listing 1.2. Pseudocode for the optimized line detection

```
ExtendInitialLineSegments:
  fit line to line pixels
  if line has a slope between +1 and -1
    while extension of the line possible
      calculate next y-value at end of line segment, based on line equation
      if gradient magnitude at [x, floor(y)] greater than value at [x, ceil(y)]
        set y to floor(y)
      else
        set y to ceil(y)
      calculate distance between [x, y] and line
      if gradient magnitude at [x, y] equal 0 or distance greater than threshold
        increment number of outliers
        if number of outliers greater than outlier threshold
          stop extending the end of the line
        fit line to line pixels
      else
        add [x, y] to line pixels
        set number of outliers to 0
      [...] // the beginning of the line segment is extended analogously
  else
  [...] // similar: use equation x = my + c
```

3 Evaluation

3.1 Evaluation of Detection Accuracy on Fisheye Images

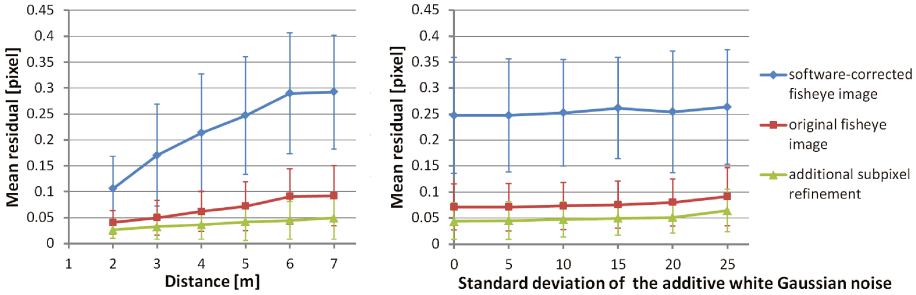
To examine the detection and localization accuracy, synthetic test data were generated that contain simple geometric shapes, e.g. triangles and rectangles (Fig. 5). The configuration of the scene corresponds to images of 3 m large objects from different positions from a distance of 5 m to 7 m. The camera has a fisheye lens with an aperture angle of 120°. Apart from a slight smoothing (5×5 Gaussian filter, $\sigma = 1$) the data contains no errors except from the discretization. For each distance 50 images were generated.

The edges of the objects were detected by the proposed method. To evaluate the detection accuracy, the points of intersection of the detected lines are calculated, and the residuals to the real corner points are determined.

The evaluation is performed on the generated fisheye images without and with the additional subpixel localization. Furthermore, the conventional approach is used for comparison. Software-corrected images are calculated for that

(a) 2 m (b) 7 m

Fig. 5. Two examples of the generated images from different distances



(a) Influence of the distance (b) Effects of Gaussian noise (at 5 m)

Fig. 6. Detection accuracy (in pixel) of the methods in comparison

and used as input for the normal EDLines algorithm. The software-corrected images are calculated by warping the generated fisheye image with a reverse distortion. Here, the same distortion model is used as for the generation of the fisheye images. This way, no calibration error occurs in the evaluation.

The results (Fig. 6(a)) show that the mean residual is significantly larger on the software-corrected image data. The causes of the rather large deviation in the software-corrected images are partly due to the fact that the focal length is adjusted during the warping to get a large part of the fisheye image without the image size increasing excessively. Thus, the resolution is reduced in the center of the image.

The detection accuracy is less precise at increasing distances, i.e for shorter lines, especially for the software-corrected images.

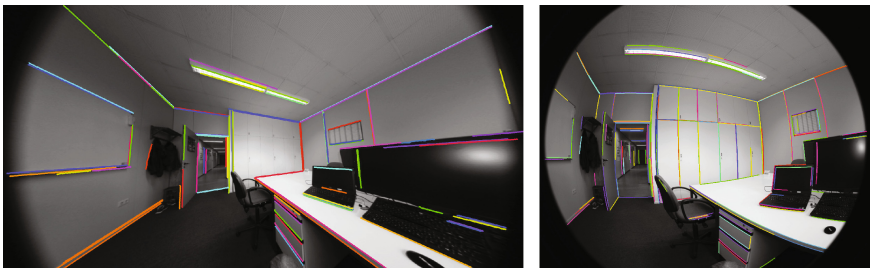
The localization accuracy can be further improved by the additional subpixel refinement. In particular, at large distances, i.e. for shorter lines, the improvement is bigger. The precise localization of each pixel is important in this case to achieve a high accuracy because the line is not fitted over many pixels.

A precise detection is important for a reconstruction of the lines with SfM, because the detection error greatly affects the accuracy of triangulated lines, since there is a quadratic relationship.

To analyze the effects of noise on the line detection, another synthetic data set was generated. The distance is fixed at 5 m, but we add different levels of additive white Gaussian noise. The standard deviation of the noise is varied from 0 to 25. We use 8-bit images, i.e. the pixel values are between 0 and 255.

The evaluation (Fig. 6(b)) shows that the methods are quite robust against noise. The detection accuracy hardly deteriorates even with a high level of noise. This is partly due to the fact that one of the first steps of the algorithms is the Gaussian smoothing of the image.

Number and Quality of the Detected Lines. Figure 7 shows the difference between the line detection on a software-corrected fisheye image and the direct detection on the original fisheye image with our method. On the cabinet and on the rear wall of the room significantly more lines are detected. In total 81 lines are found on the software-corrected image. With our approach 165 lines can be found.



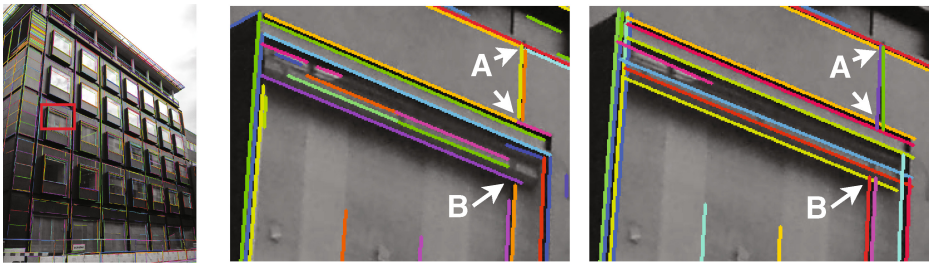
(a) Software-corrected image (b) Our approach

Fig. 7. Qualitative results of the line detection on a real fisheye image

3.2 Qualitative Evaluation of the Optimized Line Detection

For the evaluation of the optimized line detection, we proposed in Sect. 2.3, we use the public data set from [9]. It contains 9 sets of perspective images of building facades. We use one image from each sequence. Figure 8(a) shows the detected lines on one image from the data set.

For comparison, we use the EDLines algorithm again. With both methods, lines are detected in similar areas. A detailed view of a part of the image shows the differences. At intersections lines are often divided in two line segments by the EDLines algorithm (mark A in Fig. 8(b)). With our approach, these lines are detected as one continuous line segment (Fig. 8(c)). The reason for the splitting of the lines is that the edge detector follows the locally strongest gradient magnitude and not the direction of the line. Our approach prevents this by predicting the search direction. Another difference is recognizable in areas of low contrast (mark B). In these areas lines often break because of gaps in the gradient. With our approach, gaps can be bridged. So, the lines are more reliably and robustly detected over the entire length.



(a) (b) (c)

Fig. 8. Qualitative results of the line detection on one image from the dataset from [9]. (a) Entire Image. (b) Detail view (marked in (a)) of the detected lines with the EDLines algorithm. (c) Detail view of the detected lines with our approach

Table 1. Evaluation of the number and the average length of the detected lines on the data set of [9] and the ratio of the average lengths between our approach and EDLines

	EDLines		Our approach		Ratio of avg. lengths
Building	Number	Avg. length [pixel]	Number	Avg. length [pixel]	
1	2582	93.65	1761	168.41	1.80
2	2011	79.66	1133	170.59	2.14
3	2365	86.24	1209	186.68	2.16
4	2214	83.39	1380	158.39	1.90
5	3335	63.08	1238	180.62	2.86
6	7037	47.29	1603	174.08	3.68
7	1772	81.48	991	170.27	2.09
8	1044	93.00	537	208.56	2.24
9	2461	95.10	1247	201.42	2.12
Mean	2758	80.32	1233	179.89	2.24

For a statistical analysis, we have determined the number and the average length of the detected lines for all buildings of the data set. The results are shown in Table 1. With our approach, the number of detected lines is lower, but the average lengths are almost twice as long. As seen already in the images, the lines are detected reliably and fragmented significantly less into several sub-segments.

4 Conclusion

In this paper we have presented a method to detect lines in fisheye and distorted perspective images. The detection accuracy can be significantly improved by detecting lines in the original images without warping the image with a reverse distortion. The combination of the edge and line detection to a single step and the prediction of the search direction, provides a more robust and more reliable detection of larger line segments. This is an advantage in the reconstruction because short line segments often lead to inaccurate triangulation results and the matching of many small line segments is time-consuming and error-prone.

References

1. Akinlar, C., Topal, C.: EDLines: a real-time line segment detector with a false detection control. Pattern Recogn. Lett. **32**(13), 1633–1642 (2011)
2. Ballard, D.H.: Generalizing the Hough transform to detect arbitrary shapes. Pattern Recogn. **13**(2), 111–122 (1981)
3. Bay, H., Ferrari, V., Van Gool, L.: Wide-baseline stereo matching with line segments. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2005, vol. 1, pp. 329–336. IEEE (2005)

4. Bazin, J.C., Demonceaux, C., Vasseur, P.: Fast central catadioptric line extraction. In: Martí, J., Benedí, J.M., Mendonça, A.M., Serrat, J. (eds.) IbPRIA 2007. LNCS, vol. 4478, pp. 25–32. Springer, Heidelberg (2007). doi:10.1007/978-3-540-72849-8_4
5. Boutteau, R., Savatier, X., Bonardi, F., Ertaud, J.Y.: Road-line detection and 3d reconstruction using fisheye cameras. In: 2013 16th International IEEE Conference on Intelligent Transportation Systems-(ITSC), pp. 1083–1088. IEEE (2013)
6. Bukhari, F., Dailey, M.N.: Automatic radial distortion estimation from a single image. J. Math. Imaging Vis. **45**(1), 31–45 (2012)
7. Burns, J.B., Hanson, A.R., Riseman, E.M.: Extracting straight lines. IEEE Trans. Pattern Anal. Mach. Intell. PAMI **8**(4), 425–455 (1986)
8. Canny, J.: A computational approach to edge detection. IEEE Trans. Pattern Anal. Mach. Intell. PAMI **8**(6), 679–698 (1986)
9. Ceylan, D., Mitra, N.J., Zheng, Y., Pauly, M.: Coupled structure-from-motion and 3D symmetry detection for urban facades. ACM Trans. Graph. (TOG) **33**(1), 2 (2014)
10. David, P., DeMenthon, D.: Object recognition in high clutter images using line features. In: Tenth IEEE International Conference on Computer Vision, ICCV 2005, vol. 2, pp. 1581–1588 (2005)
11. Desolneux, A., Moisan, L., Morel, J.M.: From Gestalt Theory to Image Analysis, Interdisciplinary Applied Mathematics, vol. 34. Springer, New York (2008)
12. Fernandes, L.A.F., Oliveira, M.M.: Real-time line detection through an improved hough transform voting scheme. Pattern Recogn. **41**(1), 299–314 (2008)
13. Fischler, M.A., Bolles, R.C.: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Commun. ACM **24**(6), 381–395 (1981)
14. von Gioi, R.G., Jakubowicz, J., Morel, J.M., Randall, G.: LSD: a line segment detector. Image Process Line **2**, 35–55 (2012)
15. Kannala, J., Brandt, S.S.: A generic camera model and calibration method for conventional, wide-angle, and fish-eye lenses. IEEE Trans. Pattern Anal. Mach. Intell. **28**(8), 1335–1340 (2006)
16. Taylor, C.J., Kriegman, D.J.: Structure and motion from line segments in multiple images. IEEE Trans. Pattern Anal. Mach. Intell. **17**(11), 1021–1032 (1995)
17. Wang, L.L., Tsai, W.H.: Camera calibration by vanishing lines for 3-D computer vision. IEEE Trans. Pattern Anal. Mach. Intell. **13**(4), 370–376 (1991)
18. Zhang, L., Koch, R.: Structure and motion from line correspondences: representation, projection, initialization and sparse bundle adjustment. J. Vis. Commun. Image Represent. **25**(5), 904–915 (2014)
19. Zhang, L., Xu, C., Lee, K.-M., Koch, R.: Robust and efficient pose estimation from line correspondences. In: Lee, K.M., Matsushita, Y., Rehg, J.M., Hu, Z. (eds.) ACCV 2012, Part III. LNCS, vol. 7726, pp. 217–230. Springer, Heidelberg (2013)
20. Zhang, M., Yao, J., Xia, M., Li, K., Zhang, Y., Liu, Y.: Line-based multi-label energy optimization for fisheye image rectification and calibration. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 4137–4145, June 2015