Vanishing Point Detection based on Line Set Optimization

Xiaoyun An¹, Tongzhou Zhao², Shanju Jin³, Chengwan Yang⁴

College of Computer Science and Engineering, Wuhan Institute of Technology, Wuhan, Hubei, 430205, China
*21807010001@stu.wit.edu.cn.

Abstract. Vanishing point detection plays an important role in camera calibration and 3D scene reconstruction. There are usually a lot of parallel lines in the real scene. Vanishing point is the intersection point of these spatial parallel lines projected onto the image. Commonly used Hough algorithm to detect vanishing points, which has high complexity and low efficiency. This paper proposes a vanishing point detection algorithm based on optimization of line set. Firstly, the LSD algorithm is used to detect the line. Secondly, the extracted line set is optimized to remove the invalid interference line in the image, which improves the accuracy of vanishing point detection. Thirdly, K-means algorithm is used to cluster and group the optimized line set, which improves the overall efficiency of the algorithm. Finally, random sampling fitting algorithm is used to fit the grouped line set to calculate the precise vanishing point. Compared with Hough algorithm, the running speed of this algorithm is improved by 19% in the actual scene. The experimental results show that the algorithm has low complexity and short running time.

1. Introduction
Under perspective projection, parallel straight lines in the three-dimensional space are mapped onto the image plane and intersect at a point, which is called the vanishing point[1]. The vanishing point can simplify the description and understanding of the scene to a large extent. It has a wide range of applications in robot navigation, 3D reconstruction, camera calibration, etc. It is an important research topic in the field of computer vision[2]. The accuracy of the vanishing point extraction is directly related to the correct understanding of the scene geometry. Image noise, straight line extraction errors and other factors will cause the deviation of the vanishing point. Therefore, how to effectively deal with these influencing factors and accurately extract vanishing point features is the focus of research.

In this paper, line information is used to estimate vanishing points on image plane. It includes three parts, line set extraction and optimization, line set grouping and vanishing point extraction. Firstly, the LSD algorithm is used to extract the line, and then the line is optimized to increase the parallel line feature to make it more suitable for vanishing point detection. Secondly, assuming that each line set corresponds to a vanishing point direction, k-means method is used to cluster the optimized line set. Finally, RANSAC algorithm is used to fit the clustered line set to get the vanishing point

2. Related Work
The commonly used vanishing point detection algorithms can be divided into three categories. The first type is a detection algorithm based on space transformation, which transforms the information on the image into a limited space[3]. For example, the Gaussian sphere transform maps the image plane to the unit sphere centre on the camera origin. The intersection of the straight lines in the image is mapped to the Gaussian sphere to form an intersection along the spherical curve. After transformation, the finite
vanishing point and the infinite vanishing point are transformed into equivalent. However, after the transformation, this method reduces the space position information of the line segment and vanishing point, and loses the distance between the line segment and the vanishing point. At the same time, the accuracy of the transformation method is affected by the accuracy of the accumulated area[4].

The second method uses the information of straight line to detect vanishing point on image plane, avoiding the loss of distance information caused by exchange[5]. The general idea is to calculate all possible straight line intersections, and then use the least square method to solve it. The efficiency of this algorithm is relatively low. Generally, constraints are added according to the characteristics of the scene, such as coplanar and equal spacing between parallel lines, and the algorithm complexity is relatively high.

The third category uses statistical estimation methods to estimate the parameters of the straight line based on the edge feature points on the image[6]. Then, the vanishing point is estimated according to the obtained parameters, or the vanishing point and edge feature points are used to construct the cost function, and the straight line and the vanishing point are estimated at the same time. These statistical algorithms have certain advantages in theory, but in practical applications, the algorithm has high complexity, low computational efficiency, and little application value.

3. Core idea of algorithm

3.1. Line extraction algorithm

Straight line features are important clues to perceive information and are often used in related applications of computer vision. The current popular line detection algorithm is mainly the Hough transform, which is not affected by image rotation and is easy to perform rapid transformation of geometric images. However, its time complexity and space complexity are very high, and the direction of the straight line can only be determined in the detection process, and the length information of the line segment is lost.

This article mainly uses LSD algorithm to extract straight lines in the image. LSD is a partial extraction algorithm, which runs faster than Hough transform. LSD can quickly detect the straight line segments in the image, and then design a fast algorithm according to the geometric characteristics of the target to quickly determine the suspected target area[7]. LSD is a straight line detection and segmentation algorithm, which can obtain sub-pixel precision detection results in a linear time. The algorithm is designed so that no parameter adjustment is required on any digital image. It can control the number of false detections by itself.

3.2. Line set optimization algorithm

Correcting the slope of a straight line, it is for correcting the slope of a straight line at the two edges of double-edged objects, such as door frame edges, windows, poles and other strip-shaped objects. After being processed by the straight line optimization algorithm, straight line features will be obtained at the edges on both sides. Due to the influence of light noise and other factors, the direction of some straight line segments may be inconsistent, and the slope of these straight line segments needs to be corrected. The slope correction process of the straight line segment is shown in Figure 1.

![Figure 1. Straight line slope correction.](image)

If the two straight lines have overlapping parts and the distance between the straight lines is less than a given threshold, it can be determined that the two straight lines have an inclusive relationship, so that the slope can be corrected.
Continuing the broken straight lines, Due to the influence of noise or lighting changes, the continuous straight lines in the real world may form multiple broken short straight lines after being extracted by the algorithm. Moreover, these short straight lines may deviate from the original straight trajectory, and these short straight lines need to be recalibrated and connected to make them more in line with the actual straight line. The straight-line connection process is shown in Figure 2.

As shown in Figure 2, line L4 is on the extension line of line L3, and the distance between them is less than the given threshold value. Therefore, it can be determined that these two lines are broken line segments, and can be connected and merged into a straight line.

Remove too short straight line segments, If there are short isolated straight line segments less than a given threshold length in the straight line collection, these straight lines can be considered to be interference straight line segments formed by light or noise, and they can be directly removed from the straight line collection.

It is necessary to judge whether the corresponding processing of correction and connection can be performed according to the distance between the two straight lines. For straight lines AB and CD, the distance measurement is defined as shown,

\[
\text{Dist}(AB, CD) = \begin{cases} 
\max(LAA', LBB'), & \text{if } L(MN) \leq (L(AB) + L(CD)) \\
\infty, & \text{if } L(MN) > (L(AB) + L(CD)) 
\end{cases}
\]  

(1)

where, \(AA' \perp CD\), the vertical point is \(A'\), \(BB' \perp CD\), the vertical point is \(B'\), \(L(AA')\) is the length of the line \(AA'\), \(M\) is the midpoint of the line \(AB\), \(N\) is the midpoint of the line \(CD\), \(Dist(AB, CD)\) is the distance from the straight line \(AB\) to \(CD\). In the same way, the distance from the straight line \(CD\) to \(AB\) is obtained, as shown,

\[
\text{Dist}(CD, AB) = \begin{cases} 
\max(LCC', LDD'), & \text{if } L(MN) \leq (L(AB) + L(CD)) \\
\infty, & \text{if } L(MN) > (L(AB) + L(CD)) 
\end{cases}
\]  

(2)

The distance between the straight lines \(AB\) to \(CD\) as shown,

\[
\text{DistOfEach}(AB, CD) = \min(\text{Dist}(AB, CD), \text{Dist}(CD, AB))
\]  

(3)

If the distance between two straight lines is less than a given threshold, the slope of the straight line is corrected. The correction equation as shown,

\[
P = \text{Dist}(AB, CD) / (\text{Dist}(AB, CD) + \text{Dist}(AB, CD))
\]  

(4)

\[
K_{\text{new}} = K_{AB} * P + K_{CD} * (1 - P)
\]  

(5)

where, \(P\) is the proportion of the slope of the straight line \(AB\) in the straight line correction, \(K_{\text{new}}\) is the new straight line slope after correction.

For the straight lines \(AB\) and \(CD\), the slopes are corrected to \(K_{\text{new}}\) respectively, and the two corrected straight lines can be obtained after passing through their original midpoints. After the straight line is corrected, if the two straight lines are disconnected lines, the straight lines will be connected to generate a new straight line \(L_{\text{continue}}\), define the straight line connection as shown,
After optimizing the initial set of straight lines, the slope of the parallel straight lines is corrected, the disconnected straight line segments are connected, and straight lines whose length is too short can be effectively removed for the vanishing point detection, thereby reducing the interference of line segments. The effect of vanishing point detection result accuracy. It can be clearly observed that the optimized line set can conform to the real line features in the picture to a greater extent.

3.3. Clustering all straight lines

There are many kinds of clustering algorithms, and the K-means algorithm is a commonly used one, which is a classic partition-based clustering method[8]. The biggest feature of K-means algorithm is simple, easy to understand, and fast. In this paper, one-dimensional K-means algorithm is used to cluster and group the optimized set of straight lines, thereby reducing the computational complexity.

The key of the K-means algorithm is the calculation of cluster centers and distances[9]. In this paper, the angle between the straight line segment and the positive direction of the $x$-axis is used to determine the type of straight line. Since this paper is one-dimensional clustering, the choice of initial centroid has little effect on the clustering results. Therefore, this paper uses a randomization algorithm to select $K$ different angles as the initial centroid, and defines the equation for the distance between the centroid and the straight line, as shown,

$$D(\text{Center, } l_i) = \min[|\text{Center} - \text{angle}(l_i)|, 180 - |\text{Center} - \text{angle}(l_i)|]$$

where, $D(\text{Center, } l_i)$ is the distance between the center of mass and the line, $\text{Center}$ represents the center of mass, and $l_i$ represents a straight line in the collection of lines, $\min[a, b]$ represents the minimum value of $a, b$. $\text{angle}(l_i)$ represents the angle between the straight line and the positive direction of the $x$-axis, and the value range is $[0, 180]$.

The new definition of centroid is calculated as shown,

$$\text{New}_\text{center} = \sum_{l_i \in LS}(X/n)$$

$$X = \begin{cases} \text{angle}(l_i), & |\text{angle}(l_i) - \text{Center}| \leq 90 \\ 180 - \text{angle}(l_i), & |\text{angle}(l_i) - \text{Center}| > 90 \text{ and } \text{angle}(l_i) > 90 \\ 180 - \text{angle}(l_i), & |\text{angle}(l_i) - \text{Center}| > 90 \text{ and } \text{angle}(l_i) \leq 90 \end{cases}$$

where, $\text{New}_\text{center}$ represents the new centroid, $\text{Center}$ represents the initial centroid before the update. $LS$ is the line included in the cluster before the update, $n$ is the number of lines included in the cluster, $\text{angle}(l_i)$ is the line and the positive $x$-axis direction, and the value range is $[0, 180]$. It can be seen from equation 10 that the distance between the center of mass and the straight line is defined as the angle difference between the two, and there is no absolute relationship with the relative position between the straight lines.

3.4. Estimated vanishing point

When the sample is large and there are outliers, if the vanishing point is determined by the intersection of two lines in the set, the calculation amount of the algorithm is too large, and the time complexity is $O(n^2)$, where $n$ is the number of lines in the set. RANSAC algorithm can provide a good solution, which can effectively reduce the impact of outliers on vanishing point detection. The RANSAC algorithm is different from the traditional smoothing process[10]. The traditional method is to use as much data as
possible to obtain a relatively primitive solution, and then try to use some optimization algorithms to eliminate invalid data points. RANSAC uses a relatively small data set, and then uses consistent data as much as possible to expand the original initialization data set. The input of the RANSAC algorithm is a set of observation data, a parameterized model used to interpret the observation data and some credible parameters. The purpose of fitting is achieved by repeatedly selecting a set of random subsets of the data.

After K-means clustering, the optimized line set $L$ can be divided into $k$ sub-line sets. When the number of lines in the line set is relatively large, the corresponding vanishing point is calculated for each subset, and one iteration will cost a lot time. In addition, due to the existence of outliers, the intersection point obtained from the intersection of two straight lines may change in a larger area, which will increase the workload of vanishing point selection. Therefore, it is not feasible to find the vanishing point by intersecting straight lines. In this paper, RANSAC algorithm is used to polynomial fit the clustering line set, which not only improves the running speed, but also reduces the influence of outliers on vanishing point detection.

4. Experimental results and analysis

In order to reflect the characteristics of the algorithm more realistically, this paper performs vanishing point detection on actual indoor scenes and outdoor scenes. The overall experimental results of vanishing point detection in indoor scenes are shown in Figure 3. The display sequence of the experimental results are, original image, Hough algorithm line extraction, LSD algorithm line extraction, line set optimization, line clustering, and vanishing point detection.

Figure 3. Algorithm running results in indoor scenarios.
Observing Figure 3, it can be seen that compared with the Hough algorithm, the LSD algorithm can extract better straight line features, and the straight line is more in line with the actual scene, thus providing a powerful guarantee for accurate detection of the vanishing point.

This paper also gives the running time of Hough algorithm and LSD algorithm, the comparison result is shown in Table 1.

| Algorithm | Running Time (s) | Improved Speed (%) |
|-----------|------------------|--------------------|
| Hough     | 0.12992          | 19.21              |
| LSD       | 0.02496          |                    |

According to the results of Table 1, in the actual indoor scene. The running time of LSD algorithm is better than Hough algorithm, and the computing speed is increased by 19.21%.

The overall experimental results of vanishing point detection in outdoor scenes are shown in Figure 4. The display order of the experimental results are, original image, Hough algorithm line extraction, LSD algorithm line extraction, line set optimization results, line clustering results, and vanishing point detection results.

It can be seen from Figure 4 that the algorithm in this paper is also suitable for vanishing point detection in actual outdoor scenes. The effect is the same as the indoor scene experiment, and the vanishing point can also be detected very accurately.
In terms of operating speed, the LSD algorithm is still better than the Hough algorithm. The running time comparison is shown in Table 2.

| Algorithm | Running Time (s) | Improved Speed (%) |
|-----------|-----------------|--------------------|
| Hough     | 0.15696         | 19.73              |
| LSD       | 0.03098         |                    |

From the calculation results in Table 2, it can be seen that in outdoor scenes, the running time of the LSD algorithm is also better than that of the Hough algorithm. The calculation speed of this experiment increased by 19.73%.

5. Conclusion

This paper analyzes the previous vanishing point detection algorithms and existing problems, and proposes a vanishing point detection algorithm based on straight line optimization. Firstly, the LSD algorithm is used to detect the lines. Secondly, the extracted lines are optimized. Thirdly, according to the line characteristics of vanishing points, K-means algorithm is used to cluster and group the optimized line set, which improves the overall efficiency of the algorithm. Finally, random sampling fitting algorithm is used to fit the grouped line sets. This method can effectively eliminate the invalid interference lines in the image, reduce the amount of calculation in the subsequent steps, and improve the accuracy of vanishing point detection, so as to calculate the accurate vanishing point. Experimental results show that the algorithm can detect vanishing points quickly and efficiently.

Acknowledgements

This work was supported by the Wuhan Institute of Technology Software engineering laboratory and the eleventh Graduate Innovation Fund of Wuhan Institute of Technology (CX2019250, CX2019254).

References

[1] Chu J, Xiao X, Liang C, Un-Calibrated Single Visual Image of Orthogonal Vanishing Point Detection Algorithm [J]. Journal of Graphics, 2016, 37(06):783-790.
[2] Luo X, S, Fang B, Yang W. B. Road Vanishing Point Detection Using Weber Local Descriptor [J]. Journal of Computer Applications, 2014, 34(S1):219-222.
[3] Mei X, Xia L. Z, et, al. A Vanishing Point Detection Algorithm for 3D Scene [J]. Journal of Signal Processing, 2007(06):924-926.
[4] Hu P. C. Research on Camera Calibration of 3D Information Extration[D]. Nanjing University of Aeronautics and Astronautics, 2007.
[5] Li H. A Study Line And Vanishing Point Detection Algorithm On Indoor Scenes Photo[D]. Xidian University, 2014.
[6] Grompone von Gioi Rafael, Jakubowicz, Morel Jean-Michel, et al. LSD, A Fast Line Segment Detector with A False Detection Control[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence. 2010, 32(4):722-732.
[7] Liu Y, Xie Z. W, Liu H. LB-LSD, A Length-Based Line Segment Detector For Real-Time Applications[J]. Pattern Recognition Letters. 2019, 128:247-254.
[8] Cuneyt Akinlar, Cihan Topal. ED Lines, A Real-Time Line Segment Detector with A False Detection Control[J]. Pattern Recognition Letters. 2011, 32(13):1633-1642.
[9] Wilhelm T, Wohler C. Boundary Aware Image Segmentation with Unsupervised Mixture Models[C]. Proc of IEEE International Conference on Image Processing. Piscataway, NJ, IEEE Press, 2017:3325-3329.
[10] Xue L. J. Research on 3D Point Cloud Based Object Size and Orientation for Mobile Robot [D].Shanghai Jiao Tong University, 2018.