Polygon graphic recognition based on improved fast corner detection

Jinfeng Gu¹, Weizhou Liu²

¹Guangxi Normal University, Guangxi, China
²Nanjing Xiaozhuang University, Nanjing, China

Corresponding author and e-mail: Weizhou Liu, 435767883@qq.com

Abstract. The accuracy of polygon graphic recognition based on chain code features and Hough transform is low, and the computation is limited. Therefore, this paper proposes a polygon graphic recognition method based on improved features from accelerated segment test (FAST) corner detection. First, hole filling and Freeman chain code are used to segment the image, and the regular geometric features are obtained. Second, in order to improve the performance of the algorithm, an improved FAST corner detection combined with DBSCAN (Density-Based Spatial Clustering of Applications with Noise) is proposed. Through clustering, false corner in the image can be eliminated, and the feature points are quickly filtered by local NMS (non-maximum suppression). Finally, polygon graphic recognition is realized by feature points, experimental results illustrate that this method has high calculation efficiency and recognition rate.

1. Introduction

As an important feature to describe target features, corners are widely used in image processing and computer vision tasks, such as feature extraction, target matching and target tracking. However, there is no clear mathematical definition of corner. It is generally considered that corner points are either points with sharp change in gray level or extreme points with maximum curvature on the edge curve of the image [1-2]. Generally, there are three main methods for corner detection: corner detection based on gray-scale image, corner detection based on binary image and corner detection based on contour curve.

The corner detection method based on the gray-scale image need to convert the image into gray-scale mode, and determines the corner point by calculating the change of gray-scale intensity in the field [3-4]. The Moravec algorithm is the earliest corner detection algorithm. Moravec found that there are obvious differences in homogeneous regions, edges and corners. The gray difference between the current pixel window and the eight directions around it is calculated, and the minimum value is taken as the response value, then use the local maximum suppression algorithm to eliminate the false corners. However, the algorithm is sensitive to edges and does not have rotation invariance characteristics. Harris and Stephens proposed the well-known corner detection algorithm Harris, which modified Moravec’s interest operator, used the first derivative to approximate the second derivative, and calculated the Harris matrix M (auto-correlation matrix) by Eigenvalues $\lambda_1, \lambda_2$. Using these two eigenvalues to construct a response function R to determine the corner point. Drummond et al. proposed a fast corner detection algorithm and used the number of decisions to speed up the corner
detection process [5]. In order to further improve the performance of the FAST corner detector, Biadgie and Sohn improve the computational efficiency and repeatability of the algorithm through multi-scale space and multi-mask.

The binary corner detection method transforms the image into a binary image. Liu and Zhu et al. proposed a corner detection algorithm for binary image based on morphological skeleton. In this algorithm, the binary image is regarded as a polygon, and corners are detected by calculating the points in the skeleton whose maximum circle radius is zero. The algorithm has low computational complexity and can be well applied to hardware.

The contour-based corner detection method detects corners by calculating the maximum curvature on the image edge curve. Mokhtarian et al. proposed a corner detection method based on curvature scale space (CSS). The edge was extracted from the image by the Canny detector, and then the threshold was set to separate the maximum curvature from the contour [6-7]. Awrangjeb and Lu proposed a corner detection algorithm that accumulates the chord-to-point distance, which reduces false corners while performing corner detection in the image.

In this paper, FAST corner detection algorithm is adopted. Compared with other corner detection algorithms, this algorithm not only has a fast calculation speed, but also has a relatively high accuracy, which has become very popular in recent years.

2. Improved FAST corner detection

FAST corner point algorithm, by extracting pixel point p and 16 pixels around point P, assume that the gray value of pixel point P is LP, if the gray value of n consecutive points in 16 points is higher than LP or lower than LP, then set this point as the corner point. Due to the FAST algorithm 1) The selected corner points are not the corner points of the geometric figure, and may be noise points. 2) Too much candidate points are obtained, which has a great impact on the final result of polygon graphic recognition.

Based on the task of polygon image recognition, this paper proposes an effective method to improve fast corner detection algorithm [8-9]. Firstly, in order to eliminate the interference of complex background and geometry, Freeman chain code is used to extract the connected area, and then the hole filling algorithm is used to eliminate the influence of irregular lines in the geometry. Then, in order to improve the performance of the algorithm, according to the characteristics of geometry, an improved fast corner detection method combined with DBSCAN algorithm is proposed [10]. The algorithm can quickly eliminate the false corners. Finally, the category of geometry is judged by the number of corners.

2.1. Extract geometry

Traditional Freeman chain code is eight consecutive pixels with different directions adjacent to the current pixel, as shown in Figure 1, where point P is the current pixel, and eight directions are formed by the current point pointing to adjacent pixels 0-7. Therefore, a line can be digitized to form n chains, and each chain points to different directions. Freeman direction chain code algorithm is used to track the image edge, and the direction chain code of multiple contours will be obtained and stored in an array. Since the geometric figure has a certain perimeter and area, the contour approximation algorithm is used to remove the complex background by setting threshold. As shown in Figure 2(b), the hooks, forks and curves in Figure 2(a) are eliminated.

![Figure 1](image1.png)

**Figure 1.** Eight Freeman directional chain codes of point P.

Next, put each outline extracted into a blank picture. The hole filling algorithm is used to eliminate the holes in the outline of the geometric figure. Binarize the picture first, and then using the area
growth algorithm to fill the background. As shown in Figure 2(c), through hole filling, the geometry is cut out from the image for candidate corner extraction in Chapter 2.2.

![Figure 2](image) (a) Triangle image (b) Triangle contour extraction (c) Hole filling algorithm

**Figure 2.** The process of extracting geometry.

### 2.2. Candidate corner extraction

In this paper, we can quickly extract the position of corners in geometry by improving the traditional fast corner detector. Since there are a lot of acute angles, right angles and obtuse angles in the geometry, the candidate corners are obtained by using the information of 16 pixels around the current pixel \( P \). Common corner conditions are shown in Figure 3. Figure (a) is the correct corner extracted, namely acute corner, right angle corner and obtuse corner. Figure (b) and Figure (c) are possible wrong corner points, which often appear on the oblique edge of geometric figure, especially Figure (c).

![Figure 3](image) (a) Correct corners (b) Thin-band pixel (c) Salient pixel

**Figure 3.** Candidate corners that may be extracted from the image.

By observing the graphs (a) (b) (c) in Figure 3, we find that in Figure (a), the acute, right, and obtuse angles occupied up to two pixels at the 0, 4, 8, 12 positions. The improved FAST algorithm firstly conducts rapid filter the candidate corners by sampling the surrounding pixels 0, 4, 8, 12 of the current pixel \( P \). Next, observing Figure (a)(c), We find that 2 times of the number of pixels in the non-corner region of figure (a) must be greater than 16, and that of figure (c) is less than or equal to 16. The pixels in the non-corner region of figure (b) are difficult to follow regularly, and the false corners generated by them will be discussed in Chapter 2.3.

Since the original image is converted into a binary image, the relationship between point \( P \) and surrounding pixels can be divided into two types, as shown in (1). The gray value of current pixel \( P \) is \( IP \), and the gray value of surrounding pixel \( X \) is \( IX \). if the gray value \( IX \) is less than \( IP \), it means that pixel \( X \) is darker than point \( P \). if the gray value \( IX \) is greater than or equal to \( IP \), it means that pixel \( X \) is similar to \( P \) or is lighter than \( P \).
\[ S_x = \begin{cases} I_x < I_p & \text{darker} \\ I_p \leq I_x & \text{similar or brighter} \end{cases} \]  

(1)

Extract the surrounding pixels of the current pixel \( P \) in the way of circle: extract a group of pixels with radius of 3, and set the number of pixels as \( N \). According to the characteristics of corners of geometry analyzed above, when the number of pixels \( n \) in non-corner area of \( N \) pixels is greater than \( N \), that is, when formula (2) is satisfied, the current pixel is considered as a candidate corner. Where \( \epsilon \) is the error value, which can be used to fine tune the number of surrounding pixels in non-corner area.

\[ 2n + \epsilon > N \]  

(2)

Through the above method, the candidate corners of the geometric figure can be quickly obtained. As shown in Figure 4, the original image is shown in Figure (a), and the candidate corner points obtained by the improved FAST corner detector are shown in Figure (b).

2.3. False corner elimination

The final candidate corners in Figure 4(b) are generated by the improved FAST detector. By observing the results, we find that the candidate corners at the corner positions are relatively concentrated. The corner points on the edge are relatively scattered, and most of these corner points are generated by the thin-band pixels in Figure 3(b). In order to eliminate false corners on the edge, the DBSCAN algorithm is used to further optimize the algorithm.

The DBSCAN algorithm is a density-based clustering algorithm. Unlike partitioning and hierarchical clustering methods, it defines clusters as the maximum set of points connected by density. Without knowing the number of clusters, the area with high density can be divided into clusters, and it can effectively robust to the noise points away from the density core.

As shown in Figure 5, all candidate corners in Figure 4(b) are drawn. The dark points in the image are the cluster formed by the core points, and the light points are the candidate corner on the edge. Next, the DBSCAN algorithm is used to form multiple class clusters, as shown in Figure 6, in which the category -1 represents the edge points (noise points), and 0-5 is the correct candidate corners.

Using the cluster formed by DBSCAN algorithm, the pixels marked as -1 (noise points) are eliminated, and the remaining pixels in the cluster are suppressed to achieve the final corner detection. As shown in Figure 7, the corner position is marked with a blue circle.
3. Experiment and analysis of results

In order to verify the effectiveness of the algorithm, in the paper, the same geometry are used in the experiment and compare the experimental results with the results of FAST algorithm and Harris algorithm. The experimental equipment parameters are: processor Intel Core i7-7500u 2.7GHz, memory 8G). In the experiment, the software platform are: Python 3.7, OpenCV 3.4.7. This algorithm is implemented using python language programming.

Figure 7. The final corner detection.

Figure 8. Polygon geometry.

Figure 8 shows the original image data. Figure 9 shows the comparison of corner extraction of two geometric figures under different corner detector. The experimental process is to extract the candidate corners of two images. For FAST algorithm (Figure 9(a)) and Harris algorithm (Figure 9(b)), NMS algorithm is used to suppress false corners and extract key points. Our algorithm (Figure 9(c)) is to extract key points by using DBSCAN algorithm. The key points are marked by circles.

Figure 9. The comparison of corner extraction under different corner detector.
As shown in Table 1, compared with Harris corner detector and fast corner detector, our proposed method takes moderate time and has the highest accuracy. It shows the efficiency of the proposed method in extracting feature points.

|       | Running Time(s) | Detected corners | Correct corners | Recognition accuracy | Corner accuracy |
|-------|----------------|------------------|----------------|----------------------|----------------|
| FAST  | 0.08           | 28               | 26             | 0.33                 | 0.92           |
| Harris| 4.32           | 44               | 40             | 0.75                 | 0.91           |
| Our algorithm | 0.86 | 40               | 40             | 1                    | 1              |

4. Conclusions
In this paper, an improved fast corner detection algorithm is used to quickly extract candidate corner points, and DBSCAN algorithm is used to eliminate false corners. To a certain extent, the efficiency of the algorithm is improved and the calculation accuracy is ensured. The algorithm makes full use of the gray information and corner position of geometric figures in the image. The experimental results show that the algorithm can realize the rapid recognition of geometric figures, and its recognition accuracy and speed are better than the traditional corner algorithm, and the robustness of the algorithm is improved.

Acknowledgement
Thanks to Professor Weizhou Liu for his help in the experiment

References
[1] Xing Y, Zhang D, Zhao J, et al. Robust fast corner detector based on filled circle and outer ring mask [J]. IET Image Processing, 2016, 10(4): 314-324.
[2] Biadgie Y, Sohn K A. Speed-up feature detector using adaptive accelerated segment test [J]. IETE Technical Review, 2016, 33(5): 492-504.
[3] Y. Ren, "A survey of corner detection algorithms", Mechanical Engineering and Automation, vol. 2009, no. 1, pp. 198-200, 2009.
[4] H. Moravec, "Towards automatic visual obstacle avoidance", Proceedings of 5th International Joint Conference on Artificial intelligence, pp. 584-584, 1997.
[5] Harris, Christopher G., and Mike Stephens. "A combined corner and edge detector." In Alvey vision conference, vol. 15, no. 50, pp. 10-5244. 1988.
[6] Rosten, E. and Drummond, T., 2006, May. Machine learning for high-speed corner detection. In European conference on computer vision (pp. 430-443). Springer, Berlin, Heidelberg.
[7] W. Liu and G. Zhu, "Corner detection for binary image using morphology skeleton", Signal Processing, vol. 16, no. 3, pp. 276-280, 2000.
[8] Rosenfeld, Azriel, and Joan S. Weszka. "An improved method of angle detection on digital curves." IEEE Transactions on Computers 100.9 (1975): 940-941.
[9] Mokhtarian, Farzin, and Riku Suomela. "Robust image corner detection through curvature scale space." IEEE Transactions on Pattern Analysis and Machine Intelligence 20.12 (1998): 1376-1381.
[10] Han J H, Poston T. Chord-to-point distance accumulation and planar curvature: a new approach to discrete curvature [J]. Pattern Recognition Letters, 2001, 22(10): 1133-1144.