Improved Harris Corner Detection Algorithm Based on Canny Edge Detection and Gray Difference Preprocessing

Chuan Luo1,2, a, Xiaoliang Sun1,2, b*, Xiangyi Sun1,2, c and Junyao Song1,2

1 College of Aerospace Science and Engineering, National University of Defense Technology, Changsha, Hunan, 410073, China
2 Hunan Provincial Key Laboratory of Image Measurement and Vision Navigation, Changsha, Hunan, 410073, China
a E-mail: luochuan@nudt.edu.cn
b*Corresponding author’s e-mail: alexander_sxl@nudt.edu.cn
c E-mail: sunxiangyi_hn@163.com

Abstract. Harris corner detection algorithm has been widely used in many computer vision allocations. However, it has low efficiency and accuracy, poor noise immunity and needs to set an artificial threshold. In this paper, an improved algorithm based on Canny edge detection and gray difference preprocessing is proposed. Firstly, Canny edge detection and gray difference preprocessing are used for corner prescreening to improve the detection efficiency, anti-noise, and rotation invariance. Secondly, non-maximum suppression is applied to the screened corners to reduce the number of false corners. Finally, the average of adjacent points method is used to solve the problem of corner cluster, and the detection results are compared the measurement accuracy is improved to sub-pixel level. Experimental results indicate that the proposed algorithm can accurately extract the corners in the image and remove the false corners and corner clusters. It achieves superior performance than existing methods.

1. Introduction

Corners are crucial local feature information in an image. Corners are generally considered to be the points where the brightness of the image changes dramatically or the intersection of the image contour boundaries. It has been widely used in computer vision applications, e.g., 3D scene reconstruction, motion estimation, object recognition, image registration [1-4].

The existing corner detection algorithms can be roughly summarized into two categories: one is based on gray image corner detection, the other is based on contour curve corner detection [5]. These algorithms have their advantages and disadvantages in efficiency, accuracy, and robustness. In comparison, Harris corner detection is widely used, and the calculation results are stable. However, there are also many shortcomings, such as the need to manually set the threshold, false corners, corner clusters, inaccurate corner positioning, and low detection efficiency. Zhang, J. et al. [6] used the B-spline function to replace the Gaussian smoothing function in the original Harris algorithm, which improved the problem of difficult selection of window size and improved the positioning accuracy. Zou, Z.Y. et al. [7] proposed an adaptive threshold method to avoid the limited corner distribution caused by manual selection of the threshold. Zhang, J.S. et al. [8] improved the corner cluster phenomenon through image block processing and adaptive threshold setting in each image block. Dong, L.H. et al. [9] proposed an
improved circle algorithm based on Sobel edge detection and Cai, X.Z. et al. [10] proposed the improved algorithm to quickly eliminate many non-feature points through the FAST algorithm, which both significantly improved the operation efficiency.

Aiming to improve the efficiency, anti-noise, and rotation invariance of the original Harris algorithm, this paper modifies the original algorithm using Canny edge detection [11] and gray difference preprocessing [12]. Non-maximum suppression is used to reduce the number of false corners. The corner cluster is removed by taking the average of adjacent corner positions in the neighborhood to improve the corner extraction accuracy to sub-pixel level. Experimental results indicate that the proposed method achieves superior performance than existing methods.

2. Materials and Methods

2.1. Principle of Harris corner detection algorithm

Harris corner detection algorithm is based on a gray image. Its basic principle is to set a square window in the gray level image plane, and let the window traverse the image. Suppose that the window translation is \((u, v)\), and the resulting gray level change is \(E(u, v)\), which is called image gray level autocorrelation function [13]. It is represented by the following function:

\[
E(u, v) = \sum_{x,y} \omega_{x,y}(I_{x+u,y+v} - I_{x,y})^2 = \sum_{x,y} \omega_{x,y} \left[ u \frac{\partial I}{\partial x} + v \frac{\partial I}{\partial y} + o(u^2, v^2) \right]^2
\]

Where \(\omega_{x,y}\)—window function; \(I_{x+u,y+v}\)—gray value after window translation; \(I_{x,y}\)—gray value before window translation. Perform Gaussian smoothing on the image to improve noise immunity.

\[
\omega_{x,y} = \exp \left[ -\frac{1}{\sigma^2}(x^2 + y^2) \right]
\]

After rounding off the small quantity of higher order, it can be obtained from function (1):

\[
E(u, v) = \sum_{x,y} \omega_{x,y} \begin{bmatrix} I_x & I_y \end{bmatrix} \begin{bmatrix} I_x & I_y \end{bmatrix}^T = \sum_{x,y} [u \quad v] \omega_{x,y} \begin{bmatrix} I_x^2 & I_xI_y \ I_xI_y & I_y^2 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix}
\]

Let \( M = \sum_{x,y} \omega_{x,y} \begin{bmatrix} I_x^2 & I_xI_y \ I_xI_y & I_y^2 \end{bmatrix} \). \( M \) is called autocorrelation matrix, then

\[
E(u, v) = [u \quad v] M \begin{bmatrix} u \\ v \end{bmatrix}
\]

The eigenvalues of the matrix \( M \) can represent the extreme curvature of the image grayscale autocorrelation function at a certain point. Suppose the two eigenvalues of the matrix \( M \) are both relatively large. It means that the extreme curvatures in the two orthogonal directions of the image grayscale self-phase function at this point are both large so that this point can be considered as a corner point. Let the trace and determinant of matrix \( M \) be \( T_r(M) \) and \( \text{Det}(M) \). Define the corner response function:

\[
R = \text{Det}(M) - t[T_r(M)]^2
\]

\( t \) is a constant and usually takes \([0.04,0.06]\) according to empirical values. \( R \) is only related to the characteristic value of \( M \). It is a corner point when \( R \) is a large positive number. It is an edge when \( R \) is a large negative number, and when \( R \) is a small value, it represents a flat area. During corner detection, threshold processing is performed on the corner response function \( R \). If \( R \) is greater than the given threshold, the detection window's center point is a corner point.

2.2. Harris algorithm steps and shortcomings

The process can be divided into three steps by understanding and analyzing Harris corner detection, as shown in Figure 1. The Harris corner detection algorithm has many shortcomings, such as low efficiency and accuracy, poor noise immunity and needs to set an artificial threshold.
2.3. Improved Harris corner detection algorithm

To solve the problems, the proposed algorithm uses Canny edge detection and gray difference preprocessing to prescreening corners before calculating the autocorrelation matrix to improve the detection efficiency, noise resistance, and rotation invariance; then, the screened corners are processed non-maximum value suppression, reducing the number of false corners; finally, the neighboring point average method is used to solve the corner cluster problem and improve the detection accuracy to sub-pixel level. The proposed algorithm flow is shown in Figure 2.

2.3.1. Canny edge detection screens candidate corners

The corner point should be on the edge of the image. So, the Canny edge detection algorithm is used to detect the edge of the image, and the detected edge is regarded as a candidate point to reduce the amount of calculation for the next step of gray difference preprocessing. In this paper, the building block diagram is used as the original experimental image (as shown in Figure 3(a)). The original image contains 256*256=65536 pixels, and 3198 edge points are filtered by the Canny edge algorithm (as shown in Figure 3(b)). The screening result is much smaller than the number of pixels in the original image, which is about 4.88% of the original, which greatly reduces the running time.
2.3.2. Preprocessing of gray difference to filter candidate corners
Let the absolute value of the gray difference between the center pixel and any pixel in the neighborhood be $T$, and set the threshold $m$. If $T \leq m$, the center point is similar to the point. Similarly, whether the center point is similar to other points in the neighborhood can be judged. The algorithm uses a $3 \times 3$ rectangular detection window, and the similarity between the center point and its 8 neighboring pixels should be considered. Let the number of similar points in the center point and its neighborhood be $n(i, j)$, then the maximum value of $n(i, j)$ is 8, and the minimum value is 0. Various situations are discussed as follows:

1) When $n(i, j) = 0$, it means that there is no similar point about the center store, so the center point is an isolated point or a noise point and cannot be used as a candidate point.

2) When $n(i, j) = 7$, it is possible that the corner point should be a point in the forward direction of the target pixel or a point in an oblique direction, and the target pixel should not be a candidate point for the corner point.

3) When $n(i, j) = 8$, it means that the current target pixel is all similar pixels in the 8-neighborhood, so the pixel should be a point inside a region, so this type of pixel should also be excluded pixel.

4) When $1 \leq n(i, j) \leq 6$, in this case, the target pixel may be a corner point, so it can be a candidate point.

Based on Canny edge detection, further gray-scale difference processing is performed to obtain 1972 corner points (as shown in Figure 3(c)). The number of runs is 61.66% of the previous step, which is 3.01% of the original, which further reduces the running time.

2.3.3. Adaptive threshold
When using the Harris algorithm to extract corners of a picture, it is necessary to continuously debug by setting a fixed threshold, and better results can be obtained. However, the set threshold is not necessarily applicable to other pictures and needs to be readjusted. Considering the versatility of the algorithm, the algorithm itself is required to have the ability to calculate an appropriate threshold. The proposed algorithm sets the threshold to $q$ times the maximum value of the corner response function, $R \geq q \times R_{max}$. When $q$ takes the empirical value [0.005, 0.015], basically all corner points will be detected, and the number of false corner points detected at the same time is small.

2.3.4. Removing false corners and corner clusters
After the corner prescreening and threshold screening, there will be some corner clusters and pseudo corners. Two or more corners may appear in a specific field, which may lead to inaccuracy for some subsequent processing, such as image matching and pose estimation. Therefore, this paper uses the non-maximum suppression method to remove the false corners. Only one value is allowed in a specific local range; then, the arithmetic mean of the corners in the adjacent range is taken as the corner position, removing the corner cluster, and improving the position accuracy to the sub-pixel level at the same time.

3. Results & Discussion
Perform verification experiments on the algorithm in this paper and the original algorithm. The experimental hardware environment is as follows: CPU intel(R)Core(TM)i7-8565U, memory is 8GB, 64-bit Windows10 operating system; software implementation platform is MATLAB R2018b. Exclude the points on the image frame that we are not interested in. the parameters were selected as $q=0.005$, $q=0.010$, and $q=0.015$. The experimental results are shown in Figure 4.
3.1. Algorithm time comparison
The original algorithm and the proposed algorithm are compared under different $q$ values. The efficiency of the algorithm is shown in Table 1. It can be seen from the experimental results that the average running time is reduced from 2.46s to 1.94s, which is about 78.86% of the traditional algorithm.

| Detection algorithm | $t_1$ | $t_2$ | $t_3$ | $t_4$ | $t_5$ | Average time |
|---------------------|-------|-------|-------|-------|-------|--------------|
| Original            | 2.42  | 2.48  | 2.47  | 2.55  | 2.36  | 2.46         |
| Proposed            | 1.83  | 2.06  | 1.95  | 1.96  | 1.88  | 1.94         |

3.2. Accuracy comparison
The algorithm's accuracy (number of right corner points / (number of correct corner points + false corner number + missed detection number)) is compared in Table 2. The original Harris corner detection algorithm has many false corners, but the proposed algorithm significantly reduces the false corners. The accuracy of the proposed algorithm is also better. Taking $q=0.005$ as an example. The accuracy is improved from 68.49% of the original algorithm to 81.03%.

| $q$ | Detection algorithm | Correct corner points | false corner number | missed detection number | Accuracy  |
|-----|---------------------|-----------------------|---------------------|-------------------------|-----------|
| 0.005 | Original            | 50                    | 16                  | 7                       | 68.49%    |
|      | Proposed            | 47                    | 1                   | 10                      | 81.03%    |
| 0.01  | Original            | 49                    | 7                   | 8                       | 76.56%    |
|      | Proposed            | 45                    | 1                   | 12                      | 77.59%    |
| 0.015 | Original            | 47                    | 5                   | 10                      | 75.81%    |
|      | Proposed            | 44                    | 0                   | 13                      | 77.19%    |
3.3. Noise immunity comparison

Add Gaussian white noise with $\Phi=0$ and $\delta=0.001$ to the original image [14]. Taking $q=0.005$ as an example to compare the traditional algorithm with the algorithm in the text. As shown in Figure 5, this proposed algorithm dramatically improves the influence of corner clusters and pseudo corners and has strong noise resistance compared with the original algorithm.

![Original](image1.png) ![Proposed](image2.png)

(a) Original                                           (b) Proposed

Figure 5. Experimental comparison after adding noise

3.4. Rotation invariance comparison

The original image is rotated as an experimental image, and the "Error Ratio" ((false corner number + missed detection number)/ correct number) is used to compare the proposed algorithm with the original algorithm. The greater the error ratio value make the detection effect worse. The experimental results are shown in Table 3. Taking $q=0.005$ and a rotation angle of $45^\circ$ as an example, the experimental results are shown in Figure 6. The results show that the proposed algorithm's Error Ratio is far lower than the original algorithm.

| Rotation angle | Detection algorithm | Correct corner points | false corner number | missed detection number | the Error Ratio |
|----------------|---------------------|-----------------------|----------------------|-------------------------|----------------|
| 25°            | Original            | 50                    | 21                   | 7                       | 56.00%         |
|                | Proposed            | 50                    | 5                    | 7                       | 24.00%         |
| 45°            | Original            | 50                    | 21                   | 7                       | 56.00%         |
|                | Proposed            | 50                    | 1                    | 7                       | 16.00%         |
| 70°            | Original            | 52                    | 20                   | 5                       | 48.08%         |
|                | Proposed            | 51                    | 1                    | 6                       | 11.76%         |
| 90°            | Original            | 50                    | 16                   | 7                       | 46.00%         |
|                | Proposed            | 48                    | 1                    | 9                       | 20.83%         |
| 120°           | Original            | 50                    | 19                   | 7                       | 52.00%         |
|                | Proposed            | 50                    | 4                    | 7                       | 22.00%         |
| 150°           | Original            | 51                    | 22                   | 6                       | 54.90%         |
|                | Proposed            | 52                    | 1                    | 5                       | 11.54%         |
4. Conclusions
Aiming at the Harris corner detection algorithm's problems, such as, the manual setting of thresholds, poor noise resistance, prone to false corners and corner clusters, inaccurate corner positioning, and low detection efficiency, an improved Harris algorithm based on Canny edge detection and gray difference preprocessing is proposed. The experimental results indicate that compared with the original algorithm, the proposed algorithm has increased the detection accuracy about 5%; the detection efficiency has increased by 21.14%; the false corner points have been effectively reduced. When \( q = 0.005 \), the improved algorithm's accuracy has reached more than 80%; corner positioning can reach sub-pixel level; it has stronger robustness in noise resistance and rotation invariance. However, the proposed algorithm's efficiency is not significantly improved, and there are more missed points. How to further improve the efficiency and reduce the missed points is the future research direction.

References
[1] Hong, G.Y., Rui, T.X., Yu, W.G., et al. (2017) Optimization algorithm of Harris corner detection. J. Computer system application, 26(4):169-172
[2] Guan, B., Vasseur, P., Demonceaux, C., and Fraundorfer, F. (2018) Visual Odometry Using a Homography Formulation with Decoupled Rotation and Translation Estimation Using Minimal Solutions. In: 2018 IEEE International Conference on Robotics and Automation (ICRA). 2320-2327.
[3] Guan, B., Zhao, J., Li, Z., Sun, F., and Fraundorfer, F. (2020) Minimal Solutions for Relative Pose With a Single Affine Correspondence. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). 1926-1935.
[4] Guan, B., Zhao, J., Li, Z., Sun, F., and Fraundorfer, F. (2021) Relative Pose Estimation with a Single Affine Correspondence. In: IEEE Transactions on Cybernetics, doi: 10.1109/TCYB.2021.3069806.
[5] Zitová, B., Kautsky, J., Peters, G., et al. (1999) Robust Detection of Significant Points in Multiframe Images. J. Pattern Recognition Letters, 20(2):199-206.
[6] Zhang, J., Li, Y.L., Wang, Y.P. (2013) Improved Harris corner detection algorithm based on B-spline function. J. Journal of Lanzhou Jiao tong University,32(6):32-35.
[7] Zou, Z.Y., An B.W., Cao, F., et al. (2015) Adaptive corner point detection of infrared images. J. Laser&Infrared, 45(10):1272-1276.
[8] Zhang, J.S., Zhang, H.M., Luo, Y.T., et al. (2017) An improved Harris corner detection image registration method. J. Laser&Infrared, 47(2):230-233.
[9] Dong, L.H., Peng Y.X., Fu L.M. (2019) A circular Harris corner detection algorithm based on Sobel edge detection. J. Journal of Xi'an University of Science and Technology, 39(02):374-380.
[10] Cai, X.Z., Liu, Y.Y. (2020) A fast image registration algorithm based on adaptive screening Harris corner detection. J. Semiconductor Optoelectronics, 41(06):875-878.
[11] Wang, Z., He, S.X. (2004) An adaptive edge detection method based on Canny theory. J. Journal of Image and Graphics, (08):65-70.
[12] Wang, H.Y., Chen, D.L., Shuai, Y.S., et al. (2019) Improved Harris corner detection algorithm based on Gray difference pre-processing. J. Electronic Technology and Software Engineering, (09):72-73.
[13] Han, S.Q., Yu, W.B., Yang, H.T., et al. (2018) Improved Harris corner detection algorithm. J. Journal of the Changchun University of Technology, 39(05):470-474.
[14] Li, Y.B., Li, J.J. (2011) Harris corner detection algorithm based on improved contourlet transform. J. Procedia Engineering, 15:2239-2243.