IET Image Processing

Research Article

IET Journals
The Institution of Engineering and Technology

Automatic chessboard corner detection method

ISSN 1751-9659
Received on 14th February 2015
Revised on 16th June 2015
Accepted on 5th July 2015
doi: 10.1049/iet-ipr.2015.0126
www.ietdl.org

Yu Liu[1] ✉, *Shuping Liu*[1], *Yang Cao*[1], *Zengfu Wang*[1,2]

[1]*Department of Automation, University of Science and Technology of China, Hefei 230026, People's Republic of China*
[2]*Institute of Intelligent Machines, Chinese Academy of Sciences, Hefei 230031, People's Republic of China*
✉ *E-mail: liuyu1@mail.ustc.edu.cn*

Abstract: Chessboard corner detection is a necessary procedure of the popular chessboard pattern-based camera calibration technique, in which the inner corners on a two-dimensional chessboard are employed as calibration markers. In this study, an automatic chessboard corner detection algorithm is presented for camera calibration. In authors' method, an initial corner set is first obtained with an improved Hessian corner detector. Then, a novel strategy that utilises both intensity and geometry characteristics of the chessboard pattern is presented to eliminate fake corners from the initial corner set. After that, a simple yet effective approach is adopted to sort the detected corners into a meaningful order. Finally, the sub-pixel location of each corner is calculated. The proposed algorithm only requires a user input of the chessboard size, while all the other parameters can be adaptively calculated with a statistical approach. The experimental results demonstrate that the proposed method has advantages over the popular OpenCV chessboard corner detection method in terms of detection accuracy and computational efficiency. Furthermore, the effectiveness of the proposed method used for camera calibration is also verified in authors' experiments.

1 Introduction

Camera calibration is an important technique which has been widely used in many machine vision applications such as stereo vision [1], visual surveillance [2] and intelligent transportation [3]. In the last few decades, various camera calibration methods have been introduced, which can be generally grouped into three categories: self-calibration, two-dimensional (2D) planar pattern-based calibration and 3D object pattern-based calibration. Among them, the calibration methods using 2D planar pattern like circles [4, 5], grid squares [6–8] and concentric circles [9, 10] are very popular for their practical convenience. In particular, the planar chessboard with black-and-white squares is one of the most commonly used patterns for its flexibility. All the chessboard pattern-based calibration methods are on the premise of the accurate detection of chessboard inner corners, which are also known as X-corners for their intuitive perception. Datta *et al.* [11] verified that the accurate detection of these control points is of great importance to camera calibration. However, compared with the great concentration on developing robust algorithms for the calculation of camera parameters, less attention has been paid to the bottleneck of X-corner detection.

In general, the existing X-corner detection methods can be broadly classified into two categories: intensity-based methods [12–15] and geometry-based methods [16–18]. The intensity-based methods mainly focus on the greyscale characteristics of the local regions around X-corners. This category of methods usually relies on a generic or specific corner detector. Harris corner detector [19], smallest univalue segment assimilating nucleus (SUSAN) [20], Hessian matrix and 2D Hilbert transform are employed in publications [12–15]. In these methods, a response map is generated to decide whether a pixel is a corner with a given threshold. Due to the impacts of noise, illumination and clutter, it is usually difficult to set the response threshold. As a result, some real X-corners are often missed while some fake corners are misdetected, which will negatively affect the calibration accuracy. The geometry-based methods aim to detect the X-corners using the geometry characteristics of the chessboard pattern, such as the rectilinear distribution of X-corners and the cross-connection of the black-and-white squares. In [16], Hough transform is performed to detect straight lines for corner detection. The characteristic that black-and-white squares appear in sequence on a chessboard is fully considered by Ha [17]. Chu *et al.* [18] exploited the property of chessboard pattern after a morphological dilation operation. Since the prior knowledge of the chessboard pattern is well utilised, the detection results of these methods are often more accurate. However, this category of methods tends to produce unstable results when the distortion of camera lens is too severe or the projection angle of chessboard plane is too large. Furthermore, there usually exist many free parameters that need to be set manually in these methods, which is not an easy task in practical applications.

Currently, there are two widely used camera calibration interfaces. The first one is a MATLAB toolbox developed by Bouguet [21]. This toolbox is very effective for it has many powerful functions. However, its X-corner detection step asks users to click on the four extreme corners of the chessboard for each calibration image, which limits its practicability to a large extent. The second one is the camera calibration module included in the OpenCV library [22]. In this module, a chessboard corner detection function named *findChessboardCorners* is offered. Unlike the manual approach adopted in the MATLAB toolbox [21], this OpenCV function only requires users to input the size of chessboard (the number of X-corners in 2D). Generally, the OpenCV method can achieve satisfactory results even in the scene with complex background. However, it usually fails in working when the area of square is too small or the projection angle is too large. Meanwhile, the computational efficiency of this method will significantly decrease when the number of X-corners increases.

In this work, we present a new automatic X-corner detection method for camera calibration. In our algorithm, a Hessian corner detector proposed in [14] is first improved to construct an initial corner set. Then, a novel strategy that takes both intensity and geometry characteristics of the chessboard pattern into account is employed to eliminate fake corners from the initial corner set. After that, a simple yet effective approach is adopted to sort the

detected corners into a meaningful order. Finally, the sub-pixel location of each corner is calculated. Like the OpenCV method, our method only requires users to provide the chessboard size, while all the other parameters can be adaptively calculated with a statistical approach.

A preliminary version of this work was appeared in [23], where we only introduced an incomplete pixel-level X-corner detection approach. In [23], the detected corners were not meaningfully sorted and precisely located. We also did not make an attempt on camera calibration. In this paper, the whole detection process is completed. Moreover, we conduct more experiments on location precision and calibration accuracy to further confirm the effectiveness of the proposed method. The rest of this paper is structured as follows. The relation to prior work is presented in Section 2. Section 3 describes the detailed detection algorithm. The experimental results and discussions are given in Section 4. Finally, Section 5 concludes the paper.

2 Related work

The proposed method is improved and extended from Chen and Zhang's [14] corner detection method, in which they introduced a novel X-corner detector based on Hessian matrix. In this section, we first have a brief review of the Hessian corner detector and then list the main contributions of this paper.

Supposing that $f(x, y)$ is the original chessboard image and $r(x, y)$ is its Gaussian blur version, the Hessian matrix of $r(x, y)$ is defined as

$$\boldsymbol{H} = \begin{pmatrix} r_{xx} & r_{xy} \\ r_{xy} & r_{yy} \end{pmatrix}, \tag{1}$$

where r_{xx}, r_{xy} and r_{yy} are the second-order derivatives of $r(x, y)$. As X-corners locate at the centres of black-and-white squares, one can easily find that they are the saddle points in $r(x, y)$ (the purpose of Gaussian blur is to construct these saddle points). It is well known that Hessian matrix is a suitable tool to detect the saddle points of a 2D function, and in [14] the determinant of Hessian matrix is used to construct the response map

$$S = \det(\boldsymbol{H}) = r_{xx} \cdot r_{yy} - r_{xy}^2. \tag{2}$$

For an X-corner, its response should be a negative local minimum in S. Instead of directly searching for the negative local minima in Spixel by pixel, the method in [14] searches for the local minima in S with the following constraint:

$$\lambda_1 > 0 \text{ and } \lambda_2 < 0, \tag{3}$$

where $\lambda_1, \lambda_2 = 1/2(r_{xx} + r_{yy} \pm \sqrt{(r_{xx} - r_{yy})^2 + 4r_{xy}{}^2})$ are the larger and smaller eigenvalues of H, respectively. Actually, since $S = \lambda_1 \cdot \lambda_2$, the above criterion is totally equivalent to only checking the pixels with $S < 0$ whether they are local minima or not. Finally, the X-corners are obtained with a fixed response threshold.

Compared with some generalised detectors such as Harris detector [21] and SUSAN [22], the Hessian detector is a specific approach for the detection of X-corner pattern. However, the method in [14] still has two main shortcomings:

i. The constraint in (3) is somewhat too weak, which may cause that many pixels in the 'flat' regions of an image also satisfy the constraint. As a result, the computational efficiency will significantly decrease. In fact, λ_1 and λ_2 represent the maximum and minimum of the directional derivatives of the image, respectively. If the value of λ_1 or λ_2 is approximate to zero, it is almost impossible for the corresponding pixel to be an X-corner although it may be indeed a negative local minimum.
ii. Although the Hessian detector makes good use of the prior information of the X-corner pattern, it is still impossible to find a threshold to ensure that all the real X-corners are detected with no fake corner mingled. This is mainly because the local greyscale distribution of some pixels may be very similar to an X-corner, especially when the background is complex. As fake corners have adverse impacts on the subsequent calibration steps, it is worthwhile to eliminate them in the detection step.

We address the above two issues in this work, especially the latter one. The main contributions of this paper are summarised as follows:

i. An improved Hessian detector is presented on the basis of the method in [14].
ii. A robust approach for fake corner elimination is proposed.
iii. A simple yet effective approach is introduced to sort the detected X-corners into a meaningful order.

3 The proposed method

3.1 Improved Hessian detector

To overcome the first shortcoming mentioned above, the constraint in (3) is replaced by the following one:

$$\lambda_1 > \varepsilon \text{ and } \lambda_2 < -\varepsilon, \tag{4}$$

where ε is a small positive number. In our algorithm, it is set to $C \cdot \lambda_{\max}$, where C is a constant and $\lambda_{\max}$ is the maximum value of all the pixels in mapλ_1. Fig. 1 gives an example which exhibits the effect of this constraint with different constant Cranging from 0 to 0.05 by a stride of 0.01. Please note that the red dots in Fig. 1 denote the local minima in the response map obtained by (2) with the new constraint (for each pixel in the response map, we first check whether it satisfies the constraint. Only when the condition is valid, we further check whether it is a local minimum), so they are just a subset of the pixels which satisfy the constraint. On one hand, it can be seen from Fig. 1 that when $C = 0$, which indicates the original situation in (3), a large number of pixels in 'flat' regions (the wall) are recognised. Actually, as mentioned above, much more pixels satisfy the constraint (they may be not local minima), leading to a time-consuming process. On the other hand, to avoid missing real X-corners, the value of C also cannot be too large. In this example, when C ranges from 0 to 0.05, the number of detected dots are 10,129, 1155, 356, 221, 175 and 151, respectively. Through a quantity of tests, we find that $C = 0.03$ is always a reasonable choice, so we fix it to 0.03 in our algorithm. With this stronger constraint, most of the pixels in 'flat' regions will not be viewed as candidates, so the subsequent procedure of verifying local minima is not required. Thus, after applying the new constraint, the computational efficiency can be greatly improved, especially when the area of 'flat' regions in the scene is large.

For the second problem, we would rather involve some fake corners than miss any real X-corner since there are elimination approaches later. Therefore, the response threshold is set conservatively. Empirically, the threshold can be adaptively set to the value which makes the number of initial X-corners twice that of the real X-corners. After applying the improved Hessian detector, an initial corner set can be obtained, and the next step to eliminate the fake corners in it.

3.2 Fake corner elimination

Three properties of chessboard pattern, namely, centrosymmetry property, distance property and angle property are developed to accomplish the elimination task in this work. For the convenience of description and understanding, an illustration to describe the elimination process is shown in Fig. 2. The red dots denote the obtained X-corners after applying the centrosymmetry property in Section 3.2.1. Some of them are named via uppercase letters for method description. The intersectional arrows illustrate angles. The subfigure embedded in the top right is a circular mask employed in Section 3.2.1.

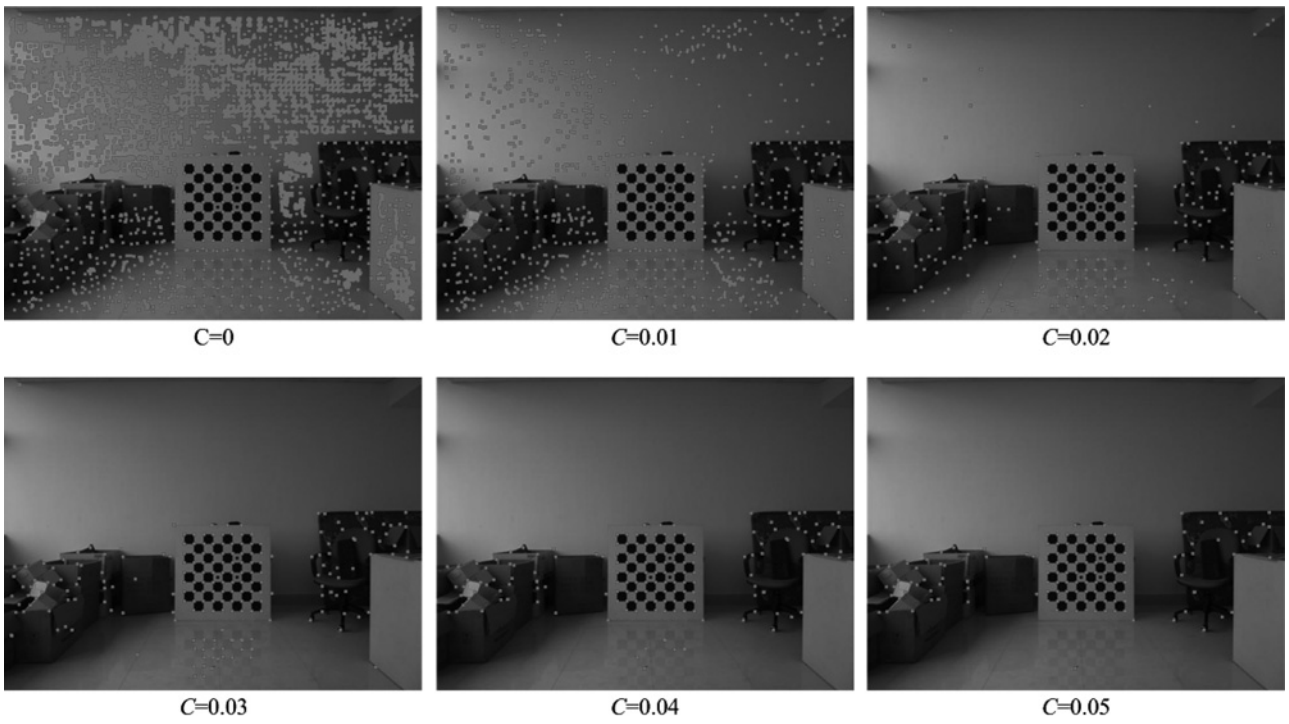

Fig. 1 *Effect of the constraint in (4) with different constant C ranging from 0 to 0.05 by a stride of 0.01. The red dots denote the local minima in the response map obtained by (2) with the new constraint. When C ranges from 0 to 0.05, the number of detected dots are 10,129, 1155, 356, 221, 175 and 151, respectively*

3.2.1 Centrosymmetry property: As shown in Fig. 2, it is easy to see that the local intensity distribution around an X-corner is approximately centrosymmetric. In our method, a circular mask shown in the top right in Fig. 2 is used to verify whether a candidate corner satisfies this property.

Let $\bar{I}_i$ ($i \in \{1, \ldots, 8\}$) denote the average intensity values of the eight equal sectors in the circular mask. When the centre of the circular mask locates at an X-corner, the difference between $\bar{I}_i$ and $\bar{I}_{i+4}$($i \in \{1, \ldots, 4\}$) will be small while the difference between $\bar{I}_i$ and $\bar{I}_{i+2}$($i \in \{1, \ldots, 4\}$) will be large. Consequently, the criterion is

$$(D_1 < pD_3 \text{ and } D_2 < pD_3) \text{ or } (D_4 < pD_6 \text{ and } D_5 < pD_6), \quad (5)$$

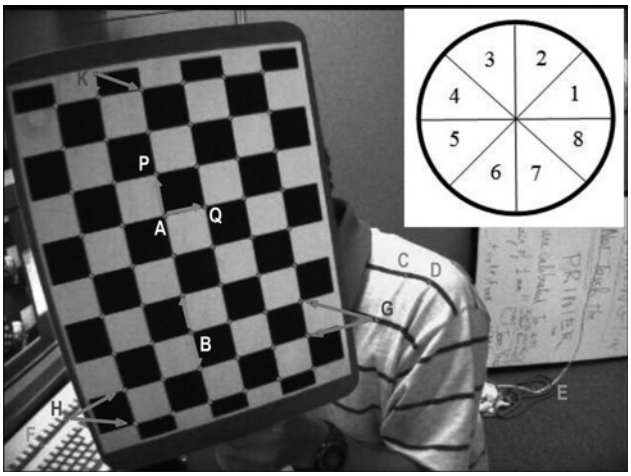

Fig. 2 *Illustration of fake corner elimination. The red dots denote the obtained X-corners after applying the centrosymmetry property in Section 3.2.1. Some of them are named via uppercase letters. The intersectional arrows illustrate angles. The subfigure embedded in the top right is a circular mask employed in Section 3.2.1*

where p is a ratio factor ranging from 0 to 1 and

$$D_1 = |\bar{I}_1 - \bar{I}_5|,\ D_2 = |\bar{I}_3 - \bar{I}_7|,\ D_3 = |\bar{I}_1 + \bar{I}_5 - \bar{I}_3 - \bar{I}_7|/2,$$
$$D_4 = |\bar{I}_2 - \bar{I}_6|,\ D_5 = |\bar{I}_4 - \bar{I}_8|,\ D_6 = |\bar{I}_2 + \bar{I}_6 - \bar{I}_4 - \bar{I}_8|/2. \quad (6)$$

For each corner in the initial set obtained from the Hessian detector, it will be eliminated if the criterion in (5) is invalid. An adaptive calculation approach for the circular mask's radius and the parameter p will be presented in Section 3.5. It should be noted that the two terms in (5) are connected with 'or' rather than 'and'. This is because if the constraint is too strong, some real X-corners will be eliminated since the noise and illumination changes always exist in practice.

It can be seen from Fig. 2 that all the real X-corners (such as A, B) are preserved after using this property, but several stubborn fake corners (C, D, E, F, G, H, K) still exist because their local intensity distribution are also approximately centrosymmetric.

3.2.2 Distance property: We can see from Fig. 2 that for a real X-corner, there are at least three neighbour X-corners around it. Therefore, some isolated fake corners can be rooted out if they have less than three neighbours. Therefore, for each candidate cornerc_i, the criterion is

$$\#\{c_j | j \in \{1, \ldots, N\},\ j \neq i,\ ||c_i - c_j||_2 < d\} \geq 3, \quad (7)$$

where N is the size of current corner set and d is a distance threshold. In Fig. 2, four fake corners (C, D, E, F) are successfully eliminated after applying this property. The calculation of d is also presented in Section 3.5.

3.2.3 Angle property: Finally, for an arbitrary candidate corner (such as A), we can search for its nearest two corners (P, Q) and the intersection angle (PAQ) is employed to distinguish real X-corners. We can see from Fig. 2 that for a fake corner (G, H, K), this angle is usually a small acute angle (e.g. 30°). While for a real X-corner, this angle is usually significantly larger even when the projection angle of the chessboard plane to camera is large.

IET Image Process., 2016, Vol. 10, Iss. 1, pp. 16–23

Considering the cosine function is monotonically decreasing from 0° to 180°, the criterion is

$$\cos\theta < t, \tag{8}$$

where θ is the intersection angle and t is a cosine threshold. The calculation of t is also given in Section 3.5. If the criterion in (8) is invalid, the corresponding corner will be viewed as a fake corner. As shown in Fig. 2, the last three fake corners are eliminated with this property.

It should be noted that the angle property must be iteratively used. This is because sometimes a few fake corners tend to get together in a small area, which makes it impossible to eliminate all of them at once. For example, if several fake corners locate one by one closely in a line (this situation is very common in practice), then only the two terminal points can be eliminated after the first iteration. Furthermore, to accelerate the computational process, the iteration procedure is set from the distance property to the angle property in our method.

3.3 Corner ordering

A reliable X-corner set can be obtained after eliminating the fake corners, but the corners should be sorted into a meaningful order before they are used for camera calibration. We introduce a simple yet effective approach to solve this problem here. Fig. 3 shows the illustration of the corner ordering process. As shown in Fig. 3, for a chessboard, its upper left, upper right, lower left and lower right corners are denoted as A, B, C and D, respectively. The subfigure embedded in the top right contains four equations used for determining the four extreme corners, where $c_i$is a candidate in the detected corner setΛ. In addition, c_i^x and c_i^y are the horizontal and vertical coordinates of c_i, respectively.

To sort the detected X-corners, we first identify the four extreme ones by comparing the sums and differences of the two coordinate values over all of the corners. For example, the upper left one owns the minimal sum while the upper right one owns the maximal difference. The situations of the other two are similar. Fig. 3 provides the related mathematical description. In most cases of practical camera calibration, the above approach works well. The ordering of X-corners will be an easy task after obtaining the four extreme points. Considering the example shown in Fig. 3, we can first search for the corners between A (B) and C (D) via the linearity, and then get the inside corners on each row (there are nine rows in Fig. 3). As a result, all the detected corners can be correctly ordered. Please note that the order shown in Fig. 3 is not unique since the chessboard has different number of rows and columns. In practical applications, the order should be adjusted according to the calibration requirement, but the approach remains the same.

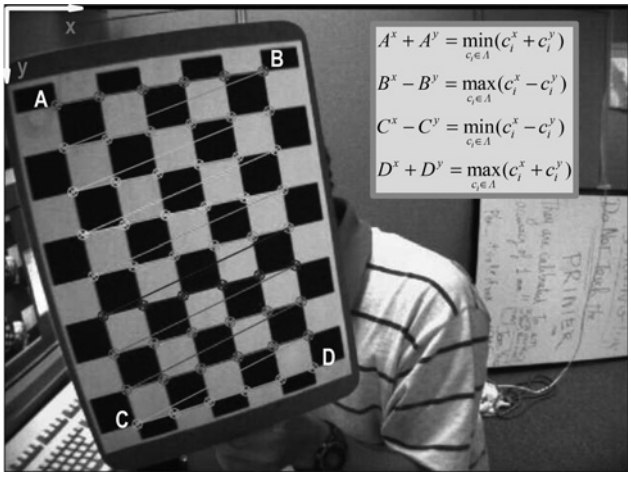

Fig. 3 *Illustration of corner ordering. For a chessboard, its upper left, upper right, lower left and lower right corners are denoted as A, B, C and D, respectively. The subfigure embedded in the top right contains four equations used for determining the four extreme corners. The meanings of the variables in these equations are given in the text*

3.4 Precise location

Finally, we use the squared greyscale centroid method [24] to obtain the sub-pixel location for each corner. Specifically, the sub-pixel location (x_0, y_0) of a corner c is calculated by

$$x_0 = \frac{\sum_{(i,j)\in R} i \cdot I^2(i,j)}{\sum_{(i,j)\in R} I^2(i,j)}, y_0 = \frac{\sum_{(i,j)\in R} j \cdot I^2(i,j)}{\sum_{(i,j)\in R} I^2(i,j)}, \tag{9}$$

Where R is a small circular window located at c and $I(i, j)$ indicates the intensity value at pixel (i, j).

3.5 Adaptive parameter setting

The proposed method has four main parameters: the radius r of the circular mask in Fig. 2, the ratio factor p in (5), the distance threshold d in (7) and the cosine threshold t in (8). The radius of window R is also set to r. To make the method more practical, we present an effective approach to adaptively estimate these parameters.

Considering the perspective transform, we use $a_{\min}$ and $a_{\max}$ to represent the shortest and longest side lengths of all the squares, respectively. To guarantee the correctness of the above three properties, it is obvious that both $r < a_{\min}$ and $d > \sqrt{2}a_{\max}$ should be satisfied. Furthermore, the parameters p and t can also be estimated by the two side lengths. When the difference between the two side lengths is larger, that is, the extent of perspective transform is stronger, both p and t should increase to make the constraints weakened. In our algorithm, the parameters are set as follows:

$$r = 0.7a_{\min},\ p = 0.3a_{\max}/a_{\min},\ d = 2a_{\max},\ t = 0.4a_{\max}/a_{\min}. \tag{10}$$

In our method, $a_{\min}$ and $a_{\max}$ are estimated using the initial corner set obtained by the improved Hessian detector in Section 3.1. For each candidate, we can find its nearest neighbour from all the other candidates and calculate the corresponding distance. Therefore, a histogram that describes the distribution of nearest distance can be generated. A typical example is shown in Fig. 4. Then, we get the peak of the histogram and a reliable subset of distance data shown

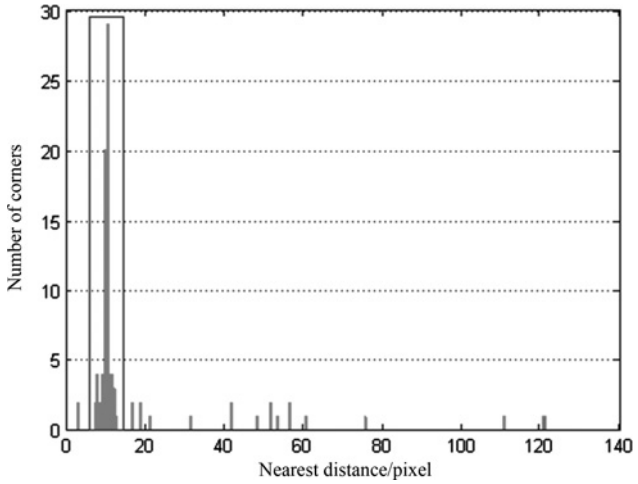

Fig. 4 *Typical distance histogram used for parameter estimation. The data contained in the rectangle is used for side-length estimation*

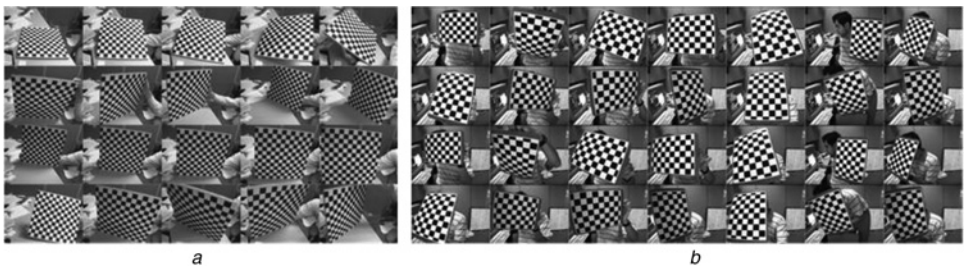

Fig. 5 *Two public data sets for camera calibration*

a Set contains 20 images, and the chessboard has$13 \times 12 = 156$ corners
b Set contains 14 pairs of stereo images, and the chessboard has $9 \times 6 = 54$ X-corners

Table 1 Detection rate of the OpenCV method and the proposed method

	Image set 1, %	Image set 2, %
OpenCV	20	100
proposed	90	100

as the rectangle in Fig. 4 can be obtained. Particularly, we extend the subset from the peak to two sides step by step until the number of elements in the subset reaches 80% of the total number of candidates. Finally, by employing the Gaussian distribution model, $a_{\min}$ and $a_{\max}$ are calculated by

$$a_{\min} = \bar{a} - 3\sigma, \ a_{\max} = \bar{a} + 3\sigma, \tag{11}$$

where $\bar{a}$ and σ are the mean value and the standard deviation of the distance data in the obtained subset, respectively.

4 Experiments

4.1 Experimental setups

As shown in Fig. 5, two public image data sets for camera calibration from website [21] are mainly used in our experiments. The first data set shown in Fig. 5*a* consists of 20 calibration images, and the size of chessboard is $13 \times 12 = 156$. The second data set shown in Fig. 5*b* contains 14 pairs of stereo images, with $9 \times 6 = 54$ X-corners in the chessboard. The spatial resolution of each image in both the two data sets is 640×480 pixels.

To confirm the effectiveness of our method, we mainly compare it with the popular OpenCV detection method [22]. The function *findChessboardCorners* in OpenCV *calib3d* module can

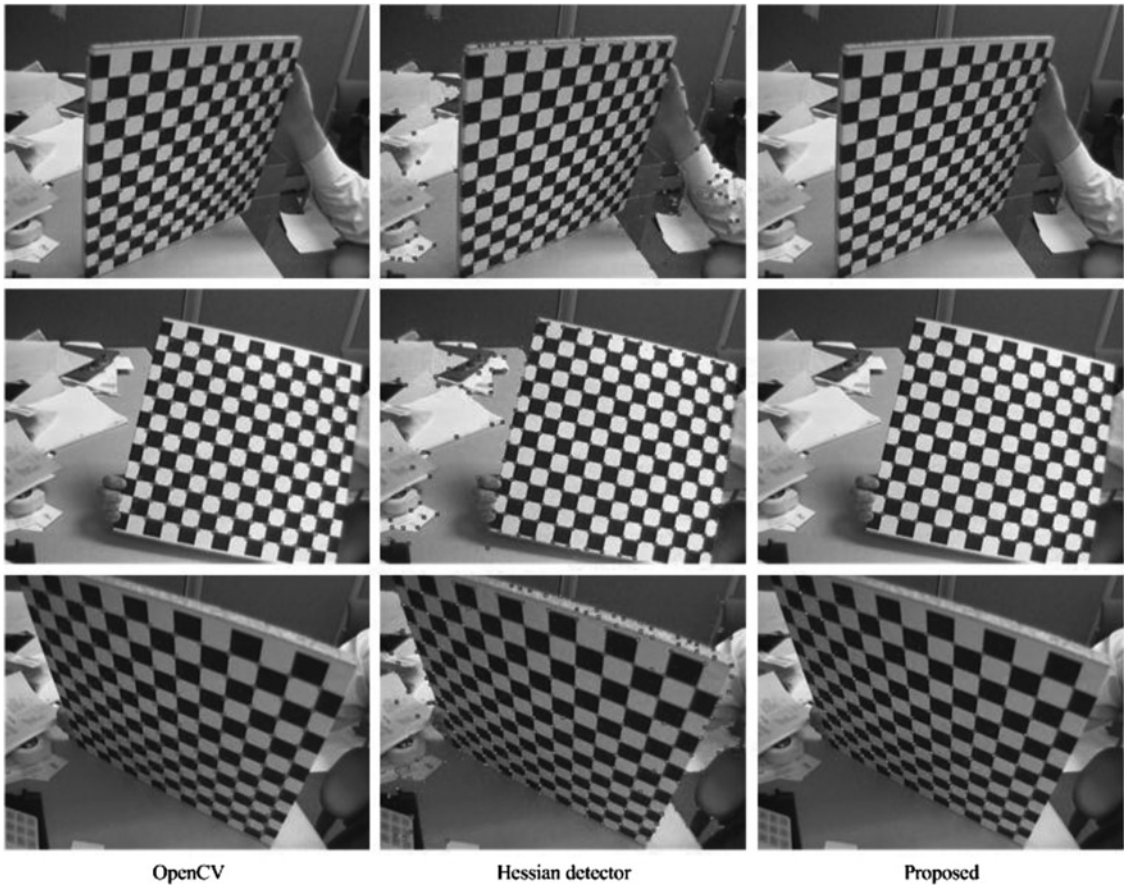

Fig. 6 *Three examples of X-corner detection results. The left, middle and right columns show the results of the OpenCV method, the Hessian detector and the proposed method, respectively*

IET Image Process., 2016, Vol. 10, Iss. 1, pp. 16–23

Table 2 Average distance between the refined locations of the OpenCV method and the proposed method (unit: pixel)

	Image set 1	Image set 2 – left	Image set 2 – right
distance	0.013	0.010	0.009

Table 3 Average distance between the reference location and the location obtained by the OpenCV method or the proposed method (unit: pixel)

	Image set 1	Image set 2 – left	Image set 2 – right
OpenCV	0.552	0.438	0.310
proposed	0.465	0.386	0.354

automatically detect and sort the X-corners at sub-pixel level, and it also just requires the users to input the size of chessboard. Thus, the inputs and outputs of the proposed method and the OpenCV method are completely the same, which guarantees the fairness of the comparison.

The comparison is conducted on four aspects, namely, *detection rate, location precision, calibration accuracy* and *computational efficiency*. For each image set, the detection rate equals the ratio of successfully detected images in the whole set. An image is successfully detected means that all the real X-corners are detected with no fake corner involved. The location precision is measured by the average distance between an obtained sub-pixel location and its reference location. The reference location is calculated by the function *cornerSubPix* in OpenCV *imgproc* module. This function adopts an iterative strategy to find the accurate sub-pixel location of corners. Actually, to refine the location precision for calibration, the OpenCV library suggests users to apply this function after obtaining corners by the function *findChessboardCorners*. The feasibility of this function used to calculate the reference location will be experimentally verified later. The calibration accuracy is measured by camera intrinsic parameters and the re-projection error. The re-projection error indicates the average distance between an observed point and its corresponding projected point (using the calculated camera intrinsic and extrinsic parameters) from the chessboard pattern. To compare the computational efficiency, we measure the average running time of the two methods' corner detection parts. All the experiments are carried on a PC with 2.8 GHz central processing unit and 4 GB random access memory.

4.2 Experimental results

Table 1 lists the detection rate of each image set. It can be seen that for the second set, both of the two methods can perfectly accomplish the detection task for all the 28 images. However, for the first set, the OpenCV method can only successfully handle four images (Nos. 9, 11, 17, 20), while our method only fails in processing two images (Nos. 5, 18). Fig. 6 shows three examples (Nos. 7, 16, 18 in the first set) of detection results of the OpenCV method, the Hessian detector and the proposed method. In all the three examples, the Hessian detector can obtain all the real X-corners but many fake corners are involved at the same time. In the first example, the OpenCV method misses four real X-corners (three are extreme ones) and wrongly detects a fake corner. The proposed method can successfully detect all the real X-corners with no fake corner involved. In the second example, the OpenCV method misses two extreme corners, and we can clearly see that the locations of many corners are not very accurate. The proposed method also successfully finishes the detection task. In the third example, the projection angle of the chessboard plane is larger than the first two examples. As a result, the OpenCV method misses several corners in the lower left region. The proposed method obtains all the real corners, but one fake corner near the chessboard is involved because all of the above three properties fails in eliminating it.

Table 5 Average computational time of the two methods on one image (unit: s)

	Image set 1	Image set 2
OpenCV	1.027	0.133
proposed	0.126	0.104

To measure the location precision, as mentioned above, we use the function *cornerSubPix* to obtain the reference location for each corner. The feasibility of this approach is first verified. For the first image set, only four images (Nos. 9, 11, 17 and 20) take part in the calculation since others are not successfully processed by the OpenCV method or the proposed method. For the second image set, we further divide it into two subsets: 14 left images and 14 right images. For each new data set, the OpenCV method and the proposed method are employed to detect the X-corners. Then, we apply *cornerSubPix* to refine the obtained results. For each set, let L_1 and L_2 denote the refined locations of the OpenCV method and the proposed method, respectively. The average Euclidean distance of two corresponding locations in L_1 and L_2 is given in Table 2. It can be seen that the distance is only about 0.01 pixel for each set, which means that the function *cornerSubPix* is not sensitive to the inputs. Therefore, it is reasonable to use the refined results as the reference. In our experiments, for an X-corner, its reference location is set to the average value of the two refined results. Table 3 lists the average Euclidean distance between the reference location and the location obtained by the OpenCV method or the proposed method for each image set. It can be seen that our method slightly outperforms the OpenCV method on the first image set. For the second set, the location precisions of the two methods are comparable.

In this work, calibration results are obtained by the OpenCV function *calibrateCamera*, which is implemented based on the calibration method [6]. For either the OpenCV method or the proposed method, the original detection results and the refined results by *cornerSubPix* are employed for calibration, respectively. Since the successfully detected images in the first image set are not enough, only the second set are used to test here, and we also divide it into two parts as before. The camera calibration results of the two methods are listed in Table 4, where f_x, f_y are the local lengths and (c_x, c_y) are the principal points. It can be seen that the results of the OpenCV method and the proposed methods are very approximate when the refinement is performed. When there is no refinement, the proposed method outperforms the OpenCV method

Table 4 Camera calibration results of different methods (unit: pixel)

Data	Methods	Refinement	f_x	f_y	c_x	c_y	Re-projection error
left image set	OpenCV	no	530.693	531.430	342.640	237.703	0.619
	OpenCV	yes	532.801	532.965	341.773	234.101	0.199
	proposed	no	533.365	533.298	341.376	235.343	0.492
	proposed	yes	532.843	532.990	341.805	234.110	0.198
right image set	OpenCV	no	537.115	536.842	327.776	250.073	0.454
	OpenCV	yes	537.147	536.722	326.941	249.142	0.211
	proposed	no	536.872	536.107	326.649	250.495	0.498
	proposed	yes	537.124	536.705	326.958	249.137	0.210

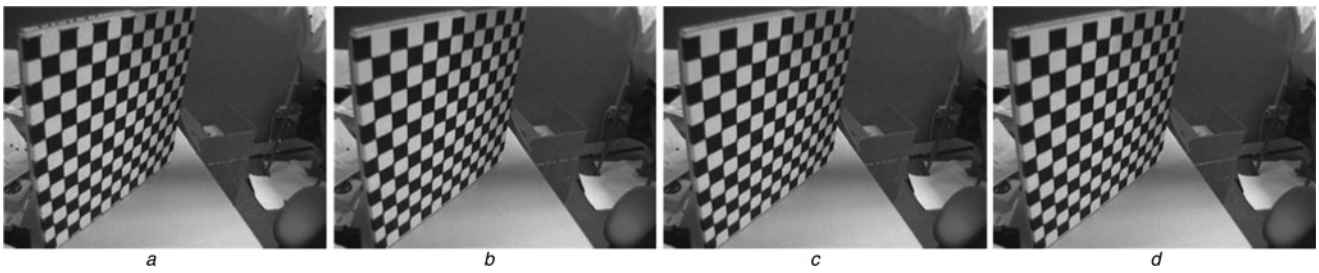

Fig. 7 *Example of fake corner elimination*

a Detection results after applying improved Hessian matrix detector
b Detection results after applying the centrosymmetry property
c Detection results after applying the distance property (the first iteration)
d Final detection results

for the left image set, but the situation of the right image set is just on the contrary. Please note that the results in Table 4 are in accord with those in Table 3.

Table 5 lists the average computational time of the two methods on one image. It can be seen that the proposed method is more efficient than the OpenCV method even though our program is not well optimised. Furthermore, the computational efficiency of the OpenCV method significantly decreases when the number of X-corners increases from 54 to 156. However, our method can still maintain a high speed in this situation.

In summary, the above experimental results demonstrate that the proposed method owns clear advantages over the OpenCV method in terms of detection rate and computational efficiency, which is the highlight of the proposed method. Furthermore, the location precision of the detected corners by our method is at least comparable with the OpenCV method before refinement, which ensures that the locations after refinement can result in a reliable calibration performance. In practical calibration applications, the refinement stage using the function *cornerSubPix* can be combined into our method to further improve location precision and calibration accuracy.

4.3 Further discussions on fake corner elimination

The primary contribution of this paper is the proposed strategy for fake corner elimination. In this subsection, we have a brief review of the elimination approach. The centrosymmetry property is an intensity-based property, which is strictly valid for affine transforms and approximately valid for perspective transforms. In real-world camera calibration applications, this property can always work well as long as the distortion of camera lens is not too severe. Moreover, the condition in (5) is not very strong, so the real X-corners have a very high probability to be completely preserved after using this property. However, for the same reason, a few stubborn fake corners that have similar local intensity distributions may still exist. Thus, we present two subsequent properties, which make good use of the geometry characteristics of chessboard pattern. The distance property can eliminate isolate corners, and it is valid even for non-linear transform (lens distortion). However, if several fake corners distribute closely, it will lose effectiveness. Fortunately, the angle property is an effective tool to solve this problem, and it can gradually disassemble and eliminate those fake corners. Fig. 7 provides a typical example of the elimination process. For more examples as well as the experimental results on some other calibration image sets, please refer to our supplementary material which is available on http://yuliuustc.weebly.com/.

5 Conclusions

In this paper, we propose an automatic chessboard corner detection algorithm for camera calibration. We first improve the Hessian corner detector to make it more efficient. Then, we present an effective strategy which simultaneously employs the intensity and geometry characteristics for fake corner elimination. Finally, a simple yet effective approach is introduced to accomplish corner ordering task. To make the algorithm more practical, we design a statistical approach to adaptively estimate the free parameters. The popular OpenCV corner detection method is used for comparison in the experiments. The effectiveness of the proposed method is verified from several aspects such as detection rate, location precision, calibration accuracy and computational efficiency. In the future, we will develop some new intensity and geometry characteristics of the chessboard pattern to make the detection algorithm more robust.

6 Acknowledgments

The authors like to thank the editor and the anonymous reviewers for their detailed review, valuable comments and constructive suggestions. This work was supported by the National Natural Science Foundation of China (no. 61472393) and the National Science and Technology Projects (no. 2012GB102007).

7 References

1 Wang, H.M., Chang, C.W., Yang, J.F.: 'An effective calibration procedure for correction of parallax unmatched image pairs', *IET Image Process.*, 2009, **3**, (2), pp. 63–74
2 Cui, Z., Li, A., Feng, G., *et al.*: 'Cooperative object tracking using dual-pan–tilt–zoom cameras based on planar ground assumption', *IET Comput. Vis.*, 2015, **9**, (1), pp. 149–161
3 Dinh, H., Tang, H.: 'Simple method for camera calibration of roundabout traffic scenes using a single circle', *IET Intell. Transp. Syst.*, 2014, **8**, (3), pp. 175–182
4 Heikkila, J.: 'Geometric camera calibration using circular control points', *IEEE Trans. Pattern Anal. Mach. Intell.*, 2000, **22**, (10), pp. 1066–1077
5 Bergamasco, F., Cosmo, A., Albarelli, A., *et al.*: 'Camera calibration from coplanar circles'. Proc. 22nd Int. Conf. on Pattern Recognition, Stockholm, Sweden, August 2014, pp. 2137–2142
6 Zhang, Z.: 'A flexible new technique for camera calibration', *IEEE Trans. Pattern Anal. Mach. Intell.*, 2000, **22**, (11), pp. 1330–1334
7 Douterloigne, K., Gautama, S., Philips, W.: 'Fully automatic and robust UAV camera calibration using chessboard patterns'. Proc. IEEE Int. Geoscience and Remote Sensing Symp., Cape Town, South Africa, July 2009, pp. 551–554
8 Ellmauthaler, A., da Silva, E., Pagliari, C., *et al.*: 'A novel iterative calibration approach for thermal infrared cameras'. Proc. 20th IEEE Int. Conf. on Image Process., Melbourne, Australia, September 2013, pp. 2182–2186
9 Jiang, G., Quan, L.: 'Detection of concentric circles for camera calibration'. Proc. Tenth IEEE Int. Conf. on Computer Vision, Beijing, China, October 2005, pp. 333–340
10 Zhang, B.W., Li, Y.F., Chen, S.Y.: 'Concentric-circle-based camera calibration', *IET Image Process.*, 2012, **6**, (7), pp. 870–876
11 Datta, A., Kim, J.S., Kanade, T.: 'Accurate camera calibration using iterative refinement of control points'. Proc. 12th IEEE Int. Conf. on Computer Vision, Workshops, Kyoto, Japan, September 2009, pp. 1201–1208
12 Yang, G., Peng, F., Zhao, K.: 'Steady corner detection for calibration in underwater environment'. Proc. Third Int. Symp. on Systems and Control in Aeronautics and Astronautics, Harbin, China, June 2010, pp. 97–100
13 Zhu, W., Ma, C., Xia, L., *et al.*: 'A fast and accurate algorithm for chessboard corner detection'. Proc. Second Int. Congress on Image and Signal Processing, Tianjin, China, October 2009, pp. 1–5
14 Chen, D., Zhang, G.: 'A new sub-pixel detector for X-corners in camera calibration targets'. Proc. 13th Int. Conf. in Central Europe on Computer Graphics,

Visualization and Computer Vision, Plzen – Bory, Czech, January 2005, pp. 97–100
15 da Silva Tavares, P.J., Vaz, M.A.: 'Accurate subpixel corner detection on planar camera calibration targets', *Opt. Eng.*, 2007, **46**, (10), pp. 207205-1207205-8
16 Escalera, A., Armingol, J.M.: 'Automatic chessboard detection for intrinsic and extrinsic camera parameter calibration', *Sensors*, 2010, **10**, (3), pp. 2027–2044
17 Ha, J.E.: 'Automatic detection of chessboard and its applications', *Opt. Eng.*, 2009, **48**, (6), pp. 067205-1–067205-8
18 Chu, J., GuoLu, A., Wang, L.: 'Chessboard corner detection under image physical coordinate', Optics & Laser Technology, 2013, **48**, (1), pp. 599–605
19 Harris, C., Stephens, M.: 'A combined corner and edge detector'. Proc. Fourth Alvey Vision Conf., University of Manchester, UK, August 1988, pp. 147–152
20 Smith, S.M., Brady, J.M.: 'SUSAN – a new approach to low level image processing', *Int. J. Comput. Vision*, 1997, **23**, (1), pp. 45–78
21 Bouguet, J.Y.: 'Camera calibration toolbox for Matlab'. Available at http://www.vision.caltech.edu/bouguej/calib_doc, accessed October 2014
22 Laganière, R.: 'OpenCV 2 computer vision application programming cookbook' (Packt Publishing, Birmingham, UK, 2011)
23 Liu, Y., Liu, S., Cao, Y., *et al.*: 'A practical algorithm for automatic chessboard corner detection'. Proc. 21th IEEE Int. Conf. on Image Process., Paris, France, October 2014, pp. 3449–3453
24 Shortis, M.R., Clarke, T.A., Short, T.: 'A comparison of some techniques for the subpixel location of discrete target images'. Proc. SPIE: Videometrics III, . 2350, 1994, pp. 239–250