A PRACTICAL ALGORITHM FOR AUTOMATIC CHESSBOARD CORNER DETECTION

Yu Liu* Shuping Liu* Yang Cao* Zengfu Wang†

* Department of Automation, University of Science and Technology of China, Hefei, China
† Institute of Intelligent Machine, Chinese Academy of Sciences, Hefei, China

{liuyul, fengya}@mail.ustc.edu.cn, {forrest, zfwang}@ustc.edu.cn

ABSTRACT

Chessboard corner detection is a fundamental work of the popular chessboard pattern-based camera calibration technique. In this paper, a fast and robust algorithm for chessboard corner detection is presented. In our method, an initial corner set is obtained with an improved Hessian corner detector. And then, a novel strategy which takes both textural and geometrical characteristics of a chessboard into consideration is employed to eliminate fake corners in the initial corner set. The proposed algorithm only requires a user-input of the total number of chessboard inner corners, while all the other parameters can be adaptively calculated with a statistical approach. Experimental results on two public data sets demonstrate that the proposed method can outperform the most commonly used OpenCV method in terms of both detection rate and computational efficiency.

Index Terms— camera calibration, chessboard corner detection, Hessian corner detector, fake corner elimination

1. INTRODUCTION

Camera calibration is an important technique which has been widely used in many machine vision applications. Recently, various camera calibration methods have been introduced. Among them, the methods using planar patterns, such as circles [1], grid squares [2] and concentric circles [3], are very popular for their practical convenience. In particular, the planar chessboard pattern with black-and-white squares is the most commonly used one for its simplicity and accuracy. All the chessboard pattern-based calibration methods are on the premise of the detection of chessboard inner corners, which are also known as X-corners for their intuitive perception. 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 automatic X-corner detection.

In general, the existing X-corner detection methods can be broadly classified into two categories: texture-based methods [4, 5, 6] and geometry-based methods [7, 8]. The texture-based methods are usually based on keypoint detectors, such as Harris [9], SUSAN [10] and Hessian matrix [6]. In these methods, the response map of the detector is used to decide whether a pixel is a corner with a given threshold. Due to the noise, illumination and clutter, it is always difficult to set the response threshold. As a result, some real X-corners are missed while some fake corners are detected. On the other hand, the geometry-based methods make use of the geometrical characteristics of the chessboard pattern, and their detection results are often more accurate. However, the geometry-based methods tend to be unstable when the distortion of chessboard is severe.

Although some previous publications have noticed that the textural and geometrical characteristics can be combined for corner detection, this topic has not been put sufficient efforts and there is still large room for further improvement.

Currently, there are two camera calibration interfaces which are widely used in academia. The first one is a MATLAB toolbox developed by Bouguet [11]. 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 image, which limits its practicability to a large extent. The second one is a chessboard corner detection function named findChessboardCorners which is offered by the OpenCV library [12]. Unlike the MATLAB toolbox [11], this function only requires users to input the size of chessboard (the number of rows and columns). Generally, the OpenCV method can achieve satisfactory results even in the scene with complex background. However, it usually fails in working when either the area of square is small or the number of X-corners is large. Furthermore, the computational efficiency will significantly decrease when the number of X-corners increases.

In this paper, we present a fast and robust algorithm for automatic X-corner detection. In our algorithm, a Hessian corner detector proposed in [6] is first improved to construct an initial corner set. And then, a novel strategy which takes both textural and geometrical characteristics into account is employed to eliminate fake corners in the initial corner set. The proposed method only requires users to input the total number of X-corners, while all the other parameters can be adaptively calculated with a statistical approach. Experimental results on two public data sets demonstrate that the proposed method can outperform the OpenCV method in terms of both detection rate and computational efficiency. The rest of this paper is organized as follows. The relation to prior work is presented in Section 2. Section 3 describes the detailed detection algorithm. The experimental results are shown in Section 4. Section 5 concludes the paper and puts forward some future work.

2. RELATION TO PRIOR WORK

This work can be viewed as an improved and extended version of the work [6] of Chen and Zhang. In [6], they presented a novel texture-based 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 image and $r(x, y)$ is its Gaussian blur version, the Hessian matrix of $r(x, y)$ is defined as:

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

(1)
where \( r_{xx}, r_{yy} \) and \( r_{xy} \) are the second order derivatives of \( r(x, y) \). As X-corners locate at the centers 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 mathematical tool to detect saddle points in a 2D function, and in [6] the determinant of Hessian matrix is used to construct the response map:

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

For an X-corner, its response should be a negative local minimum in \( S \). Instead of directly searching for the negative local minima in \( S \), the method in [6] searches for the local minima in \( S \) with the following constraint:

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

where \( \lambda_1, \lambda_2 = \frac{1}{2}(r_{xx} + r_{yy} \pm \sqrt{(r_{xx} - r_{yy})^2 + 4r_{xy}^2}) \) 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 can be obtained from all the negative local minima with a fixed response threshold.

Compared with some generalized detectors such as Harris [9] and SUSAN [10], the Hessian detector in [6] is a special approach for the detection of X-corner pattern. However, it also has two main shortcomings:

1. The constraint in Eq. (3) is somewhat too weak, which causes that many pixels in the “flat” regions of an image also satisfy the constraint. As a result, the computation efficiency will significantly decrease. In fact, the \( \lambda_1 \) and \( \lambda_2 \) represent the maximum and minimum of the directional derivatives of the image, respectively. If either the value of \( \lambda_1 \) or \( \lambda_2 \) 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.

2. 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 can be detected with no fake corner mingled. This is mainly because some pixels has the similar local intensity distributions with an X-corner. As fake corners have very bad impacts on the subsequent calibration steps, it is worthwhile to eliminate them in the detection step.

In this paper, we address the above two issues, especially the latter one. The two main contributions are:

1. An improved Hessian detector is presented.
2. A practical approach for fake corner elimination is proposed.

The flowchart of the proposed algorithm is shown in Fig.1.

### 3. THE PROPOSED METHOD

#### 3.1. Improved Hessian detector

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

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

where \( \varepsilon \) is a small positive number. In our algorithm, it is set to 0.03\( \lambda_{\text{max}} \), where \( \lambda_{\text{max}} \) is the maximum value of all the pixels in map \( \lambda_1 \). With this new constraint, most of the pixels in “flat” regions will not be viewed as candidates, thereby the computation efficiency is improved.

#### 3.2. Elimination of fake corners

As shown in Fig.1, three properties of chessboard pattern, named as centrosymmetry property, distance property and angle property, are applied to the elimination task.

##### 3.2.1. Centrosymmetry property

As an example shown in Fig.2, it is easy to see that the local intensity distribution around an X-corner is approximately centrosymmetric. In our algorithm, a circular mask shown in the upper right corner is used to construct the response map:

\[
I(x; y) = \frac{1}{(D_1 + 4D_2 + \cdots + 8D_8)} \sum_{i=1}^{8} I(i \cdot D_1 + i \cdot D_2 + \cdots + i \cdot D_8) \tag{5}
\]

where \( D_1 < p \cdot D_4 \) and \( D_2 < p \cdot D_5 \) or \( D_4 < p \cdot D_6 \) and \( D_5 < p \cdot D_8 \), respectively, where \( p \) is a ratio factor ranging from 0 to 1 and

\[
D_1 = |I_1 - I_5|, \quad D_2 = |I_3 - I_7|, \quad D_3 = |I_1 + I_5 - I_3 - I_7|/2, \quad D_4 = |I_2 - I_6|, \quad D_5 = |I_4 - I_8|, \quad D_6 = |I_2 + I_6 - I_4 - I_8|/2.
\]

For each corner in the initial set obtained from the Hessian detector, it will be eliminated if the criterion in Eq. (5) is invalid. It should be noted that the two terms in Eq. (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.

Fig.2 shows the detection results (denoted as red dots) after applying the centrosymmetry property. We can see that all the real X-corners (such as A, B, P, Q) are preserved, but several stubborn fake corners (C, D, E, F, G, H, K) still exist because their local textures are also approximately centrosymmetric.
3.2. Distance property

We can see from Fig. 2 that for an X-corner in a chessboard, there are at least three neighbor X-corners around it. Therefore, some isolated fake corners can be rooted out if they have less than three neighbors. Therefore, for each candidate corner \( c_i \), the criterion is:

\[
\# \{ c_j | j \in \{1, \ldots, N\}, j \neq i, \| c_i - c_j \|_2 < d \} \geq 3, \tag{7}
\]

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.

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 real X-corner (A, B), this angle is approximately 90° or 180°. While for a fake corner (G, H, K), it is usually a small acute angle. Thus, the criterion is:

\[
\cos \theta < t, \tag{8}
\]

where \( \theta \) is the intersection angle and \( t \) is an angle threshold. If the criterion in Eq. (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.

As shown in Fig. 1, there is an iteration procedure from the distance property to the angle property. 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 simultaneously. 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.

3.3. 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 Eq. (5), the distance threshold \( d \) in Eq. (7), and the angle threshold \( t \) in Eq. (8). To make the method more practical, we present an effective approach to adaptively estimate these parameters.

3.4. Why the elimination approach works

In this subsection, we have a brief review of the elimination approach. The centrosymmetry property is a texture-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...
contains 14 pairs of stereo images, with than the OpenCV method even though our program has not been is large. On the other hand, the proposed method is more efficien-
of the black-and-white square is small and the number of X-corners 18 in the set). Therefore, our method is more robust when the area set), while our method only fails in processing 2 images (No. 5, can only successfully process 4 images (No. 9, 11, 17, 20 in the 28 images. However, for the first image set, the OpenCV method can perfectly accomplish the detection task for all the methods can both perfectly accomplish the detection step. Actually, if all the X-corners are correctly detected, the ordering of X-corners will be an easy task. We first find the four extreme X-corners by comparing the coordinate values and then all the other X-corners can be easily ordered. Since it is not the focus of this work, here we don’t introduce it in detail.

For each data set, the number of successfully detected images and the average computational time for one image (only the successful processed images take part in the calculation, and the two algorithms are implemented on a same platform using C++) are listed in Table 1. We can see that for the second image set, the two methods can both perfectly accomplish the detection task for all the 28 images. However, for the first image set, the OpenCV method can only successfully process 4 images (No. 9, 11, 17, 20 in the set), while our method only fails in processing 2 images (No. 5, 18 in the set). Therefore, our method is more robust when the area of the black-and-white square is small and the number of X-corners is large. On the other hand, the proposed method is more efficien-
t than the OpenCV method even though our program has not been optimized. We can also see that for the OpenCV method, the computational efficiency 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.

Table 1. Performance comparisons between the proposed method and the OpenCV method on the two data sets.

<table>
<thead>
<tr>
<th></th>
<th>Method</th>
<th>Data set 1</th>
<th>Data set 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Successful</td>
<td>OpenCV</td>
<td>4</td>
<td>28</td>
</tr>
<tr>
<td></td>
<td>Proposed</td>
<td>18</td>
<td>28</td>
</tr>
<tr>
<td>Computational Time/s</td>
<td>OpenCV</td>
<td>1.0268</td>
<td>0.1327</td>
</tr>
<tr>
<td></td>
<td>Proposed</td>
<td>0.1053</td>
<td>0.0980</td>
</tr>
</tbody>
</table>

5. CONCLUSION

Contribution- This paper presents a fast and automatic algorithm for chessboard corner detection. An improved Hessian corner detector is first introduced to make the detection process more efficient. Then, by simultaneously considering both textural and geometrical characteristics of a chessboard, we propose a novel approach for fake corner elimination. Furthermore, the proposed algorithm only requires a user-input of the total number of chessboard inner corners, while all the other parameters can be adaptively calculated. Experimental results on two public image sets demonstrate that the proposed method exhibits clear advantages over the popular OpenCV method in terms of both detection rate and computational efficiency.

Limitation- It should be noted that this paper has not exhibited a completed work. First, the detection results of the proposed algorithm in this paper are just at pixel level. Second, this work has not employed the final calibration results to further verify the effectiveness of the proposed detection method.

Future Work- Considering the above limitations, we will extend the proposed algorithm to sub-pixel level and complete the subsequent calibration work to further confirm the effectiveness of our method in the future.

Fig. 5. An example of X-corner detection results. (a) The OpenCV method. (b) The Hessian detector. (c) The proposed method.
6. REFERENCES


