Robust Recognition of Checkerboard Pattern for Deformable Surface Matching in Multiple Views

Weibin Sun  Xubo Yang  Shuangjiu Xiao  Wencong Hu
DALAB Shool of Software
Shanghai Jiao Tong University, Shanghai, China 200240
Email: {skyend, yangxubo, xiaosj, ycguyue}@sjtu.edu.cn

KEYWORDS
Recognition checkerboard deformable surface matching

ABSTRACT
Checkerboard pattern can be used in many computer vision areas by matching the pattern as a surface, such as camera calibration, stereo vision, projector-camera system and even surface reconstruction. However, most existing checkerboard pattern recognition methods only work in planar and fine illuminating circumstances. A robust recognition method for checkerboard pattern is proposed in this paper to deal with those arbitrary surface deformation and complex illumination problems. Checkerboard internal corners are defined as special conjunction points of four alternating dark and bright regions. A candidate corner’s neighbor points within a rectangular or a circular window are treated as in different one-point-width layers. By processing the points layer by layer, we transform the 2D points distribution into 1D to detect corners, which simplifies the regions amount counting and also improves the robustness. After corner detection, the pre-known checkerboard grids rows and columns amounts are used to match and decide the right checkerboard corners. Regions boundary data produced during the corner detection also assist the matching process. We compare our method with existing corner detection methods, such as Harris, SUSAN, FAST and also with the widely adopted checkerboard pattern recognition method, FindChessboardCorners function in OpenCV, to show the robustness and effectiveness of our approach.

INTRODUCTION
Checkerboard pattern (fig. 1) can be useful in many vision systems. Its alternating bright and dark grids and grid corners features can be a very strong features to detect and recognize. In most checkerboard applications, the internal corners (marked by circles in fig. 1), which are conjunctions of every four close grids, are the main features to use. Each checkerboard grid is a solid intensity region where the internal corner is surrounded by four alternating bright and dark regions, so we call these internal corners the region corners in this paper. By detecting and locating region corners of checkerboard pattern surfaces in two scenes, the mapping between those corresponding corners on the deformed surfaces of different scenes can be calculated. The mapping between those two deformed surfaces actually consists of many homographies between the approximate planar quadrangle grids of checkerboard patterns in different scenes. If both of the two scenes are camera views of a static checkerboard pattern on an object, the mapping can be used to model the object in this case, which is actually a modeling technique in stereo vision. If one of those scenes is a camera view of a planar checkerboard pattern and the other is the original checkerboard image, the mapping can be used in camera calibration case (Zhang, 2000). If the checkerboard pattern in the camera view in the second case is projected onto a surface by a projector, the mapping can be the foundation of geometry registration (Brown et al., 2005) in projector-camera augmented reality system (Bimber and Raskar, 2006; Raskar et al., 2002) case. What’s more, in the geometry registration case, by treating each checkerboard grid as a quadrangular patch, we can reconstruct the projection surface, then checkerboard pattern can be a special structure light (Aggarwal and Wang, 1988; Stockman and Hu, 1986) type.

However, to recognize the checkerboard pattern is very difficult on deformable surfaces under natural lighting condition, which are very common environment problems in various computer vision tasks. This paper will deal with these problems.

RELATED WORKS
The recognition process is actually the locating process of checkerboard region corners. All checkerboard region corners should be detected and located without false corners and corners that do not belong to the checkerboard.

To detect corners, (Harris and Stephens, 1988) devel-
oped (Moravec, 1977)'s idea into the famous Plessey corner detector. This method is based on the first-order derivatives and has good behavior with respect to detection. (Smith and Brady, 1997) applied a circular mask to detect corners, that is, the so called SUSAN detector, which has the advantages of being robust to noises and yielding accurate results at a reasonable computation speed. The SUSAN principle is based on the fact that the center pixel should be a corner point if the number of the pixels that have the same brightness as the center pixel in the circular mask is below a threshold. (Rosten and Drummond, 2006)developed their FAST method for corner detection based on their previous works(Rosten and Drummond, 2005). FAST can perform efficient corner detection at high speed, however, its threshold to decide the dark and bright areas in a circle around a candidate corner weaken the robustness against complex illumination. Apart from this method, there are various other ones being proposed in (Trajkovic and Hedley, 1998; Zheng et al., 1999) to find corners or features. But when those methods were applied for checkerboard recognition under complex illumination and deformation conditions, many noise and redundant corner points were detected. Both problems would bring great trouble to identify the real checkerboard corners.

Besides those corner detection approaches, there are some specialized methods for checkerboard recognition. In (Zhang, 2000), the corners are found by intersecting lines. The drawback of this approach is that edges may be in general curved due to radial distortions or deformation. Furthermore, the subsequent ordering of the corners into a regular grid can be complex and unreliable. (Bouguet, 2000) proposed an interactive method to find the internal corners of planar checkerboard pattern image. This tool is especially convenient when working with a large number of images. However, the user needs to click on the four extreme corners on each rectangular checkerboard pattern image in order to calculate the data of the corners. The OpenCV function, cvFindChessBoardCorner, which is widely used, can do automatic corner extraction, but the algorithm fails rather often under complex illumination and deformation. (Wang et al., 2007) proposed an approach to automatically recognize and locate the internal target corners of the planar checkerboard pattern image. The proposed approach is based on the characteristics of local intensity and the grid line architecture of the planar checkerboard pattern image. (Shu et al., 2003) proposed a method, which is based on the algorithm of (Watson, 1981), that exploits the topological structure of the checkerboard pattern. The main idea is to use Delaunay triangulation (Bern and Eppstein, 1992) to connect the corner points. It can deal with different lighting conditions but also only planar checkerboard pattern.

In this paper, we propose a robust automatic method that can recognize checkerboard pattern under complex illumination and deformation. The main idea is to treat a certain point's neighbor points within a rectangular or a circular window as in different one-point-width layers and transform the 2D points distribution into 1D to detect regions. Corners are correlated and clustered by the region boundary data to recognize the checkerboard corners.

**NOTATIONS AND MOTIVATION**

Let \( I \) be the image containing the checkerboard pattern.

\[
I = \{\vec{p}\}, \quad \vec{p} = [x, y] \text{ denotes the point in } I. \quad (1)
\]

For a vector point \( \vec{p} \), we use \((x_p, y_p)\) to represent its Cartesian coordinates, \((\rho_p, \theta_p)\) to represent its polar coordinates. In a grayscale image, we make \( I (\vec{p}) \) or \( I (x_p, y_p) \) be the intensity of point \( \vec{p} \), \( I_B (\vec{p}) \) or \( I_B (x_p, y_p) \) be the intensity of point \( \vec{p} \) in a binary image.

A checkerboard internal corner, which we call it the region corner(fig. 2(a)), has four alternating dark and bright regions around it. We define a rectangular and a circular window covering the four regions feature surrounding a candidate corner \( \vec{p} \) to be \( R(\vec{p}, w) \) and \( C(\vec{p}, w) \) in eq. 2.

\[
R(\vec{p}, w) = \{ p_i | |p_i - x_p| \leq w, |y_p - y_p| \leq w \} \\
C(\vec{p}, w) = \{ p_i | ||p_i - \vec{p}|| \leq w \} \quad (2)
\]

\( w \) here is always an non-negative integer limit the window’s scope. \( R(\vec{p}, w) \) is a set of points that are in a rectangular, actually a \( 2w + 1 \) points width square window(we still call it rectangular to be more general in the rest of this paper). \( C(\vec{p}, w) \) is a set containing the points around \( \vec{p} \) within a circular window whose radius is \( w \) points and the center is \( \vec{p} \). The circular window can be generated by a Bresenham (Bresenham, 1977) circle when implementing to be efficient and accurate.

The four alternating dark and bright regions around a corner can be deformed seriously on arbitrary surface. The isotropy against deformation can be achieved if points within a window are iterated by circumambulating the corner from the outer to the inner. The layer is defined in eq. 3 to represent the points being checked in each circumambulating iteration.

\[
\mathbb{L}_r(\vec{p}) = \langle \vec{p}_1, \ldots, \vec{p}_i, \ldots, \vec{p}_n \rangle, \quad i \in [1, n_r] \\
\text{for circular window } \vec{p}_i \in C(\rho, r) - C(\rho, r - 1) \\
\text{for rectangular window } \vec{p}_i \in R(\rho, r) - R(\rho, r - 1) \\
\theta_{\vec{p}_j} < \theta_{\vec{p}_{j+1}} \quad j \in [1, n_r - 1] \quad (3)
\]

In a window \( \mathbb{L}(p, w) \) or \( \mathbb{C}(p, w) \), there are \( w \) layers from \( \mathbb{L}_1(\vec{p}) \) to \( \mathbb{L}_w(\vec{p}) \). \( n_r \) is the points amount in \( \mathbb{L}_r(\vec{p}) \). For a \( \mathbb{L}_r(\vec{p}) \) in \( \mathbb{L}(p, w) \), \( n_r \) can be calculated by eq. 4.

\[
n_r = 4(2r + 1) - 4 = 8r \quad (4)
\]

For circular window layers \( \mathbb{L}_r(\vec{p}) \) in \( \mathbb{C}(p, w) \), the Bresenham circle algorithm will decide the \( n_r \). To be used later, we define the first order derivative in a layer \( \mathbb{L}_r(\vec{p}) \) to be:

\[
I_L(\vec{p}_i) = I(\vec{p}_{i+1}) - I(\vec{p}_i) \quad (5)
\]

Note that the coordinate index in \( \mathbb{L}_r \) should be \( i \) in \( \vec{p}_i \), this would make the coordinates in \( \mathbb{L}_r \) be 1D. A layer
Regions are detected by checking whether there are four alternating dark and bright regions around a candidate corner within a window scope. To be efficient and robust, firstly, the image I is resized to a range \([I_{S_{\text{min}}}, I_{S_{\text{max}}}]\) without changing the width and height ratio. According to the common camera resolutions from \(320 \times 240\) to \(2048 \times 1536\), the range is set to be \([300, 200), (2100, 1600)\]. There is no difference between the rectangular window and the circular one to achieve isotropy against deformations when circumambulating the points in layers around the candidate corner. We will use the rectangular one in rest of this paper.

The window size parameter, \(w\) in \(R(\vec{p}, w)\) is decided by the checkerboard grid size. To detect the four alternating dark and bright regions, the window should not cover more than four regions, which are actually four grids surrounding a corner. So window width \(2w + 1\) should be less than two times of the grid width (we only consider checkerboard with square grid) \(g_{\text{width}}\). To be efficient, we decide the two windows of two neighbor corners belonging to the same grid should not intersect too much. So \(2w + 1 < g_{\text{width}}\) and \(w \approx g_{\text{width}}/2\). We also define a window size range \([W S_{\text{min}}, W S_{\text{max}}]\) to limit the window size for extreme large or small grid. \(g_{\text{width}}\) can be calculated by the amount of checkerboard grids rows \(g_r\) and columns \(g_c\) and the image size \(w_{\text{height}}\) and \(w_{\text{height}}\) according to eq. 7 to be a probable value.

\[
g_{\text{width}} = \min\left(\frac{w_{\text{height}}}{g_r}, \frac{w_{\text{height}}}{g_c}\right) \tag{7}
\]

Here \(g_r\) and \(g_c\) should be known before recognition. eq. 8 shows the calculating of the window size parameter \(w\).

\[
w = \frac{w'}{2}, w' \text{ is calculated by eq. 9.} \tag{8}
\]

\[
w' = \begin{cases} W S_{\text{min}} & \text{if } g_{\text{width}} < W S_{\text{min}} \\ g_{\text{width}} & \text{if } W S_{\text{min}} \leq g_{\text{width}} \leq W S_{\text{max}} \\ W S_{\text{max}} & \text{if } g_{\text{width}} > W S_{\text{max}} \end{cases} \tag{9}
\]

The window size parameter does not need high precision, we just ignore the 1 in \(2w + 1\) window width in eq. 8. The window size range \([W S_{\text{min}}, W S_{\text{max}}]\) reflects the minimum scope in which a region corner can be identified. The region borders around a corner may be smoothed or extended to blocks rather than border lines to lose sharpness because of noise, low camera capturing quality and complex lighting. To get robustness, the window should be able to cover all four regions to detect them. However, too large window size will cause low performance. We set this range be \([11, 21]\) to tolerate the low quality corners in most common camera cases. The \([W S_{\text{min}}, W S_{\text{max}}]\) with value of \([11, 21]\) limits the window widths from 11 to 21 of \(R(\vec{p}, w)\). This range may be changed when implementing, however, a points block with over 21 points long width should not be recognized to be a border line even in human’s eyes, so for most applications, the range can be fixed to our values without manual change.

For each layer, a mean value of the layer points is calculated to do thresholding as in eq. 10. Let \(L_r^B(\vec{p})\) be the
the 2D to 1D transformation. An example is shown in fig. 4.

A checkerboard corner should have four regions, so \( \alpha \) should be less than \( \frac{1}{4} \). To avoid incorrect noise reduction, layers with small width parameter \( r < 2 \) will not do this operation. What's more, because the last point and the first point of a layer in 1D form are actually continuous in 2D form, a region may break here because of the 2D to 1D transformation. An example is shown in fig. 5. These broken regions may be treated as fake regions when noise reducing because of their short widths. To avoid this problem, we modify the binary morphology operations by elongating the layer to repeat it one more time behind its last point to be cycle-compatible, these modified operations are called the ring-morphology as in eq. 13.

Dilation: \( A \circ \text{ring} B = \{ x \mod |A| \mid (B)_x \cap (A \cup (A)|A|)^c \neq \emptyset \} \)

Erosion: \( A \bullet \text{ring} B = \{ x \mod |A| \mid (B)_x \cap (A \cup (A)|A|)^c \neq \emptyset \} \)

Opening: \( A \sqcap \text{ring} B = (A \circ \text{ring} B) \bullet \text{ring} B \)

Closing: \( A \bullet \text{ring} B = (A \circ \text{ring} B) \circ \text{ring} B \) (13)

The ring-morphology treats a 1D image as a ring that the head and the end are joined. By performing ring-morphology opening and closing on the binary layers \( I^B_{L'}(\vec{p}) \) after thresholding, we can reduce the fake regions. Then regions count can be calculated by summing absolute value of the first order derivatives \( I^r_{L'}(\vec{p}_r) \) of this binary(only 0 and 1 values) layer. To avoid the similar region breaking problem, we let the first order derivative \( I^r_{L'}(\vec{p}_r) \) at \( \vec{p}_r \) of a layer \( L^B_{L'}(\vec{p}) \) to be:

\[
I^r_{L'}(\vec{p}_r) = I_B(\vec{p}_i) - I_B(\vec{p}_{n_r})
\]

In checkerboard pattern, one layer should have 4 regions. If all layers of a candidate corner have 4 regions, this candidate should be a corner. However, to get high robustness, we define an acceptance threshold value \( \alpha \) to allow some noises. If there are \( \alpha \) or more portions of layers containing 4 regions, the candidate corner is accepted to be a checkerboard corner. Too large \( \alpha \) will cause too many noise corners while too small \( \alpha \) also reduces the robustness. This acceptance threshold value can be determined by the acceptance degree of human’s eyes and image noise degree. We set it to be 0.7 to tolerate 30% noise layers in a window for common camera images.

Till now the corner detection result including the region corners is produced. Let the result be \( \mathbb{N} \). Moreover, the regions boundaries positions of a corner are recorded for the later checkerboard match step. Since there are more than one layers within a corner’s surrounding window, only the most outer layer is recorded. We represent the region boundary of a corner \( \vec{p} \) within its window \( \mathbb{R}(\vec{p}, w) \) to be(note that the derivative \( I^r_{L'}(\vec{p}) \) is on the thresholded binary layer \( L^B_{L'}(\vec{p}) \)):

\[
B_{\vec{p}} = \{ \vec{p}_i \mid |I^r_{L'}(\vec{p}_i)| = 1, \forall \vec{p}_i \in L^B_{w_{max}}(\vec{p}) \}
\]

\[
w_{max} = \max \{ w_j \mid w_j \in [1, w], \sum_{\vec{p}_k \in L^B_{I^r_{L'}(\vec{p})}} |I^r_{L'}(\vec{p}_k)| = 4 \}
\]

CHECKERBOARD MATCH

The corner detection step can produce noise corners and redundant corners because of the acceptance threshold
Because mass centers of some clusters may be not in solved by calculating the mean region boundary positions not recorded in corner detection. This problem can be the region boundary positions of those mass centers are.

When doing noise reduction on the corner detection result $N$, each corner $\vec{c}'$’s neighbors within a window $\mathbb{R}(\vec{c}', w)$ is checked to see whether there are enough corners in the window. If there are enough (more than a given $\tau$) corners, all corners within the window are reserved, otherwise the checked corner is removed. This operation is similar to binary erosion, we also need a $(2w_{\text{se}} + 1) \times (2w_{\text{se}} + 1)$ structure element $SE_{cr}$ to perform the neighbors checking. By defining $SE_{cr}$ in eq. 16, the refined result $N'$ can be calculated by eq. 17, in which we also use the translation operation and set cardinality defined in eq. 6.

$$SE_{cr} = \{\vec{p}_{i,j} \mid i \in [1, 2w_{\text{se}} + 1], j \in [1, 2w_{\text{se}} + 1], x_{\vec{p}_{i,j}} = i - w_{\text{se}} - 1, y_{\vec{p}_{i,j}} = j - w_{\text{se}} - 1\}$$  \hspace{1cm} (16)

$$N' = \{\vec{p} \mid |(SE_{cr}) \cap N| \geq \tau, \forall \vec{p} \in N\}$$ \hspace{1cm} (17)

The $\tau$ to reduce noise in eq. 17 should be no larger than $w_{\text{se}}$. We define $w_{\text{se}}$ to be 2 to check a $5 \times 5$ neighbors patch and $\tau$ to be 2 to eliminate the isolated noisy corner, which is the only one corner within that neighbors patch when $\tau$ is 2.

Redundant corners are reduced by clustering. Corners result $N'$ after noise reduction is clustered according to their distances between each other. The result $N'$ is treated to be a nodes set and each corner in $N'$ is a node. If two corners have a distance no larger than $SE_{cr}$’s width $2w_{\text{se}} + 1$, there is an edge connecting them. By defining nodes and edges, we get an undirected graph $G'$ (eq. 18).

$$G' = (N', E')$$

$$E' = \{(\vec{p}_i, \vec{p}_j) \mid \|\vec{p}_i - \vec{p}_j\| \leq 2w_{\text{se}} + 1, \forall \vec{p}_i, \vec{p}_j \in N'\}$$ \hspace{1cm} (18)

Each connected component $G'_{c}(N'_{c}, E'_{c})$ in $G'$ is a cluster. For each cluster $G'_{c}$, the mass center or the point $\vec{p}_{c}$ with minimum distances sum to other points of the same cluster is calculated and made to be the new corner to replace others in the cluster. $\vec{p}_{c}$ is calculated by eq. 19. There may be more than one $\vec{p}_{c}$, just pick a random one when implementing.

$$\vec{p}_{c} : \vec{p}_{c} \in N'_{c} \text{ and } \sum_{\vec{p}_j \in N'_{c}} \|\vec{p}_j - \vec{p}_{c}\| = \min\{\sum_{\vec{p}_k \in N'_{c}} \|\vec{p}_k - \vec{p}_{c}\| \mid \forall \vec{p}_k \in N'_{c}\}$$ \hspace{1cm} (19)

Because mass centers of some clusters may not be in $N'$, the region boundary positions of those mass centers are not recorded in corner detection. This problem can be solved by calculating the mean region boundary positions of corners in a cluster. $\vec{p}_{c}$ does not have this problem although the mass center is always more accurate in position than $\vec{p}_{c}$ to represent the cluster’s position. So when implementing, $\vec{p}_{c}$ will be an efficient selection. We let $N''$ be the corners result after redundant corners reduction.

After noise and redundancy reduction, the region boundary positions data of each found corner are used to calculate the connectedness of corners. We also use the graph theory to assist this process. Each corner in $N''$ is a node. An edge connecting two corners will exist if those two corners are on a same region boundary line. An edge actually connects two checkerboard corners sharing the same boundary of one checkerboard grid. These two corners are neighbor corners of a grid. The edge set is defined by eq. 20.

$$E'' = \{(\vec{p}_i, \vec{p}_j) \mid \exists \vec{p}_k \in \mathbb{B}_{\vec{p}_i}, \vec{p}_j \in \mathbb{B}_{\vec{p}_j} :$$

$$\|\vec{p}_i - \vec{p}_k\|, \|\vec{p}_j - \vec{p}_k\|, \|\vec{p}_j - \vec{p}_i\| \in [-\epsilon, \epsilon]\}$$ \hspace{1cm} (20)

$\epsilon$ here should be very close to 1. $[-\epsilon, \epsilon]$ defines an acceptance range to tolerate the inaccurate region boundaries. The inaccuracy is mainly caused by the serious deformation within a checkerboard grid. If two corners $\vec{p}_i$ and $\vec{p}_j$ are on a same boundary of a checkerboard grid, $[-\epsilon, \epsilon]$ defines the cosine value (calculated by $\|\vec{p}_i - \vec{p}_k\|, \|\vec{p}_j - \vec{p}_k\|$) range of the angle between two region boundaries vectors: $(\vec{p}_i - \vec{p}_k), (\vec{p}_j - \vec{p}_k)$, here $\vec{p}_k$ and $\vec{p}_j$ are on the same grid boundary on which $\vec{p}_i$ and $\vec{p}_j$ locate. eq. 20 ignores the relative position of the two corners when deciding edges. In most application cases such as geometry registration, surface reconstruction, checkerboard usually has large amount of grids and a grid is treat to be a plane without deformation and distortion, deformation and distortion only occur among different grids. However, considering other cases that can tolerate the non-planar grid and to be robust under serious deformation, we decide $\epsilon$ to be $-0.8$ to tolerate a $\pm \arccos(-0.8) \approx \pm 36^\circ$ bending deformation of straight boundaries within a grid.

Now the graph $G'' = (N'', E'')$ is produced. With the assistance of checkerboard grids rows $g_r$ and columns $g_c$, we can find the checkerboard pattern corners by finding the connected component $G''_{c}$ that has exact $(g_r - 1) \cdot (g_c - 1)$ nodes. If there is the only one $G''_{c}$ having $(g_r - 1) \cdot (g_c - 1)$ nodes, it is the set that contains all the right checkerboard corners.

**RESULT AND COMPARISON**

We compare our method with Harris, SUSAN and FAST corner detector to evaluate our corner detection of checkerboard region corners. We also compare our method with FindChessboardCorner function in OpenCV to evaluate our recognition of the checkerboard pattern.

Our comparison focuses on the corner detection robustness and checkerboard pattern recognition correctness. To evaluate the robustness against complex illum-
We adjust this value to 10 and 20 old illumination in our test images (fig. 6), FAST with thresh-
demo supplied by the authors of FAST. Under complex environment light. General cameras always have seri-
ous color incorrectness that they can not restore the exact color of the object it captured, which will cause various noises. fig. 7 shows our comparison results.

For Harris and SUSAN, both of them fail to deal with the darker area in fig. 7(b) and 7(a) while too many noises and redundance are produced in fig. 7(g) and 7(f).

FAST method’s threshold weakens the robustness against complex illumination significantly. We use both the default value 20 and an adjusted value 10 in the FAST demo supplied by the authors of FAST. Under complex illumination in our test images (fig. 6), FAST with threshold 20 will miss many region corners (fig. 7(e) and 7(j)). We adjust this value to 10, however, it will produce many redundant and noise corners (fig. 7(d) and 7(i)). It is difficult to distinguish the four different corners on the boundaries of a grid.

OpenCV’s FindChessboardCorner will do thresholding on the global image to transform the dark and bright grids into black and white. This thresholding can always fail even with adaptive method under complex illumination. The results in fig. 7(c) and 7(h) show the failed detection.

fig. 7(k) and 7(l) show that our method can find all checkerboard region corners and match them within the checkerboard pattern successfully while other methods fail. fig. 6(a) and 6(b) are just two representative images of a large number of checkerboard images captured by general cameras in our test. We test our method with plenty of checkerboard images captured by general cameras under complex illumination and deformation and find it is robust to recognize the checkerboard pattern correctly in those images. We also present some of them (fig. 7(m), 7(n), 7(o)) in this paper. The circumambulating iteration around a corner can get isotropy against deformation. Thresholding in 1D layers locally will get robustness against various illumination. The fake region reduction can ensure the correctness of corner detection. The acceptance threshold of layers and other parameters with the values defined in our method according to most general cases will reduce the effects caused by noise or deformation. Noise reduction in checkerboard match and the redundant corner reduction by clustering can ensure the corners without redundance and noise. Then corners connected by region boundaries will produce the right checkerboard corners set.

CONCLUSION AND FUTURE WORKS

We present a robust recognition method for checkerboard pattern. This method addresses the problem of recognizing checkerboard pattern under complex illumination and deformation in computer vision systems to do camera calibration, stereo vision, geometry registration or surface reconstruction. The checkerboard internal corners (region corners) surrounded by four alternating dark and bright regions and their boundaries are the features to find in our recognition. By detecting corners with the help of our own robust region corner detection approach and then do checkerboard matching mainly by clustering, our method can be automatic and robust with the parameters values we defined according to most general cases rather than manually-setting parameters. The experiment results in (Section EVALUATION) show that our method can deal with most common checkerboard images captured by general cameras.

To achieve the most robustness, we detect the region corners and match checkerboard pattern by recognizing and using almost all their features which can ensure the robustness but costs much time. Now our algorithm is $O(mn)$ time complexity ($m$ for window size, $n$ for image size). In future, we will focus on the speed. Some features may be simplified to save time without losing robustness.

ACKNOWLEDGEMENTS

This work is sponsored by 863 National High Technology R&D Program of China (No.2006AA01Z307) and National Natural Science Foundation of China (No.60403044).

REFERENCES


(a) SUSAN result, fail to find all region corners.
(b) Harris result, fail to find all region corners.
(c) OpenCV FindChessboardCorner result, fail to find all region corners and match checkerboard.
(d) FAST with threshold 10. Too many redundant and noise corners to match checkerboard.
(e) FAST with default threshold 20, fail to find all region corners.
(f) SUSAN result, fail to find all region corners.
(g) Harris result, fail to find all region corners.
(h) OpenCV FindChessboardCorner result, fail to find all region corners and match checkerboard.
(i) FAST with threshold 10. Too many redundant and noise corners to match checkerboard.
(j) FAST with default threshold 20, fail to find all region corners.
(k) Our method result, success.
(l) Our method result, success.
(m) Our method result on other image, success.
(n) Our method result on other image, success.
(o) Our method result on other image, success.

Figure 7: Comparison Result on fig. 6(a), 6(b) and some of our method results on other images. Corners are marked by blue cross. Red circles mark the checkerboard pattern’s region corners. SUSAN and Harris corners are marked by red circle. FAST corners are marked by red pixels. (Enlarge images to see detail.)


**AUTHOR BIOGRAPHIES**

**Weibin Sun** was born in 1984’s China. His email is skyend@sjtu.edu.cn and his personal webpage at http://dalab.se.sjtu.edu.cn/endysun.

**Xubo Yang**’s email is yangxubo@sjtu.edu.cn.

**Shuangjiu Xiao**’s email is xiaosj@sjtu.edu.cn.

**Wencong Hu**’s email is ycgyue@sjtu.edu.cn.