Optical Engineering 47(5), 057008 (May 2008)

Corner detector based on global and local curvature properties

Xiao Chen He
University of Hong Kong
Department of Computer Science
Pokfulam Road, Hong Kong
E-mail: xche@cs.hku.hk

Nelson H. C. Yung
University of Hong Kong
Department of Electrical and Electronic Engineering
Pokfulam Road, Hong Kong

Abstract. This paper proposes a curvature-based corner detector that detects both fine and coarse features accurately at low computational cost. First, it extracts contours from a Canny edge map. Second, it computes the absolute value of curvature of each point on a contour at a low scale and regards local maxima of absolute curvature as initial corner candidates. Third, it uses an adaptive curvature threshold to remove round corners from the initial list. Finally, false corners due to quantization noise and trivial details are eliminated by evaluating the angles of corner candidates in a dynamic region of support. The proposed detector was compared with popular corner detectors on planar curves and gray-level images, respectively, in a subjective manner as well as with a feature correspondence test. Results reveal that the proposed detector performs extremely well in both fields. © *2008 Society of Photo-Optical Instrumentation Engineers.* [DOI: 10.1117/1.2931681]

Subject terms: corner detection; adaptive threshold; region of support; curvature; contour; round corner; obtuse corner.

Paper 070884R received Oct. 30, 2007; revised manuscript received Mar. 10, 2008; accepted for publication Mar. 13, 2008; published online May 22, 2008.

1 Introduction

Corners in images represent critical information in describing object features that are essential for pattern recognition and identification. There are many applications that rely on the successful detection of corners, including motion tracking, object recognition, and stereo matching.[1–3] As a result, a number of corner detection methods have been proposed in the past.

Kitchen and Rosenfeld[4] proposed a corner detection scheme based on a differential operator that determines the first and second partial derivatives of an image, from which corners are identified as local extrema. This method is sensitive to noise, and suffers from missing junctions and poor localization. Moravec[5] observed that the difference between adjacent pixels of an edge or a uniform part of the image is small and at the corner the difference is significantly high in all directions. The idea was later used by Harris[6] to develop the Plessey algorithm. This method provides good repeatability under rotation and various illuminations, and is often used for stereo matching and image database retrieval. Unfortunately, it is sensitive to quantization noise and suffers from a loss in localization accuracy, particularly at certain junction types. Smith and Brady[7] proposed a detector named SUSAN using a circular mask for corner and edge detection. Although SUSAN's corner localization and noise robustness are better than those of the previously mentioned algorithms, it is time-consuming in obtaining an area (called the USAN) and finding corners in large windows. Another formulation of USAN was proposed in Ref. 8, in which two oriented cross operators, called crosses as oriented pair (COP), were used instead of circular mask as in Ref. 7.

0091-3286/2008/$25.00 © 2008 SPIE

Other corner detectors are described in Ref. 9–23 . In summary, most of them are single-scale detectors and work well if the image has similar-size features, but are ineffective otherwise. As a result, either the fine or the coarse features are poorly segmented, which is unacceptable because natural images normally contain both kinds of features.

To alleviate this problem, Rattarangsi and Chin[24] proposed a multiscale algorithm based on curvature scale space (CSS), which can detect corners of planar curves. Although it can detect multiple-size features, the algorithm is computationally intensive due to parsing features are the entire scale space. Moreover, it detects false corners on circles. Some other multiscale approaches do not check all the scales, e.g., the technique for smoothing a curve adaptively based on its roughness in the region, as proposed in Ref. 25. Given that the CSS technique is suitable for recovering invariant geometric features of a planar curve at multiple scales,[26] Mokhtarian et al. proposed two CSS corner detectors[27,28] for gray-level images. These CSS detectors perform well in corner detection and are robust to noise, but they have problems too.

To begin with, we quote the definition of curvature, K, from Ref. 26 as follows:

$$K(u,\sigma)=\frac{\dot{X}(u,\sigma)\ddot{Y}(u,\sigma)-\ddot{X}(u,\sigma)\dot{Y}(u,\sigma)}{[\dot{X}(u,\sigma)^2+\dot{Y}(u,\sigma)^2]^{1.5}}, \tag{1}$$

where $\dot{X}(u,\sigma)=x(u)\otimes\dot{g}(u,\sigma)$, $\ddot{X}(u,\sigma)=x(u)\otimes\ddot{g}(u,\sigma)$, $\dot{Y}(u,\sigma)=y(u)\otimes\dot{g}(u,\sigma)$, $\ddot{Y}(u,\sigma)=y(u)\otimes\ddot{g}(u,\sigma)$, and $\otimes$ is the convolution operator, while $g(u,\sigma)$ denotes a Gaussian of width σ, and $\dot{g}(u,\sigma)$, $\ddot{g}(u,\sigma)$ are the first and second derivatives of $g(u,\sigma)$, respectively. The following steps are used by the original CSS algorithm[27]to detect corners of an image:

1. Apply Canny edge detection to the gray-level image, and obtain a binary edge map.
2. Extract edge contours from the edge map. When the edge reaches an end point, fill the gap and continue the extraction if the end point is nearly connected to another end point, or mark this point as a T-junction corner if the end point is nearly connected to an edge contour, but not to another end point.
3. From each contour, compute curvature values at a high scale, σ_{high}. Then consider the local maxima as initial corners whose absolute curvatures are above the threshold t and twice as high as one of the neighboring local minima; t in this case is selected manually.
4. Track the corners from the highest scale to the lowest scale to improve the localization error.
5. Compare the T-junction corners with other corners, and remove one of any two corners that are close to each other.

There are a number of problems associated with this algorithm. Firstly, a single scale is used in determining the number of corners (step 3), and multiple scales are used only for localization. Not surprisingly, it misses true corners when σ_{high} is large and detects false corners when σ_{high} is small. If the algorithm is applied to a complex image, this effect becomes more prominent, and choosing an appropriate σ_{high} becomes challenging. Secondly, as local maxima of the absolute curvature function make up the set of corner candidates, a corner candidate can be a true corner, a rounded corner, or noise. Mokhtarian and Suomela in Ref. 27 asserted that the curvature of a true corner has a higher value than that of a round corner or noise, but in practice it is very easy to find a corner due to noise that has higher curvature value than an obtuse corner. Thirdly, the performance of the algorithm depends on the selection of threshold value t, the proper value of which may change from image to image, or even from one edge contour to another. Lastly, tracking is performed to improve localization by computing curvature at a lower scale and examining the corner candidates in a small neighborhood of previous corners. When multiple corner candidates exist in the small neighborhood, the corners may be mismatched. This situation is likely to result in a poor localization performance.

The enhanced CSS algorithm[28] dealt with some of these problems, by using different scales of the CSS for contours with different lengths, and smoothing the curvature function for long contours to remove false maxima. However, the criterion for selecting contour lengths is not explicit. Such a criterion is obviously important, for it determines the success of the algorithm. On the other hand, it is reasonable to believe that the meaningful scale value does not necessarily depend on the contour length. The contour length is not a major attribute of a curve, since the algorithm for edge contour extraction can alter it. In fact, different-size features, which need different scales, can exist on the same contour. Although the enhanced CSS offers better results than the original CSS, there is much room for improvement.

Our survey revealed that corner detection involves extracting corners in gray-level images and also in digital curves, which can be extended to gray-level images by edge detection and contour extraction. The former approach regards a corner as an individual feature, and detects corners only according to their local properties (curvature, gradient magnitude, etc.), while the latter approach has the potential to detect corners according to their global properties by considering the relationship between neighboring features in the contours. Another observation is that conventional corner detectors are not able to distinguish round corners from obtuse corners. Broadly, a round corner is a point on an arc, which has the highest curvature among the points on the arc, but the curvature differences between these points are small. On the other hand, an obtuse corner, whose absolute curvature may be similar to that of a round corner, always has a prominent point whose curvature is significantly larger than the curvature of its neighboring points. Obtuse corners are much more valuable and useful for representing the shape of objects than round corners, but they often are not appropriately distinguished by existing corner detectors. Furthermore, there is no explicit criterion to distinguish round corner and obtuse corner.

To summarize, the goals of this paper are: (1) to consider corners to be defined by global and local curvature properties, (2) to distinguish round corners from obtuse corners, and (3) to parameterize the approach.

This paper proposes a new and improved corner detection method, of which a preliminary version has been described in Ref. 29. It relies on an edge map from which absolute curvature is computed at a relatively low scale to retain almost all corners, true or false. All the local maxima of the absolute curvature function are regarded as corner candidates. We assume that true corners are completely included in this set of corner candidates, together with some false corners. This assumption is only true when the edge map is extracted using a low threshold and the scale used is low enough. In fact, both conditions are easy to achieve.

Since a local maximum may represent a true corner, a round corner, or simply noise,[27] two criteria are adopted to remove the latter two from the initial list of corner candidates To do that, we first compare the corner candidates using an adaptive local threshold (automatically calculated) instead of a single global threshold to remove the round corners. Second, the angles of the remaining corner candidates are evaluated to eliminate any false corners due to quantization noise and trivial details. The evaluation is based on a dynamic region of support, which varies from corner to corner according to adjacent corner candidates. We also introduce an end-point handling method to ensure that end points are appropriately dealt with.

The proposed detector has been tested and evaluated over a number of images with multiple-size features and compared with popular corner detectors on planar curves as well as on gray-level images. It is found that the proposed method outperforms the rest and is more consistent from image to image.

In Sec. 2, our proposed corner detection method is presented in detail. Section 3 depicts and discusses the experiment results. The conclusions are presented in Sec. 4.

2 Proposed Method

2.1 *Overview*

Traditional single-scale algorithms (e.g., Refs. 4 and 7) detect corners by considering their local properties, and either miss fine features or detect noise as false corners. The philosophy of the proposed method is to utilize global and local curvature properties, and balance their influence when extracting corners. With this philosophy and the problems of traditional corner detectors in mind, a new corner detector is proposed as follows:

1. Detect edges using the likes of a Canny edge detector to obtain a binary edge map.
2. Extract contours as in the CSS method.
3. After contour extraction, compute the curvature at a fixed low scale for each contour to retain the true corners, and regard the local maxima of absolute curvature as corner candidates.
4. Compute a threshold adaptively according to the mean curvature within a region of support. Round corners are removed by comparing the curvature of corner candidates with the adaptive threshold.
5. Based on a dynamically recalculated region of support, evaluate the angles of the remaining corner candidates to eliminate any false corners.
6. Finally, consider the end points of *open* contours, and mark them as corners unless they are very close to another corner. Open and closed contours are defined by Eq. (3).

2.2 *Initial List of Corner Candidates*

Let us first define the j'th extracted contour as

$$A^j = \{P_1^j, P_2^j, \dots, P_N^j\}, \tag{2}$$

where $P_i^j = (x_i^j, y_i^j)$ are pixels on the contour, N is the number of pixels on the contour, and x_i^j, y_i^j are the coordinates of the i'th pixel on the j'th contour. We further define the contour as *closed* if the distance between its end points is small enough, and otherwise *open*:

$$A^j \text{ is } \begin{cases} \text{closed} & \text{if } |\overline{P_1^j P_N^j}| < T, \\ \text{open} & \text{if } |\overline{P_1^j P_N^j}| > T, \end{cases} \tag{3}$$

where the threshold T is used to determine whether two end points are close enough. A typical value of T is 2 or 3 pixels.

For a closed contour, circular convolution can be applied directly to smooth the contour. For an open contour, however, a certain number of points should be symmetrically compensated at both ends of the contour when it is smoothed. The contour convolved with the Gaussian smoothing kernel g is denoted by

$$A_{\text{smooth}}^j = A^j \otimes g, \tag{4}$$

where g is a digital Gaussian function with width controlled by σ. A value $\sigma=3$ has been used in all the experiments presented in this paper. After that, the curvature value of each pixel of the contour is computed using

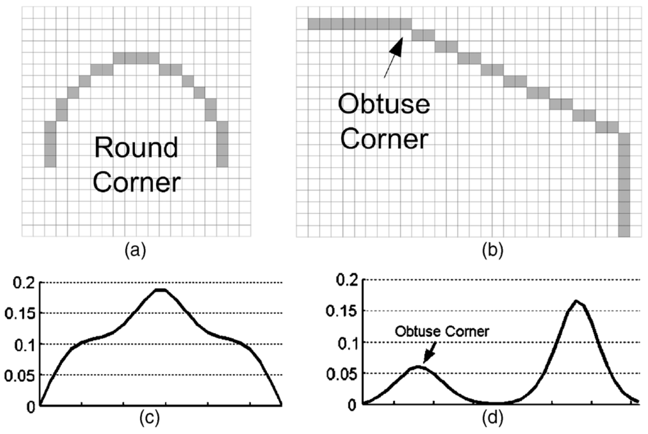


Fig. 1 Examples of round corner and obtuse corner: (a) round corner, (b) obtuse corner, (c) curvature plot of round corner in (a), (d) curvature plot of obtuse corner in (b).

$$K_i^j = \frac{\Delta x_i^j \Delta^2 y_i^j - \Delta^2 x_i^j \Delta y_i^j}{[(\Delta x_i^j)^2 + (\Delta y_i^j)^2]^{1.5}} \quad \text{for } i = 1, 2, \dots, N, \tag{5}$$

where $\Delta x_i^j = (x_{i+1}^j - x_{i-1}^j)/2$, $\Delta y_i^j = (y_{i+1}^j - y_{i-1}^j)/2$, $\Delta^2 x_i^j = (\Delta x_{i+1}^j - \Delta x_{i-1}^j)/2$, and $\Delta^2 y_i^j = (\Delta y_{i+1}^j - \Delta y_{i-1}^j)/2$. From Eq. (5), all the local maxima of the curvature function are included in the initial list of corner candidates.

2.3 *Corner Evaluation*

2.3.1 *Round-corner removal*

As defined in Sec. 1, although the curvature of a round corner is the largest among its neighbors, the actual difference may not be significant, as depicted in Fig. 1(a) and 1(c) On the other hand, the curvature of an obtuse corner [Fig. 1(b)] may have similar or even lower absolute maximum than a round corner. Its magnitude is often significantly larger than its neighbors', and its neighbors' overall, or global, curvature characteristics usually vary more abruptly, as depicted in Fig. 1(d). In order to utilize this global curvature characteristic of the neighbors to eliminate round corners yet not the obtuse corners, we define the term *region of support* (ROS).

In this section, the ROS of a corner is defined by the segment of the contour bounded by the corner's two nearest curvature minima. The ROS of each corner is used to calculate a local threshold adaptively, where u is the position of the corner candidate on the contour, L_1+L_2 is the size of the ROS centered at u, and R is a coefficient:

$$T(u) = R \times \bar{K} = R \times \frac{1}{L_1 + L_2 + 1} \sum_{i=u-L_2}^{u+L_1} |K(i)|, \tag{6}$$

where $\bar{K}$ is the mean curvature of the ROS. If the curvature of the corner candidate is larger than $T(u)$, then it is declared a true corner; otherwise it is eliminated from the list. The reason why this can eliminate round corners is that for an obtuse corner, the curvature drops faster over L_1+L_2 then does that of a round corner over a similar ROS. As a result, the mean curvature of an obtuse corner is smaller than that of a round corner. A round corner tends to have absolute curvature smaller than $T(u)$, while an obtuse corner tends to have absolute curvature larger than $T(u)$, even if their absolute curvatures are similar. Obviously, the truth of this depends on how R is selected.

In theory, by controlling R appropriately we should be able to differentiate round corners from obtuse corners and eliminate various kinds of round corners as well. However, round corners are ill defined by nature, and there is no explicit criterion to distinguish them. For instance, every point on a circle has the same curvature, and a circle has no obvious corner. However, for an ellipse, it could be argued that the vertices may be considered as true corners. Therefore, whether a round corner should be regarded as a true corner is determined by how round or sharp it is. So, it is worthwhile investigating the relationship between the nation of round corner and the coefficient R.

Suppose an ellipse is given by $f(x)=[b^2-(bx/a)^2]^{1/2}$, with $x\in[-a,a]$ and $b>a$. The vertex $(0,b)$ of the ellipse will be a curvature maximum and therefore a likely true corner. The absolute curvature of every ellipse point can be calculated as

$$K(x)=\left|\frac{f'(x)}{[1+f'(x)^2]^{3/2}}\right|=\frac{ba^4}{[(bx)^2-(ax)^2+a^4]^{3/2}},$$

and we have $K_{\max}=K(0)=b/a^2$, $K_{\min}=K(a)=a/b^2$. Because the area under the curvature function is given by

$$\int K(x)\,\mathrm{d}x=\int\frac{ba^4}{[(bx)^2-(ax)^2+a^4]^{3/2}}\,\mathrm{d}x$$
$$=\frac{bx}{[(b^2-a^2)x^2+a^4]^{1/2}},$$

we have the mean curvature given by $\bar{K}=\int_{-a}^{a}\kappa(x)\,\mathrm{d}x/2a$ $=[1-(-1)]/2a=1/a=K_{\max}\cdot a/b$, and the adaptive threshold is given by $T=R\times\bar{K}=R\cdot K_{\max}\cdot a/b$ From this equation, it can be deduced that

$$K_{\max}\begin{cases}>T & \text{if } b/a>R,\\ <T & \text{if } b/a<R.\end{cases}\qquad(7)$$

In other words, for the vertex of an ellipse, if the ratio of its major axis to its minor axis is lower than R, it is to be regarded as a round corner. Given this relationship, we are able to use R to define the round corners to be eliminated, and filter them out from the initial list of corner candidates.

2.3.2 *False-corner removal*

Generally speaking, a well-defined corner should have a relatively sharp angle. As argued in Ref. 11, if we knew the angle of each corner on a contour, it would be easier to differentiate true corners from false corners. The key to the success of this approach is to correctly define the angle of a corner In particular, the angle of a corner can be an ambiguous quantity that varies according to its definition and the extent over which the angle is defined. For instance, Fig. 2 depicts five points labeled on a curve, all of which represent local curvature maxima and can be regarded as corners. Taking point 3 as an example, if the angle of a corner is defined as an acute angle, point 3 will fall within

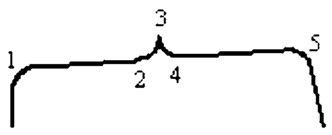


Fig. 2 Illustration of an ambiguous case.

this definition. Moreover, if we consider point 3 within the range of points 2 and 4, then it will be classified as a true corner too. However, if we consider point 3 within the range of points 1 and 5, then points 2, 3, and 4 may all be classified as false corners, the reason being that points 1 and 5 form almost a straight line, which indirectly implies points 2, 3, and 4 are the result of some local variations on the curve. When the global curvature characteristic of a contour is not known *a priori*, it can be challenging to decide on the range over which a potential corner candidate should be considered. This motivates us to propose a method for determining an appropriate range when evaluating potential corner candidates based on our previously defined ROS.

In Sec. 2.3.1, we defined the ROS as the segment of the contour bounded by the corner's two nearest curvature minima. In this section, we extend this definition to include the two neighboring corners of the corner in question. Using the same illustration in Fig. 2, if all five points labeled are corner candidates after round-corner removal, then point 3 will have a new ROS spanning from points 2 to 4 and will be classified as corner, because its angle is acute. On the other hand, points 2 and 4 might potentially be removed after round-corner removal, and as a result, the new ROS for point 3 would span from point 1 to 5. In this case, point 3 would likely be classified as a false corner, because its angle would be obtuse. Furthermore, some corners do not have two neighboring corners, such as the corner nearest to an end point of an open contour (points 1 and 5). End points of an open contour are used as additional corners to define the ROS.

After determining the ROS of corner candidates, the angle of a corner candidate can be defined as that between the lines joining the corner point concerned and the two centers of mass on both sides of the ROS,[29] where the center of mass is defined as the mean position of all the pixels on one arm of the ROS. This definition enables the removal of point 3 on a straight line as depicted in Fig. 2. However, it fails when dealing with local variations along an arc, as depicted in Fig. 3, which is illustrated in the following. After round-corner removal, points C as depicted in Fig. 3 would most likely be considered as potential true corners. The ROS of point C is then defined over points E and F according to our definition, where points E and F could be corner candidates or end points. Based on the definition of angle in Ref. 29 [Fig. 3(a)], $\angle C$ may not be obtuse enough to be considered for removal. It does not help if the arc extends longer (larger ROS), for $\angle C$ would then become sharper.

To alleviate this problem, we redefine the angle of a corner using tangents instead. For any point in the arc, the tangent directions on its two sides form an angle at that point. Similarly, a straight line can be regarded as an arc with infinite radius of curvature, so the tangent direction of

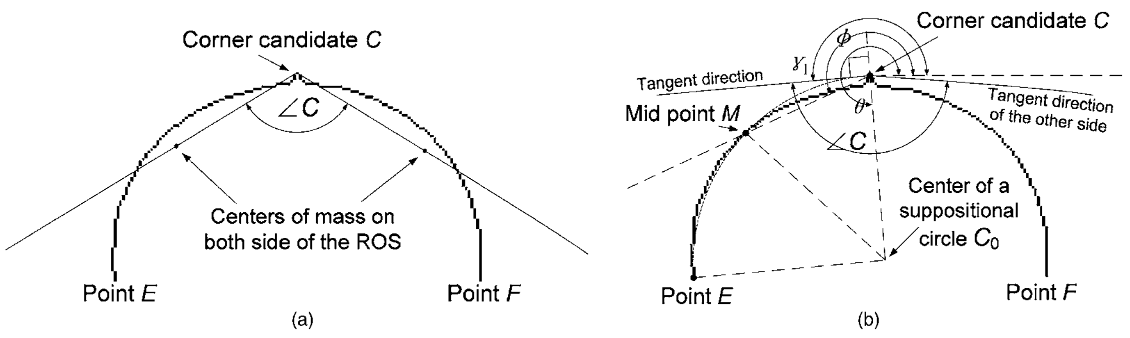


Fig. 3 Angle definitions of a corner: (a) center-of-mass definition, (b) tangent definition.

any point on the line is the same as the line direction. In this respect, straight lines and arcs can be treated in exactly the same way. To calculate the tangent, a circle is best-fitted to the pixels on each arm of the ROS of the corner candidate, as shown in Fig. 3(b). The traditional way is to minimize the mean squared Euclidean distance from the circle to the pixel points. Unfortunately, there is no closed-form solution for that.[30] All known algorithms involve either approximation or costly iteration.[31] Because optimal fitting is unnecessary in this case, a simple three-point method is employed to determine the circle. This three-point method is detailed below, with reference to Fig. 3(b).

First, on one arm of an ROS (from C to E, say), three points (C, the mid point M, and E) are selected. If these three points are collinear, the tangent direction of this ROS arm is defined from C to E else the center of a suppositional circle C_0 is deduced as follows, which has the same distance (radius of curvature of this ROS) to the three points. Let $C=(x_1,y_1)$, $M=(x_2,y_2)$, $E=(x_3,y_3)$, and $C_0=(x_0,y_0)$; we have

$$x_0=\frac{(x_1^2+y_1^2)(y_2-y_3)+(x_2^2+y_2^2)(y_3-y_1)+(x_3^2+y_3^2)(y_1-y_2)}{2\cdot[(x_1(y_2-y_3)+x_2(y_3-y_1)+x_3(y_1-y_2)]},$$

$$y_0=\frac{(x_1^2+y_1^2)(x_2-x_3)+(x_2^2+y_2^2)(x_3-x_1)+(x_3^2+y_3^2)(x_1-x_2)}{2\cdot[(y_1(x_2-x_3)+y_2(x_3-x_1)+y_3(x_1-x_2)]}. \tag{8}$$

Second, a line is drawn from C to C_0, and θ is used to represent the direction from C to C_0, which could be calculated by a four-quadrant inverse-tangent function. Similarly, we use ϕ to denote the direction from C to M. Then we can have the tangent of C at this side of ROS as follows:

$$\gamma_1=\theta+\text{sign}(\sin(\phi-\theta))\cdot\frac{\pi}{2}, \tag{9}$$

where sign is a signum function. Third, the tangent of the ROS from C to F is determined similarly, and is denoted by γ_2. Fourth, the two tangent lines form the angle of the corner:

$$\angle C=\begin{cases}|\gamma_1-\gamma_2| & \text{if } \quad |\gamma_1-\gamma_2|<\pi,\\ 2\pi-|\gamma_1-\gamma_2| & \text{otherwise.}\end{cases} \tag{10}$$

Finally, the corner checking criterion is given as follows:

$$\begin{aligned}&C_i \text{ is true corner} \quad \text{if } \angle C_i\leq\theta_{\text{obtuse}},\\ &C_i \text{ is false corner} \quad \text{if } \angle C_i>\theta_{\text{obtuse}}.\end{aligned} \tag{11}$$

The parameter θ_{obtuse} designates the maximum obtuse angle that a corner can have and still be considered as a true corner.

False corners are marked and removed after all corner candidates have been checked. Because the set of corner candidates will change after this step, further iterations are performed until there are no further possible changes of the corner list. The number of iterations is typically two or three. Using this criterion, isolated corner candidates due to quantization noise and trivial details can be eliminated, but the dominant corners are retained.

2.3.3 *Advantage of considering global properties*

Traditional single-scale methods detect corners only according to their local properties. They are ineffective for objects with multiple-size features. We find that the global property of curvature can be used to determine a more appropriate ROS for accurate detection. For example, the round corner in Fig. 1(a) and the contour vertex in Fig. 4(a) have similar local curvature. Taking a global view, the former is more an arc than a corner, while the latter can be regarded as a true corner. The absolute curvature values of the maxima of these two contours are also similar, but the latter has a larger region of support, and therefore a lower mean curvature. So the contour vertex in Fig. 4(a) has the tendency to be detected as a true corner by using the proposed method. Furthermore, by using the angle definition over a dynamic ROS as described in Sec. 2.3.2, the latter will be evaluated as a sharper angle and more likely be regarded as true corner than the former.

Another example is depicted in Fig. 4(c): an image contains two contours having three maxima each, as shown. Among these corner candidates, C_1 and C_4 have similar local properties. From a global view, C_1 is a dominant point, because it represents the shape of the contour. On the contrary, C_4 is more of a trivial detail in the whole contour, since it is unimportant in representing the shape of the contour. By checking the angle in the self-determined ROS, the longer the contour is, the less likely it will be regarded as a true corner.

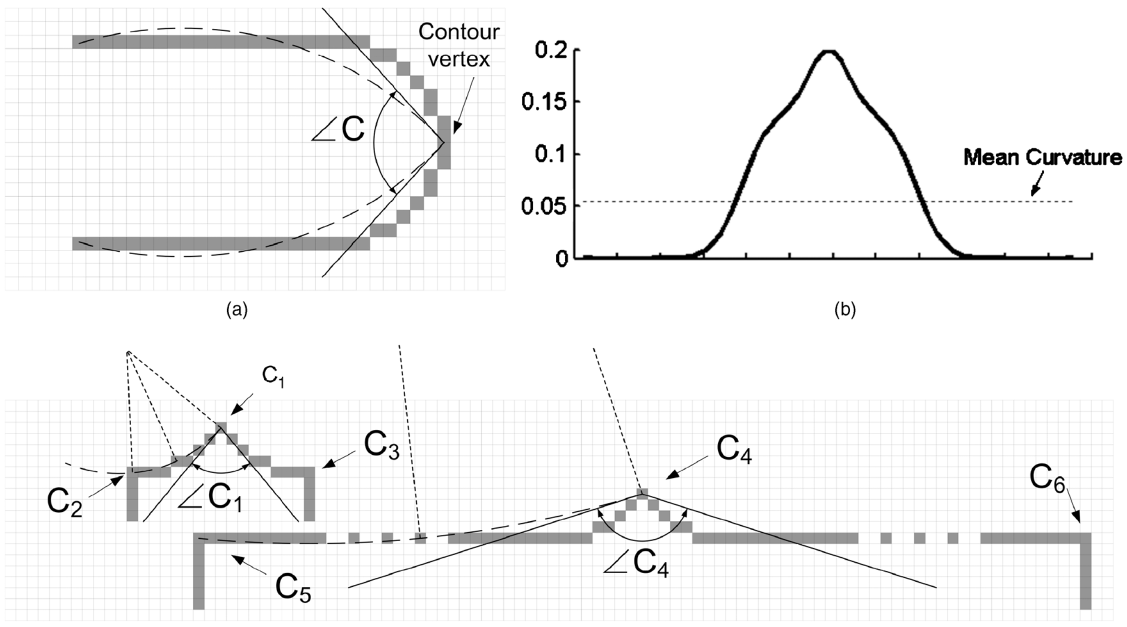


Fig. 4 Feature of different sizes: (a) extended version of round corner; (b) curvature plot of (a); (c) corners with similar local properties but different global properties.

2.4 *End-Point Consideration*

In general, a closed contour does not have end points. The end points of an open contour, on the other hand, are peculiar points. In the original CSS algorithm,[27] if an end point is nearly connected to an edge contour, it is regarded as a T junction and marked as a true corner. Extending this idea, we argue that even if an end point of an open contour is not close to any edge contours, it should be considered as a true corner.[32] Therefore, in the proposed method, at the final stage, the end points of open contours are checked, and are marked as true corners unless they are very close to another true corner, in which case one of them will be eliminated. In the implementation depicted in the following section, a 5×5 neighborhood was used to define closeness.

3 Experiment and Parameter Analysis

In this section, detection results in each stage of the proposed method are presented first, and then its performance is compared with popular detectors on planar curves as well as on gray-level images. A feature correspondence test gives an objective evaluation of the proposed method by comparing it with other popular detectors. The final subsection discusses the computational time requirements.

3.1 *Results at Different Stages*

To illustrate how the proposed method works, an image with a large number of acute, obtuse, and round corners is used for the test. Figure 5 depicts the processed images at various stages. Figure 5(a) depicts the Canny edge map. Figure 5(b) depicts the initial corner candidates. After applying round-corner removal based on an adaptive threshold, corner candidates with curvature similar to their neighborhood are eliminated, and the result is shown in Fig. 5(c). The round corners in the arcs at the right-hand center of the image have all been eliminated, such as corner candidate 1. However, at the top image boundary, many false corners are still present. By checking the angle of corner candidates, these false corners (due to noise and local variations) can also be eliminated, and the result is shown in Fig. 5(d). By treating end points of open contours as true corners, we obtain the final detection result as shown in Fig. 5(e). In Fig. 5(f), corner candidate 1 is an example of a round corner; it appeared only in the initial corner stage, and was eliminated at the round-corner removal stage. Another example is corner candidate 2: As a trivial feature in the middle of a long straight line, it was eliminated at the false-corner removal stage through angle checking, and it was recovered at the end-point stage as a corner at a T junction. This is why it is included in Fig. 5(c) and 5(e), but not in Fig. 5(d).

3.2 *Test Results on Planar Curves*

To evaluate the performance of the proposed method on planar curves, we have chosen some published test shapes of different sizes and features from Chetverikov and Szabo[10] and selected the two best-performing detectors, BT87 and IPAN99, from their comparative work on the following corner detectors: RJ73,[11] RW75,[12] FD77,[13] BT87,[14] and IPAN99.[10] We then compared them with ACORD[25] and our proposed method; the test results are shown in Fig. 6. It should be noted that in Fig. 6(a)–6(c), parameter values are tuned to get the best result of each shape. On the other hand, in Fig. 6(d), the same parameter values are used for the proposed method to process the whole set of input shapes. Among all these detectors, BT87 tends to miss some of the obvious true corners, as can be seen in the rightmost shape in the second row of Fig. 6(a), and is not able to suppress some of the round corners, as depicted in shape second from the right in the first row of Fig. 6(a). On the other hand, IPAN99 can suppress round

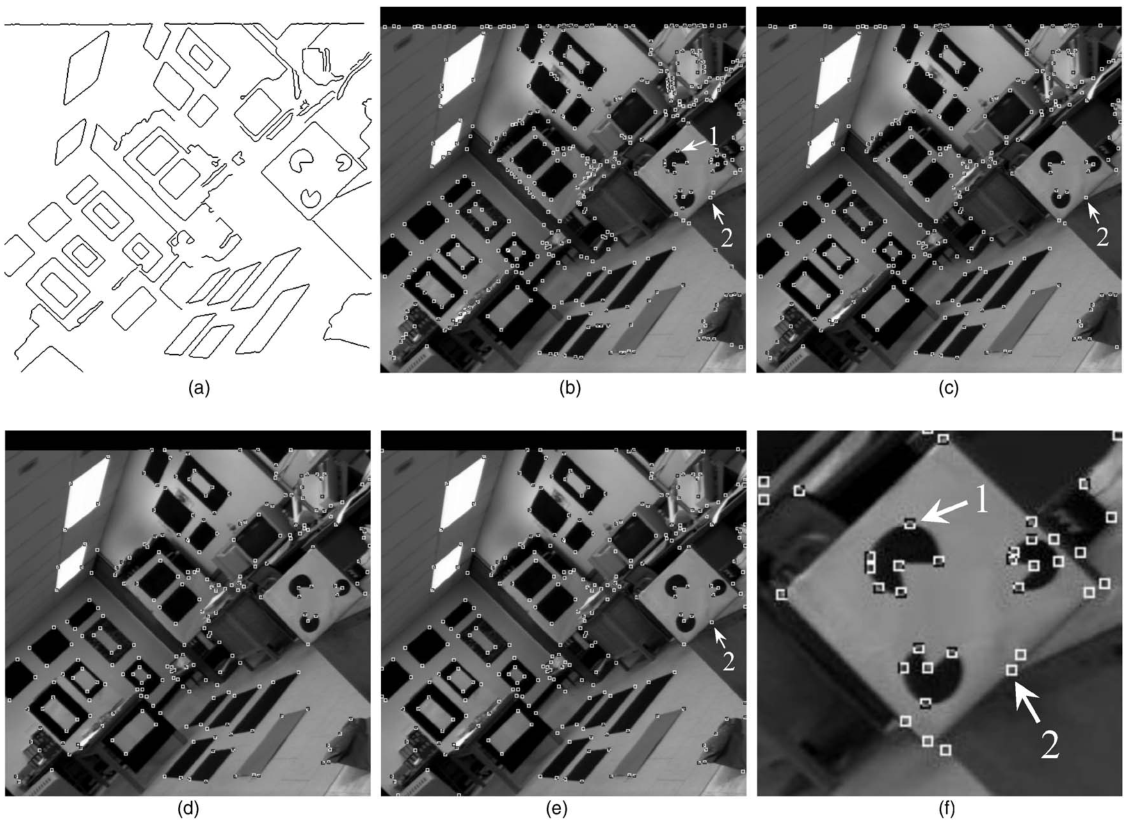


Fig. 5 Corner detection of the "Lab" image by the proposed method: (a) detected edge contours, (b) initial corner candidates, (c) round corners removed, (d) other false corners removed, (e) end-point consideration, (f) magnified image of (b).

corners effectively in simple shapes. However, in order to keep obtuse corners, it has to adjust parameters, causing it to detect local variations (false corners) as corners. This can be seen in the shape second from the left in the second row of Fig. 6(b). The ACORD method in this case suffers from multiple detections around a corner point, which can be seen in the shape second from the right in the first row of Fig. 6(c). The proposed method has the best corner detection performance visually, in that it detects almost all the dominant features, including the fine features of the plane's engines [rightmost shape in the second row of Fig. 6(d)] and suppresses noise and round corners. Although it has

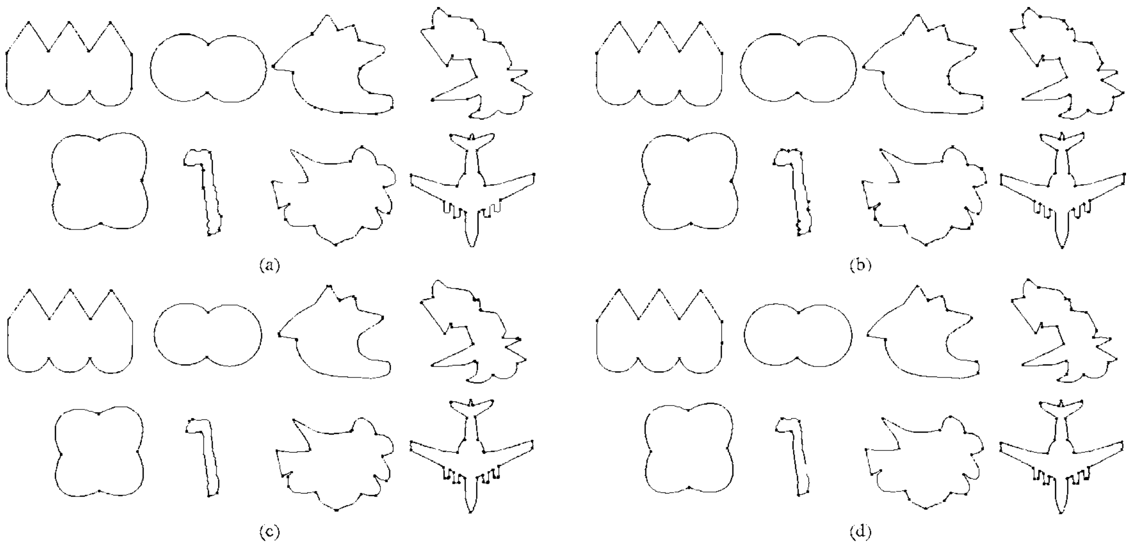
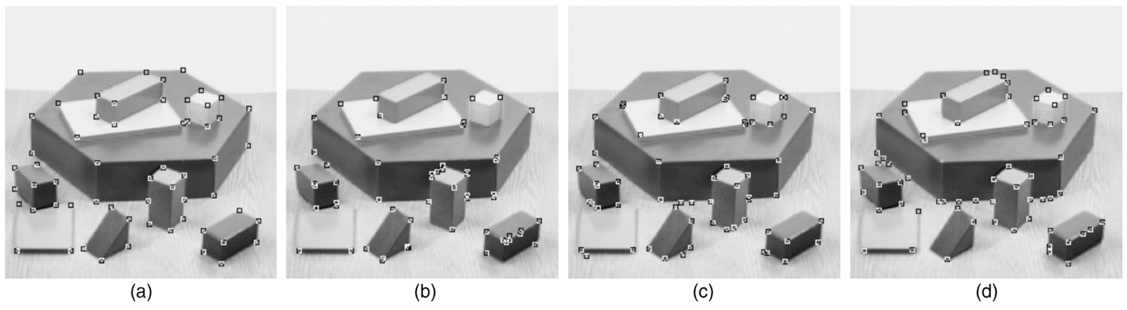


Fig. 6 Detection results on planar curves: (a) BT87, (b) IPAN99, (c) ACORD, (d) proposed method with $R=1.5$, $\theta_{\text{obtuse}}=162$.



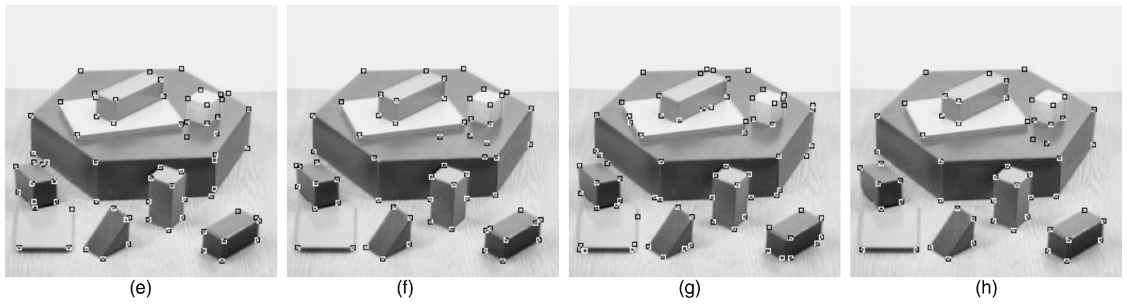


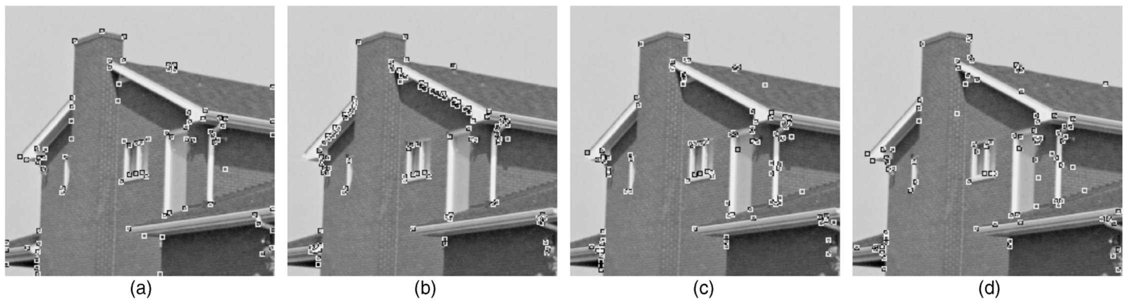
Fig. 7 Detection results on the "Blocks" image: (a) reference solution, (b) Plessey, (c) Kitchen-Rosenfeld, (d) SUSAN, (e) original CSS, (f) enhanced CSS, (g) COP, (h) proposed method.

detected a false corner in the leftmost shape in the first row of Fig. 6(d), this incorrect detection can be removed by adjusting R and θ_{obtuse} appropriately. In summary, the proposed detector works quite well on different-size features using the same R and θ_{obtuse}.

3.3 *Test Results on Gray-Level Images*

In this subsection we have chosen Mokhtarian et al.[27,28,33] comparison as our evaluation basis and extended it further. To perform an objective evaluation, a reference solution for each test image is needed. In our research the reference solutions were manually generated, and corners were identified in an appropriately magnified version of the image. Since it is often difficult to decide whether or not a point should be classified as a corner, only the very obvious corners are included in the reference solutions. Figure 7(a) and 8(a) depict the reference corner solutions of the "Blocks" and "House" images, respectively.

The method of evaluation adopted in this research is described as follows: Let C_{REF}, C_{DET} denote the set of cor-



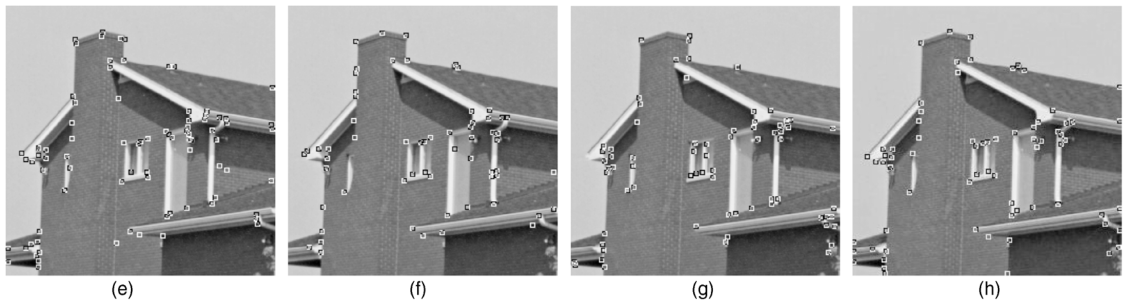


Fig. 8 Detection results on the "House" image: (a) reference solution, (b) Plessey, (c) Kitchen-Rosenfeld, (d) SUSAN, (e) original CSS, (f) enhanced CSS, (g) COP, (h) proposed method.

Table 1 Evaluation results for the "Blocks" image.

Detector	True corners	Missed corners	False corners	Localization error
Plessey	39	20	19	1.6045
Kitchen-Rosenfeld	48	11	14	1.5180
SUSAN	44	15	19	1.5453
Original CSS	55	4	15	1.8502
Enhanced CSS	54	5	9	0.9755
COP	51	8	26	1.6918
Proposed	**55**	**4**	**2**	**1.4010**

ners from the reference solution and the set of corners found by a particular detector, respectively. Let $d_{i,j}$ be the distance difference between the i'th corner in the reference list (C_i) and the j'th corner (C_j) in the detected list. If $d_{i,j}$ between C_i and C_j is minimum for $\forall i,j$, and if $d_{i,j} \leq D_{\max}$, then C_j is labeled as a *correct* detection of C_i, and is also termed a *true* corner; otherwise, C_i is labeled as a *missed* corner. Here $D_{\max}$ is defined to be the maximum admissible distance difference between C_i and C_j. In other words, the localization error can be up to $D_{\max}$, which is set to 4 pixels in the following evaluation. The corners labeled as missed in C_{REF} are considered as true corners not detected, and the remaining corners in C_{DET} are considered as *false* corners. The localization error is calculated as the mean of all the distances $d_{i,j}$ for those correctly detected corners.

Using this evaluation method, the detection results of the proposed method are compared with those six other corner detectors (Plessey,[6] Kitchen-Rosenfeld,[4] SUSAN,[7] original CSS,[27] enhanced CSS,[28] and COP[8]). The results are summarized in Tables 1 and 2 and in Figs. 7 and 8. From Table 1, the number of true corners is 59 in the "Blocks" reference solution. The proposed method detected 55 true corners (representing 93%), while it has the least number of false corners (only 2), although the localization error is only the second best, behind the enhanced CSS method. The two CSS methods performed quite well too, with the original CSS performing ever so slightly better than the enhanced CSS on true corners, but with a lot more false corners (15 versus 9). On the other hand, the original CSS has almost twice as much localization error as the enhanced CSS, which has the largest error in this group of detectors. The other corner detectors performed substantially poorer in that their percentage of success for true corners ranges from 68% to 86%, with many more false corners detected. Similar results are shown in Table 2. Due to the complexity of the "House" image, all the corner detectors had a lower success rate in detecting true corners (the best is 82% for the proposed method, followed by the original CSS at 78%). The false-corner detection rate has also gone up for all the detectors, except for COP. The fact remains that the proposed method has only 5 false corners, while the next best is the enhanced CSS with 12 false corners. The localization error of the proposed method is just under 1 pixel on average, while the rest are substantially higher. In summary, the proposed method remains the most successful in detecting true corners, with the least number of false corners and amount of localization error. Both sets of results highlight the importance of considering the local and global curvature properties of corners and achieving a balance between the two in detection.

Table 2 Evaluation results for the "House" image.

Detector	True corners	Missed corners	False corners	Localization error
Plessey	53	28	50	1.5773
Kitchen-Rosenfeld	59	22	36	1.6561
SUSAN	60	21	29	1.8428
Original CSS	63	18	18	1.5285
Enhanced CSS	49	32	12	1.3251
COP	52	29	18	1.8980
Proposed	**66**	**15**	**5**	**0.9901**

3.4 *Objective Measure Based on Feature Correspondence*

The results in Secs. 3.2 and 3.3 are subjective in that they illustrate the similarity between a detector's and a human's view of what a corner is. In this section, we consider a more objective measure by applying the proposed algorithm to a feature correspondence system and comparing the feature matching rate with those of SUSAN,[7] GFtT,[34] and SIFT.[35]

As depicted in Fig. 9, with two different frames in a real traffic scene as input, our testing system extracts corners of a target vehicle in the near frame, and then finds their correspondence in the far frame. Correspondences are sought by hierarchical block-based matching algorithms (HBMAs), scaling the target vehicle in the far frame to the same size as the one in the near frame, and inversely scaling after matching to fit its original position in the far frame. It should be noted that since the SIFT algorithm includes feature matching as well as feature extraction, the preceding feature correspondence procedure is not necessary for it. The difficulty of the correspondence problem is that shadows or reflections on vehicle surfaces could change the vehicle's appearance significantly, especially in regions of highly reflective material, such as the windscreen and windows of a vehicle. These features may be detected and classified as features of a vehicle, since substantial edges or junctions may appear in the captured image even in homogeneous region. However, since light sources may be continually changing and vehicles are also moving, a vehicle's appearance may be quite different between two consecutive images. Those features due to shadows or reflections are not stable enough to provide accurate matching.

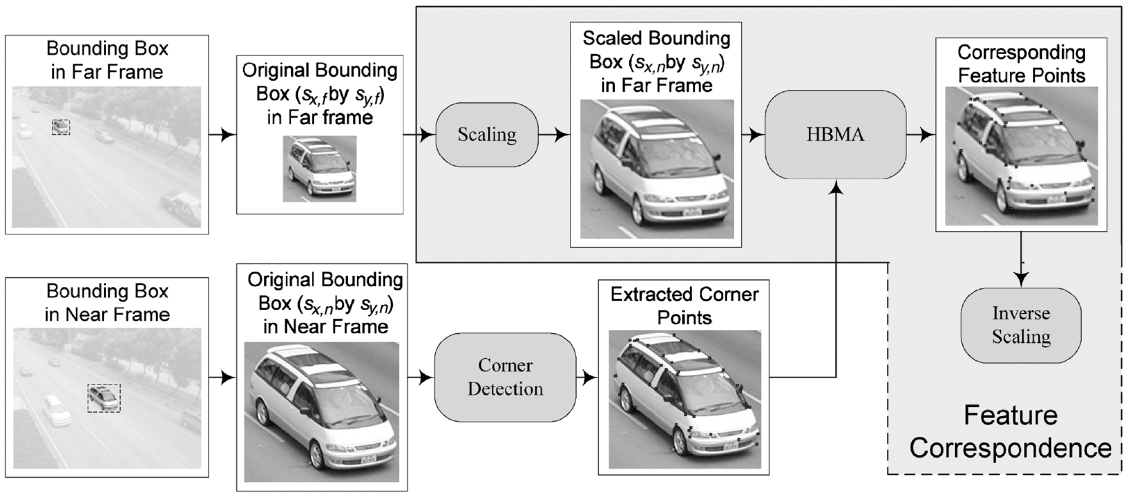

Bounding Box in Far Frame
Original Bounding Box ($s_{x,f}$ by $s_{y,f}$) in Far frame
Scaling
Scaled Bounding Box ($s_{x,n}$ by $s_{y,n}$) in Far Frame
HBMA
Corresponding Feature Points
Inverse Scaling
Bounding Box in Near Frame
Original Bounding Box ($s_{x,n}$ by $s_{y,n}$) in Near Frame
Corner Detection
Extracted Corner Points
Feature Correspondence

Fig. 9 Diagram of feature correspondence system.

In this experiment, the correspondence results of different detectors are checked manually and labeled as match or mismatch; obvious localization errors are also labeled as mismatch. Then the matching rate is calculated, which is defined as the ratio of matched features to detected features, and is used to evaluate the performance of the four feature extraction algorithms in this system. Parameters of the SUSAN and GFtT were tuned to include features that represent the main structure of vehicle and exclude features in homogeneous regions, which tend to mismatch. The brightness threshold of SUSAN was set as 45; the quality level and minimum distance of GFtT were set as 0.2 and 5 respectively. The SIFT function has many parameters; the default values were chosen to emulate Lowe's original implementation. The parameters of the proposed corner detector were set as the defaults, that is, $R=1.5$ and $\theta_{\text{obtuse}}=162$ deg. Usually, the proposed corner detector is applied to the Canny edge map. In this experiment, it was also applied to the boundaries of a segmentation result[36] to enhance the matching performance.

The experiment was performed on 16 vehicles in different traffic scenes. Typical feature correspondence examples are presented in Fig. 10, and correspondence results are summarized in Table 3.

SUSAN detected the most features in this experiment. But some of these features lie on the straight part of a boundary [e.g., points 12, 13, 66 in Fig. 10(a)], and some others lie on the vehicle windows or windscreen [e.g., points 10, 19 in Fig. 10(a)]; these represent most of the mismatched features due to SUSAN. GFtT exhibited very promising performance in this experiment: As shown in Fig. 10(b), only a few features have localization error [e.g., points 30 and 32 in Fig. 10(b)]. SIFT seems unsuitable for this kind of data, for it detected the least features, and

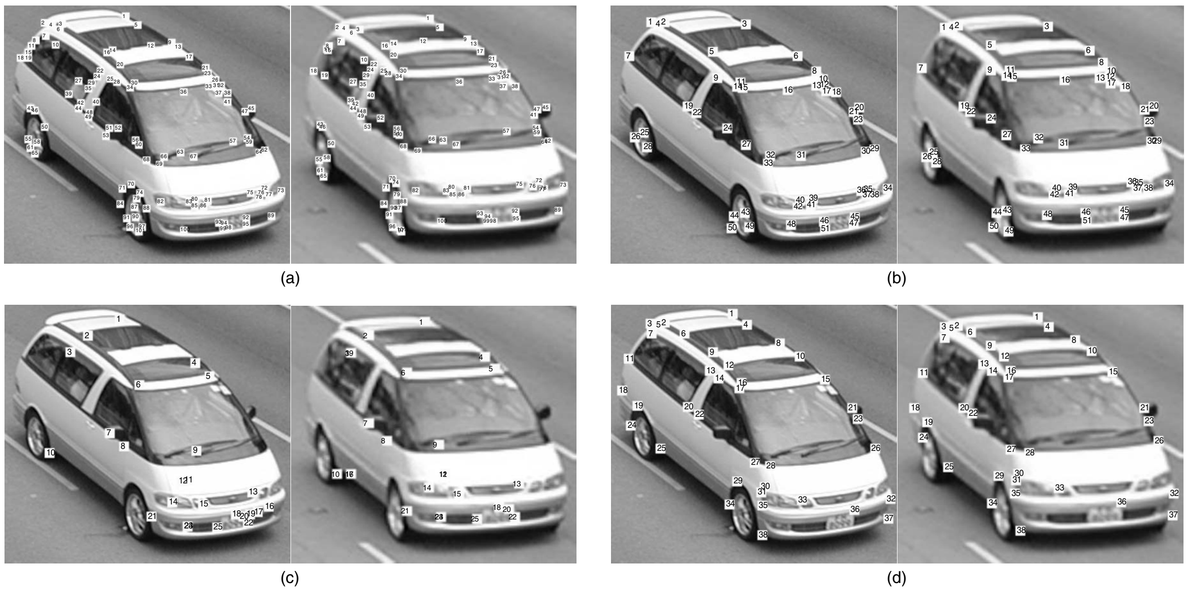


Fig. 10 Feature corresponding results: (a) SUSAN, (b) GFtT, (c) SIFT, (d) proposed method on segmentation boundaries.

Table 3 Evaluation on corresponding results.

Detector	Detected features	Matched features	Mismatched features	Matching rate (%)
SUSAN	2408	2102	306	87.29
GFtT	1227	1164	63	94.87
SIFT	603	505	98	83.75
Proposed:	1178	1087	91	92.28
On Canny				
On segmentation boundary	913	896	16	98.14

missed some significant corners of the vehicle, as well as producing some erratic mismatches, [e.g., points 16, 17, 19 in Fig. 10(c)]. The proposed corner detector could suppress noise and trivial details on planar curves, and therefore achieved reasonable correspondence performance. However, when it was applied to the Canny edge map, it also detected some unstable features in windows or windscreens, which led to some mismatches. When we applied the proposed corner detector to the segmentation boundary, as depicted in Fig. 10(d), it detected all the key corners of a vehicle without any ambiguous features. Of course, the well-segmented input contributes a lot to this excellent correspondence result. We believe utilizing a sophisticated segmentation algorithm to suppress trivial information is advantageous to curve-based corner detectors.

To sum up, in terms of matching rate, the proposed detector on the segmentation boundary (98.14%) has the best performance, and the proposed detector using Canny (92.28%) is worse than GFtT (94.87%). They all perform better than SUSAN (87.29%) and SIFT (83.75%) in this evaluation.

3.5 *Processing Speed*

The proposed detector has been implemented in Matlab. The source code can be obtained from

http://www.mathworks.com/matlabcentral/fileexchange/loadFile.do?objectId=7652&objectType=File

The proposed detection algorithm was executed 100 times on a 1.8-GHz PC with 256 Mbyte of memory, and mean execution times were measured. The processing speed of the subcomponents of the proposed algorithm is depicted in Table 4. The processing time for the planar curves shown in Fig. 6, excluding edge detection and contour extraction, varied from 0.015 to 0.046 s. According to Table 4, the "Block" image and "House" image required similar time, while the "Lab" image required more time, which is reasonable in view of their difference in sizes. In the three subcomponents of the algorithm, corner detection consumes much less time than edge detection and contour extraction, since the proposed corner detection algorithm is only performed on one scale instead of parsing features across the entire scale space like multiscale algorithms. This enables the proposed method to be deployed in real-time applications.

4 Conclusions

The main contribution of this paper is the consideration of both global and local curvature of corners in the detection, in which the use of adaptive threshold and dynamic ROS to identify corners helps to take both properties into account. As a result, different parameters can be automatically determined for different images, different curves, and different kinds of corner candidates.

To summarize, the advantages of the proposed method are that it: (1) increases the number of true corners detected and reduces the number of false corners detected for the images tested; (2) produces relatively low localization errors; (3) supports two controllable parameters (R and θ_{obtuse}) to achieve consistent detection performance from image to image; and (4) identifies corners not only according to their local curvature but also according to their global curvature, detects dominant feature of different sizes, and ignores trivial details. The proposed corner detector could potentially be utilized in many applications, e.g., motion estimation, object tracking, stereo matching, camera calibration, and 3-D reconstruction. So far, its implementations realized by the author include a camera calibration system using road lane markings,[37,38] a visual vehicle speed estimation system,[39] and a vehicle 3-D wire-frame reconstruction system.[40]

Acknowledgments

We would like to express our thanks to Farzin Mokhtarian and Dmitry Chetverikov for making their work available to us for comparison.

Table 4 Time requirements.

Task	Time (s)		
	Blocks (256×256)	House (256×256)	Lab (512×512)
Edge detection	0.681	0.658	2.616
Contour extraction	0.580	0.564	2.960
Corner detection	0.088	0.114	0.358

References

1. G. S. K. Fung, N. H. C. Yung, and G. K. H. Pang, "Vehicle shape approximation from motion for visual traffic surveillance," in *Proc. IEEE 4th Int. Conf. on Intelligent Transportation Systems*, pp. 201–206 (2001).
2. G. S. Manku, P. Jain, A. Aggarwal, A. Kumar, and L. Banerjee, "Object tracking using affine structure for point correspondence," in *Proc. 1997 IEEE Comput. Soc. Conf. on Computer Vision and Pattern Recognition*, pp. 704–709 (1997).
3. B. Serra and M. Berthod, "3-D model localization using high-resolution reconstruction of monocular image sequences," *IEEE Trans. Image Process.* **6**(1), 175–188 (1997).
4. L. Kitchen and A. Rosenfeld, "Gray level corner detection," *Pattern Recogn. Lett.* **1**, 95–102 (1982).
5. H. P. Moravec, "Towards automatic visual obstacle avoidance," in *Proc. Int. Joint Conf. on Artificial Intelligence*, p. 584 (1977).
6. C. Harris and M. Stephens, "A combined corner and edge detector," in *Fourth Alvey Vision Conf.*, pp. 147–151 (1988).
7. S. Smith and J. Brady, "SUSAN—a new approach to low-level image processing," *Int. J. Comput. Vis.* **23**(1), 45–48 (1997).
8. S. C. Bae, I. S. Kweon, and C. D. Yoo, "COP: a new corner detector," *Pattern Recogn. Lett.* **23**(11), 1349–1360 (2002).
9. H. C. Liu and M. D. Srinath, "Corner detection from chain-code," *Pattern Recogn.* **23**, 51–68 (1990).
10. D. Chetverikov and Z. Szabo, "Detection of high curvature points in planar curves," http://visual.ipan.sztaki.hu/corner/index.html (1999).
11. A. Rosenfeld and E. Johnston, "Angle detection on digital curves," *IEEE Trans. Comput.* **22**, 875–878 (1973).
12. A. Rosenfeld and J. S. Weszka, "An improved method of angle detection on digital curves," *IEEE Trans. Comput.* **24**, 940–941 (1975).
13. H. Freeman and L. S. Davis, "A corner finding algorithm for chain-coded curves," *IEEE Trans. Comput.* **26**, 297–303 (1977).
14. H. L. Beus and S. S. H. Tiu, "An improved corner detection algorithm based on chain-coded plane curves," *Pattern Recogn.* **20**, 291–296 (1987).
15. K. Rangarajan, M. Shah, and D. V. Brackle, "Optimal corner detector," *Comput. Vis. Graph. Image Process.* **48**, 230–245 (1989).
16. F. Arrebola, A. Bandera, P. Camacho, and F. Sandoval, "Corner detection by local histograms of contour chain code," *Electron. Lett.* **33**(21), 1769–1771 (1997).
17. W. C. Chen and P. Rockett, "Bayesian labeling of corners using a grey-level corner image model," in *Proc. IEEE Int. Conf. on Image Processing*, pp. 687–690 (1997).
18. F. Chabat, G. Yang, and D. Hansell, "A corner orientation detector," *Image Vis. Comput.* **17**, 761–769 (1999).
19. A. Quddus and M. Fahmy, "Fast wavelet-based corner detection technique," *Electron. Lett.* **35**(4), 287–288 (1999).
20. C. Achard, E. Bigorgne, and J. Devars, "A sub-pixel and multispectral corner detector," in *Proc. 11th Int. Conf. on Pattern Recognition*, pp. 971–974 (2000).
21. F. Shen and H. Wang, "Real time gray level corner detector," in *Proc. 6th Int. Conf. on Control, Automation, Robotics and Vision* (2000).
22. F. Shen and H. Wang, "Corner detection based on modified Hough transform," *Pattern Recogn. Lett.* **23**(8), 1039–1049 (2002).
23. C. Urdiales, C. Trazegnies, A. Bandera, and F. Sandoval, "Corner detection based on an adaptively filtered curvature function," *Electron. Lett.* **39**, 426–428 (2003).
24. A. Rattarangsi and R. T. Chin, "Scale-based detection of corners of planar curves," *IEEE Trans. Pattern Anal. Mach. Intell.* **14**(4), 430–449 (1992).
25. B. K. Ray and R. Pandyan, "ACORD—an adaptive corner detector for planar curves," *Pattern Recogn.* **36**, 703–708 (2003).
26. F. Mokhtarian and A. K. Mackworth, "A theory of multi-scale curvature-based shape representation for planar curves," *IEEE Trans. Pattern Anal. Mach. Intell.* **14**(8), 789–805 (1992).
27. F. Mokhtarian and R. Suomela, "Robust image corner detection through curvature scale space," *IEEE Trans. Pattern Anal. Mach. Intell.* **20**(12), 1376–1381 (1998).
28. F. Mokhtarian and F. Mohanna, "Enhancing the curvature scale space corner detector," *Proc. Scandinavian Conf. on Image Analysis*, pp. 145–152 (2001).
29. X. C. He and N. H. C. Yung, "Curvature scale space corner detector with adaptive threshold and dynamic region of support," in *Proc. 17th Int. Conf. on Pattern Recognition*, pp. 791–794 (2004).
30. H. Spath, "Least-squares fitting by circles," *Computing* **57**, 179–185 (1996).
31. G. Taubin, "Estimation of planar curves, surfaces and nonplanar space curves defined by implicit equations, with applications to edge and range image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.* **13**(11), 1115–1138 (1991).
32. P. l. Rosin, "Multiscale representation and matching of curves using codons," *CVGIP: Graph. Models Image Process.* **55**, 286–310 (1993).
33. F. Mokhtarian, "Image corner detection through curvature scale space," http://www.ee.surrey.ac.uk/Research/VSSP/demos/corners/ (2001).
34. J. B. Shi and C. Tomasi, "Good feature to track," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 593–600 (1994).
35. D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.* **60**(2), 91–110 (2004).
36. X. C. He, N. H. C. Yung, K. P. Chow, F. Y. L. Chin, R. H. Y. Chung, K. Y. K. Wong, and K. S. H. Tsang, "Watershed segmentation with boundary curvature ratio based merging criterion," in *Proc. Ninth IASTED Int. Conf. on Signal and Image Processing*, pp. 7–12 (2007).
37. G. S. K. Fung, N. H. C. Yung, and G. K. H. Pang, "Camera calibration from road lane markings," *Opt. Eng.* **42**(10), 2967–2977 (2003).
38. X. C. He and N. H. C. Yung, "A new method for solving ill-condition in vanishing point based camera calibration," *Opt. Eng.* **46**(3), 037202 (2007).
39. X. C. He and N. H. C. Yung, "A novel algorithm for estimating vehicle speed from two consecutive images," in *Proc. Eighth IEEE Workshop on Applications of Computer Vision*, p. 12 (2007).
40. X. C. He, "Feature extraction from two consecutive traffic images for 3D wire frame reconstruction of vehicle," PhD thesis, Univ. of Hong Kong (2006).

Xiao Chen He received his BEng and MEng degrees from the University of Science and Technology of China in 1999 and 2002, respectively, and his PhD degree from the University of Hong Kong in 2007. He is currently a research associate with the Department of Computer Science, University of Hong Kong. His research interests include pattern recognition, digital image processing, and visual traffic surveillance.

Nelson H. C. Yung received his BSc and PhD degrees from the University of Newcastle-upon-Tyne, where he was a lecturer 1985 to 1990. From 1990 to 1993, he worked as senior research scientist in the Department of Defence, Australia. He joined the University of Hong Kong in late 1993 as associate professor. He is the founding director of the Laboratory for Intelligent Transportation Systems Research at HKU, and also deputy director of HKU's Institute of Transport Studies. Dr. Yung has coauthored five books and book chapters, and has published more than 100 journal and conference papers in the areas of digital image processing, parallel algorithms, visual traffic surveillance autonomous vehicle navigation, and learning algorithms. He serves as reviewer for the *IEEE Transactions on SMC*, *CASVT*, *Vehicular Technology, and Signal Processing*; *Optical Engineering*; *the Journal of Electronic Imaging*; *HKIE Proceedings*; *Microprocessors and Microsystems*; and *Robotics and Autonomous Systems*. He was a member of the Advisory Panel of the ITS Strategy Review, Transport Department, HKSAR; regional secretary of the IEEE Asia-Pacific region; council member of ITS-HK; and chair of the Computer Division of the International Institute for Critical Infrastructures. He is a chartered electrical engineer; member of the HKIE and IEE, and senior member of the IEEE. His biography has been published in *Who's Who in the World* (Marquis, USA) since 1998.