Volume 2, Issue 4, April 2012 ISSN: 2277 128X

International Journal of Advanced Research in Computer Science and Software Engineering

Research Paper

Available online at: www.ijarcsse.com

Comparison and Error Finding of 2D Frontal Facial Images Between Twins

Rajib Saha[1], Debotosh Bhattacharjee[2], ShombhuNath Ghosh[3], Prasenjit Das[4], Dona Ghosh[5]

Department of Computer Science and Engineering

[1,3,5]RCC Institute Of Information Technology Kolkata, India

[2]Jadavpur University Kolkata, India

rajibsaha_4u@yahoo.co.in

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. This project proposes a method to extract the feature points from faces automatically. It provides a feasible way to locate the positions of two eyeballs, near and far corners of eyes, midpoint of nostrils and mouth corners from face image. This approach would help to extract useful features on human face automatically and improve the accuracy of face recognition. The experiments show that the method presented in this paper could locate feature points from faces exactly and quickly and the goal of this project is to compare and finding error of images of twins.

Keywords – Canny Method, Active ASM, Face recognition, Corner detection, Adaptive Threshold, Region of support, Curvature, Contour, Round Corner, Obtuse corner.

I. INTRODUCTION

Face recognition has become one of the most important applications of image analysis and computer vision in recent years. The methods[1-2] of face recognition are divided into two main categories:

Holistic approaches and geometric-based approaches.

1) In the holistic methods, the recognition is achieved by modeling or representing the intensity values of the pixels in facial images.

2) While in the geometric methods, the geometry of the face and the locations of salient points, known as facial features, are taken into account.

Facial feature extraction consists in localizing the most characteristic face components (eyes, nose, mouth, etc.) within images that depict human faces. This step is essential for the initialization of many face processing techniques like face tracking, facial expression recognition or face recognition.

This paper presents an automatic method for facial feature extraction that use for the initialization of face recognition technique. To extract the Facial components equal to locate certain characteristic points, e.g. the center and the corners of the eyes, the nose tip, etc. Particular emphasis will be given to the localization of the most representative facial features, namely the eyes, and the locations of the other features will be derived from them.

SUSAN's [7] corner localization and noise robustness, a circular mask for corner and edge detection, are better corner and edge detection 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 which two oriented cross operators, called crosses as oriented pair (COP) [8] , were used instead of circular mask.

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 the above problem, a multiscale algorithm[23] based on curvature scale space (CSS) also proposed, 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[24]. . The CSS technique is suitable for recovering invariant geometric features of a planar curve at multiple scales.

The following steps are used by the original CSS algorithm [26] 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. 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 these 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.

1. 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 high scale intensity is large and detects false corners when high scale intensity is small. If the algorithm is applied to a complex image, this effect becomes more prominent, and choosing appropriate high scale intensity becomes challenging.
2. 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. 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.
3. 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.
4. 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[27] 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.

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.

II. EXPERIMENTS

This paper proposes a new and improved corner detection method, of which a preliminary version has been described. 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. It assumes 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.

The deferent steps of the developed algorithm are described below:-

1. Detect edges of the 1st image of twins using the likes of a Canny edge detector to obtain a binary edge map.
2. Detect edges of the 2nd image of twins using the likes of a Canny edge detector to obtain a binary edge map.
3. Extract contours of both images as in the CSS method.
4. 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.
5. 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.
6. Based on a dynamically recalculated region of support, evaluate the angles of the remaining corner candidates to eliminate any false corners.
7. 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 (2).

8. Finally compare each corresponding points of tow images and consider their edge, corner &curve detection time to compute the error between their faces.

A. *Initial List of Corner Candidates*

Let us first define the j^{th} extracted contour as:

$$A^j = \{P^J_1, P^J_2, P^J_3, \ldots\ldots, P^J_N\} \quad (1)$$

Where $P^j{}_i=(x_i{}^j, y_i{}^j)$ 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 is closed if $|P^j{}_1 P^j{}_N| < T$
and
A^j is open if $|P^j{}_1 P^j{}_N| > T$ (2)

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

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

$$A^j{}_{smooth} = A^j \times g \quad (3)$$

where g is a digital Gaussian function with width controlled by σ. A value σ=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

$$K_i^j = \frac{\Delta x_i^j \, \Delta^2 y_i^j - \Delta^2 x_i^j \, \Delta y_i^j}{[(\Delta x_i^j)^2 + (\Delta y_i^j)^2]^{1.5}} \quad \text{for i=1,2,3,…….N} \quad (4)$$

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 (4) all the local maxima of the curvature function are included in the initial list of corner candidates.

B. *Corner Evaluation*

As defined in Sec. I, although the curvature of a round corner is the largest among its neighbors, the actual difference may not be significant. On the other hand, the curvature of an obtuse corner 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. In order to utilize this global curvature characteristic of the neighbors to eliminate round corners yet not the obtuse corners, in this paper 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, $L1+L2$ is the size of the ROS centered at u, and R is a coefficient:

$$T(u)=R \times \bar{K} = R \times \frac{1}{L1+L2+1} \sum_{i=u-L2}^{u+L1} |K(i)| , \quad (5)$$

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 $L1+L2$ then does that of a round corner over a similar ROS.

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 $f(x) = \sqrt{\left[b^2 - \left(\frac{bx}{a}\right)^2\right]}$ 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

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

and we have $K_{max}=K(0)=b/a^2$, $K_{min}=K(a)=a/b^2$. (6)

C. *False-corner removal*

A well-defined corner should have a relatively sharp angle. If the angle of each corner on a contour is kown, 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.-1 depicts five points labeled on a curve, all of which represent local curvature maxima and can be regarded as corners. Taking point 3 and 2 as an example, if the angle of a corner is defined as

an acute angle, point 3 and 2 will fall within this definition. Moreover, if we consider point 3 and 2 within the range of points 1 and 4, then it will be classified as a true corner too.

Fig. 1 Illustration of an ambiguous case.

However, if we consider point 3 or point 2 within the range of points 5, then points 2 and 3 may all be classified as false corners, the reason being that points 1and 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.

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 [30] , 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.-1.

.

However, it fails when dealing with local variations along an arc, as depicted in Fig. 2, which is illustrated in the following. After round-corner removal, points *C* as depicted in Fig. 2 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 [Fig.-2(a)], $\angle C$ has not been 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, the angle of a corner has been redefine 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 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.-2(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. [29] 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-2(b) :-

1. On one arm of an ROS (from *C* to *A*, say), three points (*C*, the midpoint *M*, and *A)* are selected. If these three points are collinear, the tangent direction of this ROS arm is defined from *C* to *A* 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

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

$$y_0 = \frac{(x_1^2+y_1^2)(x_2-x_3)+(x_2^2+y_2^2)(x_3-x_1)+(x_3^2+y_3^2)(x_1-x_2)}{2[y_1(x_2-x_3)+y_2(x_3-x_1)+y_3(x_1-x_2)]} \quad (7)$$

2. 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 $\emptyset$ to denote the direction from *C* to *M*. Then we can have the tangent of C at this side of ROS as follows:

$$\gamma_1 = \theta + sign(\sin(\emptyset - \theta))\frac{\pi}{2},$$

Where sign is a signum function.

3. The tangent of the ROS from *C* to *F* is eterminedsimilarly, and is denoted by γ_1.

4. Fourth, the two tangent lines form the angle of the corner:

$$\angle C = \begin{cases} |\gamma_1-\gamma_2| & if\ |\gamma_1-\gamma_2| < \pi \\ 2\pi - |\gamma_1-\gamma_2| & otherwise \end{cases} \quad (8)$$

5. The corner checking criterion is given as follows:

C_i is true corner if $\angle C_i \leq \theta_{\text{obtuse}}$,

C_i is false corner if $\angle C_i > \theta_{\text{obtuse}}$, (9)

The parameter θobtuse 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.

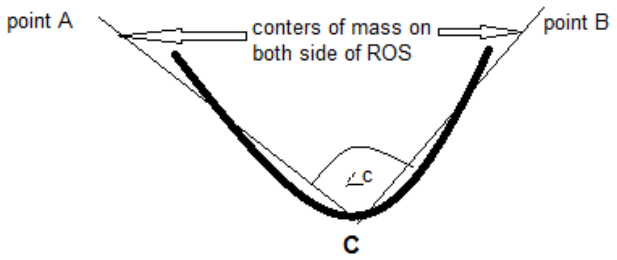

Fig. 2(a) Angle definitions of a corner:

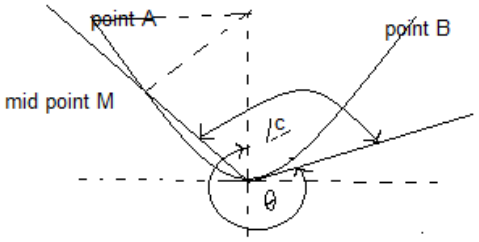

Fig. 2(b) Tangent definition:

D. 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, [26] 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. [33] 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.

III. 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.

A. Results at Different Stages

a) Images of identical Twins:

i) Overview:

To illustrate how the proposed method works, two set of images of two twins with a large number of acute, obtuse, and round corners is used for the test. Fig.-3 depicts the processed images at various stages. Fig.- 3(b) depicts the Canny edge map. 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.-3(c).

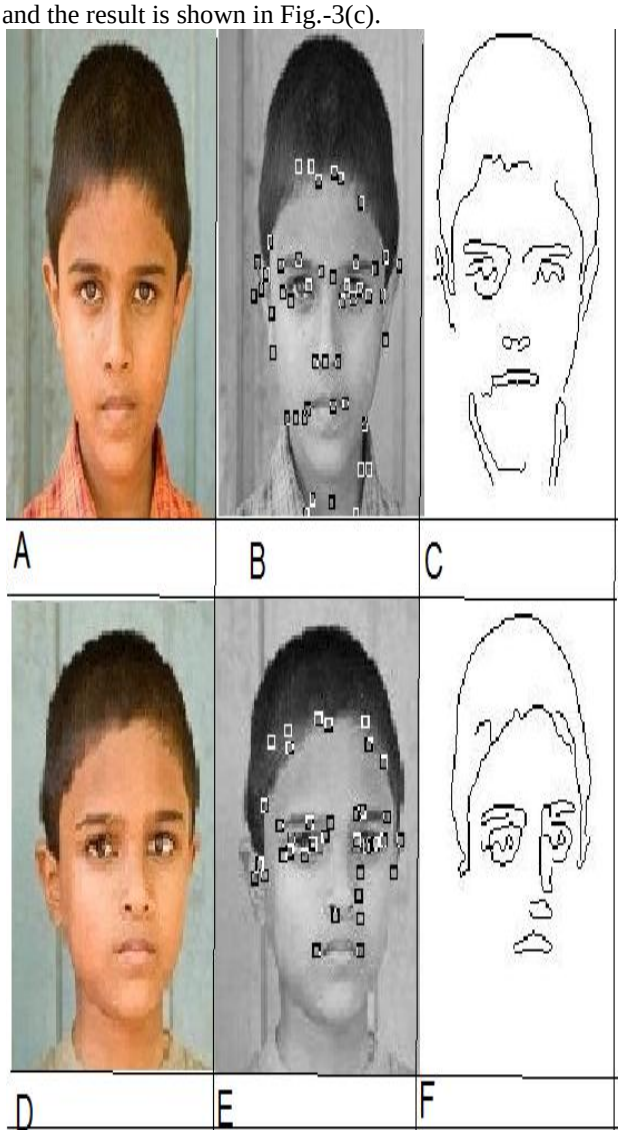

Fig. 3 Corner detection of the "Lab" image by the proposed method: (a)1st original image(b) detected edge contours of 1st image, (c)end-point consideration of 1st image, (d)2nd original image,(e)detected edge contours of 2nd image.(f)end point consideration of 2nd image.

ii) Result:

The deferent time to detecting edges,extracting curves & detecting corners of both the images are given below:-

Time for detecting edge for 1st image = 0.0811 sec
Time for extracting curve for 1st image = 0.1328 sec
Time for detecting corner for 1st image = 0.0386 sec

Time for detecting edge for 2nd image = 0.0516sec
Time for extracting curve for 2nd image= 0.1263sec
Time fordetecting corner for 2nd image= 0.0313sec

The detected corner points of both the image are given as:

TABLE I
Evaluation results for the1st set of images

CORNER POINTS OF 1st IMAGE	CORNER POINTS OF 2nd IMAGE
X Y	X Y
94 27	99 24
43 62	45 68
52 92	47 44
85 46	84 66
90 45	91 54
124 61	124 87
112 72	113 87
86 85	87 96
75 88	77 103
86 101	86 111
51 34	53 45
80 39	79 57
97 87	88 91

iii) Analysis of Result:

From the above results it clear that both the images have so many similarities of their faces as the detected corner points are very similar to each others also the time required to detected their edges, curves, corner point are also very near.
So, based on the above result it is clear that the images are vary similar to each other.

b) Images of non identical Twins:

Now we apply this method on two images of dissimilar twins , to show that the process is work properly on deferent stage of input.

i) *Overview:*

Figure 4 depicts the processed images at various stages. Figure 4(b) depicts the Canny edge map. 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. 4(c).

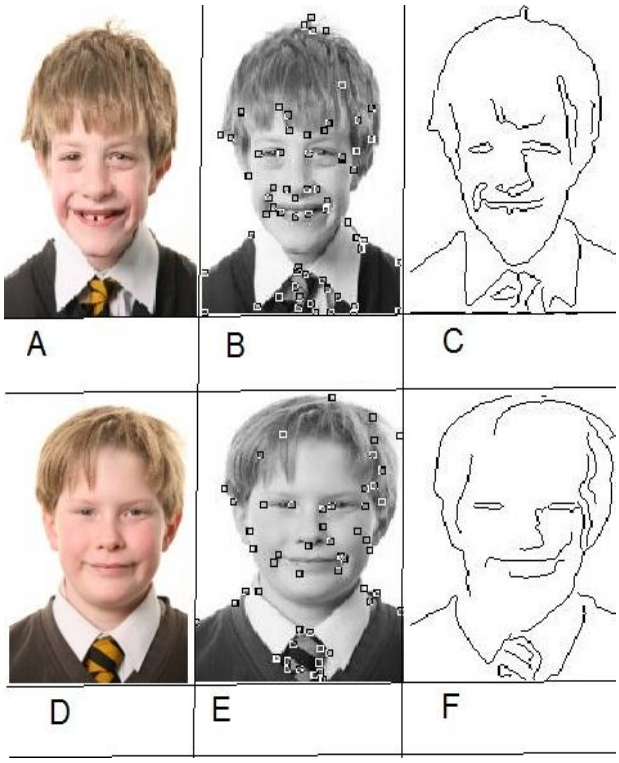

Fig. 4 Corner detection of the "Lab" image by the proposed method: (a)1st original image(b) detected edge contours of 1st image, (c)end-point consideration of 1st image, (d)2nd original image,(e)detected edge contours of 2nd image.(f)end point consideration of 2nd image.

ii) *Result:*

The deferent time to detecting edges,extracting curves & detecting corners of both the images are given below:-

Time for detecting edge for 1st image = 0.0414 sec
Time for extracting curve for 1st image = 0.0772 sec
Time for detecting corner for 1st image =0.0299sec

Time for detecting edge for 2nd image = 0.0391sec

Time for extracting curve for 2nd image= 0.0851sec
Time fordetecting corner for 2nd image= 0.0259sec
The detected corner points of both the image are:

TABLE II
Evaluation results for the 2nd set of images

CORNER POINTS OF 1st IMAGE		CORNER POINTS OF 2nd IMAGE	
X	Y	X	Y
196	38	174	65
86	13	88	20
171	66	19	85
19	76	11	75
99	39	98	54
127	46	120	49
136	54	139	64
121	72	121	58
86	72	83	84
121	79	137	73
180	83	194	75
101	96	99	73
200	88	201	104
199	135	168	135
154	111	159	109
196	105	190	93
76	107	54	96
175	2	200	2
112	32	129	86
68	53	73	59
190	53	190	67
127	64	134	85
176	71	185	86
142	100	152	106
97	106	108	104
69	116	89	116

iii) *Analysis of Result:*

From the above results it clear that both the images have so many dissimilarities of their faces as the deference of their detected corner points are vary from each others also the time required to detected their edges, curves, corner point are not similar.

So, based on the above result it is clear that the images are dissimilar to each other.

IV. CONCLUSSION

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.

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. And (5) find the no of error and compare the images of twins and give the result as time taken to extract their images and the extracted corner points as output. 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 camera calibration system using road lane markings, [34,35] a visual vehicle speed estimation system [36] and a vehicle 3-D wire-frame reconstruction system [37] .

V. REFERENCES

[1] B. Serra and M. Berthod, "3-D model localization using highresolution reconstruction of monocular image sequences," *IEEE Trans. Image Process.* **6**(1), 175–188 (1997).

[2] L. Kitchen and A. Rosenfeld, "Gray level corner detection," *Pattern Recogn. Lett.* **1**, 95–102 (1982).

[3] H. P. Moravec, "Towards automatic visual obstacle avoidance," in *Proc. Int. Joint Conf. on Artificial Intelligence*, p. 584 (1977).

[4] G. S. K. Fung, N. H. C. Yung, and G. K. H. Pang, "Vehicle shapeapproximation from motion for visual traffic surveillance," in *Proc. IEEE 4th Int. Conf. on Intelligent Transportation Systems*, pp. 201–206 (2001).

[5] 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).

[6] H. C. Liu and M. D. Srinath, "Corner detection from chain-code," *Pattern Recogn.* **23**, 51–68 (1990).

[7] R.Saha,, D. Bhattacharjee. "Memory Efficient Human Face Recognition Using fiducial Points",IJARCSSE,Vol 2, Issue 1,January,2012

[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] D. Chetverikov and Z. Szabo, "Detection of high curvature points in planar curves," http://visual.ipan.sztaki.hu/corner/index.html (1999).

[10] A. Rosenfeld and E. Johnston, "Angle detection on digital curves," *IEEE Trans. Comput.* **22**, 875–878 (1973).

[11] A. Rosenfeld and J. S. Weszka, "An improved method of angle detection on digital curves," *IEEE Trans. Comput.* **24**, 940–941 (1975).

[12] H. Freeman and L. S. Davis, "A corner finding algorithm for chaincoded curves," *IEEE Trans. Comput.* **26**, 297–303 (1977).
[13] 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).
[14] K. Rangarajan, M. Shah, and D. V. Brackle, "Optimal corner detector," *Comput. Vis. Graph. Image Process.* **48**, 230–245 (1989).
[15] 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).
[16] 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).
[17] F. Chabat, G. Yang, and D. Hansell, "A corner orientation detector," *Image Vis. Comput.* **17**, 761–769 (1999).
[18] A. Quddus and M. Fahmy, "Fast wavelet-based corner detection technique," *Electron. Lett.* **35**_4_, 287–288 (1999).
[19] 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).
[20] F. Shen and H. Wang, "Real time gray level corner detector," in *Proc. 6th Int. Conf. on Control, Automation, Robotics and Vision* (2000).
[21] F. Shen and H. Wang, "Corner detection based on modified Hough transform," *Pattern Recogn. Lett.* **23**_8_, 1039–1049 (2002).
[22] C. Urdiales, C. Trazegnies, A. Bandera, and F. Sandoval, "Corner detection based on an adaptively filtered curvature function," *Electron. Lett.* **39**, 426–428 (2003).
[23] 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).
[24] B. K. Ray and R. Pandyan, "ACORD—an adaptive corner detector for planar curves," *Pattern Recogn.* **36**, 703–708 (2003).
[25] 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).
[26] F. Mokhtarian and R. Suomela, "Robust image corner detection through curvature scale space," *IEEE Trans. Pattern Anal. Mach. Intell.* **20**_12_, 1376–1381 (1998).
[27] F. Mokhtarian and F. Mohanna, "Enhancing the curvature scale space corner detector," *Proc. Scandinavian Conf. on Image Analysis*, pp. 145–152 (2001).
[28] 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).
[29] H. Spath, "Least-squares fitting by circles," *Computing* **57**, 179–185 (1996).
[30] 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).
[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] X. C. He, "Feature extraction from two consecutive traffic images for 3D wire frame reconstruction of vehicle," PhD thesis, Univ. of Hong Kong (2006).
[33] P. l. Rosin, "Multiscale representation and matching of curves using codons," *CVGIP: Graph. Models Image Process.* **55**, 286–310 (1993).
[34] 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).
[35] 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).
[36] 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).
[37] X. C. He, "Feature extraction from two consecutive traffic images for 3D wire frame reconstruction of vehicle," PhD thesis, Univ. of Hong Kong (2006)

