# Corner Detection using Difference Chain Code as Curvature

Neeta Nain, Vijay Laxmi, Bhavitavya Bhadviya * and Nemi Chand Singh †

*Abstract*—**Detection of discontinuity plays an important role in image registration, comparison, segmentation, time sequence analysis and object recognition. In this paper we address some aspects of analyzing the content of an image. A point is classified as a corner where there are abrupt changes in the curvature of the curve. This paper presents a new approach for corner detection using first order difference of chain codes called shape numbers. Since the proposed method is based on integer operations it is simple to implement and efficient. Preliminary results are presented and evaluation with respect to standard corner detectors is done as a benchmark.**

***Keywords: Corners, Chain Code, Difference Code, Curvature.***

## 1 Introduction

An edge [1], [2], [3] by definition, is that where drastic change in value from one spot to the next either in intensity level of an image or in one's own eye takes place. In the digital world, an edge is a set of connected pixels that lie on the boundary between two regions. Similarly corner detection [4], [5], [6], [7], [8], [9], [10], [11], [12], [13], [14], [15], [16], etc., is identification of high curvature points on planar curves which is referred to as boundary of an image. A corner can also be defined as the intersection of two edges or, as a point for which there are two dominant and different edge directions in a local neighborhood. Corners play a dominant role in shape perception by humans. They play crucial role in decomposing or describing the object. They are used in scale space theory, image representation, reconstruction, matching and as preprocessing phase of outline capturing systems.

This paper proposes a new difference chain encoding algorithm for corner detection in images. The task of finding corners is formulated as a three step process. In the first step we extract one pixel thick noiseless boundary by using various morphological operations. The extracted boundary is chain encoded using $m$-connectivity [3]. In the second step we apply chain smoothing procedures to remove false corners by suppressing regular intensity changes. The final step calculates shape numbers and then abrupt changes are identified in shape numbers [1], [2] to detect significant corners.

## 2 Existing Techniques

Many corner detection algorithms have been proposed in the literature. Rosenfeld and Johnston [4] calculate curvature maxima points using $k$-cosines as corners. Rosenfeld and Wezka [5] proposed a modification of [4] in which averaged k-cosines were used. Freeman and Davis [6] found corners at maximum curvature change in which they move a straight line segment along the curve. Angular differences between successive segments was used to measure local curvature. Beus and Tiu [7] algorithm was similar to [6] except they proposed arm cutoff parameter $\tau$ to limit length of straight line. Smith's algorithm [8] involves generating a circular mask around a given point in an image and then comparing the intensity of neighboring pixels with that of the center pixel, and repeating the procedure for each pixel within the image. Harris corner detector [9] computes locally averaged moment matrix computed from the image gradients, and then combines the eigenvalues of the moment matrix to compute a corner "strength", of which maximum values indicate the corner positions. Yung's [10] algorithm starts with extracting the contour of the object of interest, and then computes the curvature of this contour with Gaussian derivative filters at various scales. Local extremes of the product of the curvatures at different scales are reported as corners when the value of the product exceeds a threshold. There are various other techniques like wavelets, Hough transform, neural networks etc using which corners can be extracted as described in [11], [12], [13], [14], [15], [16], [17], [18], [19], [20],

## 3 Corner Detection

It has long been known that information about shape is conveyed via the changes in the slopes of an object's boundary. The information content is greatest where this change also called curvature is strongest. We present a method of estimating the curvature of a planar curve, and detecting corners also called high-curvature points on such curves. Digital images are acquired and pro-

*Department of Computer Engineering, Malaviya National Institute of Technology Jaipur, Jaipur, India - 302017. Tel: +91 141 2529140 Email: neetanain@yahoo.com

†Rajasthan Swayat Sashan Mahavidyalaya, Tonk Road, Jaipur, India - 302015.

ISBN: 978-988-98671-8-8

cessed in a grid format with equal spacing in the x and y direction. To extract corners we need precise, compact and continuous in a segment boundary. One-pixel thick m-connected boundary is extracted using our own morphological algorithm [3].

One-pixel thick, and m-connectivity avoids redundancy in chain codes. This boundary is *Chain-Encoded* using 8−way chain encoding [6] method. When 8-way connectivity is used, the 360° is divided into eight directions. Each direction specifying 45° angle from the previous. Thus a chain code could be generated by following a boundary in, say, an anti-clockwise direction and assigning a direction code to the segments connecting each pair of pixels. The chain-code varies from $0, 1, 2, 3, 4, 5, 6, 7$ in anti-clockwise direction, where 0 means moving one unit in $x$ direction making angle 0° with the $x-$axis, code 1 represents 45° from $x-$axis and so on. The chain code thus codes the slope of the curve.

In this proposed corner detection algorithm we perform chain coding on a boundary extracted image. The boundary chain($BC$) code specifies the direction in which a step must be taken to go from the present boundary point to the next one. Thus the BC's represent the slope of the boundary at that position. The BC codes are further modified, called Modified Chain($MC$) codes. The modification step smoothens the boundary, removes noise and thus removes false corners from boundary. From the MC codes we calculate first order Difference Codes($DC$). The first order difference codes represents the turning angles from the previous point or the curvature of the boundary curve at a point. The maximum curvature change occurs where two straight line segments of length $l$ meet, where $l \geq threshold$. The threshold for our experiments when taken as

$$l = (BoundaryLength)^{\frac{1}{4}} \qquad (1)$$

fourth root of the boundary length gives excellent results. If $l$ is taken less than our threshold then false corners will be generated as it will capture all the small curvature changes also. While a larger value will miss significant corners.

The proposed technique of corner detection is based on finding vertex points on the boundary curve where two straight line segments meet. The length of the straight line segment is taken as a parameter $l$ called pixel neighborhood length. The algorithm identifies the points where the rate of change of the slope is highest in a neighborhood of $\pm l$ pixels. The BC's represent the slope and the DC's represents the rate of change of slope. Thus the sequence of 0's in DC specifies a straight line and isolated non-zero codes in a sequence of zeros specify the vertex points where two line segments meet. These vertex points may be true corners if the length of the line segment $\geq l$. But the search for corners is not so easy. Due to side effects of scan-conversion or aliasing effects, a straight line in the pixel form is not drawn as a straight line, but it has some regular changes also called stair-step effects. The problem gets more complicated when the radius of curvature is small on the boundary, or in other words if the slope changes fast in a small neighborhood of points. Then if we capture all these slope changes then we generate false corners. We need to identify only abrupt changes in slope on the boundary to detect significant corners. This is taken care of Modified Chain codes. The MC codes smooth the boundary in the neighborhood of $\pm l$ pixels, which is equivalent of convolving with a $l$ x $l$ averaging mask.

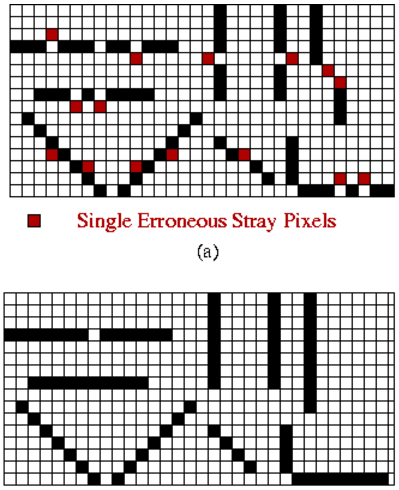


Figure 1: (a)Various erroneous single stray pixel cases, (b) Single stray pixels removed after smoothing step

The detailed proposed algorithm is as follows:

1. Extract one-pixel thick m-connected boundary.

2. Calculate slope: Assign BC codes by chain encoding the boundary using 8-way connectivity. The BC codes represent the slope of the current pixel with the previous. The slope at current pixel $i$ is thus

$$\phi_i = \tan^{-1}(\frac{\pi}{4}BC_i) \qquad (2)$$

3. Smooth the Boundary:

   - Remove spurious single pixels from the boundary. Figure 1 depicts all such cases.
     - One pixel breaks in straight line, where there is only one isolated pixel above the break. A $V$ made by a set of pixels.
     - One pixel thick diagonal line with one extra pixel in 4−neighborhood of any of the pixel of this diagonal line.

   The following substitutions are done to handle the above case. For current pixel position $i$ check the next two pixels' $BC_i$. Substitute the current series of the $BC$'s according to Table 1. Figure 1(b) shows the result of the smoothing step on Figure 1(a).

| Current Series of $BC's$ | $BC's$ After Replacement |
|---|---|
| (*0*, 7, 1) / (0, 1, 7) | (0, 0, 0) |
| (*2*, 3, 1) / (2, 1, 3) | (2, 2, 2) |
| (*4*, 5, 3) / (4, 3, 5) | (4, 4, 4) |
| (*6*, 5, 7) / (6, 7, 5) | (6, 6, 6) |
| (*1*, 2, 0) / (1, 0, 2) | (1, 1) |
| (*3*, 4, 2) / (3, 2, 4) | (3, 3) |
| (*5*, 6, 4) / (5, 4, 6) | (5, 5) |
| (*7*, 0, 6) / (7, 6, 0) | (7, 7) |

Table 1: Smoothing single stray pixels from the boundary

- Align stray single pixels along the dominant slope in a 3 x 3 neighborhood.
  For current pixel position $i$
  - Check previous two pixels and next two pixel's $BC$.
    If $BC_{i\pm n(n=1,2)}$ are same and $BC_i \neq BC_{i\pm n}$ then $BC_i = BC_{i+1}$.
  - Check previous two pixels' and next pixels' BC or next two pixels' and previous pixels' BC.
    If ($BC_{i-n(n=1,2)} = BC_{i+1}$ AND $\neq BC_i$) OR ($BC_{i+n(n=1,2} = BC_{i-1}$ AND $\neq BC_i$) then $BC_i = BC_{i+1}$ (or $BC_i = BC_{i-1}$).
  - Check previous two BC and next to next BC of current pixel or next two BC and previous to previous pixels' BC
    If ($BC_{i-n(n=1,2)} = BC_{i+2}$ AND $BC_i \neq BC_{i+2}$) OR ($BC_{i+n(n=1,2)} = BC_{i-2}$ AND $BC_i \neq BC_{i-2}$) then $BC_i = BC_{i+2}$ (or $BC_i = BC_{i-2}$).

  This step removes all the spurious codes on the curve and aligns the stray pixels along the dominant slope of the line.

4. Avoid False Corners: Check $MC$ codes. For current pixel position $i$, If $MC_i \neq MC_{i-1}$, then this is called the first occurrence of $MC$ code. If the first occurrence $MC_i = MC_{i-n(n=1,2,...,l)}$, then the $i^{th}$ position is not a corner. For example, let $l = 4$, and if the $MC$ code is *2222222333222222222244433333333*, then its first occurrence $MC$ code positions are $1, 8, 11, 21, 24$. The $MC$ code at $11^{th}$ position already exists in previous codes within a neighborhood of $l = 4$, and thus is a false corner.

5. Calculate Curvature by First Order Difference Codes: The curvature at current pixel $i$ is

$$\delta\phi_i = \phi_{i+1} - \phi_i \qquad (3)$$

The first order difference code ($DC$) on the $MC$ codes represents this curvature or turning angle from the previous point. It is calculated as

$$DC_i = (MC_{i+1} - MC_i + 8) \ mod \ 8 \qquad (4)$$

where the $MC_i$ś are the modified boundary chain codes. The first order difference on the $MC$ codes represents the turning angles from the previous point.

6. Find True Corners: Traverse the $DC's$. Let $P_i$ denotes the position of isolated non-zero DC in a sequence of 0 $DC's$. If $P_i - P_{i-1} > l$ (the difference between consecutive positions of isolated non zero DC's). Then there are true corners at $P_i$ and $P_{i-1}$ positions. For example, if the $MC$ code = *11111222222224444444*, its DC = *0000100000002000000*. The non-zero $DC$ positions $P_i$ are $8, 13$. The difference between non zero $DC$ positions $= 5 > l$, then there are corners at $5^{th}$ and $13^{th}$ positions.

## 4 Test Results

The proposed corner detector gives single, localized and accurate response to corners. A corner detection algorithm must be tested on a variety of shapes for its proper evaluation. Our test image includes most variety of variations in curvature, corner sharpness and noise/irregularities along the boundary curves. Such variations are expected in real life images. Comparison is also done with two popular algorithms namely Harris [9] and Yung [10]. Table 2 and Fig 2 shows the results. Both Harris and Yung miss true corners on small curvature curves. Our corner detector gives excellent results even in the presence of noise like Gaussian, Poisson, and Speckle etc., while both Harris [9] and Yung [10] are not invariant to noise. They eliminate some true corners and also introduce false corners due to noise. Our algorithm is invariant to noise and extracts all true corners as shown in Fig 3.

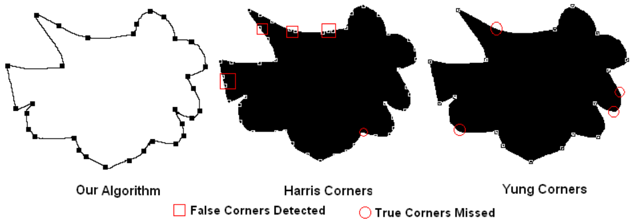
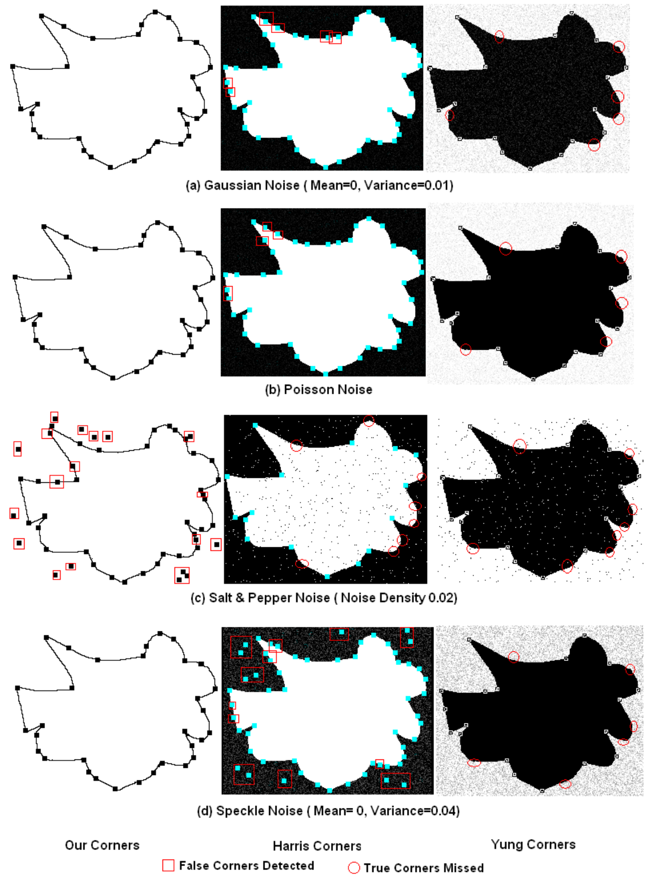


Figure 2: Test image with regular curvature changes

## 5 Affine Transformation Invariance

Corner detection should not be influenced by affine transformation effects like translation, rotation, shearing and minor size variations. The proposed algorithm is affine transformation invariant. Corners are calculated on the first order differences of the chain codes. The first difference in the chain codes makes the boundary translation invariant as they represent the turning angles from the



(a) Gaussian Noise ( Mean=0, Variance=0.01)

(b) Poisson Noise

(c) Salt & Pepper Noise ( Noise Density 0.02)

(d) Speckle Noise ( Mean= 0, Variance=0.04)

Figure 3: Corners extracted on noisy images (a) Gaussian Noise, (b) Poisson Noise, (c) Speckle Noise

previous point. To expand or to contract a chain by a specified scale factor, one must appropriately scale each chain link and then re quantize. In down sampling, a number of links may merge into one, and in up-sampling one link may cause a string of many links to be generated. The generation of these new links is most easily handled by using Bresenham scan conversion technique. Compared to up sampling, down sampling is robust to aliasing effects. We down sample the boundary to a constant size before chain coding. This re sampling makes the corner detector scale invariant.The boundary is aligned with minimum angle with the $x-$axis. This is done by circularly shifting the DC's and starting with the first difference of smallest magnitude. Circular shift makes the boundary rotation invariant and the corners extracted on this boundary will be rotation invariant.Transformation invariant test results of the corner detection algorithm on the standard test image is shown in Fig 4.

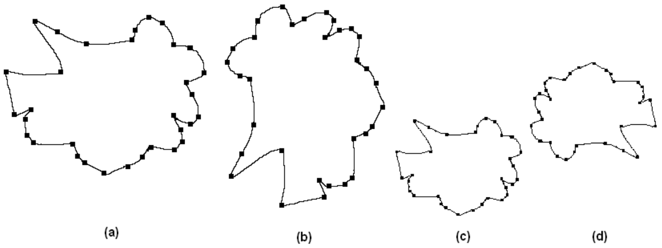


Figure 4: Transformation invariance of corner extractor(a) Original (b) Rotated by 270°, (c) Scaled 50% (d) Scaled 50% and Rotated 90°

## 6 Shape Reconstruction from Linear Interpolation of Extracted Corners

Corners extracted from this technique could be used in various shape analysis, size measurement and boundary reconstruction applications. From the extracted corners it is very easy to reconstruct an approximation to the boundary by fitting straight line segments between corners. For polygonal figures having straight line segments the reconstructed shape gives a 100% similarity measure with the original as shown in Figure 5. For curved figures it helps in identifying the convex hull of the shape as shown in Figure 6 and Figure 7, which could also be used as a filter for shape comparison applications. Many scalar shape descriptors [21] like compactness, rectangularity, perimeter, area, elongatedness etc., can be easily calculated from the convex hull.

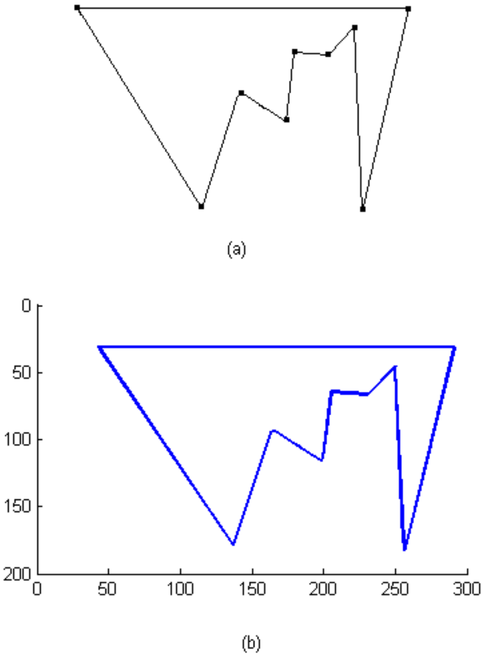
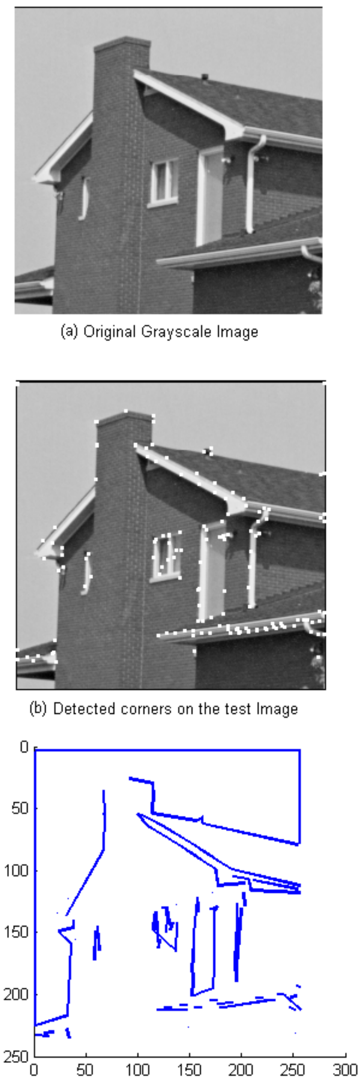


Figure 5: Image Reconstruction(a) Original Image (b) Reconstructed by fitting straight lines between corners

## 7 Conclusions

In this paper a novel simple, efficient and transformation invariant method for corner detection is proposed which uses only integer arithmetic for approximation of curvature on curve boundary. The novelty of the approach



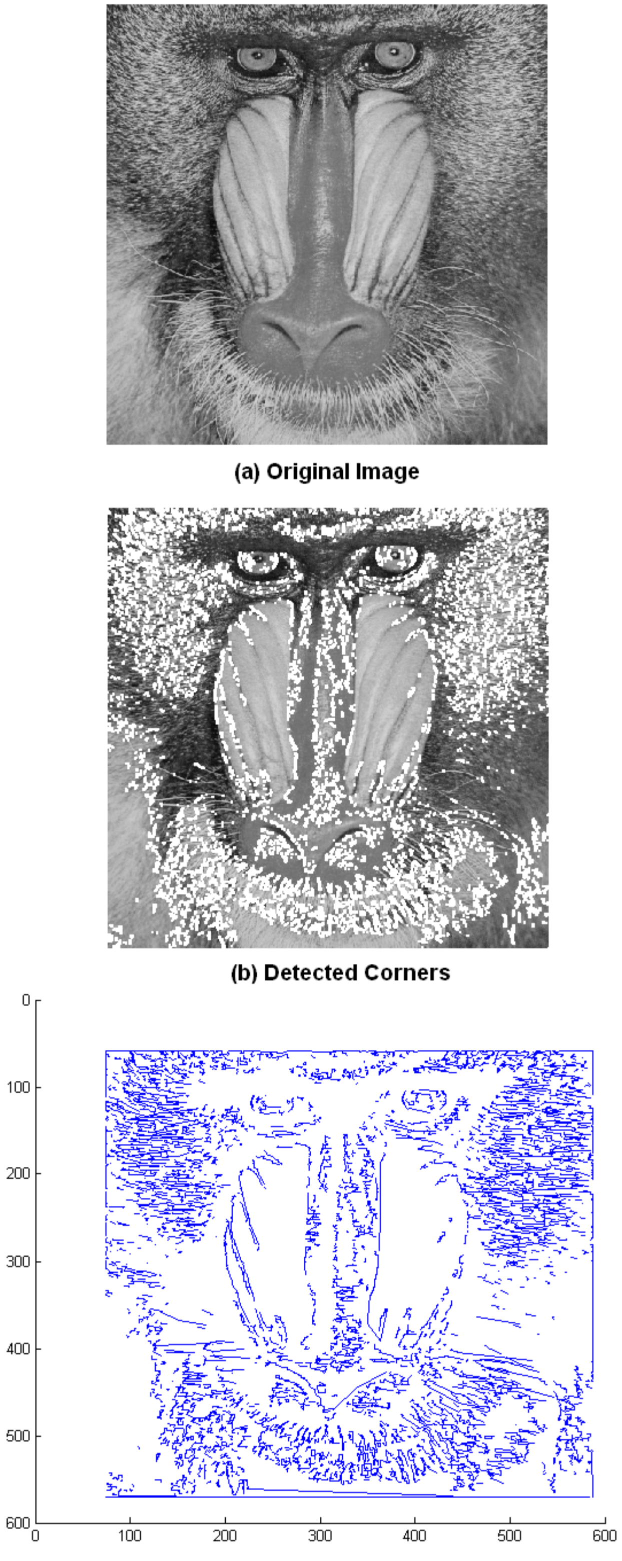
(a) Original Grayscale Image

(b) Detected corners on the test Image

(c) Reconstructed boundaries of the grayscale image

Figure 6: Image Reconstruction(a) Original Image (b) Extracted corners (c) Reconstructed by fitting straight lines between corners

lies in its simplicity and usage of integer arithmetic. The algorithm depends on only one parameter $l$. The parameter $l$, can be assigned a constant value as per personal preference of neighborhood, and it is also adaptive to the length of the boundary curve. Experimental results show that $l$ taken as fourth root of boundary length is sufficient for extracting good corners. The parameter $l$ is therefore both local and global in nature. Comparative study on various noiseless and noisy images has been done. Finally, the comparison between the proposed approach and other corner detectors show that the approach is very competitive with respect to Harris [9] and Yung [10] under similarity and affine transforms. Moreover, a number of experiments also illustrate that the corner detection approach is more robust to noise.



(a) Original Image

(b) Detected Corners

(c) Reconstructed Image

Figure 7: Image Reconstruction(a) Original Image (b) Extracted corners (c)Reconstructed image by fitting straight lines between corners

Corners extracted from this technique could be used in various shape analysis, size measurement and boundary reconstruction applications. From the extracted corners we can easily reconstruct the boundary by fitting straight line segments between corners. This can be used as the convex hull or an approximation to the object's shape.

| Corner Detector | Results |
|---|---|
| Yung | True corners missed on curved edges |
| Harris | Poor true corners even<br>on significant curvature change |
| Our Algorithm | All true corners, No false corners<br>Better approximation of curvature |

Table 2: Comparison of corner detectors on Fig 2 and Fig 3

## References

[1] Rafael C. Gonzalez and Richard E. Woods. *Digital Image Processing.* Addison Wesley Longman, 2nd ed, 2000.

[2] V. Hlavac M. Sonka and R. Boyle. *Image Processing Analysis and Machine Vision.* Chapman and Hall, 1993.

[3] Ankur Kumar Jain Neeta Nain, Vijay Laxmi and Rakesh Agarwal. In *Morphological Edge Detection and Corner Detection Algorithm Using Chain-Encoding*, volume II, pages 520–525. The 2006 International Conference on Image Processing, Computer Vision and Pattern Recognition, Las Vegas, Nevada, USA, 2006.

[4] A. Rosenfeld and E. Johnston. In *Angle Detection on Digital Curves*, volume C-22, pages 875–878. IEEE Transactions on Computers, 1973.

[5] A. Rosenfeld and J. S. Wezka. In *An Improved Method of Angle Detection on Digital Curves*, volume C-24, pages 940–941. IEEE Transactions on Computers, 1975.

[6] H. Freeman and L. S. Davis. In *A Corner Finding Algorithm for Chain Coded Curves*, volume 26, pages 297–303. IEEE Transaction in Computing, 1977.

[7] Beus. H. L. and S. S.H Tiu. In *An Improved Corner Detection Based on Chain Coded Plane Curves*, volume 20, pages 291–296. Pattern Recognition Letters, 1987.

[8] Stephen M. Smith and J. Michael Brady. Susana new approach to low level image processing. *International Journal for Computer Vision*, 23(1):45–78, 1997.

[9] C. Harris and M. Stephens. In *A Combined Corner and Edge Detector*, volume 23, pages 189–192. Proceedings of 4th Alvey Vision Conference, Manchester, 1988.

[10] X.C. He and N. H. C. Yung. In *Curvature Scale Space Corner Detector with Adaptive Threshold and Dynamic Region of Support*, volume 2, pages 791–794. International Conference in Pattern Recognition, IEEE Computer Society, 2004.

[11] Chang S. P and J. H. Horng. In *Corner Point Detection Using Nest Moving Average*, volume 27(11), pages 1533–1537. Pattern Recognition Letters, 1994.

[12] H.C Liu and M.D. Srinath. In *Corner Detection from Chain Code*, volume 23, pages 51–68. Pattern Recognition Letters, 1990.

[13] H.T. Hou Tsai. D.M and H. J. Su. In *Boundary Based Corner Detection Using Neural Networks*, volume 20, pages 31–40. Pattern Recognition Letters, 1999.

[14] E.Rosten and T. Drummong. Machine learning for high-speed corner detection. European Conference on Computer Vision, Citeseer, May 2006.

[15] Azhar Quddus and Moncef Gabbouj. Wavelet-based corner detection technique using optimal scale. *Pattern Recogn. Lett.*, 23(1-3):215–220, 2002.

[16] Fei Shen and Han Wang. Corner detection based on modified hough transform. *Pattern Recognition Letters*, 23(8):1039–1049, 2002.

[17] Eduard Sojka. A new algorithm for detecting corners in digital images. In *SCCG '02: Proceedings of the 18th spring conference on Computer graphics*, pages 55–62, New York, NY, USA, 2002. ACM Press.

[18] Gustavo Olague and Benjamín Hernández. A new accurate and flexible model based multi-corner detector for measurement and recognition. *Pattern Recognition Letters*, 26(1):27–41, 2005.

[19] R. Bergevin and A. Bubel. Detection and characterization of junctions in a 2d image. *Computer Vision and Image Understanding*, 93(3):288–309, 2004.

[20] R. Bergevin and A. Bubel. Object-level structured contour map extraction. *Computer Vision and Image Understanding*, 91(3):302–334, 2003.

[21] Neeta Nain, Vijay Laxmi Manish Khandelwal and Deepak Agarwal. Transformation invariant robust shape analysis. In *The 2007 International Conference on Image Processing, Computer Vision and Pattern recognition*, pages 275–281, Nevada, USA, 2007. CSREA Press.