CODE: An Adaptive Algorithm for Detecting **Co**rners and **D**irections of Incident **E**dges

Partha Bhowmick
Computer Science & Technology Department
Bengal Engineering College, India
partha@becs.ac.in

Bhargab B. Bhattacharya
Center for Soft Computing Research
Indian Statistical Institute, Kolkata, India
bhargab@isical.ac.in

Abstract

Conventional derivative-based corner detection is often crippled by missing junctions, poor localizations, and showing up of false negatives and false positives with change in the control parameters. This paper presents a novel algorithm that efficiently detects corners along with the directions (orientations) of the edges associated with each corner. The algorithm is marked by some adaptive features that strengthen the robustness and usefulness of the algorithm. Experimental results on a varied set of images exhibit the stability and efficiency of the algorithm.

1. Introduction

Corners are important geometric features of a digital image. In a gray-level image, corners are formed at boundaries between two significantly dissimilar image brightness regions, where the boundary curvature is sufficiently high. Detection of corners is often required in many computer vision applications such as shape analysis, object recognition, optical flow computation, 3D scene reconstruction from stereo image pairs, etc. Being sparse features, mere presence of them are considered sufficiently informative. Hence, much of the work on two-dimensional features of an image is focused on detection of corners.

Corner detection can be broadly categorized as contour-based method or gray-level based method. In the contour-based method [3, 6, 9, 11], a segmented boundary is followed and the rate of change of the contour angle is watched to detect and locate corners. Thus, this method is quite susceptible to the adopted segmentation procedures. As an improvement on corner detection, CSS (curvature scale-space) method [14] is proposed recently by Mokhtarian and Suomela, where corners are detected, tracked, and localized through the curvature analyses based on multiple scales. On the contrary, the gray-level based method has been suggested to detect corners directly from the gray-scale images without any prior segmentation. The use of angle-based templates is proposed by Rangarajan et al. [15]. There also exist gradient-based methods [7, 8, 10] for corner detection by identifying curvature changes by differential analysis without the need for prior segmentation. A literature survey by Zheng et al. [18] summarizes the existing gray-level corner detection methods. In recent times, several corner detection techniques have been proposed [1, 2, 4, 5, 12, 13, 17]. One of the widely referred corner detection algorithms is SUSAN [16], where, a circular mask of fixed diameter (7 pixels) is used to extract the local structural information based on the USAN area in order to judge the candidature of the mask's nucleus as a corner. The algorithm proposed here differs from SUSAN in several aspects, viz., adaptively augmenting the annular window depending on a number of conditions, design of an adaptive annular filtering scheme, adaptive thresholding of brightness parameter, and subpixel-precision evaluation of incident edge directions at each corner.

2. Proposed corner detection

Let $I(x, y) \in \{0, 1, 2, \ldots, 255\}$ be the grayscale intensity of any point (x, y) in a grayscale digital image I with m rows and n columns. Then the gradient of intensity at any point $P(i, j) \in I$ is given by two components in the right-hand side of equation 1.

$$\nabla I(i,j) = \left\langle \frac{\partial I}{\partial x}(i,j), \frac{\partial I}{\partial y}(i,j) \right\rangle \tag{1}$$

The magnitude of $\nabla I(i, j)$ is a measure of the strength of edge, if any, passing through (i, j), and can be expressed in several suitable forms, customized to the requirement, one of which is as follows:

$$|\nabla I(i,j)| = \left|\frac{\partial I}{\partial x}(i,j)\right| + \left|\frac{\partial I}{\partial y}(i,j)\right| \tag{2}$$

The corresponding second order difference $\nabla^2 I(i, j)$ is used to find the zero-crossing across the edge, guided by the direction ϕ as given by equation 3.

$$\phi = tan^{-1} \frac{\left|\frac{\partial I}{\partial x}(i,j)\right|}{\left|\frac{\partial I}{\partial y}(i,j)\right|} \quad (3)$$

Usage of above directional differences (first order and second order) is a traditional and popular approach to find the gray level topological features (viz. edges, corners, etc.) of an image, but this has some severe problems, some of which are as follows:

(i) *Inappropriate* $\nabla I(i,j)$ *and* $\nabla^2 I(i,j)$: In the discrete domain, $\nabla I(i,j)$ and $\nabla^2 I(i,j)$ merely considers the neighboring pixels (4-N or 8-N) of (i,j), thereby weakening the relevance of their definitions in the real domain, especially for the case where the gray level transition in and around the concerned point (i,j) is spread over a larger region, which often leads to improper location and erratic detection of desired features.

(ii) *Zero crossing problem*: For the gray level transition associated with a ramp edge, which is the most prevalent type of edge in a natural gray scale image, the zero crossing procedure at a point sometimes yields a wide range of solution (when $|\nabla I(i,j)|$ is same for 3 or more consecutive pixels along the same "ramp") or a staggered edge point (when the zero crossing(s) lies away from the middle of the same "ramp"), posing further analysis in extracting the true edge point and the best edge direction at the concerned point.

(iii) *Noise*: Presence of noise is a very common characteristic in a digital image, which, without any noise cleaning, when convoluted with the gradient operators, $\nabla I(i,j)$ and $\nabla^2 I(i,j)$, often produces impure results. The refinement process (e.g. Gaussian filtering, median filtering, etc.), in order to get rid of the noise from the image, in turn, blurs the image in general, and the edges in particular, thereby worsening the situation of detecting any image feature.

In order to circumvent the aforesaid problems, in this work, therefore, we have not used the directional difference operators for detecting the corners. We have not even applied any standard filtering on the input image, since any filtering mask of some defined size, say $w \times w$, (viz. 3×3, or, 5×5 gaussian mask) may include, in worst case, the gray values of $(w^2 - w)$ non-edge pixels for a true edge pixel, which will affect our feature detection procedure. Instead, we have designed an adaptive annular filtering scheme as discussed below.

2.1. Adaptive annular filtering

Let $P(x,y)$ be any point in the image I, and $\mathcal{C}_r^{(x,y)}$ be the ordered list of pixels of the (digital) circle of radius r centered about P, enumerated in clockwise direction starting from $(x+r, y)$, as shown in Fig. 1. Let $|\mathcal{C}_r|$ be the number of pixels in $\mathcal{C}_r^{(x,y)}$, which will be independent of (x,y) and constant for a given value of r. Now we define $\mathcal{A}_r^{(x,y)}$ as the ordered list of image gray values of the points

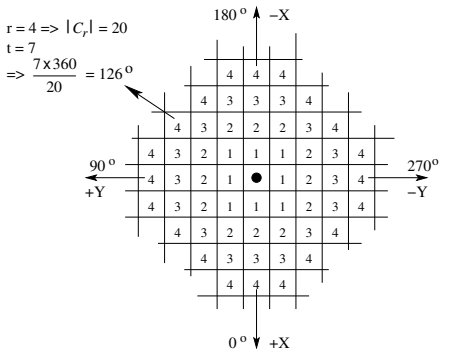


Figure 1. Annular lists $\mathcal{C}_1^{(x,y)}, \mathcal{C}_2^{(x,y)}, \ldots$, where the pixels labeled by r are in the list $\mathcal{C}_r^{(x,y)}$ and $P(x,y)$ is shown by '•'.

in $\mathcal{C}_r^{(x,y)}$, such that for each k, $0 \le k \le |\mathcal{C}_r| - 1$, the k-th entry in $\mathcal{A}_r^{(x,y)}$ is the gray value of the k-th point in $\mathcal{C}_r^{(x,y)}$. In the implementation of our algorithm, $|\mathcal{C}_r|$, and the angular tolerance in degrees, T_θ, are obtained from Look Up Table LUT-1 as shown below, to reduce the execution time. Construction of LUT-1 is independent of all parameters used in the algorithm.

LUT-1 required for $|\mathcal{C}_r|$ and T_θ.

r	1	2	3	4	5
$\lvert\mathcal{C}_r\rvert$	8	12	16	20	28
T_θ	23	15	11	9	6

It may be observed that, if P lies on some gray level edge or corner in the image, then the gray level transition across the respective edges would be reflected in the pattern of gray level values in $\mathcal{A}_r^{(x,y)}$, which can be exploited to extract the location and direction of the edge. Strengthening and appropriating the procedure of finding the concerned edge directions requires the reduction of noise present in $\mathcal{A}_r^{(x,y)}$, which is done by convoluting each element in $\mathcal{A}_r^{(x,y)}$ with a mask $\mathcal{W}_r$ = $\boxed{\omega_1}\boxed{\omega_2}\boxed{\ldots}\boxed{\omega_{\alpha_r}}\boxed{\ldots}\boxed{\omega_2}\boxed{\omega_1}$ to create a filtered list of gray values $\mathcal{F}_r^{(x,y)}$, in accordance with the following equation:

$$\mathcal{F}_r^{(x,y)}(k) = \frac{\sum_{i=-\alpha_r}^{\alpha_r} \mathcal{W}_r(\alpha_r + i) \cdot \mathcal{A}_r^{(x,y)}(k+i)}{\sum_{i=-\alpha_r}^{\alpha_r} \mathcal{W}_r(\alpha_r + i)} \quad (4)$$

where, $\mathcal{A}_r^{(x,y)}(k+i)$ denotes the $\big((k+i)$ **mod** $|\mathcal{C}_r|\big)$-th element in $\mathcal{A}_r^{(x,y)}$, for $k = 0, 1, \ldots, |\mathcal{C}_r| - 1$. Since the 2nd order difference of the gray value transition along the maximum gray value gradient for a ramp edge resembles the 1-d gaussian function, we resort to the gaussian mask $\mathcal{W}_r$ of size $2\alpha_r + 1$. The speciality of the mask $\mathcal{W}_r$ adopted in our algorithm is that, with increase in the value of r, i.e., higher augmentation of $\mathcal{C}_r^{(x,y)}$, the size of $\mathcal{W}_r$ = $2\alpha_r + 1$ is also increased to suit the slant of the ramp edge, since a lower

value of r encompasses a part of the gray value transition, whereas a sufficiently high value of r ensures the inclusion of the entire gray value transition of the edge in $\mathcal{C}_r^{(x,y)}$. A sample image is cropped and magnified as an example in Fig. 2 that demonstrates the increase in the length of the ramp edge, where, there exist two ramp edges, whose ramp lengths are given in Table 1.

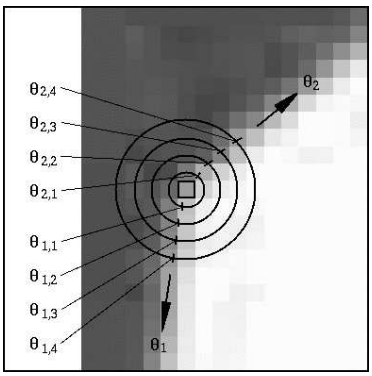


Figure 2. An example of finding the directions (θ_1, θ_2) of incident edges at a point $P(x, y)$ (shown by ▢) using the angles $\{\theta_{1,1}, \theta_{1,2}, \ldots\}$ and $\{\theta_{2,1}, \theta_{2,2}, \ldots\}$ extracted from filtered annular lists $\mathcal{F}_1^{(x,y)}, \mathcal{F}_2^{(x,y)}, \ldots$.

Table 1. Gray value ramp lengths and gray value transitions for different r in Fig. 2.

r	$\lvert\mathcal{C}_r\rvert$	ramp length				α_r	gray value trans.	
		edge-1		edge-2			edge-1	edge-2
1	8	4	4	4	4	0	92	103
2	12	6	7	5	7	1	141	133
3	16	6	7	6	6	1	165	172
4	20	5	10	5	6	2	177	175
5	28	5	9	5	6	2	177	174

2.2. Adaptive brightness thresholding

In a natural image, the gray-level value around an edge point or a corner point along the maximum gradient direction changes gradually. Thus, larger the augmenting window radius r centered about the corresponding point is, higher is the change of gray value across an edge, within some suitable range of r. Fig. 3 justifies the need for adaptively changing the brightness threshold γ_r with r. If Z represents the zero-crossing in an ideal ramp edge as shown in Fig. 3, then γ_r is practically divided into 2 equal parts, one above the abscissa line through Z and the other below it. As a result, the value of $\frac{1}{2}\gamma_r$ (γ_r, there of) increases with the window radius r, as in Eqn. 5, so that γ_r is minimum for $r = 1$ and has no appreciable change when r exceeds 3 or 4.

$$\gamma_r = \lfloor \gamma_\infty (1 - exp(-r/\kappa)) \rfloor \tag{5}$$

κ is related with the maximum gray value transition of an edge along its maximum gradient direction, i.e., slope of

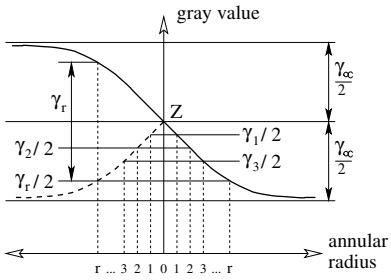


Figure 3. Change of gray level threshold with annular radius.

the curve shown in Fig. 3. Based on the observation that an entire edge transition along the maximum gradient direction (i.e. ramp length as shown in Table 1) gets captured in the annular list $\mathcal{A}_r$ for $r = 3$ or 4, we have considered $\kappa = 1$ in our experiments, where implementation of Eqn. 5 is realized by a look up table LUT-2 that is prepared once for an image for a given value of brightness threshold, γ_∞, as shown below.

An instance of LUT-2 for $\gamma_\infty = 45$ and $\kappa = 1$.

r	1	2	3	4	5
γ_r	28	38	42	44	44

2.3. Estimation of edge directions

Another important feature of the proposed work is the estimation of edge direction by mean weighted difference. This method of finding edge direction works superbly for both natural and synthetic images, which usually possess ramp and step edge transitions respectively. For step edges, usual directional derivatives $\frac{\partial I}{\partial x}$ and $\frac{\partial I}{\partial y}$ provide the necessary edge directions. But for ramp edges with low $|\nabla I|$, the derivatives produce erratic edge directions, since calculation of $\frac{\partial I}{\partial x}$ and $\frac{\partial I}{\partial y}$ deals with only 8×8 neighborhood of the concerned point and the entire transition pattern along the maximum gray value gradient is not considered while estimating the edge direction. In this approach, we have taken into consideration the entire gray-level transition corresponding to an edge and, therefore, the estimated edge direction precisely matches the true edge transition.

We define a function φ to reorder the annular list $\mathcal{F}_r^{(x,y)}$ by a cyclic forward shift 's' to produce a new list $\tilde{\mathcal{F}}_r^{(x,y)}$ in order to expedite the edge extraction procedure as follows:

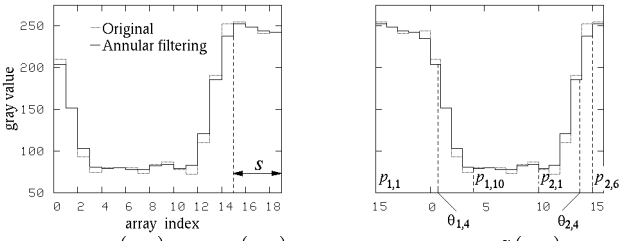
$\varphi : \{k\}_0^{|\mathcal{C}_r|-1} \mapsto \{(k+s)\mathbf{mod}|\mathcal{C}_r|\}_{k=0}^{|\mathcal{C}_r|-1}$,

such that both the following two conditions (**c1**) and (**c2**) are satisfied.

(**c1**) $\left\langle \mathcal{F}_r^{(x,y)}(k) \right\rangle_0^{|\mathcal{C}_r|-1} = \left\langle \tilde{\mathcal{F}}_r^{(x,y)}\big((k+s) \bmod |\mathcal{C}_r|\big) \right\rangle_{k=0}^{|\mathcal{C}_r|-1}$,
where, $s = \mathbf{min}\{0, 1, 2, \ldots, |\mathcal{C}_r| - 1\}$.

(**c2**) either, at least one of
$\delta(\tilde{\mathcal{F}}_r^{(x,y)}, 0, 1)$ and $\delta(\tilde{\mathcal{F}}_r^{(x,y)}, |\mathcal{C}_r| - 1, 0)$ is 0, or,

sign $\big(\delta(\tilde{\mathcal{F}}_r^{(x,y)}, 0, 1)\big) \neq$ **sign** $\big(\delta(\tilde{\mathcal{F}}_r^{(x,y)}, |\mathcal{C}_r| - 1, 0)\big)$, where, $\delta(\tilde{\mathcal{F}}_r^{(x,y)}, p, q) = \tilde{\mathcal{F}}_r^{(x,y)}(p) - \tilde{\mathcal{F}}_r^{(x,y)}(q)$, and, **sign** $(a) = -1(a < 0), 1(a > 0), 0(a = 0)$.



(a) $\mathcal{A}_4^{(x,y)}$ and $\mathcal{F}_4^{(x,y)}$ (b) $\tilde{\mathcal{F}}_4^{(x,y)}$

Figure 4. An example of finding the directions $(\theta_{1,4}, \theta_{2,4})$ of incident edges at $P(x, y)$ in Fig. 2 using the filtered annular list $\tilde{\mathcal{F}}_4^{(x,y)}$.

Let $\big\langle \tilde{\mathcal{F}}_r^{(x,y)}(p_{u,t}) \big\rangle_{t=1}^{t=e_u}$ be the u-th monotonically ascending (or, monotonically descending) subsequence of length e_u in $\tilde{\mathcal{F}}_r^{(x,y)}$. Then $\big\langle \tilde{\mathcal{F}}_r^{(x,y)}(p_{u,t}) \big\rangle_{t=1}^{t=e_u}$ corresponds to the gray value transition for a valid edge if both the following conditions (**c3**) and (**c4**) are true:

(**c3**) In $\tilde{\mathcal{F}}_r^{(x,y)}$, there exists no monotonically ascending subsequence of length exceeding e_u that contains $\big\langle \tilde{\mathcal{F}}_r^{(x,y)}(p_{u,t}) \big\rangle_{t=1}^{t=e_u}$.

(**c4**) $\Big|\delta(\tilde{\mathcal{F}}_r^{(x,y)}, p_{u,e_u}, p_{u,1})\Big| \geq \gamma_r$.

As there are $|\mathcal{C}_r|$ pixels in the annular list $\tilde{\mathcal{F}}_r^{(x,y)}$, the angular resolution at the center $P(x, y)$ of the digital circle $\mathcal{C}_r^{(x,y)}$ is $\frac{360^o}{|\mathcal{C}_r|}$. So, the angle made by the radius vector joining the point $\mathcal{C}_r^{(x,y)}(t)$ (see Fig. 1) with the +ve x-axis measured in clockwise direction is given by $\frac{t \cdot 360^o}{|\mathcal{C}_r|}$. Further, it may be observed there are e_u discrete gray-values in the subsequence $\big\langle \tilde{\mathcal{F}}_r^{(x,y)}(p_{u,t}) \big\rangle_{t=1}^{t=e_u}$, i.e. $(e_u - 1)$ gray value differences, which have a cumulative effect on the value $\delta(\tilde{\mathcal{F}}_r^{(x,y)}, p_{u,e_u}, p_{u,1})$ of overall gray-level transition corresponding to u-th edge incident at P. The dominant ones out of these $(e_u - 1)$ differences will outweigh the others in their way of deciding the resultant edge direction. Hence, the edge direction corresponding to the subsequence $\big\langle \tilde{\mathcal{F}}_r^{(x,y)}(p_{u,t}) \big\rangle_{t=1}^{t=e_u}$ is given by:

$$\theta_{u,r} = \left\lfloor \frac{360^o}{|\mathcal{C}_r|} \left(\sum_{t=1}^{e_u - 1} \frac{q_{u,r}^{(t)}}{\delta(\tilde{\mathcal{F}}_r^{(x,y)}, p_{u,e_u}, p_{u,1})} + s \right) + \frac{1}{2} \right\rfloor \tag{6}$$

where, $q_{u,r}^{(t)} = \big(\frac{(p_{u,t+1} + p_{u,t})}{2} \cdot \delta(\tilde{\mathcal{F}}_r^{(x,y)}, p_{u,t+1}, p_{u,t})\big)$.

In Fig. 4(a), an instance for $r = 4$ is shown for the sample given in Fig. 2. The minimum cyclic shift s, shown in Fig. 4(a), is required to produce the new (cyclic shifted) annular list $\tilde{\mathcal{F}}_4^{(x,y)}$ shown in Fig. 4(b). The list $\tilde{\mathcal{F}}_4^{(x,y)}$ contains one descending subsequence $\{15, 16, \ldots, 19, 0, 1, \ldots, 4\}$ and one ascending subsequence $\{10, 11, \ldots, 15\}$ (considering the indices in the original list $\mathcal{F}_4^{(x,y)}$) that satisfy conditions (**c3**) and (**c4**). Hence these two subsequences correspond to the angles $\theta_{1,4}$ and $\theta_{2,4}$, shown by their respective index values in Fig. 4(b), and actual angles overlaid in Fig. 2, evaluated from Eqn. 6. Note that, the last entry of the last valid subsequence (conforming to conditions (**c3**) and (**c4**)) may be the first element in $\tilde{\mathcal{F}}_4^{(x,y)}$, which should be checked exclusively, which occurs when the start pixel of the first edge transition coincides with the end pixel of the last edge transition (e.g., in Fig. 4(b)).

2.4. Algorithm

1. Execute the steps (2) - (12) for each point $P(x, y) \in I$.
2. Initialize $r = 1$.
3. Using the Look Up Table (LUT-1), assign $T_\theta = \left\lfloor \frac{180^o}{|\mathcal{C}_r|} + \frac{1}{2} \right\rfloor$, and construct the annular list $\mathcal{A}_r^{(x,y)}$.
4. Apply adaptive annular filtering on $\mathcal{A}_r^{(x,y)}$ to get $\mathcal{F}_r^{(x,y)}$ in accordance with Eqn. 4, and $\tilde{\mathcal{F}}_r^{(x,y)}$ there of, conforming to conditions (**c1**) and (**c2**).
5. Choose the adaptive brightness threshold γ_r from LUT-2.
6. Using conditions (**c3**) and (**c4**) and Eqn. 6, find the set of incident edge angles $\{\theta_{u,r}\}_{u=1}^{n_r}$ from $\tilde{\mathcal{F}}_r^{(x,y)}$, where, n_r is the number of distinct edges for r-th annular list.
7. If $n_r = 0$ or $n_r \neq n_1$, then P is not a corner candidate.
8. If $r = 1$, then increment r by unity, and go to step (3).
9. If $e_u(r) <= e_u(r-1), 1 \leq u \leq n_r$, then (further augmentation of r is not required) assign $R = r$, and go to step (10); else if $r = r_{max}$ ($r_{max} = 5$ in our experiments), then P is not a corner candidate; else, increment r by unity, and go to step (3).
10. Arrange the angles $\{\theta_{u,r}\}$ for $r > 1$ so that they are ordered as in $\langle \theta_{u,1} \rangle$. If $|\theta_{u,r+1} - \theta_{u,r}| \leq T_\theta$ for any u, $1 \leq u \leq n_r - 1$, then P is not a corner candidate.
11. Assign $\theta_u = \theta_{u,R}, 1 \leq u \leq n_1$.
12. If $n_1 = 2$ and $180^o - d\_acute(\theta_1, \theta_2) < T_a$ (default $22\frac{1}{2}^o \approx 23^o$ in our algorithm), then P is just an edge point, but not a corner point; otherwise, P is a corner candidate.
13. Merging: For each cluster of corners accumulated in and around a true corner, select the one having minimum variance of edge directions for $1 \leq r \leq R$, and discard the rest.

3. Experimental results

We have implemented our algorithm in C in SunOS Release 5.7 Generic of Sun_Ultra 5_10, Sparc, 233 MHz, and compared its performance with SUSAN [16], the mostly referred algorithm in recent times. In SUSAN's approach

(Figs. 5(a), 8(a), 6(a), 7(a), 9(a)), several false corners creep in, whereas, the proposed algorithm reports fewer false negatives and false positives (Figs. 5(b), 8(b), 6(b), 7(b), 9(b)). Apart from detection of corners, our algorithm also finds the directions of incident edges for each corner, and therefore, takes some additional time during execution. Hence the time taken by SUSAN cannot be compared with ours. The details of experimental results for our algorithm are shown in Table 2.

Table 2. Results for 5 sample images.

Image name	Image size	# corners		Time (secs.)		γ_∞
		A	B	C	D	
test	256×256	464	61	0.78	0.19	15
house	256×256	105	60	0.38	0.25	45
blocks	256×256	162	56	0.66	0.39	45
lab	512×512	776	248	2.13	1.07	30
logo	256×256	90	22	0.54	0.21	45

A = No. of corners before merging.
B = No. of corners after merging.
C = Time for detection of corners before merging.
D = Time for merging corners.

4. Conclusion and future work

The proposed algorithm produces uniform results for several real and synthetic images, binary as well as gray-scale, and is characterized by its adaptability with the gray-scale topology of an image. Though the computational complexity rises with the increase in radius of the annular list, but in terms of the accuracy and localization of detected corners and associated edge directions, our approach performs very well. Furthermore, the method can be extended to devise a procedure of finding the most genuine corners in any image by augmenting the window to suitably higher radius, having predetermined trade-off with computational complexities. Interpolation of windows is another avenue that needs exploration for improved results.

References

[1] S. Alkaabi and F. Deravi, Candidate Pruning for Fast Corner Detection, Electronic Letters, vol. 40, no. 1, pp. 18-19, 2004.

[2] M. Banerjee, M. K. Kundu, and P. Mitra, Corner Detection Using Support Vector Machine, Proc. ICPR, August 2004, Cambridge, UK.

[3] H. L. Beus and S. H. Tiu, An Improved Corner Detection Algorithm Based on Chain-code Planar Curves, Pattern Recognition, vol. 20, pp. 291-296, 1987.

[4] R. Elias and R. Laganiére, Cones: A New Approach Towards Corner Detection, Proc. IEEE Canadian Conf on Electrical & Computer Engg, pp. 912-916, 2002.

[5] Y. Etou, T. Sugiyama, K. Abe, and T. Abe, Corner Detection Using Slit Rotational Edge-Feature Detector, Proc. ICIP 2002, pp. 797-800.

[6] H. Freeman and L. S. Davis, A Corner Finding Algorithm for Chain-coded Curves, IEEE Trans. Comput., vol. 26, pp. 297-303, 1977.

[7] C. Harris and M. Stephens, A Combined Corner and Edge Detector, 4th Alvey Vision Conf., pp. 147-151, 1988.

[8] L. Kitchen and A. Rosenfeld, Gray-level Corner Detection, Pattern Recognition Lett., vol. 1, pp. 95-102, 1982.

[9] J. Koplowitz and S. Plante, Corner Detection for Chain Coded Curves, Pattern Recognition, vol. 28, pp. 843-852, 1995.

[10] T. Lindeberg, Junction Detection with Automatic Selection of Detection Scales and Localization Scales, Proc. 1st Intl. Conf. Image Processing, vol. 1, pp. 924-928, 1994.

[11] H. C. Liu and M. D. Srinath, Corner Detection from Chain-code, Pattern Recognition, vol. 23, pp. 51-68, 1990.

[12] N. Lüdtke, B. Luo, E. Hancock, and R. C. Wilson, Corner Detection Using a Mixture Model of Orientation, Proc. ICPR 2002, pp. 574-577.

[13] F. Mohanna and F. Mokhtarian, Robust Corner Tracking for Multimedia Applications, Proc. ICIP 2002, pp. 945-948.

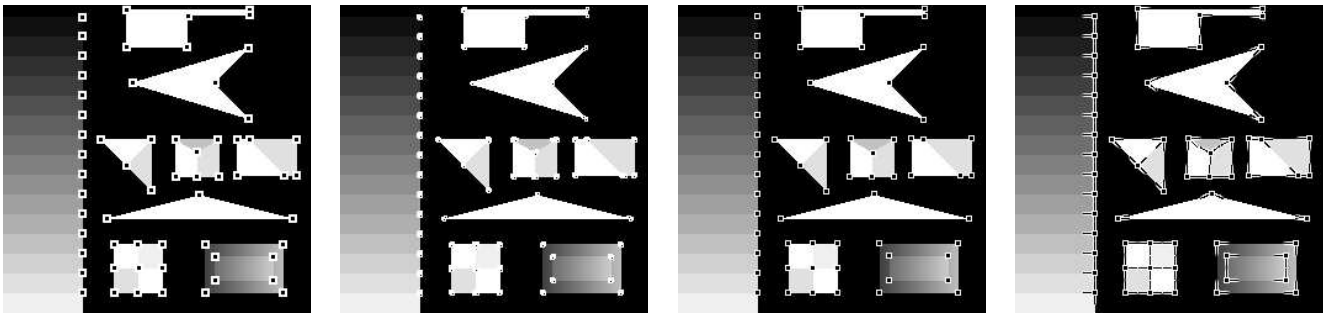
[14] F. Mokhtarian and R. Suomela, Robust Image Corner Detection through Curvature Scale Space, IEEE Trans. Pattern Anal. Machine Intell., vol. 20, pp. 1376-1381, 1998.

[15] K. Rangarajan, M. Shah, and D. V. Brackle, Optimal Corner Detection, Comput. Vision Graphics Image Process., vol. 48, pp. 230-245, 1989.

[16] S. M. Smith and M. Brady, SUSAN - A New Approach to Low Level Image Processing, Intl. J. Comput. Vision, vol. 23, pp. 45-78, 1997.

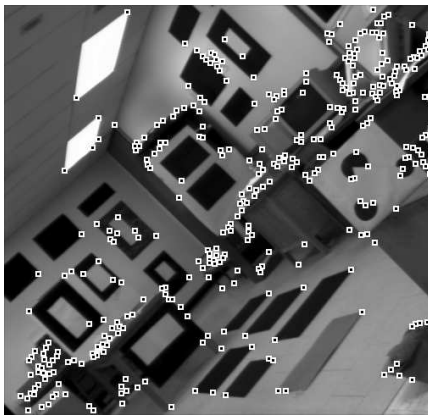
[17] C. Urdiales, C. Trazegnies, A. Bandera, and F. Sandoval, Corner detection based on adaptively filtered curvature function, Electronic Letters, vol. 39, no. 5, pp. 426-428, 2003.

[18] Z. Zheng, H. Wang, and E. K. Teoh, Analysis of Gray-level Corner Detection, Pattern Recognition Lett., vol. 20, pp. 149-162, 1999.

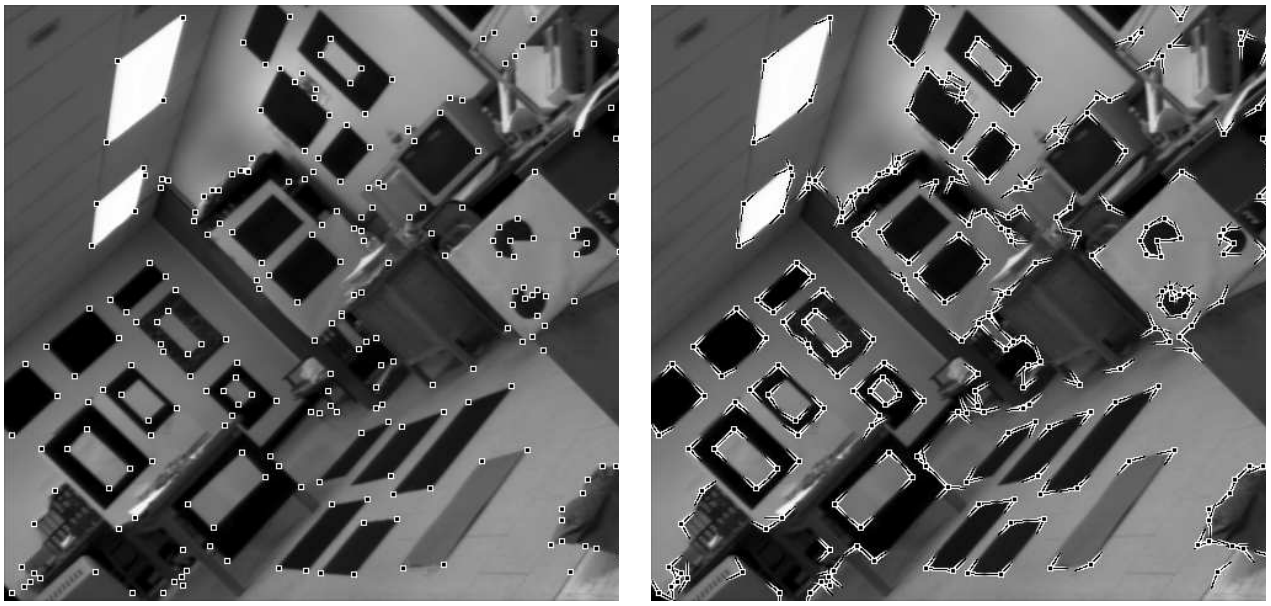


(a) Corners by SUSAN. (b) Corners before merging. (c) Corners after merging. (d) Incident Edges and Corners.

Figure 5. Outputs for Test image (b, c, d: proposed).

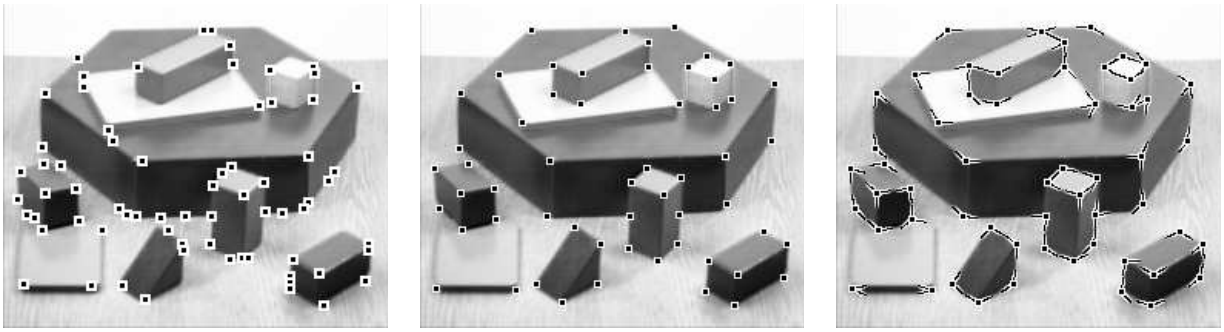


(a) Corners by SUSAN.



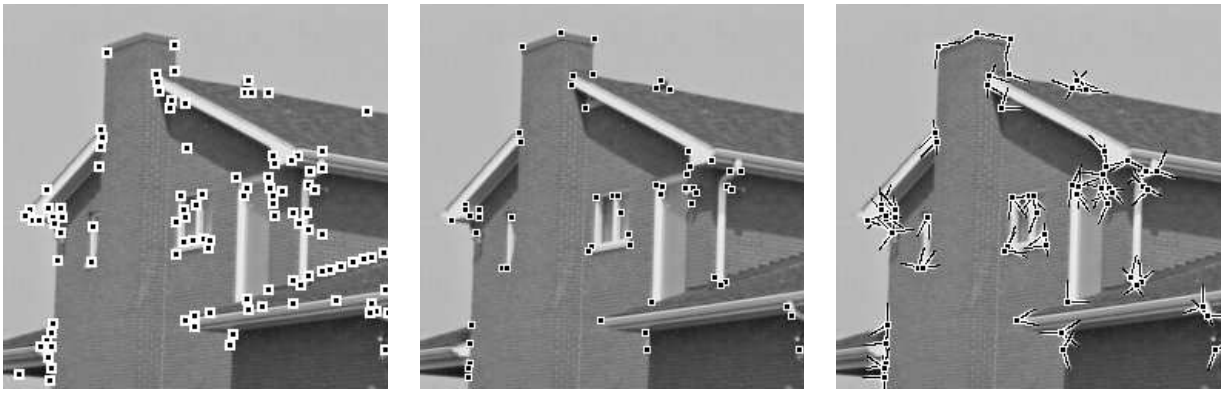
(b) Corners after merging. (c) Incident Edges and Corners.

Figure 6. Outputs for Lab image (b, c: proposed).



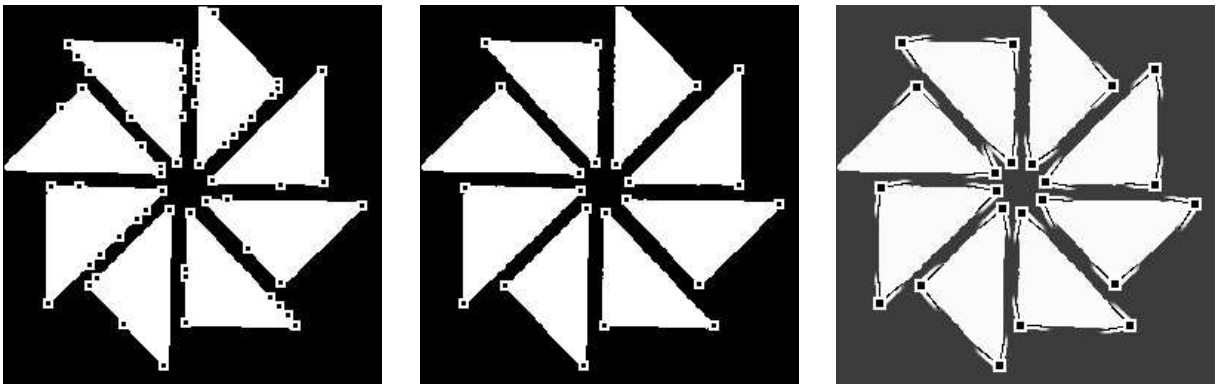
(a) Corners by SUSAN. (b) Corners after merging. (c) Incident Edges and Corners.

Figure 7. Outputs for Blocks image (b, c: proposed).



(a) Corners by SUSAN. (b) Corners after merging. (c) Incident Edges and Corners.

Figure 8. Outputs for House image (b, c: proposed).



(a) Corners by SUSAN. (b) Corners after merging. (c) Incident Edges and Corners.

Figure 9. Outputs for a sample logo image (b, c: proposed).