2014 International Conference on Advances in ICT for Emerging Regions (ICTer): 015 - 020

Locally Adaptive Isotropic Detection of Corners in Object Boundaries

U.A.A. Niroshika[1] and R.G.N. Meegama[2]

[1]*Department of IT, Faculty of Computing, Sri Lanka Institute of Information Technology, Malabe, Sri Lanka.*
[2]*Department of Computer Science, Faculty of Applied Sciences, University of Sri Jayewardenepura, Nugegoda, Sri Lanka.*

[1]aruni.n@sliit.lk, [2]rgn@sci.sjp.ac.lk

***Abstract*— Detection of corners is an important task in computer vision to capture discontinuous boundaries of objects of interest. Present operators designed to detect boundaries having sharp corners often produce unsatisfactory results because the points detected can also be an isolated point, ending of a thin line or a maximum curvature region of a planar curve. A novel corner detection operator, capable of detecting corner points that exist only on the boundary of an object, is presented in this paper. Initially, candidate corner points are detected by exploiting local neighbourhood intensity information and associated connectivity pattern around the center point within a local window. Further verification is done to confirm whether the detected corner point is on the boundary of the targeted object. As the proposed operator is isotropic, it covers all the orientations and corner angles by performing a single computation step within the local window. The performance of the operator is tested with both synthetic and real images and the results are compared with other major corner detectors.**

***Keywords*— Corner detection, Feature point detection, Object detection**

I. INTRODUCTION

Feature point detection is a challenging task in the field of image processing and numerous techniques developed to achieve this task facilitates many computer vision based applications such as motion tracking [1–4], stereo matching [5, 6], image registration [7], object recognition [8, 9], shape detection and camera calibration [10,11]. In such feature based image analysis applications, corner detection plays a major role because corners are the most prominent points on the intensity landscape among other image features. A corner is a special feature point where strong intensity diffusion occurs in more than one direction where the rate of the change of the gradient is maximum [12]. A corner detecting operator should be well localized at its true position, be able to avoid spurious detections, be repeatable and efficient [13].

Object detection is a vital task in many computer vision based applications. Examination of all the points along the boundary of the targeted object is time consuming and computationally complex. This process can be simplified if the examination is performed only on corner points of the object that are more prominent over other boundary points. For example, if the target object is triangular-shaped, the three corners are very good feature points to locate the object. Unfortunately, the existing corner detection operators are devised for general purpose applications rather than being specific to the object detection purpose. Thus, the former corner detection operators are unable to capture the corners that are located on the object boundary. On the other hand, most of the existing corner detection operators often produce unsatisfactory results since they tend to detect all the interest points present in the image rather than specifically detecting corners only. As such, the previous models may detect many spurious points such as isolated points, line endings ,and maximum curvature region of planar curves

II. RELATED WORK

Existing corner detection operators can be classified into two broader categories, namely, template-based and geometry-based corner detectors [14]. Between the given mechanisms, the template-based corner detectors attempt to match a certain corner template over the input image and on the other hand, the geometry-based detectors try to measure the differential geometry features near the corner point. In template-based mechanisms, multiple orientation corner templates are used to cover all types of corners consuming immense computational resources and time. The geometry-based corner detectors are further diversified to many formulations, such as edge related [15, 16], topological feature based [17, 18] and auto-correlation based [19-21] techniques. The edge related models consider a corner point as an intersection of two or more straight lines [22] while topology-based models consider a corner point as a geometry feature point on the image plane.

A variety of corner detection operators are revealed in recent literature and perhaps the most widely used is Harris corner detector [20] which is an extension of the Moravec operator [19]. The Harris operator is based on the local auto-correlation function of a signal in which the function measures the local change of the signal for small movements of patches in a range of directions.

Smith and Brady [21] have proposed a corner detector referred to as SUSAN (Smallest Uni-value Segment with an Assimilating Nucleus) by introducing morphological operations to the corner detection process rather than taking derivatives and image gradients. This model is capable of handling all types of corner regions and junctions and can be used to detect edges as well. Unfortunately, the SUSAN operator suffers from poor localization, low stability and false feature detection when the boundaries are blurred [13].

The FAST (Feature from Accelerated Segment Test) operator, introduced by Rosten and Drummond [23] takes a circular window around the center pixel and it is then partitioned into two groups in order to distinguish between lighter and darker neighbouring pixels. However, it examines only a set of discrete pixels within the circular window and tends to detect multiple feature points adjacent to each other.

978-1-4799-7732-1/14/$31.00 ©2014 IEEE

The SURF (Speeded Up Robust Features) corner detector [24] operator, which is based on 2D Haar wavelets, does not perform well in scale changes.

As seen, none of the above implementations provides a compact operator which avoids detecting false corner points such as isolated points, thin line endings and high curvature curve bends. Furthermore, none of the previous corner detection operators is capable of capturing the corners located only on the boundary of the object of interest while avoiding corners located outside the object.

In this paper, we propose a novel corner detection operator capable of detecting corner points that exist only on the boundary of a target object by avoiding false feature points. For this purpose, the local intensity variation of the input image together with the intensity information of the object is taken into account.

III. METHODOLOGY

A. Detection of false corner points

In derivative-based approaches, the detected corner point can be the end point of a line (or an isolated point) because such operators define a corner point as an image feature characterized by high intensity changes in the horizontal and vertical directions. On the other hand, an edge-based operator tends to detect thin line bends also as corner regions because it considers the corner point as an intersection of two or more straight lines [22]. Therefore, a corner detected using existing operators will result in false positives as indicated in Fig. 1.

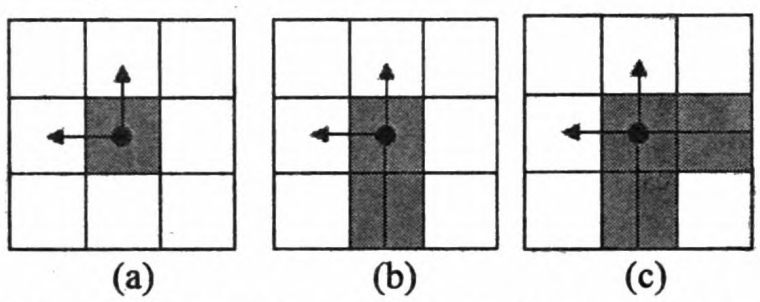

Fig. 1. False feature points: (a) an isolated point, (b) a thin line ending and (c) a high curvature curve bend.

B. Detection of false intensity extremes

Existing corner detection approaches are suitable for general purpose applications rather than specific to object detection purpose. As such, the former corner detection operators capture corners that are located outside the boundary of the object of interest. The topology-based and auto-correlation based corner detection operators tend to detect all the intensity extremes that exist inside or outside the object of interest as they realize on intensity distribution within a small window. As a result, such operators may detect spurious corners not truly connected with an object.

The proposed corner detection operator consists of three main stages as described below.

Stage 1: Detection of similar neighbors of the center pixel

The candidate corner points are detected based on the intensity diffusion within a 3x3 local window during the first two stages. The operator is initiated by exploiting local neighborhood intensity information around the center point and determining the intensity change between the center point and each of its neighboring pixels.

The intensity difference I is computed between the center point and each of its eight neighboring pixels residing within a distance $(\Delta u, \Delta v)$, as given by:

$$I = w\,|\,f(x, y) - f(x + \Delta u, y + \Delta v)\,| \tag{3}$$

where w is the local window of size 3×3 , $f(x, y)$ is the intensity of the center pixel, $f(x + \Delta u, y + \Delta v)$ is the intensity of each of the neighboring pixels within the window.

The next step is to detect the neighbours who are similar in intensity to the center pixel. The following decision is taken to extract such similar neighbours into a separate point set S:

$$D = \begin{cases} \mathbf{P}_{(x+\Delta u,\, y+\Delta v)} \in S & \text{if } I \le T_i \\ \mathbf{P}_{(x+\Delta u, y+\Delta v)} \notin S & \text{otherwise} \end{cases} \tag{4}$$

where $\mathbf{P}_{(x+\Delta u,\ y+\Delta v)}$ is a pixel belonging to the eight neighbours around the center pixel within the given 3×3 window, T_i is the user defined threshold and S is the point set.

Therefore any neighbouring pixel point $\mathbf{P}_{(x+\Delta u,\ y+\Delta v)}$, whose intensity difference I is less than or equals the threshold T_i is an element of the point set S. The point set S is defined as:

$$S = \{\, \mathbf{P}_{(x+\Delta u,\, y+\Delta v)} :\ I \le T_i \} \tag{5}$$

Based on the above predicate, the point set S holds all the neighbouring pixels whose intensities are similar to the corresponding center pixel.

Stage 2: Detection of candidate of corner regions

This stage determines whether the detected neighbouring pixel points within S and the center pixel together form a corner. A local search is performed to find connectivity patterns around the center pixel as illustrated in Fig.2. In order to assemble such patterns around the center pixel, S should satisfy following conditions simultaneously.

Condition 1: The total number of neighbouring pixels within S should be greater than one and less than five.
Condition 2: A connected neighbourhood should exist between all the neighbouring pixels within S. If there are n number of pixels within the set S, a circular array *Circular_Array* [ ] of size n is created as:

$$Circular\_Array\,[\,] = \{\mathbf{P}_{i(x+\Delta u,\, y+\Delta v)} \mid i = 0,1,2,\ldots,n-1\} \tag{6}$$

Then, consider the spatial coordinates of each pixel $\mathbf{P}_{(x+\Delta u,\ y+\Delta v)}$ within a 3×3 local window and arrange them in clockwise order around the center pixel. While preserving the clockwise order, these pixels are placed in the *Circular_Array* [ ] along with their spatial coordinates. Let all the pixel points within the *Circular_Array* [ ] be vertices of an undirected graph $G = (V, E)$. There is a connected neighbourhood $N_G(v)$ existing among all the vertices of the graph G; if each and every vertex v_i of G has at least one adjacent neighbour within a distance of one pixel (orthogonal directions). As such, each pixel in *Circular_Array* [i] should be an adjacent neighbour of the pixel at *Circular_Array* [i-1] or *Circular_Array* [i+1] or both in the coordinate space.

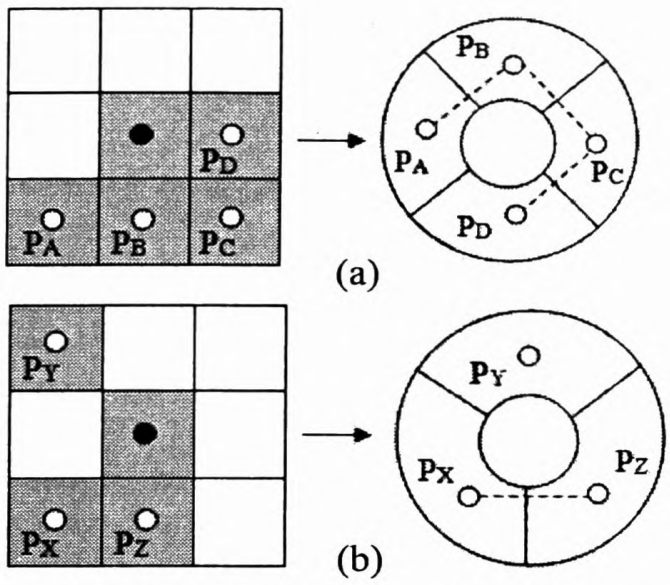

Fig. 2. (a) A possible corner region and its corresponding circular array of size 4 where each and every element has at least one adjacent neighbour and (b) Uninterested region and its corresponding circular array of size 3 with a disjointed element without any adjacent neighbors. Edges of the adjacent points within one pixel difference are defined in dotted lines

The image patches (a) and (b) shown in Fig. 2 both comply with the above condition 1. The image patch given in Fig.2.(a) complies condition 2 too because all the pixels within S are connected neighbours as every element at *Circular_Array* [ ] has at least one adjacent neighbour apart from one pixel distance within the spatial coordinate space. The image patch illustrated in Fig.2.(b) does not satisfy condition 2 as the corresponding circular array contains a disjointed element ($\mathbf{P}_y$) without any neighbors.

If S satisfies both conditions 1 and 2, then all the neighboring pixels, together with the center, form any of the local corner regions as illustrated in Fig. 3. In such a case, the center pixel $\mathbf{P}_{(x,y)}$ is considered as a corner candidate. The proposed corner detection is isotropic as the expected corner patterns are found in anywhere around the center pixel regardless of the direction.

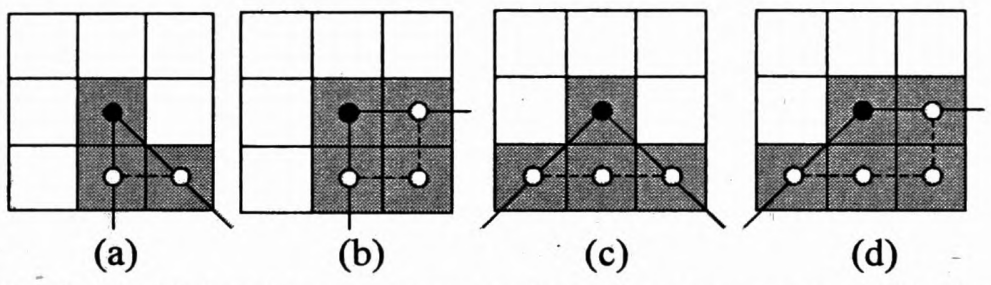

Fig. 3. (a) - (d): Local image structures of possible corner patterns within 3 × 3 window. The center pixel $\mathbf{P}_{(x,y)}$ is denoted using a filled circle, neighboring pixels within the set S are in empty circles, possible edge connections are in dotted lines and corner regions are highlighted in solid lines.

Stage 3: Determine whether the detected corner is located on object boundary

After all the possible corner candidates are detected, it is required to verify whether a detected corner point $\mathbf{P}_{(x,y)}$ is located actually on the boundary of an object. The similarity of the corner point $\mathbf{P}_{(x,y)}$ is measured relative to a pixel that resides inside the target object. The distance d between the corner point and the relative point to be examined is defined by the user and should be set according to the size and illumination factors of the target object.

The corresponding circular array *Circular_Array* [ ] of $\mathbf{P}_{(x,y)}$ is examined to obtain the head and tail points (Let $\mathbf{P}_h$ is the head and $\mathbf{P}_t$ is the tail) where both have exactly one adjacent neighbor in the spatial coordinate system. Next, the angle $\mathbf{P}_h\,\mathbf{P}_{(x,y)}\,\mathbf{P}_t$ is divided by a straight line as illustrated in Fig. 4 and traversed into the target object up to a distance d to find the intensity of a relative pixel $\mathbf{P}_{relative}$.

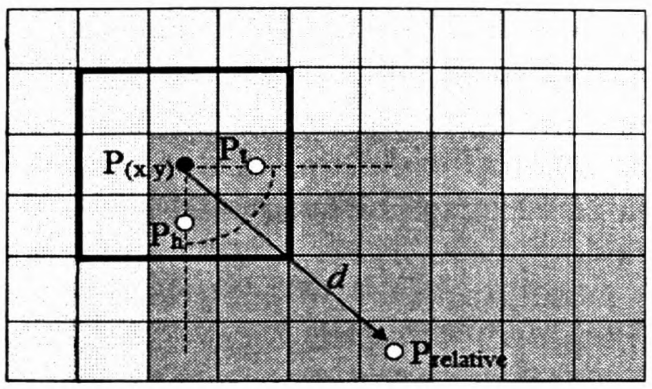

Fig. 4. : A section of a target object where $\mathbf{P}(x,y)$ is the detected candidate corner point with three similar neighbours.

Subsequently, the intensity difference I_{dif} is computed between the corresponding candidate corner point $\mathbf{P}_{(x,y)}$ and the relative pixel $\mathbf{P}_{relative}$. If $I_{dif} \leq T_d$ (user defined threshold), the candidate corner point $\mathbf{P}_{(x,y)}$ is considered as a corner point associated with the object. The value of T_d should be set according to the nature of the targeted object. For instance, if the intensity distribution of the target object is uniform, T_d should be low. The following algorithm explains the proposed initiative in brief.

ALGORITHM:

```
//w = Local window of size 3 × 3
//P (x , y) = Center pixel within the window w
//P (x + Δu, y + Δv) = neighboring pixel within the window w
//P relative = The relative pixel to be examined
//I and I dif = Intensity differences
//T i and T d = User defined threshold values
// d = Distance between the corner point and the relative pixel
// P h and P t = Head and the tail of the circular array
//P i = Any pixel point within the grayscale image of size M×N

For each pixel P i;
  P (x,y) = P i
  For each pixel point P (x,y) :
     For each neighboring pixel P (x+Δu, y+Δv) within w
        1.1: Compute I = w x, y | f (x , y) – f (x + Δu, y + Δv) |
           If I ≤ T i
                 1.1.1: Add the pixel point P (x + Δu, y + Δv) into
                        the candidate pixel set S
           End If
  End For
        1.2: Construct the circular array
        1.3: Get the total no. of elements n in the pixel set S
           If (1< n < 5 AND connected neighborhood exist in
              within the circular array)
              1.3.1: Obtain P h and P t from the
                   Corresponding circular array
              1.3.2: Bisect the angle P h P (x,y) P t
                   and traverse the distance d
              1.3.3: Obtain the intensify of P relative
              1.3.3: Compute I dif
                 If I dif ≤ T d
                   P (x,y) is a corner connected to an object
        Else
              Proceed to the next pixel, recall the Step 1.
        End If
End for
```

IV. Results and Discussion

This section presents the results obtained from the implementation of the proposed algorithm in an Intel Core 2 Duo 2.66 GHz desktop computer. The proposed model is tested on a synthetic image of simple 2D shapes and a real image of an outdoor scene. These images, consisting of sharp corners, are specially selected in order to demonstrate the ability of the proposed operator in terms of detecting corners present in an image. For all the experiments, the model parameters are empirically selected as: $T_i = 12$ and $T_d = 12$. The Gaussian filter is applied to make the edges smoother.

In order to evaluate the accuracy of the operator, we consider the total number of accurate detections, false detections, redundant multiple detections and total number of corners that are missed where the *error* is defined as the summation of false, missed and redundant detections. These parameters provide a measure of the accuracy of the corner detection with respect to a reference solution image created by detecting all the true corners of each test image manually by carefully examining the corresponding intensity matrix of the image.

The performance of the proposed corner detection operator is compared with other major landmark corner detectors, operators, namely, Harris [20], SUSAN [21], FAST [23], and SURF [24].

A. Experiments with a synthetic image

A synthetic image of 256x256 pixels, where the region of interest consists of many types of corners and junctions (L, T, X and Y), is selected as shown in Fig. 5. This test image, having altogether 32 corners, is widely used in the literature [16, 21] to test the performance of corner detection operators. In this experiment, each of the corners detected is again compared with the relative pixel (where $d = 3$) that resides inside each object. The other four models are applied on the same test image in order to compare the results.

TABLE 1.
COMPARISON OF MEASUREMENTS FOR DIFFERENT OPERATORS APPLIED ON THE IMAGE IN FIG. 5

Operator	Accurate detections	False detections	Missed corners	Redundant detections	Error
Harris	30	27	2	1	30
SUSAN	32	31	0	0	31
FAST	19	0	13	7	20
SURF	12	63	20	2	85
Proposed	30	0	2	0	2

As seen in Fig. 5, the proposed operator visually demonstrates the capability of capturing corners precisely compared to the other four operators. Table 1 gives the quantitative results taken on the outputs given in Fig. 5. As seen, the Harris, SUSAN and SURF operators ended up with too much false detection. The Harris operator responded poorly on diagonal edges while the FAST operator missed a high amount of true corners. The proposed model managed to detect 30 true corners without any redundant points and false detections. It is also noticeable that the error value is considerably low in the proposed operator. Further, the proposed model managed to detect corner points that exist only on the boundary of the object while avoiding all the intensity extremes that are located outside the object.

B. Experiments with a real image

The corners in the real images are extremely difficult to detect using most of the previous operators due to low contrast, illumination, noise, etc. The capability of the proposed operator to detect corners present in a real image is tested using an outdoor image, which is widely used in image processing and vision applications literature. There are 42 corner points the reference image. Figure 6 illustrates the output after executing each operator separately on the test image whereas Table 2 gives the corresponding measurements of this experiment.

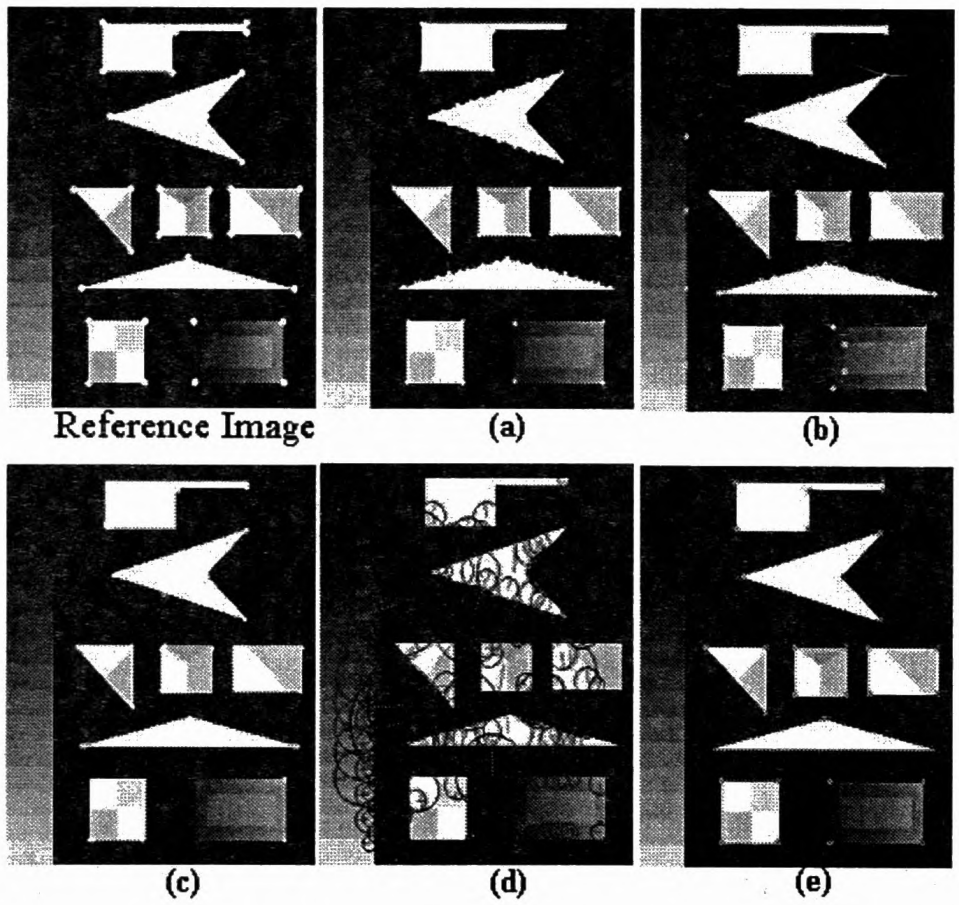

Fig. 5. : Illustrations of corner detection on a synthetic image. (a) Harris, (b) SUSAN, (c) FAST, (d) SURF and (e) proposed operator.

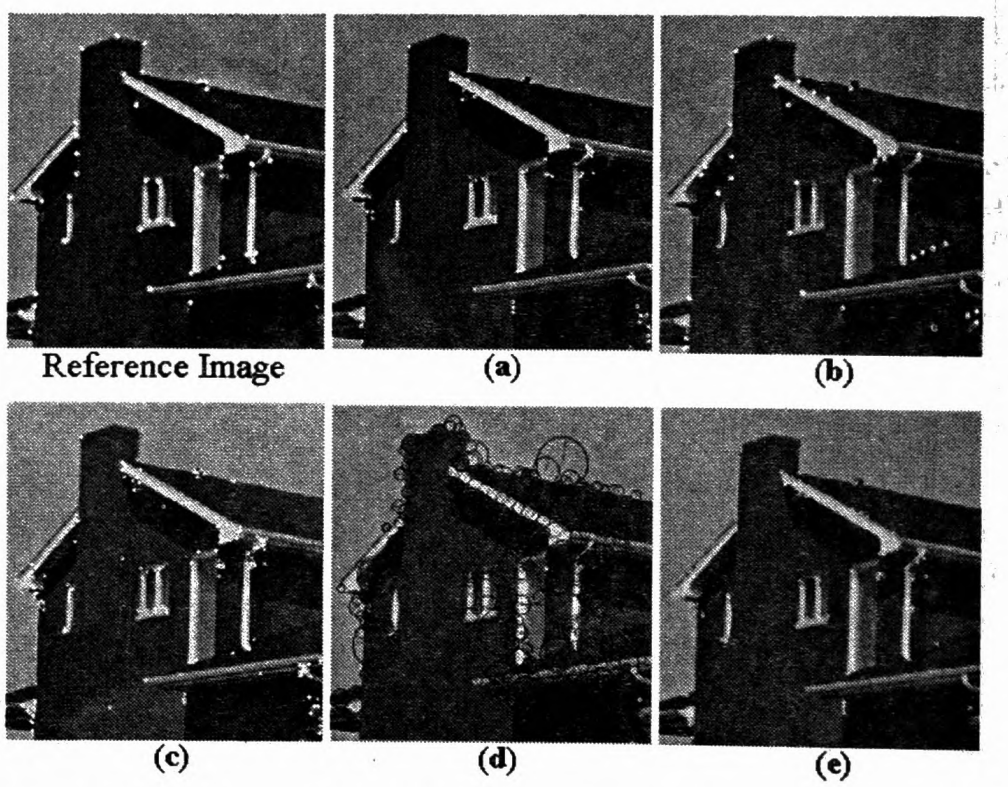

Fig. 6. : Illustrations of corner detection on a real image. (a) Harris, (b) SUSAN, (c) FAST, (d) SURF and (e) proposed operator.

TABLE 2.
COMPARISON OF MEASUREMENTS FOR DIFFERENT OPERATORS APPLIED ON THE IMAGE IN FIG. 6.

Operator	Accurate detections	False detections	Missed corners	Redundant detections	Error
Harris	31	30	11	3	44
SUSAN	16	10	26	0	36
FAST	15	74	27	7	108
SURF	29	63	13	5	81
Proposed	30	18	12	5	35

According to the values obtained in Table 2, the Harris operator has a higher detection rate but with a high rate of false detections. On the other hand, the SUSAN operator detected only a few corners. Both the FAST and SURF operators produced not only high false positives but also poor localization near detected corners. It is clearly evident that the FAST operator is not capable of detecting prominent corners in this application. In contrast, the proposed model managed to detect true corner points with fewer false detections than others.

Another real image of 256x256 pixels, where the region of interest consists of many types of corners is selected as shown in the Fig. 7. There are 66 corner points the reference image. Fig. 7 (a) – (e) shows the output images after executing five different corner detection operators on the same image whereas Table 3 gives the corresponding measurements of this experiment.

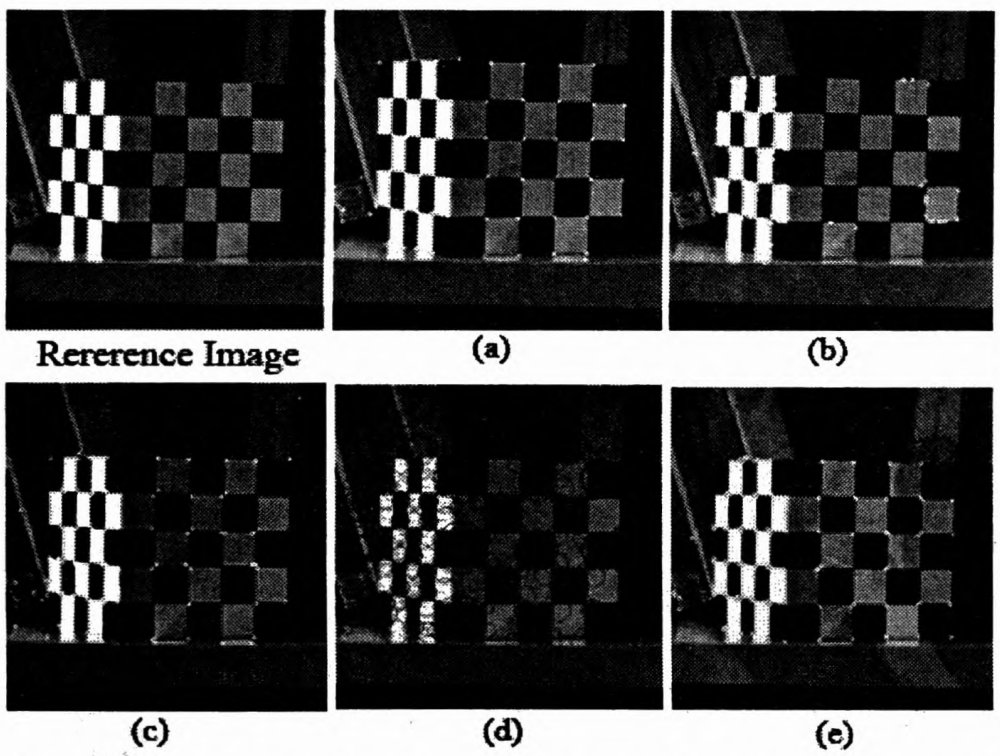

Fig. 7. : Illustrations of corner detection on a real image. (a) Harris, (b) SUSAN, (c) FAST, (d) SURF and (e) proposed operator.

TABLE 3.
COMPARISON OF MEASUREMENTS FOR DIFFERENT OPERATORS APPLIED ON THE IMAGE IN FIG. 7.

Operator	Accurate detections	False detections	Missed corners	Redundant detections	Error
Harris	61	11	5	17	33
SUSAN	28	10	38	10	58
FAST	42	74	24	12	110
SURF	30	63	36	27	126
Proposed	60	10	6	20	36

According to the values obtained in Table 3, the Harris operator has a higher accurate detection rate with a lesser error rate. As seen, the FAST and SURF operators ended up with too much false detection. On the other hand, the SUSAN operator detected only a few corners as the previous experiment. Both the Harris and proposed operators exhibit a competent rate of accurate detections. In conclusion, the final error rate of the proposed operators is better than SUSAN, FAST, and SURF.

We have further examined the computational time of the proposed operator. The other four models are applied on the same test image in Fig. 7, in order to compare the results. Table 4 shows the exact time taken to detect the corners (accurate and redundant). This experiment was done on an Intel Core 2 Duo 2.66 GHz desktop computer, without code optimization.

TABLE 4.
EXECUTION TIME COMPARISON ON THE IMAGE IN FIG. 7.

Operator	Computational Time (miliseconds)
Harris	49
SUSAN	85
FAST	52
SURF	163
Proposed	142

It is clearly noticeable that the proposed operator is slower that Harris, SUSAN, and FAST. This result can be due to the high computational effort needs to execute both Stage 2 and 3. It takes a considerable time at Stage 2, in order to carry out the local search to find the connectivity pattern around the center pixel. Consequently, the time taken to traverse into the targeted object at the Stage 3 is also high.

V. CONCLUSIONS

In this paper, we present a new corner detector that ignores corner points that exist outside the object of interest and at the same time, captures the actual corner points at a higher accuracy than previous techniques. The results obtained after applying the operator on both synthetic and real images show the performance of the proposed operator. Further studies on improving the proposed algorithm to obtain accurate results on real images having a high amount of noise and low contrast need to be explored.

REFERENCES

[1] L. Shapiro, *Affine Analysis of Image Sequences,* Cambridge University Press, 1995.

[2] C. Tomasi, T. Kanade, "Shape and motion from image streams under orthography: a factorization", *Int. Journal of Computer Vision* 9 (2) pp. 137-154, 1992.

[3] C. Harris, Structure from motion under orthographic projection, in *Proc. 1st ECCV*, pp. 118-123, 1990.

[4] L. Forlenza, P. Carton, D. Accardo, G. Fasano, and A. Moccia, "Real time corner detection for miniaturized electro-optical sensors onboard small unmanned aerial systems", *Sensors*, vol. 12, pp. 863-877, 2012.

[5] J. Basak and S. K. Pal, "PsyCOP: A psychologically motivated connectionist system for object perception", *IEE Trans. on Neural Networks*, vol.6, pp.1337-1354, 1995.

[6] D. H. Ballard and C. M. Brown, *Computer Vision*, Englewood Cliffs, NJ: Prentice-Hall, 1982.

[7] B. Zitova and J. Flusser, "Image registration methods: A Survey", *Image Vision Computing*, vol. 21, pp. 977-1000, 2012.

[8] M. H. Han and D. Jang, "The use of maximum curvature points for the recognition of partially occluded objects", *Pattern Recognition,* vol. 23, pp.1223-1233, 1990.

[9] I. D. Reid and D. W. Murray, "Active tracking of foveated feature clusters using affine structure", *Int. Journal of Computer Vision, vol.* 18, pp. 41-60, 1996.

[10] A. D. Escalera and J. M. Armingol, "Automatic chessboard detection for intrinsic and extrinsic camera parameter calibration", *Sensors, vol. 10,* pp.2027-2044, 2010.

[11] H. Weerasena, P. Bandara, J. Kulasekara, B. Dassanayake, U. A. A. Niroshika, and P. Wijenayake, "Strategic approach for high performance object tracking in a network of surveillance cameras", in Pro. IEEE 7th Int. Conference on Information Assurance and Security (IAS2011), Malacca, Malaysia. pp. 280-285, 2011.

[12] K. Rangarajan, M. Shah and D. V. Brackle, "Optimal Corner Detector", *Computer Vision, graphics and image processing* 48, pp.230-245, 1989.

[13] D. Zhou, Y. Liu, and X. Cai, "An efficient and robust corner detection algorithm", in Proc. 5th World Congress on Intelligent Control", China, pp.4020-4024, 2004.

[14] Z. Zheng, H. Wand, and E. K. Teoh, "Analysis of gray level corner detecton", *Pattern Recognition Letters*, vol. 20, no. 2, pp.149-162, 1999.

[15] L. Kitchen and A. Rosenfeld, "Gray-level corner detection", *Pattern Recognition Letters 1*, pp.95-102, 1982.

[16] H. Wang and M. Brady, "Real-time corner detection algorithm for motion estimation", *Image and Vision Computing*, 13(9), 1995.

[17] R. Deriche and G. Giraudon, " A computational approach for corner and vertex detection", *International Journal of Computer Vision* 10 (2), 101-124, 1993.

[18] P. R. Beaudet, P.R., "Rational invariant image operators", in Proc. 4th International Conference on Pattern Recognition, pp. 579-583, 1978.

[19] H. P. Moravec, "Towards automatic visual obstacle avoidance". In Proc. 5th Int. Joint Conference on Artificial Intelligence, Cambridge, MA, USA, pp. 584-589, 1977.

[20] C. Harris and M. Stephens, "A combined corner and edge detector". In Proc. Alvey vision conference, Manchester, UK, , pp. 147-151, 1988.

[21] S. M. Smith and J. M. Brady, "SUSAN- A new approach to low level image processing", *Int. Journal of Computer Vision*, vol. 23, no. 1, pp. 45-78, 1977.

[22] M. Awrangjeb, G. Lu, and C. S. Fraser, "Performance comparisons of contour-based corner detectors" , *IEEE Trans. on Image Processing*, vol. 21, pp. 4167-4179, 2012.

[23] E. Rosten, R. Porter, and T. Drummond, "Faster and Better: A machine learning approach to corner detection", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 32, No. 1, pp. 105-119, January 2010.

[24] H. bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-up robust features (SURF)", *Int. Journal of Computer Vision and Image Understanding*, vol. 110, no. 3, pp. 346-359, 2008.