Edge Detection Method Driven by Knowledge-Based Neighborhood Rules

Yavuz Çapkan[1,*], Halis Altun[2], Can Bülent Fidan[3]

[1]Department of Mechatronics Engineering, Yildiz Technical University, Istanbul, Turkey

[2]Department of Software Engineering, Istanbul Health and Technology University, Istanbul, Turkey

[3]Department of Mechatronics Engineering, Karabuk University, Karabuk, Turkey

Received 18 March 2022; received in revised form 30 April 2022; accepted 01 May 2022

DOI: https://doi.org/10.46604/ijeti.2023.9710

Abstract

Edge detection is a fundamental process, and therefore there are still demands to improve its efficiency and computational complexity. This study proposes a knowledge-based edge detection method to meet this requirement by introducing a set of knowledge-based rules. The methodology to derive the rules is based on the observed continuity properties and the neighborhood characteristics of the edge pixels, which are expressed as simple arithmetical operations to improve computational complexity. The results show that the method has an advantage over the gradient-based methods in terms of performance and computational load. It is appropriately four times faster than Canny method and shows superior performance compared to the gradient-based methods in general. Furthermore, the proposed method provides robustness to effectively identify edges at the corners. Due to its light computational requirement and inherent parallelization properties, the method would be also suitable for hardware implementation on field-programmable gate arrays (FPGA).

Keywords: image processing, edge detection, computer vision, image analysis

1. Introduction

As a fundamental operation in image and video processing, edge detection is still a vivid but demanding research area. In literature, one might come across approaches that use gradient-based traditional methods [1-3] and machine learning-based studies that have recently become very popular [4-9]. Although machine learning-based approaches perform much better than traditional gradient-based methods, the high computational load induced by these methods is a serious concern that should be considered [10]. There are a variety of application areas of edge detection, such as machine vision [4, 11], smart vehicle technologies [12-14], medical image processing [15-16], and security applications [17].

There are still efforts to improve edge detection in terms of efficiency and computational complexity. The recent trend of using machine learning in edge detection shows superior performance compared with classical gradient-based algorithms but creates a computation load [7, 10, 18]. Therefore, especially for hardware implementation, fast edge detection methods that do not require high processing power are still widely required, and new novel approaches are being proposed [19-20]. Peng-o et al. [10] propose a light algorithm that improves processing speed and energy consumption by decreasing the number of arithmetic operations for Sobel edge detection. Xuan et al. [19] have focused on the limitations of the traditional canny method. They have proposed a differential processing on the amplitude gradient histogram, which eases the limitation of the algorithm. Experiments have shown that the proposed method is resistant to noise and can successfully separate the background [19].

Günen et al. [21] proposed a backward search clustering-based edge detection algorithm to obtain edges in noisy images. They concluded that although the result statistically differs from the ground truth, it performs better than the traditional

* Corresponding author. E-mail address: yavuzcapkan@outlook.com

methods [21]. Cao et al. [22] have proposed a method to improve edge detection using MapReduce parallel programming model. Their aim is to improve the performance of Canny operator by using Otsu method to optimize the binary threshold. Furthermore, it is shown that the speed and computational cost problems have been alleviated. In their study, Mittal et al. [23] proposed an edge detection algorithm called BEdge. BEdge algorithm can provide a solution that leverages the two main limitations in edge detection; edge connectivity and edge thickness, namely. Zhang et al. [20] improved the gradient pattern in Sobel edge algorithm and implemented their approach on field-programmable gate arrays (FPGA). Al-Ghaili et al. [24] proposed a pixel density-based contrast algorithm for edge extraction. They used a mask to extract the object's edges.

The main aim of edge detection is to obtain a binary image that contains only true-edge pixels in a given image. In a gradient-based approach, this task is carried out by rejecting all non-edge pixels based mainly on the gradient information. The steps to obtain an edge in the gradient-based algorithms might be roughly categorized as follows: 1) image smoothing to reduce noise; 2) gradient calculation by extracting the gradient of the pixels; 3) detection of edge pixels by applying a threshold on the gradient; 4) rejection of the spurious edges. Finally, a binary image that represents edges is obtained at the end of the steps. In the proposed edge detection, on the other hand, a binary image is obtained at the very beginning. Then, the binary image is evaluated to detect edge pixels and reject non-edge ones, using a rule-based approach instead of gradient calculation, which requires a great deal of computational load [25].

The primary objective of this study is to reduce the computational load by removing the convolution operations to calculate gradients. Therefore, knowledge-based rules are extracted to classify the pixels as edge pixels or non-edge pixels. The rules are based on the knowledge that indicates the continuity properties of the neighborhood in the horizontal or vertical direction. It will be shown that the proposed method performs well or at least fairly comparable to the traditional gradient-based methods in terms of performance and processing speed. Two quantitative metrics, peak signal to noise ratio (PSNR) [26] and structural similarity index (SSIM) [27], which are frequently encountered quantitative metrics in the literature, are used to evaluate the results. On the other hand, the edge obtained using the proposed method and the ones from the gradient-based methods are also illustrated in the figures for qualitative evaluation. Since the proposed method does not rely on gradient calculation, it has an advantage over the gradient-based methods in terms of computational load. As a result, it is possible to obtain a much quicker algorithm. It is also shown that the proposed method is more efficient at detecting edges on the corners of patterns in an image compared to gradient-based methods.

The organization of the work is given as follows. In the next section, Section 2, the proposed algorithm is described. The section elaborates on the extraction of knowledge-based rules and implementation details. In Section 3, the datasets are described, and the metrics used to evaluate the results are introduced. The results and evaluation of the proposed method are discussed in Section 4. The conclusion is drawn in Section 5, along with the direction for future works.

2. Proposed Edge Detection Method

The proposed method relies on knowledge-based rules. The first step in the method is to obtain a binary image from a given gray-scale image. The binary image consists of candidate edge pixels, along with non-edge pixels. The aim is to reject non-edge pixels based on a set of rules and to retain only the "true" edge pixels as much as possible. A kernel of 2 pixels by 2 pixels in size is defined to extract a set of rules. The knowledge-based rules rely on the continuity properties of the edge pixels on a neighborhood region.

Fig. 1 shows a section of the binary image where couples of frames are placed that defines a neighborhood region. If one inspects the neighborhood characteristics of the pixels in the frames, it is realized that there are some discontinuity behaviors. Investigating the pixels in the frames reveals that edge and non-edge pixels have a certain type of characteristic in horizontal and vertical directions. For example, in Frame A and Frame B, the pixels do not have any discontinuity. This observation leads

to an inference that the pixels in the frames are non-edge pixels. Therefore, these pixels should be rejected and labeled accordingly. On the other hand, Frame C and Frame D contain edge pixels vertically or horizontally. Furthermore, Frame E shows a discontinuity behavior between pixels and has an edge pixel at the corner. This characteristic behavior is exploited to extract a set of knowledge-based rules formulated in terms of row and column sum. Using the knowledge-based rules, it is possible to evaluate all pixels by just using a simple arithmetic operation instead of using more complex gradient-based operations. Based on this observation, the following algorithmic approach has been extracted and given in Table 1.

A frame of 2 pixels by 2 pixels is used to scan the whole binary image. This frame is called a neighborhood frame. Starting from the top-left point, (1,1) position of the image, a scanning operation is carried out to investigate a neighborhood relation between pixels in the frame. If there is a discontinuity between the pixels in horizontal or vertical directions at the current location of the frame, the corresponding pixels are classified as edge candidates. At the end of the process, all pixels in the image are evaluated based on the knowledge-based rules in Table 1. In the algorithm, a matrix called e(i, j) is defined to keep track of the promoted pixel position in the current frame. If one of the pixels in the current frame is classified as an edge, a corresponding element in the E matrix is set accordingly. The same pixel would be present in the neighborhood in the next sweep of the frame. If the pixel still satisfies the condition of being in a neighborhood, the algorithm rechecks this condition and re-updates the corresponding pixel position in the matrix e(i, j).

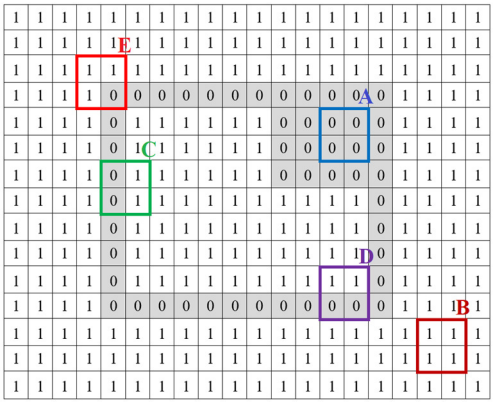

Fig. 1 Couple of frames at different locations in a binary image

Table 1 Knowledge-based rules for promoting pixels as edge candidates in a 2-by-2 neighborhood frame

a) Extract row and column sums in the frame

$$r_i = \sum_{j=1}^{2} p(i,j), \ where \ i = 1,2 \tag{1}$$

$$c_j = \sum_{i=1}^{2} p(i,j), \ where \ j = 1,2 \tag{2}$$

b) Calculate S (sum of row and column sums)

$$S = \sum_{k=1}^{2} (r_k + c_k) \tag{3}$$

c) Check the condition to promote the pixel in rows of the frame

$$\begin{aligned} &If \ S \in \{4,6\} \ and \ if \ r_i = 2 \ where \ i = 1,2 \\ &then \ e(i,j) = 1, \ where \ j = 1,2 \end{aligned} \tag{4}$$

d) Check the condition promote the pixel in columns of the frame

$$\begin{aligned} &If \ S \in \{4,6\} \ and \ if \ c_j = 2 \ where \ j = 1,2 \\ &then \ e(i,j) = 1, \ where \ i = 1,2 \end{aligned} \tag{5}$$

Note: $e(i, j)$ is a matrix to keep track of the promoted pixel position in the frame.

(a) Binary (b) Proposed case 1 (c) Proposed case 2 (d) Sobel (e) Prewitt (f) Roberts (g) Canny

Fig. 2 Results of the edge detection methods for the inverted smiley face

(a) Binary (b) Proposed (c) Sobel (d) Prewitt (e) Roberts (f) Canny

Fig. 3 Results of the edge detection methods for the non-inverted smiley face

The proposed method has been implemented in MATLAB. On the other hand, the built-in MATLAB function called `edge()` is used for the gradient-based techniques (i.e., Prewitt, Robert, Sobel, and Canny). Default values have been used for the build-in function `edge()` except for the parameter `threshold` set to 0.1.

The condition to promote a pixel requires the sum of row or column being $S \in \{4, 6\}$ as indicated in Table 1. Normally there is another condition where S = 2, which indicates a discontinuity in the neighborhood. However, in the proposed algorithm, this condition is omitted, as it does not provide considerable improvement in the performance of the proposed algorithm. Furthermore, the condition of S = 2 corresponds to the single "white" pixel in the current frame. The single "white" pixel does not hold neighborhood conditions, and it could be a "salt-pepper" type noisy spurious pixel instead of being an edge pixel. An inverted smiley face in Fig. 2(a) is taken as a case to make the concept clearer.

Edge detection methods on the inverted smiley face produce the following results given in Fig. 2. As seen, the proposed method can conserve the structure of the smiley face more successfully compared to the rest of the edge detection methods. However, because the condition S = 2 is not assumed to be a valid condition in the proposed method, the single "white" pixels have not been promoted as edge pixels. This condition results in a loss as seen Fig. 2(b). But this is not a weakness of the proposed algorithm. Instead, it is a case that the algorithm cares about. In the resultant image, "single" white pixels are omitted as they are not in a neighborhood. The proposed algorithm assumes that these could be noise-like pixels, and all the single "white" pixels are classified as non-edge pixels, as seen in Fig. 2(b).

In the second case, if the condition of S = 2 is assumed to be valid, the proposed method seems to be more successful in finding the edges, as seen in Fig. 2(c). However, in general, the single "white" pixel could be a noisy pixel and consequently results in a deteriorated performance.

In the second experiment, a non-inverted version of the smiley face in Fig. 3(a) is used. In this case, as there are no single "white" pixels in the horizontal or vertical direction, the proposed algorithm can detect the edges successfully, while the gradient-based methods fail, as shown in Fig. 3.

3. Dataset and Metrics for Comparison

In the comparison of the proposed method with respect to gradient-based ones, three different datasets and two metrics are used. The results for qualitative and quantitative comparisons are provided in tables and figures. In the first experiment, the dataset contains images from AlphabetRecognizer dataset [28]. In the experiments, the images are first of all converted to binary images using the MATLAB `imbinarize()` function. Three images for the characters a, p, and c taken from the dataset. The images have also been scaled down by 50%. Fig. 4 shows all six images and the edges produced by the edge detection methods. The reason for choosing this dataset is to draw attention to the performance of the proposed method for non-complex images, especially in detecting edge pixels in corners of patterns, and to provide a comparison with the gradient-based methods. Visual inspection along with PSNR and SSIM metrics reveals that the proposed method can detect edges more successfully than gradient-based methods. The results for the selected images are presented in Fig. 4, Table 2, and Table 3.

It is clearly seen from Fig. 4 that the proposed edge detection method can detect the edges more correctly than the rest of the methods. In particular, the performance of the proposed method is better for scaled-down images. When the results are examined qualitatively, it can be easily seen that the proposed edge detection method is clearly able to detect the edges of the letter patterns perfectly, while the rest of the methods fail to do so. The proposed edge detection method achieved the best results for all images. The results, shown in bold in Table 2 and Table 3, indicate that the proposed method outperforms the rest of the methods in this first experiment.

Furthermore, a font set from the dataset is selected, which consists of 30 letters. All images in this set have been reduced by half to obtain a scaled-down version of the images (AlphabetRecognizer/src/res/trainingData/source) [29]. The results of PSNR are provided in Table 4. All results obtained in the first experiment will be thoroughly evaluated and discussed in Section 4.

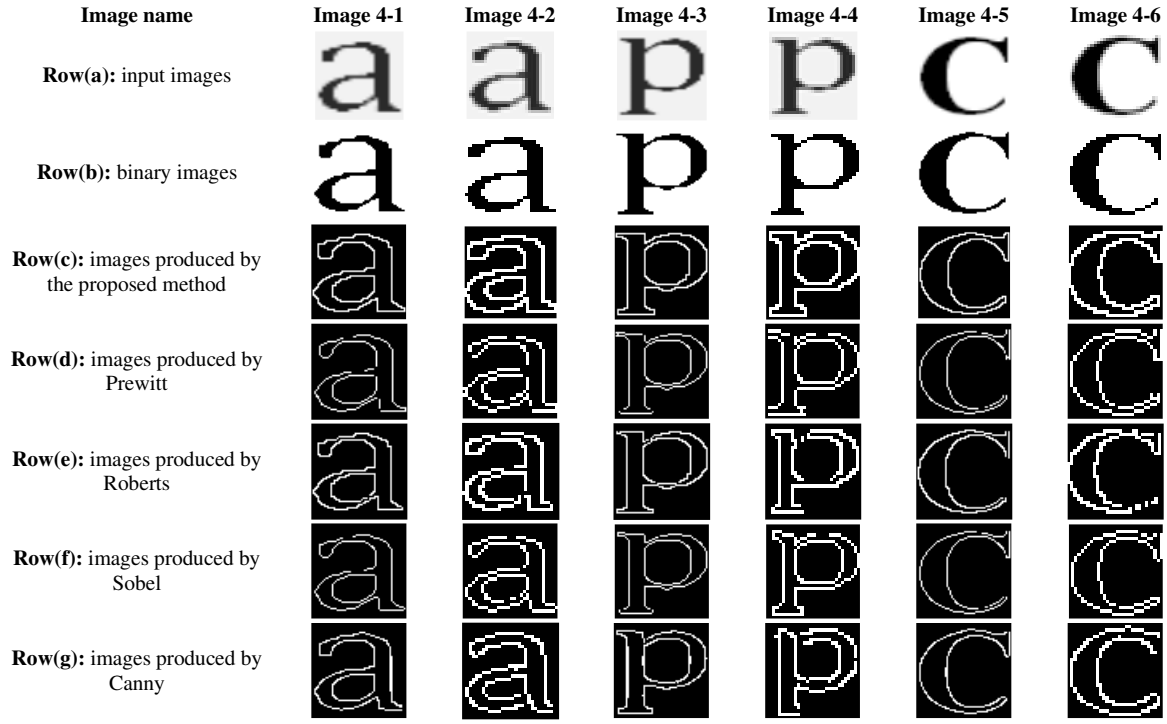

Fig. 4 Edges produced by the edge detection methods for the chosen characters

Table 2 PSNR results for the images in Fig. 4

Image name	Proposed method	Prewitt	Roberts	Sobel	Canny
Image 4-1	**2,010**	1,348	1,297	1,347	1,538
Image 4-2	**2,886**	1,646	1,294	1,611	2,123
Image 4-3	**1,976**	1,381	1,322	1,373	1,597
Image 4-4	**2,587**	1,542	1,147	1,487	1,938
Image 4-5	**1,804**	1,289	1,215	1,280	1,431
Image 4-6	**2,518**	1,604	1,217	1,555	1,775
Average PSNR	2,297	1,468	1,249	1,442	1,734
Standard deviation	0,426	0,148	0,066	0,129	0,262

Table 3 SSIM results for the images in Fig. 4

Image name	Proposed method	Prewitt	Roberts	Sobel	Canny
Image 4-1	**0,266**	0,056	0,029	0,059	0,110
Image 4-2	**0,310**	0,041	-0,061	0,038	0,161
Image 4-3	**0,265**	0,058	0,041	0,059	0,114
Image 4-4	**0,289**	0,043	-0,053	0,036	0,153
Image 4-5	**0,256**	0,087	0,055	0,087	0,116
Image 4-6	**0,300**	0,066	-0,019	0,061	0,099
Average SSIM	0,281	0,059	-0,001	0,057	0,126
Standard deviation	0,021	0,016	0,049	0,018	0,025

Table 4 PSNR results for the 30 characters in AlphabetRecognizer dataset

Image name	Proposed method	Prewitt	Roberts	Sobel	Canny
Image 1	**0,967**	0,467	0,459	0,455	0,745
Image 2	**1,018**	0,498	0,467	0,483	0,802
Image 3	**1,331**	0,752	0,790	0,744	0,822
Image 4	**0,634**	0,316	0,303	0,304	0,489
Image 5	**0,760**	0,376	0,358	0,363	0,568
Image 6	**0,954**	0,484	0,429	0,468	0,746
Image 7	**0,909**	0,473	0,505	0,467	0,602
Image 8	**0,745**	0,380	0,329	0,367	0,571
Image 9	**1,034**	0,507	0,544	0,497	0,682
Image 10	**1,145**	0,686	0,528	0,648	0,734
Image 11	**1,065**	0,565	0,578	0,554	0,716
Image 12	**0,909**	0,447	0,420	0,434	0,714
Image 13	**0,790**	0,399	0,407	0,390	0,548
Image 14	**0,574**	0,300	0,257	0,286	0,448
Image 15	**1,085**	0,606	0,529	0,578	0,730
Image 16	**0,985**	0,552	0,404	0,517	0,746
Image 17	**1,022**	0,624	0,381	0,584	0,770
Image 18	**1,290**	0,662	0,644	0,652	0,890
Image 19	**0,928**	0,484	0,521	0,481	0,615
Image 20	**0,697**	0,425	0,253	0,382	0,499
Image 21	**0,734**	0,402	0,295	0,365	0,457
Image 22	**1,020**	0,596	0,400	0,559	0,771
Image 23	**0,927**	0,479	0,396	0,460	0,722
Image 24	**0,749**	0,379	0,343	0,368	0,581
Image 25	**1,301**	0,713	0,754	0,708	0,798
Image 26	**0,974**	0,500	0,426	0,482	0,781
Image 27	**1,259**	0,652	0,609	0,633	0,880
Image 28	**0,611**	0,310	0,279	0,296	0,481
Image 29	**1,016**	0,469	0,522	0,462	0,674
Image 30	**0,956**	0,553	0,382	0,514	0,729
Average PSNR	0,946	0,502	0,450	0,483	0,677
Standard deviation	0,202	0,121	0,134	0,119	0,125

In the second experiment, 10 images are used for comparison. Five of them are images with regular geometric patterns, as illustrated in Fig. 5, which have been tailored and created for this study. The rest of the images in the dataset have been selected among the popular images widely used in image processing literature, as seen in Fig. 6. The main aim of this experiment is to shed light on the performance of the proposed method, especially on images with small-sized patterns and rich details. The results of PSNR and SSIM are provided in Table 5 and Table 6.

In the third experiment, the Berkeley dataset (BSDS500) is used. Although this database is constructed as a benchmark for contour detection, it includes interior object boundaries, background boundaries, and object contours. Therefore, it would be suitable to evaluate the edge detection methods [30]. Eight images are taken from this data set. Results produced by the proposed method and the gradient-based methods are illustrated in Fig. 7 for the first set of four images and Fig. 8 for the second set of four images. Furthermore, the quantitative metrics are extracted using the ground truth of the images in the dataset and the corresponding edge images produced by algorithms. Table 7 and Table 8 illustrate the PSNR and SSIM results for the selected images.

It is clearly seen that the ground truth of images contains only the object's contour in the image. On the other hand, the edge detection methods attempt to detect not only the contour but all details. Therefore, it is not feasible and fair to evaluate the edge detection method with respect to the contour of images. In this respect, PSNR and SSIM metrics are also evaluated using the corresponding binary images as ground truth, and the results are provided in Table 9 and Table 10. In the next section, the performance of the proposed edge detection method is thoroughly examined, and the results are discussed.

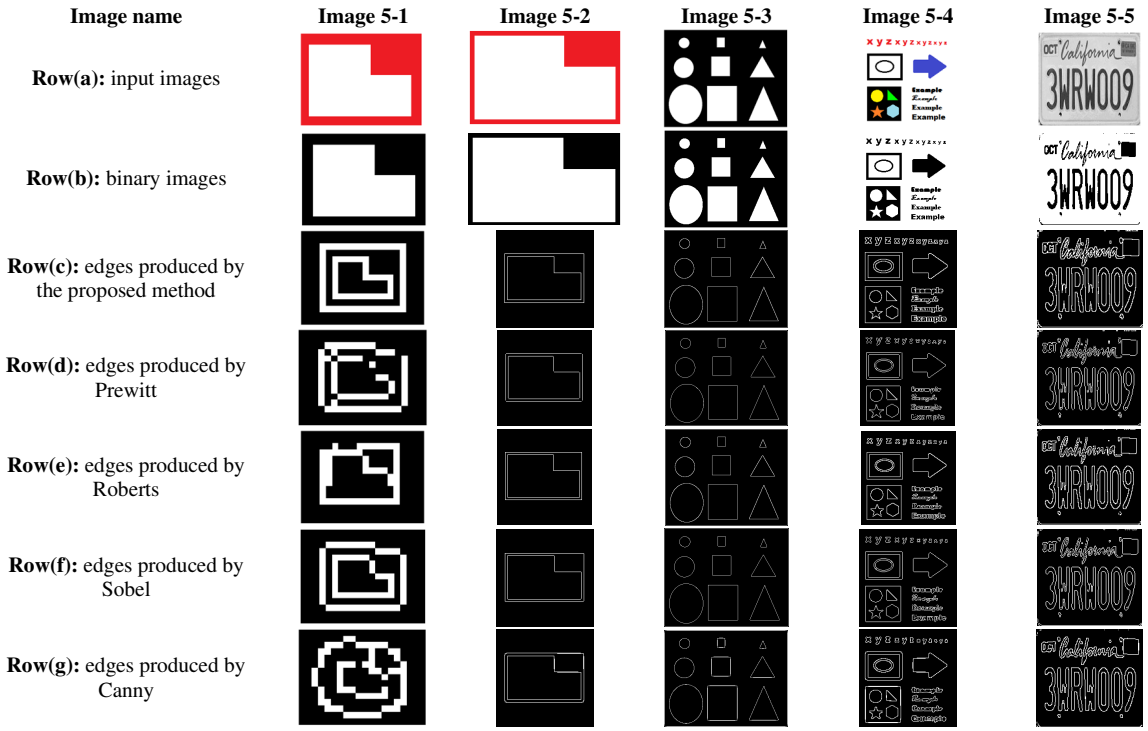

Fig. 5 Simple geometric shapes and complex patterns

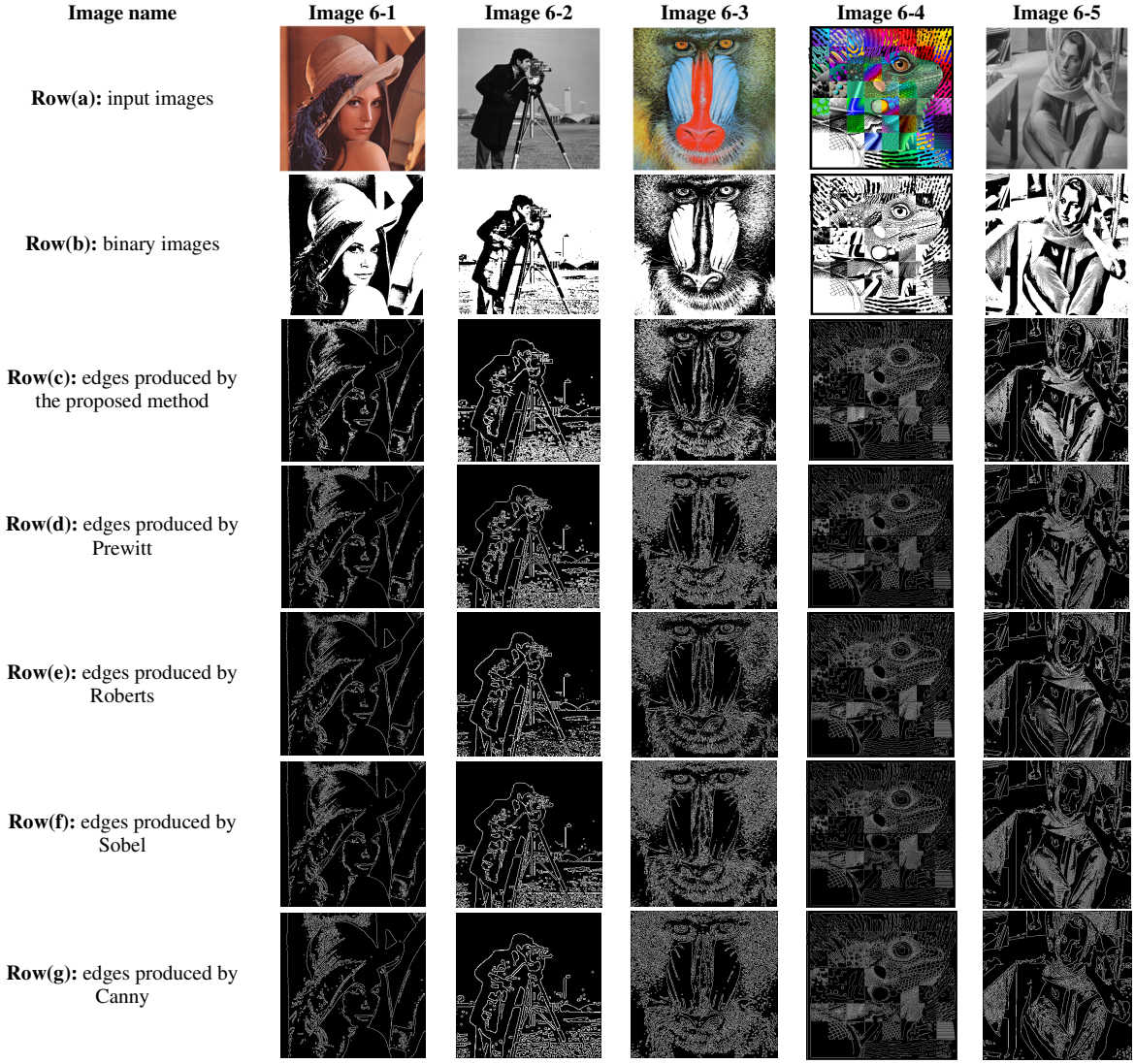

Fig. 6 Results of the edge detection methods for the images frequently used in literature

Table 5 PSNR results for the images in Fig. 5 and Fig. 6

Image name	Proposed method	Prewitt	Roberts	Sobel	Canny
Image 5-1	**2,365**	1,782	0,588	1,915	1,590
Image 5-2	**0,619**	0,495	0,495	0,495	0,590
Image 5-3	**5,258**	4,815	4,835	4,818	4,852
Image 5-4	**1,622**	1,097	1,124	1,086	1,211
Image 5-5	**2,034**	1,189	1,223	1,176	1,382
Image 6-1	**3,053**	2,164	2,090	2,137	2,167
Image 6-2	**5,618**	2,807	2,924	2,858	2,754
Image 6-3	**2,289**	1,506	1,310	1,453	1,487
Image 6-4	**9,650**	7,163	7,357	7,223	6,869
Image 6-5	**4,971**	2,421	2,320	2,407	2,352
Average PSNR	3,747	2,543	2,426	2,556	2,525
Standard deviation	2,664	2,014	2,159	2,031	1,921

Table 6 SSIM results for the images in Fig. 5 and Fig. 6

Image name	Proposed method	Prewitt	Roberts	Sobel	Canny
Image 5-1	**0,220**	0,101	-0,207	0,137	0,032
Image 5-2	**0,101**	0,047	0,047	0,047	0,080
Image 5-3	**0,665**	0,561	0,566	0,563	0,546
Image 5-4	**0,240**	0,057	0,066	0,057	0,084
Image 5-5	**0,265**	0,016	0,024	0,016	0,069
Image 6-1	**0,420**	0,166	0,192	0,175	0,169
Image 6-2	**0,602**	0,056	0,156	0,073	0,043
Image 6-3	**0,329**	0,073	0,060	0,077	0,082
Image 6-4	**0,842**	0,560	0,596	0,578	0,504
Image 6-5	**0,565**	0,036	0,069	0,046	0,033
Average SSIM	0,424	0,167	0,156	0,176	0,164
Standard deviation	0,235	0,211	0,246	0,212	0,194

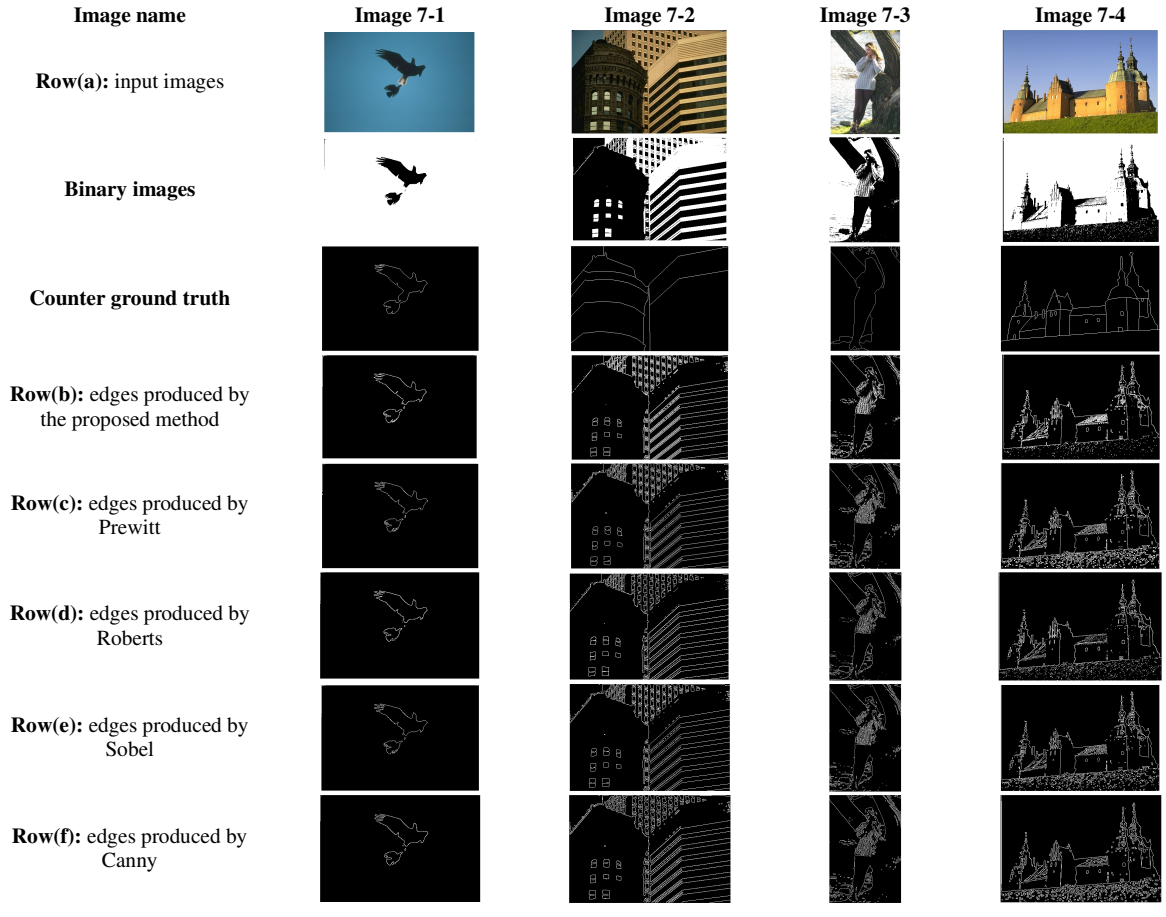

Fig. 7 Results obtained by the edge detection methods for the first set of images taken from the Berkeley dataset

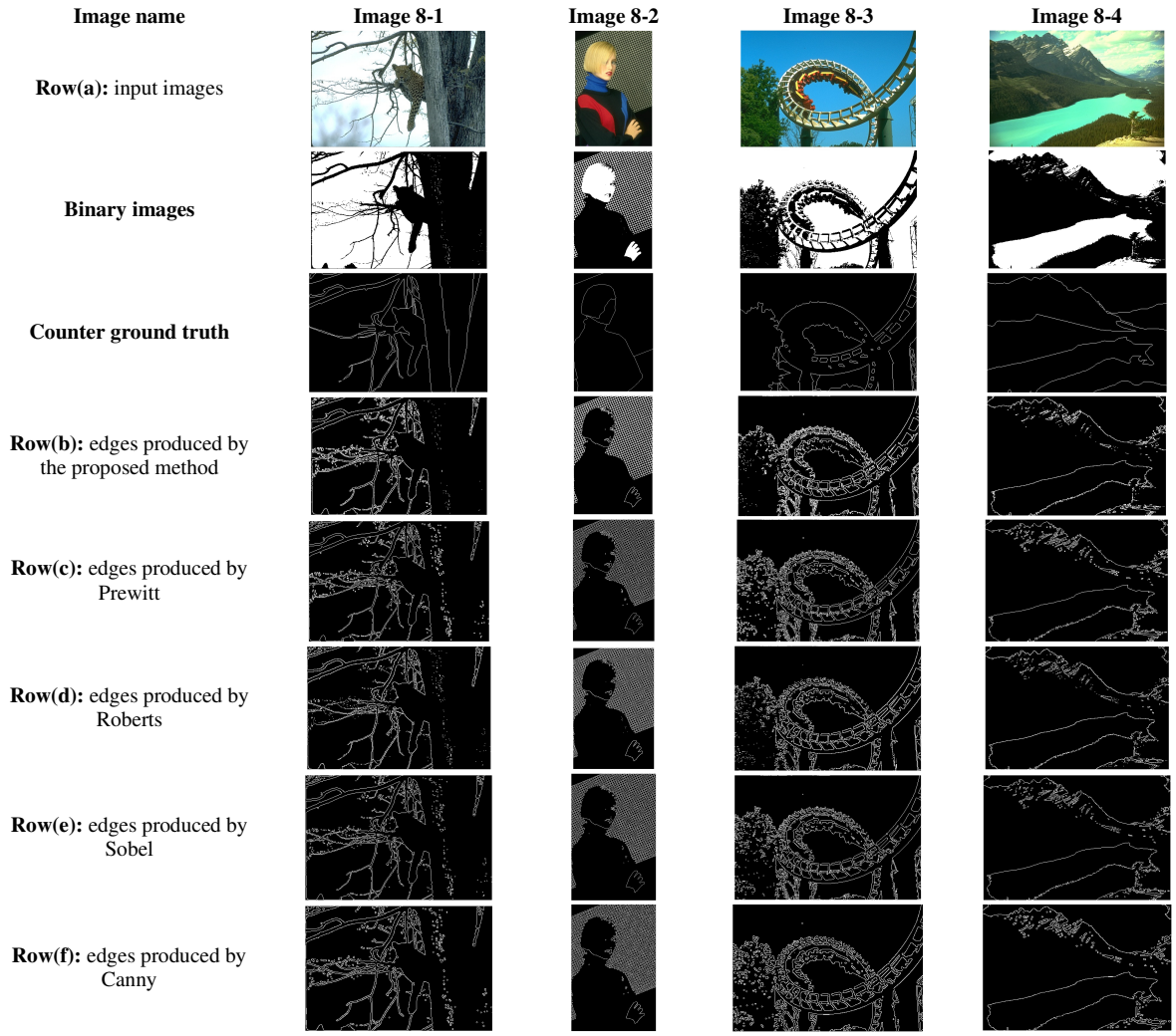

Fig. 8 Results obtained by the edge detection methods for the second set of images from the Berkeley dataset

Table 7 PSNR results for the images in Fig. 7 and Fig. 8 with respect to the corresponding ground truth images

Image name	Proposed method	Prewitt	Roberts	Sobel	Canny
Image 7-1	20,404	20,345	**21,149**	20,497	20,664
Image 7-2	10,071	10,981	11,090	**11,166**	10,815
Image 7-3	10,941	11,000	**12,101**	11,461	11,432
Image 7-4	11,829	10,529	**12,258**	11,306	10,832
Image 8-1	10,486	10,563	**11,864**	10,995	10,722
Image 8-2	7,569	**8,886**	8,169	8,706	8,529
Image 8-3	9,658	9,685	**10,342**	10,140	9,982
Image 8-4	13,508	13,105	**13,944**	13,643	13,353
Average PSNR	11,808	11,887	12,615	12,239	12,041
Standard deviation	3,871	3,627	3,833	3,610	3,736

Table 8 SSIM results for the images in Fig. 7 and Fig. 8 with respect to the corresponding ground truth images

Image name	Proposed method	Prewitt	Roberts	Sobel	Canny
Image 7-1	0,951	0,946	**0,956**	0,947	0,951
Image 7-2	**0,494**	0,486	0,492	0,489	0,485
Image 7-3	**0,653**	0,618	0,652	0,626	0,623
Image 7-4	**0,655**	0,585	0,621	0,593	0,592
Image 8-1	0,589	0,549	**0,622**	0,559	0,560
Image 8-2	**0,489**	0,483	0,488	0,485	0,483
Image 8-3	**0,536**	0,490	0,523	0,497	0,498
Image 8-4	**0,732**	0,705	0,727	0,713	0,711
Average SSIM	0,637	0,608	0,635	0,614	0,613
Standard deviation	0,153	0,157	0,154	0,156	0,158

Table 9 PSNR results for the images in Fig. 7 and Fig. 8 with respect to the corresponding binary images

Image name	Proposed method	Prewitt	Roberts	Sobel	Canny
Image 7-1	**0,245**	0,211	0,206	0,210	0,210
Image 7-2	**5,992**	4,936	4,635	4,900	4,832
Image 7-3	**4,646**	3,842	3,816	3,833	3,836
Image 7-4	**1,872**	1,398	1,509	1,448	1,404
Image 8-1	**3,659**	3,045	2,952	3,036	3,140
Image 8-2	**9,185**	4,954	5,252	4,950	3,884
Image 8-3	**2,600**	1,880	1,894	1,908	1,879
Image 8-4	**4,353**	3,890	3,962	3,914	3,869
Average PSNR	4,069	3,020	3,028	3,025	2,882
Standard deviation	2,727	1,718	1,715	1,704	1,563

Table 10 SSIM results for the images in Fig. 7 and Fig. 8 with respect to the corresponding binary images

Image name	Proposed method	Prewitt	Roberts	Sobel	Canny
Image 7-1	**0,049**	0,030	0,029	0,030	0,030
Image 7-2	**0,683**	0,499	0,477	0,501	0,483
Image 7-3	**0,607**	0,423	0,459	0,431	0,425
Image 7-4	**0,249**	0,081	0,154	0,090	0,075
Image 8-1	**0,498**	0,341	0,362	0,349	0,360
Image 8-2	**0,860**	0,413	0,464	0,415	0,247
Image 8-3	**0,364**	0,119	0,163	0,134	0,120
Image 8-4	**0,598**	0,466	0,504	0,478	0,463
Average SSIM	0,489	0,296	0,326	0,303	0,275
Standard deviation	0,259	0,189	0,184	0,189	0,183

4. Results and Discussion

In this section, the comparisons are carried out, and the algorithms' performances are evaluated based on the results obtained over three different datasets. In the first experiment, 2400 images in the AlphabetRecognizer dataset are used. Also, the second set of images is obtained by resizing the original images. The size of images has been reduced by half, and 2400 images have been obtained. As a result, in total, 4800 images are used in the first experiment [29]. For illustration purposes, 6 out of 4800 images have been selected from the dataset to visually determine the edge detection performance of the proposed method. The selected images and the reduced versions of them are given in Fig. 4. As there is no reference ground-truth, binary images (Row(b) in Fig. 4) are taken as a reference in all comparisons.

It is clearly seen from Fig. 4 that the proposed edge detection method can detect the edges efficiently compared to the rest of the methods. It successfully preserves the connectivity of the edge. In particular, the performance of the proposed method is seen clearly for small-size patterns in the scaled images in Row(c) of Fig. 4. Examining the results qualitatively reveals that the proposed edge detection method is clearly able to detect the edges of the small-size letter patterns, while the rest of the methods fail to do so. PSNR and SSIM values are given in the corresponding tables for a quantitative comparison. The proposed edge detection method achieved the best results for all images. The results shown in bold in Table 2 and Table 3 indicate that the proposed method outperforms the rest of the methods in this first experiment. The average PSNR for the proposed method is 2.297. On the other hand, Canny method obtained the second-best PSNR value of 1.734. Also, the average SSIM value for the proposed method is 0.281, while the second-best score of average SSIM is obtained by Canny method, which is 0.126.

Table 4 shows PSNR values for a set of fonts that contains 30 images reduced by half to obtain a small-sized version of letters. The proposed method produces the highest PSNR metrics compared to the other methods. If the results are carefully examined, the success of the proposed edge detection method can be easily seen in Table 4. An average PSNR value of 0.946 is obtained by the proposed method. The reason for this success of the proposed method is believed to stem from the fact that gradient-based methods fail in detecting edges at corners.

In the second experiment, a dataset with more complex patterns has been chosen to test the performance of the proposed method. The dataset contains simple geometric shapes and complex patterns, illustrated in Fig. 5 and Fig. 6. As there is no ground truth for the images in this experiment, all results have been compared with the corresponding binary image shown in Fig. 5 and Fig. 6. The results are illustrated in Table 5 and Table 6, respectively. The tables contain PSNR and SSIM metric values, and it is seen clearly that for all the cases, the proposed method outperforms the rest of the gradient-based methods.

Visual inspection of the results in Fig. 5 and Fig. 6 reveals that the proposed method successfully performs edge detection. It is also worth noticing that the method performs well for patterns with rich details, as in Fig. 5. Visual inspection of results in the case of images 6-4 and 6-5 reveals that the edge of the text patterns produced by the proposed method is well preserved with good connectivity features. On the other hand, the gradient-based methods seem to be inferior compared to the proposed method. The PSNR values obtained in Table 5 also indicate that the proposed method outperforms the rest of the gradient-based methods. The average PSNR for the proposed method is 3.747, as seen in Table 5. On the other hand, Sobel Method obtained the second-best PSNR value of 2.556. Also, the average SSIM value for the proposed method is 0.424, and the second-best score is obtained by Sobel which is 0.176.

In the third experiment, 8 images from Berkeley Data Set (BSDS500) have been selected, which are shown in Fig. 7 Row(a) and Fig. 8 Row(a), respectively. The dataset contains the ground-truth contour for each image. All images are converted to a binary image using the MATLAB `imbinarize()` function. The proposed and gradient-based methods work on the binary images to extract the edge. The evaluation of the results is conducted using PSNR and SSIM metrics. The metrics are obtained using two approaches.

In the first approach, the ground-truth contour images in the third row of Fig. 7 and the third row of Fig. 8 are used to evaluate PSNR and SSIM metrics, and the results are given in Table 7 and Table 8. In this case, in terms of the average values of PSNR, the performance of all the methods is fairly close to each other. This is not a surprise, and it is a highly expected case due to the fact that the ground truth is not the edge but the contour of the images. The metrics are evaluated against the complexity of the patterns in the images and background. However, the proposed method produces the best result for SSIM metric values. The proposed method produces the highest SSIM mean value and Roberts method is the second best. In conclusion, the performance of the proposed method is comparable to the rest of the gradient-based methods in qualitative and quantitative terms.

In the second approach, the binary images shown in the second row of Fig. 7 and the second row of Fig. 8 are used as ground-truth images, and the results are given in Table 9 and Table 10, respectively. The quantitative comparison using PSNR and SSIM reveals that the proposed method performs well in all cases in terms of PSNR and SSIM metrics.

Apart from being the best method in terms of SSIM metrics, the most distinctive feature of the proposed method over the rest of the gradient-based methods is its performance in terms of the computational load. To reach a fair comparison, the algorithms have been run 10 times using the images from Berkeley Dataset and then the mean value of CPU time has been found, which is an indication of the computational complexity exposed by the methods. The results given in Table 11 reveal that the proposed method is on average 95.61% faster than Sobel edge detection method, 125.36% faster than Prewitt, 193.44% faster than Roberts, and 482.45% faster than Canny method.

Table 11 Average CPU time required by the methods for edge detection process

Edge detection method	Proposed method	Prewitt	Roberts	Sobel	Canny
CPU time (ms)	36.75	82.82	107.84	71.89	214.09

5. Conclusions

A new rule-based method for edge detection is proposed. The rules are expressed in terms of simple arithmetic operations. As a result, a computationally efficient method is extracted. The following conclusions have been drawn:

(1) The performance of the proposed edge detection method is compared with the widely used gradient-based Sobel, Prewitt, Roberts, and Canny edge detection methods. Three different datasets have been used in the comparison. Along with the qualitative judgment, two quantitative measures, PSNR and SSIM metrics, were used to evaluate the results of the experiments. The newly developed edge detection method gives better results in images containing especially small-sized patterns. It is believed that the main reason for the phenomena lies in the fact that the proposed method evaluates the pixels on a knowledge-based neighborhood relationship. Traditional methods are relying on gradient-based operations in detecting corner points, where the gradient scattering is not uniform and smooth, which results in a deterioration in the performance.

(2) It is seen from the results given in the tables that the proposed method successfully produces comparable performance. Primarily based on SSIM values, it is reasonable to claim that the edges detected by the proposed methods provide better structural similarity. Furthermore, in terms of computational load, the proposed method outperforms the rest of the gradient-based methods, in some cases being four times faster.

(3) The proposed method utilizes a 2-by-2 frame to sweep the input image. Then, the image is scanned horizontally by moving the frame by one pixel from left to right. By increasing the frame size, one might expect to cover an extended area and consequently extract more realistic neighborhood rules. As it is shown that the proposed method is a promising approach, future work would be carried out to extend the approach of having knowledge-based rules for a kernel of 3 by 3. This extension could make it possible to consider the neighborhood properties in diagonal directions as well. Moreover, by taking advantage of being a computationally lightweight method, further improvements would be obtainable by implementing the proposed method on FPGA.

Conflicts of Interest

The authors declare no conflicts of interest.

References

[1] C. Deng, et al., “An Edge Detection Approach of Image Fusion Based on Improved Sobel Operator,” 4th International Congress on Image and Signal Processing, pp. 1189-1193, October 2011.
[2] K. Zhang, et al., “An Improved Sobel Edge Algorithm and FPGA Implementation,” Procedia Computer Science, vol. 131, pp. 243-248, May 2018.
[3] S. Taslimi, et al., “Adaptive Edge Detection Technique Implemented on FPGA,” Iranian Journal of Science and Technology, Transactions of Electrical Engineering, vol. 44, no. 4, pp. 1571-1582, March 2020.
[4] L. Luo, et al., “A Vision Methodology for Harvesting Robot to Detect Cutting Points on Peduncles of Double Overlapping Grape Clusters in a Vineyard,” Computers in Industry, vol. 99, pp. 130-139, August 2018.
[5] O. P. Verma, et al., “An Optimal Edge Detection Using Modified Artificial Bee Colony Algorithm,” Proceedings of the National Academy of Sciences, India Section A: Physical Sciences, vol. 86, no. 2, pp. 157-168, March 2016.
[6] V. Mario, et al., “Image Edge Detection: A New Approach Based on Fuzzy Entropy and Fuzzy Divergence,” International Journal of Fuzzy Systems, vol. 23, no. 4, pp. 918-936, February 2021.
[7] Y. Liu, et al., “Richer Convolutional Features for Edge Detection,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 41, no. 8, pp. 1939-1946, August 2019.
[8] N. S. Dagar, et al., “Edge Detection Technique Using Binary Particle Swarm Optimization,” Procedia Computer Science, vol. 167, pp. 1421-1436, April 2019.
[9] C. Wen, et al., “Edge Detection with Feature Re-Extraction Deep Convolutional Neural Network,” Journal of Visual Communication and Image Representation, vol. 57, pp. 84-90, November 2018.
[10] T. Peng-o, et al., “High Performance and Energy Efficient Sobel Edge Detection,” Microprocessors and Microsystems, vol. 87, Article no. 104368, November 2021.
[11] Y. Çapkan, et al., “Robotic Arm Guided by Deep Neural Networks and New Knowledge-Based Edge Detector for Pick and Place Applications,” International Conference on Innovations in Intelligent Systems and Applications, pp. 1-4, August 2021.

[12] Z. Chen, et al., "Vehicle Detection in High-Resolution Aerial Images via Sparse Representation and Superpixels," IEEE Transactions on Geoscience and Remote Sensing, vol. 54, no. 1, pp. 103-116, July 2016.
[13] Z. Chen, et al., "Vehicle Detection in High-Resolution Aerial Images Based on Fast Sparse Representation Classification and Multiorder Feature," IEEE Transactions on Intelligent Transportation Systems, vol. 17, no. 8, pp. 2296-2309, July 2016.
[14] H. Ma, et al., "Radar Image-Based Positioning for USV under GPS Denial Environment," IEEE Transactions on Intelligent Transportation Systems, vol. 19, no. 1, pp. 72-80, January 2018.
[15] D. Gupta, et al., "A Hybrid Edge-Based Segmentation Approach for Ultrasound Medical Images," Biomedical Signal Processing and Control, vol. 31, pp. 116-126, January 2017.
[16] W. C. Lin, et al., "Edge Detection in Medical Images with Quasi High-Pass Filter Based on Local Statistics," Biomedical Signal Processing and Control, vol. 39, pp. 294-302, January 2018.
[17] Z. Liu, et al., "Image Security Based on Iterative Random Phase Encoding in Expanded Fractional Fourier Transform Domains," Optics and Lasers in Engineering, vol. 105, pp. 1-5, June 2018.
[18] K. Hajipour, et al., "Edge Detection of Noisy Digital Image Using Optimization of Threshold and Self Organized Map Neural Network," Multimedia Tools and Applications, vol. 80, no. 4, pp. 5067-5086, February 2021.
[19] L. Xuan, et al., "An Improved Canny Edge Detection Algorithm," 8th IEEE International Conference on Software Engineering and Service Science, pp. 275-278, November 2017.
[20] K. Zhang, et al., "FPGA Implementation of Eight-Direction Sobel Edge Detection Algorithm Based on Adaptive Threshold," Journal of Physics: Conference Series, vol. 1678, Article no. 012105, November 2020.
[21] M. A. Günen, et al., "A Novel Edge Detection Approach Based on Backtracking Search Optimization Algorithm (BSA) Clustering," 8th International Conference on Information Technology, pp. 116-122, May 2017.
[22] J. Cao, et al., "Implementing a Parallel Image Edge Detection Algorithm Based on the Otsu-Canny Operator on The Hadoop Platform," Computational Intelligence and Neuroscience, vol. 2018, Article no. 3598284, 2018.
[23] M. Mittal, et al., "An Efficient Edge Detection Approach to Provide Better Edge Connectivity for Image Analysis," IEEE Access, vol. 7, pp. 33240-33255, March 2019.
[24] A. Al-Ghaili, et al., "Pixel Intensity-Based Contrast Algorithm (PICA) for Image Edges Extraction (IEE)," IEEE Access, vol. 8, pp. 119200-119220, June 2020.
[25] Y. Tao, et al., "A Low Redundancy Wavelet Entropy Edge Detection Algorithm," Journal of Imaging, vol. 7, no. 9, Article no. 188, September 2021.
[26] P. Ganesan, et al., "Assessment of Satellite Image Segmentation in RGB and HSV Color Space Using Image Quality Measures," International Conference on Advances in Electrical Engineering, pp. 1-5, January 2014.
[27] Z. Wang, et al., "Image Quality Assessment: From Error Visibility to Structural Similarity," IEEE Transactions on Image Processing, vol. 13, no. 4, pp. 600-612, April 2004.
[28] "Alphabetrecognizer Dataset," https://github.com/MinhasKamal/AlphabetRecognizer, June 01, 2021.
[29] "Image Dataset," https://github.com/yavuzcpkn/Knowledge-Based-Edge-Detection, January 01, 2022.
[30] A. P. Kelm, et al., "Object Contour and Edge Detection with RefineContourNet," International Conference on Computer Analysis of Images and Patterns, pp. 246-258, August 2019.