A COMPREHENSIVE COMPARATIVE EXPERIMENT OF EDGE DETECTION FOR OMR

Zahriah Sahri[1], **Zuraini Othman**[1], **Fauziah Kasmin**[1] **and Haliza Basiron**[1]

[1]Fakulti Teknologi Maklumat dan Komunikasi,
Universiti Teknikal Malaysia Melaka, Hang Tuah Jaya, 76100 Durian Tunggal, Melaka, Malaysia.

Corresponding Author's Email: [1]szahriah2511@utem.edu.my

Article History: Received 20 Mac 2022; Revised 28 April 2022; Accepted 29 May 2022

ABSTRACT: Optical mark reading (OMR) sheets are used widely to key in answers for *multiple choice questions*. Many studies have proposed automated OMR grading systems using the web or mobile devices which capture the OMR images. These captured images go through a few image processing steps to display the grades. Edge detection is one of the steps. This study compares different established edge detection algorithms to detect edges of interest in OMR answer sheets such as bubbles or rectangles containing the bubbles. The compared algorithms are Sobel, Roberts, Prewitt, Canny, and Laplacian of Gaussian (LoG). The experimental results show that LoG has accurately detected the edges in OMR images while Sobel is the least effective. Prewitt and Canny deliver almost similar performances, and Roberts falls second worst to Sobel**.**

KEYWORDS: *edge detection; edge detection algorithms, optical mark recognition*

1.0 INTRODUCTION

Optical mark reading (OMR) sheets are used to key in answers prominently for multiple-choice questions. A space (circular or oval or blank box) is designated for each option under the given question in every OMR sheet. The student is expected to shade or mark the correct option's corresponding circle. Then, the sheets are marked and graded manually by the educators themselves or by using OMR hardware-based reader [1]. Manual grading has a few drawbacks, such as improper grading, consuming educators' valuable time, and a longer response time to get results [2]. On the other hand, OMR hardware-based reader is a dedicated and expensive scanning machine and requires customized OMR sheets and OMR software [3].

Due to the drawbacks of both manual grading and dedicated OMR reader, and with the advancement of computing technology in recent years, many studies have been conducted to propose more cost-effective and automated solutions to OMR grading using web [3]–[6] or mobile technology [7]–[10]. These technologies acquire OMR images using either a traditional scanner, mobile phone camera, or webcam. Then, the automated OMR systems will read the scanned images, extract the necessary information from the images using image processing and computer vision techniques, and produce the results.

Edges constitute a significant and necessary portion of the information contained in images. Edge detection aims to identify points in a digital image where the brightness changes sharply or, more formally, has discontinuities. Image processing constitutes various steps, in which edge detection is one of them. Historically, the most common and the earliest edge detection algorithms are the Roberts operator [11] and the Sobel operator [12]. Other established algorithms are the Canny operator [13], the Prewitt operator [14], the Laplacian of Gaussian (or called Marr-Hildreth edge detector) [15], Robinson [16], Kirsch [17], and wavelet-based edge detections.

To the authors' knowledge, many studies [3], [6], [7], [9], [10], [18], [19] of OMR applications do not specifically address the edge detection process using established edge detection algorithms. Only several studies apply the Canny operator to detect edges in OMR sheets. Canny is used by [5] to detect the edges/outlines of the marked choices, and [10] uses Canny on the blurred and binary OMR images to get the essential contours. Meanwhile, [8] converted the edges into binary images using the Canny operator.

It is worth mentioning that each edge detection has its strengths and weaknesses. According to [20], edge detection using mathematical morphology is more efficient than the traditional methods (Canny, Roberts, Prewitt, and Sobel). Meanwhile, [21] concludes that Canny performs better than all these algorithms under almost all scenarios and noisy conditions. Akdemir [22] advocated using edge detection algorithms because they operate shorter and are more straightforward than other algorithms.

Therefore, this study aims to compare edge detection techniques on OMR answer sheets, learn from the shortcomings, analyze better solutions, and make future forecasts. The compared algorithms are Sobel, Roberts, Prewitt, Canny, and Laplacian of Gaussian (LoG), which are still popular today. The performance of these algorithms is compared by applying the same images, and then the results are reviewed.

The rest of this paper is organized as follows; Section II reviews related works, Section III explains the comparative experiments performed in this study, Section IV discusses the result, and Section V is focused on the conclusion.

2.0 RELATED WORK

In a digital image, an edge is a collection of the pixels whose grey value has a step or roof change, and it also refers to the part where the brightness of the image local area changes significantly. The grey profile in this region can generally be seen as a step. In a small buffer area, a grey value rapidly changes to another whose grey value is mainly different from it. Edge widely exists between objects and backgrounds, objects and objects, primitives and primitives. The edge of an object is reflected in the discontinuity of the grey. Therefore, the general method of edge detection is to study the changes of a single image pixel in a grey area use the variation of the edge neighboring first-order or second-order to detect the edge.

From several magnetic resonance imaging (MRI) thigh bone images, Meng et al. [23] applied Prewitt, Sobel, and LoG to extract the edges from the different scales of the smooth images. The extracted edges were then combined to form the multiscale edge detection. They concluded that Prewitt and Sobel operators could extract edges most similar to the original image. Laplacian could extract finely detailed edges. However, it is sensitive to noise. Albayati and Ali [24] studied different edge detection algorithms for image steganography – one of the techniques for information hiding. Comparative algorithms were Sobel, Prewitt, Kirsch, Roberts, LoG, Fuzzy Logic, and Canny. Using Mean Square Error (MSE) and Peak Signal to Noise Ratio (PSNR) as the performance metrics, they found that Canny gave the best results.

Shah et al. [25] performed comparative edge detection for image processing using Sobel, Prewitt, Roberts, LoG, and Canny. Images of different sizes, different directions, dilated and eroded, were fed to these five edge detection algorithms. Using MSE and PSNR as performance measures, they concluded that Canny detected strong and weak edges and was the most efficient but took longer. A.Ahmed [26] applied Sobel, Prewitt, and Canny operators on different images to detect edges. Their performance was measured using time intervals and accuracy (MSE and PSNR). The comparative results concluded that Canny produced the highest accuracy and lower execution time than Sobel and Prewitt.

Podder and Chanda [27] segmented thermal images using different edge detection algorithms like Prewitt, Sobel, LoG, and Canny operators. Based on the experimental results of the structure of similarity(SSIM) and PSNR metrics, it was observed that the Canny operator produces the highest PSNR, SSIM, than other operators in terms of image quality. So, the results show that canny edge detection is less erroneous than other edge detection techniques. The effectiveness of Roberts, Sobel, LoG, and Canny in detecting linear paleo-shorelines was studied by Bachofer et al. [28]. These edge detector algorithms were tasked to extract and revise those linear structures found in Synthetic Aperture Radar (SAR) images. The comparison showed that the Canny edge detector was especially suitable for images with strong speckle noise. Canny achieved relatively high accuracies compared to the other operators. The related works mentioned above show that comparative studies have been widely established for several problem domains to determine the effectiveness of different edge detection algorithms to detect edges for the specific domain. The aforementioned comparative study has not gained traction in the OMR domain to the authors' knowledge. Hence, this has spurred the interest of the authors of this study to fill the gap.

3.0 THE COMPARATIVE STUDY

Figure 1 shows the main steps for comparing different edge detection algorithms on OMR images. The details of each step are described in the respective subsections below.

Anaconda3 2019 was used as the development software to run the edge detection algorithms written in Python and calculated the performance metrics results. Python and Adobe Photoshop 2021 22.0.0.35 were used to create ground truth images of the OMR samples.

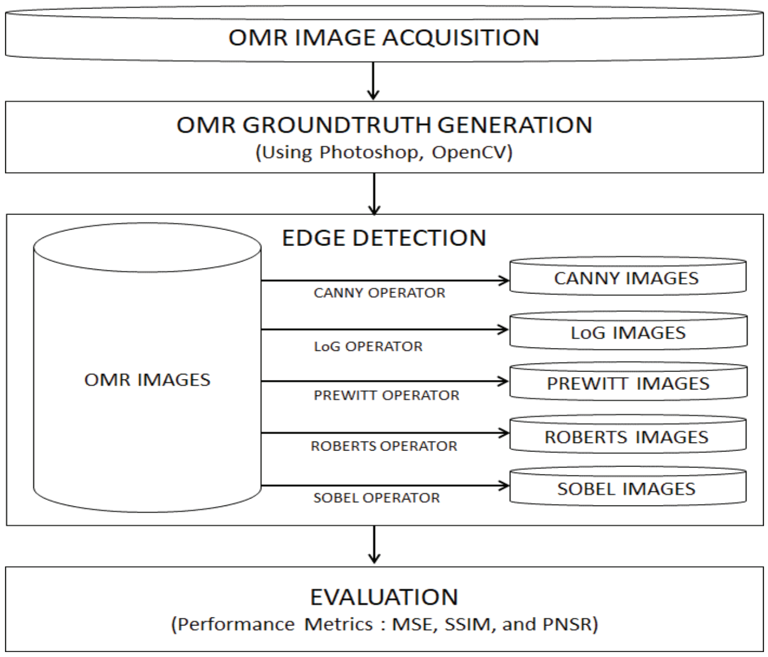


Figure 1: The proposed comparative study among edge detection algorithms

3.1 OMR Images Acquisition

The OMR images were primarily collected from the internet under the public domain license. The number of bubbles in the OMR images ranges from 5 to 300 bubbles. A few images were acquired from OMR sheets used for the national examination of Malaysian secondary students. In total, 32 OMR images were used in this comparative study. Figure 2 shows three different examples of OMR images used in this study.

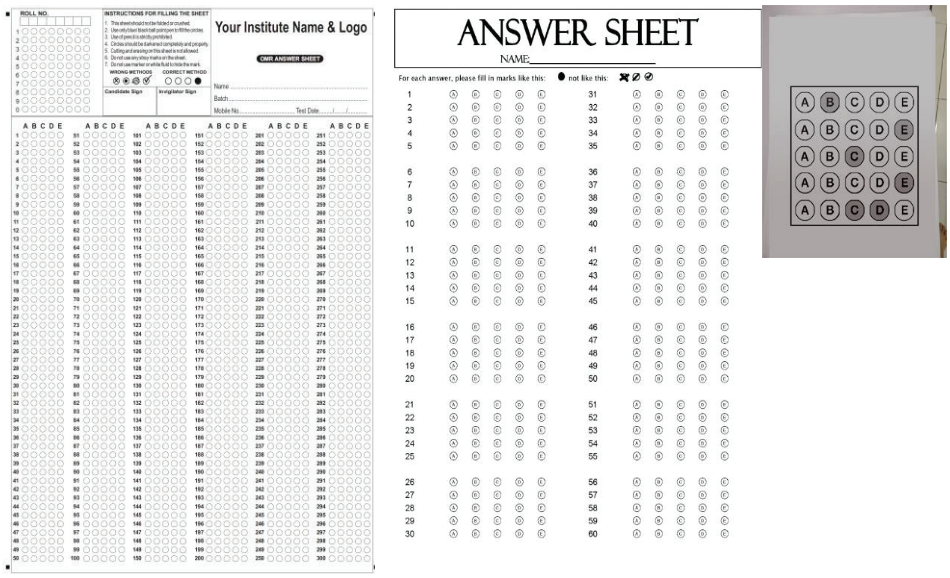

ROLL NO.
INSTRUCTIONS FOR FILLING THE SHEET
Your Institute Name & Logo
OMR ANSWER SHEET
ANSWER SHEET
NAME:
For each answer, please fill in marks like this: ● not like this:

Figure 2: Examples of OMR images used in this study

3.2 OMR Ground Truth Generation

To evaluate the performance of the five edge detection algorithms, this study compares the resultant image from each algorithm with another data source that is considered accurate. These accurate data are termed as ground truth. The ground truth of an OMR image, for instance, is the complete and precise record of every bubble. The ground-truth data has to have a sure accuracy but at least some magnitudes higher than the algorithm's precision to be evaluated to allow for a significant evaluation. Marking individual bubbles is common to generate ground truth for a particular frame. However, it is a prolonged process without automation or assistance. Consequently, generating the ground truth for hundreds of OMR images could take several weeks to complete. Therefore, this study used two software to create ground truth images, as shown in Figure 3.

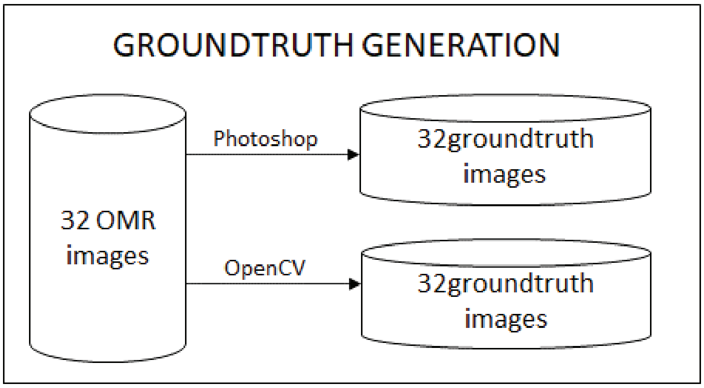


Figure 3: Ground truth generation

This study used Photoshop software to create ground truth images from the OMR samples. The OMR images were first converted to grayscale, which changed the images' colors to black, white, and shades of grey. After that, the images were duplicated, and the replicas were changed to color dodge. The current pixel's brightness value was then replaced with its surrounding pixels' greatest or lowest brightness value by using Minimum filters. This resulted in the black portions extending out and the white areas diminishing. The final step was to reverse the image's black and white color.

To further enforce the evaluation of the compared algorithms, this study also applied OpenCV Python to create ground truth images from the OMR sample. Each OMR sample was turned to grayscale, which changed the image's colors to black, white, and shades of grey. Then Gaussian blur filter was applied to reduce the noise. Finally, the Otsu Thresholding method created the ground truth image. Figure 4 shows the ground truth images obtained from OpenCV, while ground truth images from Photoshop are shown in Figure 5.

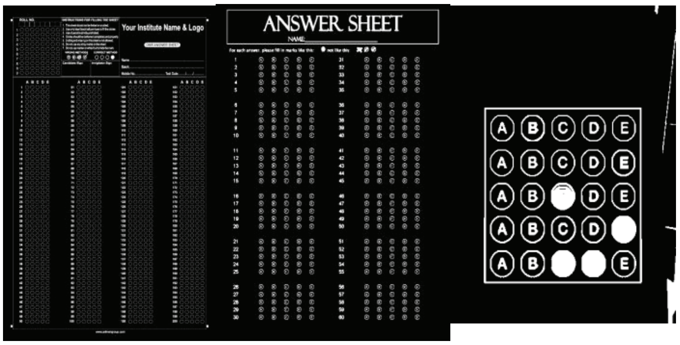


Figure 4: OpenCV ground truth images

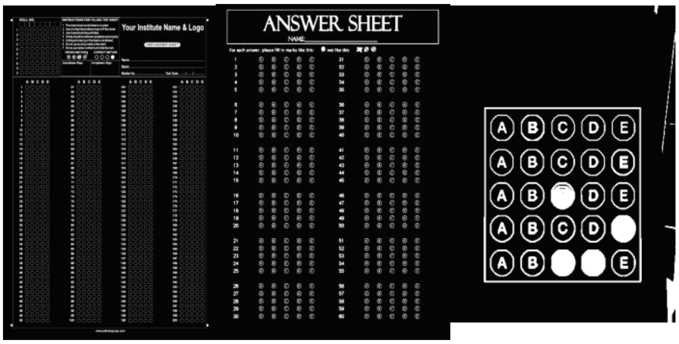


Figure 5: Photoshop ground truth images

3.3 Edge Detection Algorithms Being Compared

In this phase, each edge detection algorithm is applied to each original OMR image as shown in Figure 1. The output is five different edges detected images for each original OMR image. The working of each edge detection algorithm is described in the subsections below.

i. Roberts Operator

In 1965,Lawrence Roberts proposed the Roberts operator [11] for detecting the edges within an image. It performs a simple, quick to compute, 2-D spatial gradient measurement on an image. Pixel values at each point in the output represent the estimated absolute magnitude of the spatial gradient of the input image at that point. The operator consists of a pair of 2×2 convolution kernels, as shown in Figure 6. One kernel is simply the other rotated by 90°.

1	0
0	-1

G_x

0	1
-1	0

G_y

Figure 6: Masks used by the Roberts operator

The gradient magnitude is given by (1):

$$|G_x| = \sqrt{{G_x}^2 + {G_y}^2} \tag{1}$$

Typically, an approximate magnitude is computed using (2):

$$|G_x| = |G_x| + |G_y| \tag{2}$$

which is much faster to compute. The angle of orientation of the edge (relative to the pixel grid) giving rise to the spatial gradient is given by (3):

$$\theta = arctan\left(\frac{G_x}{G_y}\right) - \frac{3\pi}{4} \tag{3}$$

ii. Sobel Operator

In 1968, Irwin Sobel and Gary Feldman presented the Sobel operator [12]. This operator consists of a pair of 3×3 convolution kernels, as shown in Fig. 7 below. One kernel is simply the other rotated by 90°.

Sobel works almost very similar to Roberts. Using (1) and (2), the angle of orientation of the edge giving rise to the spatial gradient (relative to the pixel grid orientation) is given by (4):

$$\theta = arctan\left(\frac{G_y}{G_x}\right) \tag{4}$$

1	2	1
0	0	0
-1	-2	-1

G_x

-1	0	1
-2	0	2
-1	0	1

G_y

Figure 7: Mask used by the Sobel operator

iii. Prewitt Operator

M.S. Prewitt developed the Prewitt operator in 1970 [14]. It shows many similarities with the properties of the Sobel operator. It has two pieces of kernels, and this size is 3x3, as shown in Figure 8. It is a gradient-based edge detection operator, and it has gradient features. Using (1) and (2), the Prewitt gradient's direction is obtained using (4).

-1	0	1
-1	0	1
-1	0	1

-1	-1	-1
0	0	0
1	1	1

Figure 8: Masks for the Prewitt operator

iv. Canny Operator

Canny edge detector is an edge detection operator that uses a multi-stage algorithm to detect a wide range of edges in images. John F. Canny developed it in 1986 [13]. Before using Canny, the experimented images must be converted to grayscale pictures. The Canny edge detection algorithm is composed of 5 steps:

- Apply a Gaussian filter to smooth the image to remove the noise:

  The equation for a Gaussian filter kernel of size (2k+1)×(2k+1) is given by (5):

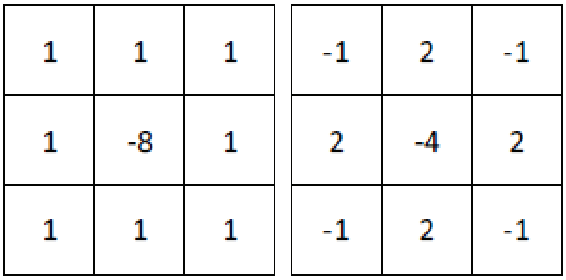
$$H_{ij} = \frac{1}{2\pi\sigma^2} exp\left(-\frac{\left(i-(k+1)\right)^2+\left(j-(k+1)\right)^2}{2\sigma^2}\right); 1 \leq i, j \leq (2k+1) \quad (5)$$

- Find the intensity gradients of the image:
  Your filters are used to detect horizontal, vertical, and diagonal edges in the blurred image. Then an edge detection operator (such as Roberts, Prewitt, or Sobel) returns a value for the first derivative in the horizontal direction (G_x) and the vertical direction (G_y). The edge gradient and direction can be determined using (1) and (4).
- Apply gradient magnitude thresholding or lower bound cut-off suppression to get rid of spurious response to edge detection: This step is to obtain thin edges using an edge-thinning algorithm. The algorithm goes through all the points on the gradient intensity matrix and finds the pixels with the maximum value in the edge directions.

- Apply double threshold to determine potential edges:
  This step filters out edge pixels with a weak gradient value and preserves edge pixels with a high gradient value to further eliminate edge pixels caused by noise and color variation.
- Track edge by hysteresis:
  Finalize the detection of edges by suppressing all the other weak edges and not connecting to strong edges.

v. Laplacian of Gaussian

It was invented by Marr and Hildreth (1980) [15]. The Gaussian filtering is combined with Laplacian to break down the image where the intensity varies to detect the edges effectively. It uses linear interpolation to determine the sub-pixel location of the edge. The digital implementation of the Laplacian function is made using the mask given in below Figure 9.



1	1	1
1	-8	1
1	1	1

-1	2	-1
2	-4	2
-1	2	-1

Figure 9: LoG operator

The operator usually takes a single gray-level image as input and produces another gray-level image as output. The Laplacian *L(x,y)* of an image with pixel intensity values *I(x.y)* is given by:

$$L(x, y) = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2} \tag{6}$$

Since the input image is represented as a set of discrete pixels, we need to find a discrete convolution kernel that can approximate the second derivatives in the definition of the Laplacian.

Figure 10, Figure 11, and Figure 12 show the original images and their respective detected edge images obtained using the compared edge detection algorithm.

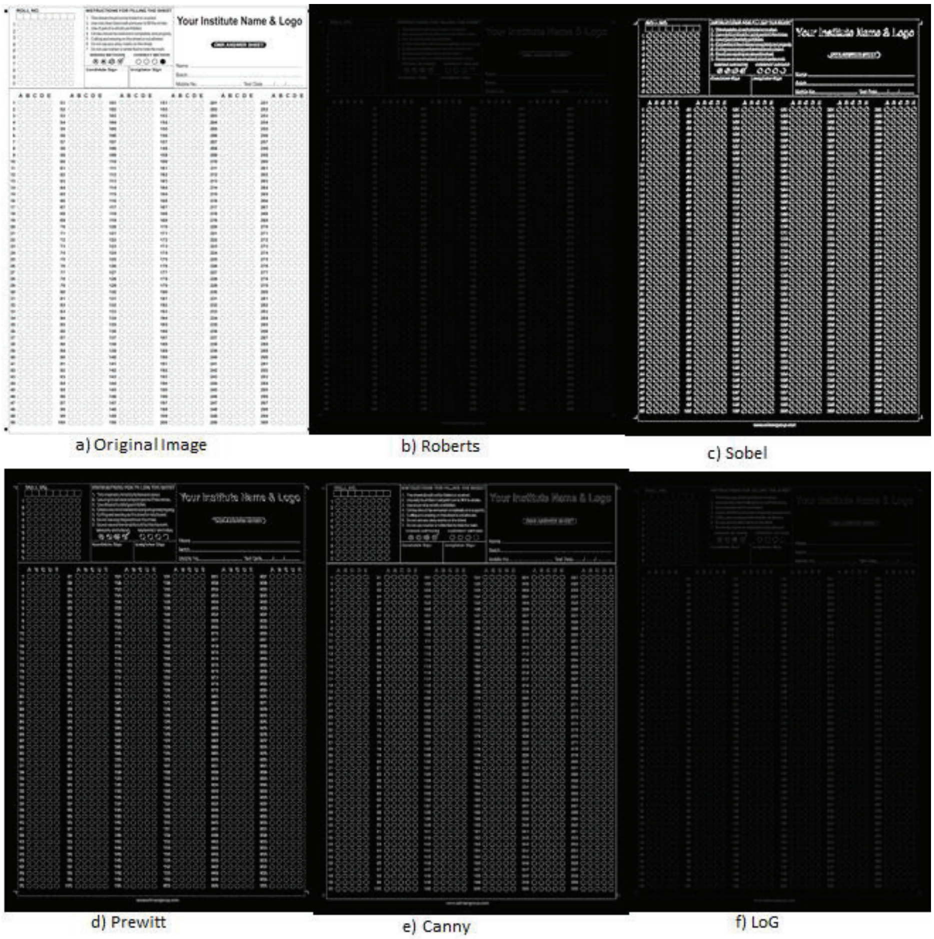


Figure 10: Original OMR Image with the Outcomes of Different Edge Detection Techniques

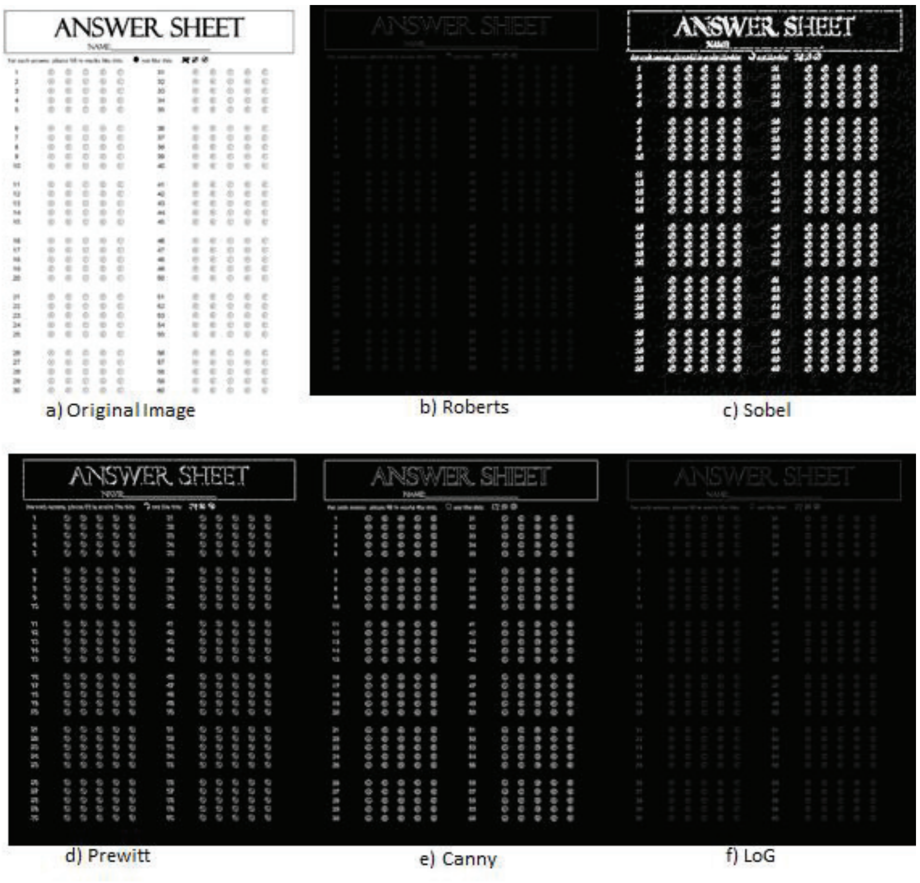


Figure 11: Original OMR Image with the Outcomes of Different Edge Detection Techniques

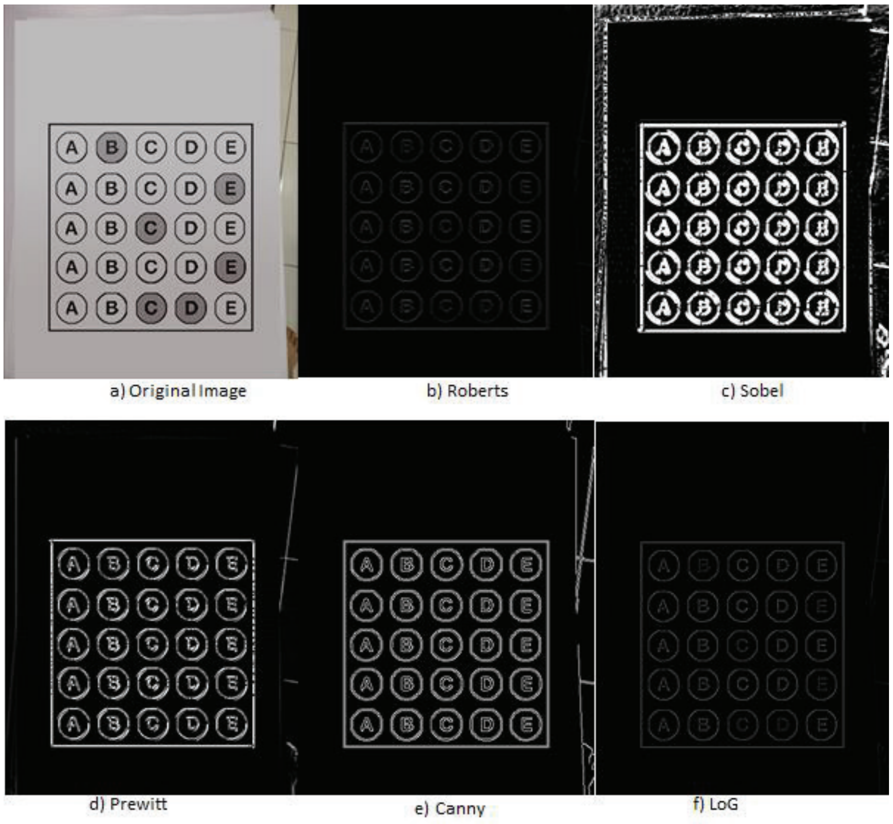


Figure 12: Original OMR Image with the Outcomes of Different Edge Detection Techniques

3.3 Performance Evaluation

The similarity between the image detected from an edge detection algorithm and its groundtruth image (generated by Python or Photoshop) is measured using MSE, SSIM, and PSNR, as shown in Figure 13.

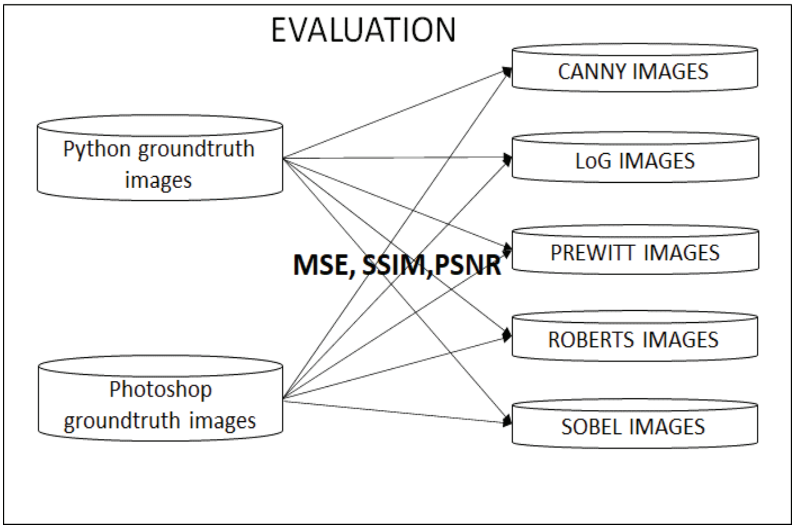


Figure 13: Evaluation of each detected image using MSE, SSIM, and PSNR

i. Mean Square Error (MSE)

MSE is defined as the Mean or Average of the square of the difference between actual and estimated values. The lower the MSE, the better the forecast. The MSE can be found from the following (7) below:

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \tilde{y}_i)^2 \tag{7}$$

where n = number of items, Σ = summation notation, y_i = original or observed y-value, $\tilde{y}_i$ = forecast or y-value from regression.

ii. Structural Similarity Index (SSIM)

SSIM is used to measure the similarity between two given images. SSIM is designed by modeling any image distortion as a combination of three factors that are loss of correlation (s), luminance distortion (l), and contrast distortion (c). The SSIM is defined as:

$$SSIM(x,y) = l(x,y).c(x,y).s(x,y) \tag{8}$$

where x, y are the two compared images, luminance distortion (l) is defined as (9), contrast distortion (c) is defined as (10), and loss of correlation (s) is defined as (11).

$$l(x,y) = \frac{2\mu_x\mu_y + c_1}{\mu_x^2 + \mu_y^2 + c_1} \tag{9}$$

$$c(x,y) = \frac{2\sigma_x\sigma_y + c_2}{\sigma_x^2 + \sigma_y^2 + c_2} \tag{10}$$

$$s(x,y) = \frac{\sigma_{xy} + c_3}{\sigma_x\sigma_y + c_3} \tag{11}$$

with μ_x = the average of x, μ_y = the average of y, σ_x^2 = the variance of x, σ_y^2 = the variance of y, $\sigma_x\sigma_y$= the covariance of x and y, $c_{1=}(k_1L)^2$ and $c_{2=}(k_2L)^2$ are two variables to stabilize the division with weak denominator, $c_{3=}\ c_2/2$,

L = the dynamic range of the pixel-values (typically this is $2^{\#bits\ per\ pixel} - 1$), $k_1 = 0.01$ and $k_2 = 0.03$.

SSIM value ranges from [0,1]. The maximum value of 1 indicates that the two signals are perfectly structurally similar, while a value of 0 indicates no structural similarity.

iii. Peak signal to noise ratios (PSNR)
Here, PSNR cites the ratio between the edge detected images, i.e., the estimator output, and the ground truth image, which is also the estimated image. PSNR can be rated by (12):

$$PSNR = 20.\,log_{10}\left(\frac{\max i}{\sqrt{MSE}}\right) \tag{12}$$

The PSNR value approaches infinity as the MSE approaches zero; this shows that a higher PSNR value provides a higher image quality. At the other end of the scale, a small value of the PSNR implies high numerical differences between images.

4.0 RESULTS AND DISCUSSION

Upon completing the evaluation step, the results are shown and discussed here. Table 1 and Figure 14, Table 2 and Figure 15, Table 3 and Figure 16 illustrate the average values of MSE, SSIM, and PSNR, respectively.

Table 1: Ground Truth Images -Vs.- Edge Detected Images

MSE					
versus	Roberts	Sobel	Prewitt	Canny	LoG
Photoshop	1879.43	8349.44	3209.41	4736.34	1322.54
OpenCV	4184.86	8781.08	4615.94	5249.07	3556.35

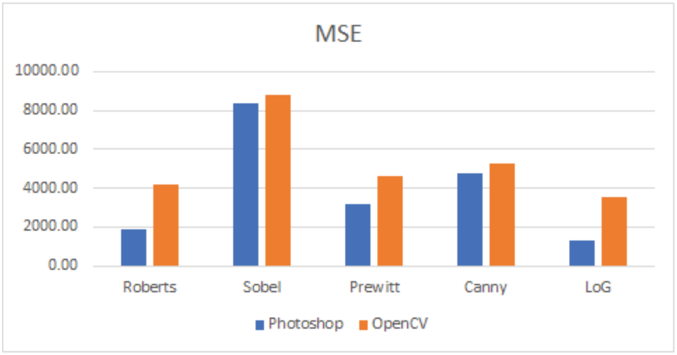


Figure 14: MSE results

Table 2: Ground Truth Images -Vs.- Edge Detected Images

SIM					
versus	Roberts	Sobel	Prewitt	Canny	LoG
Photoshop	0.72	0.54	0.65	0.67	0.83
OpenCV	0.73	0.54	0.66	0.76	0.76

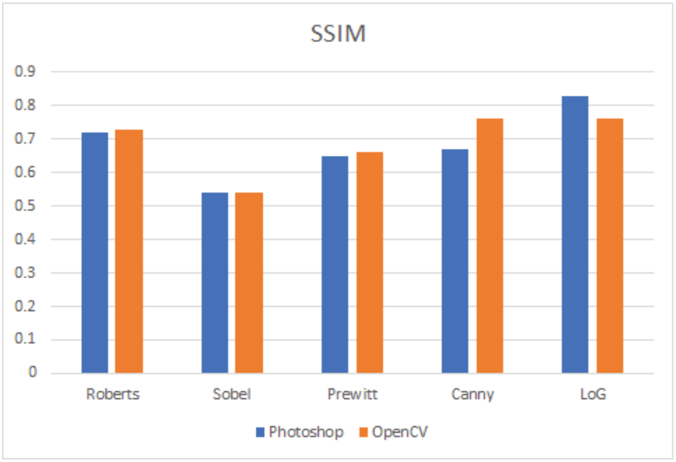


Figure 15: SSIM results

It is seen that LoG manages to detect edges as similar as the ground truth images where its MSEs are the lowest, its SSIMs are the highest, and its PSNR is the highest. Having fixed characteristics in all of the directions, detecting good edges, and its orientations [29] are some of the good qualities of LoG that help its effectiveness in detecting correct edges.

The Sobel operator is the least effective in finding similar edges to the ground truth images. It scores the highest MSE, the smallest SSIM, and the smallest PSNR values. Although this operator produces the position of edges more accurately, it has the short support of filters which causes vulnerability to noise [29], [30] which could result in its poorest performance among all operators. Prewitt and Canny show almost similar performance because both are less vulnerable to noise [29], [30]. Roberts falls second worst, which is expected because it is also susceptible to noise like Sobel [29], [30].

Kruskal-Wallis test has been used to compare the medians of all methods since the assumptions of the one-way ANOVA test are not met. Based on the results, performance measure SSIM has been selected to check whether the results for all methods are significantly different from each other or not. The hypotheses statements are as follows:

H_0: The medians of SSIM values are equal for all methods.
H_1: The medians of SSIM values are not equal for all methods

Table 3: Ground Truth Images -Vs.- Edge Detected Images

PSNR					
versus	Roberts	Sobel	Prewitt	Canny	LoG
Photoshop	9.21	9.16	13.40	11.61	15.65
OpenCV	5.38	8.94	11.75	11.13	10.52

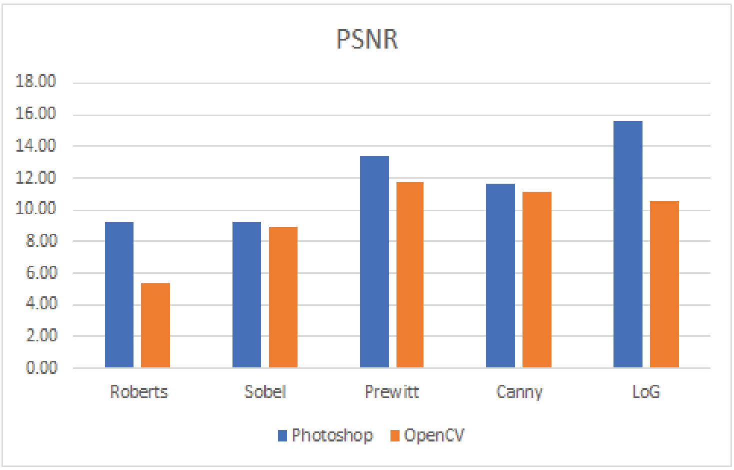
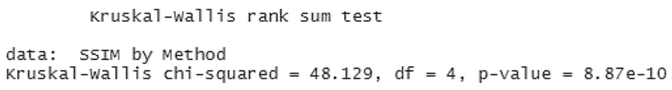


Figure 16: PSNR results

RStudio has been used to run the Kruskal-Wallis test. The result is as in Figure 17. The value of test statistics, H (4) = 48.129, p = 8.87e-10 ($p < 0.05$), which shows the medians of SSIM values are not equal for all methods significantly.



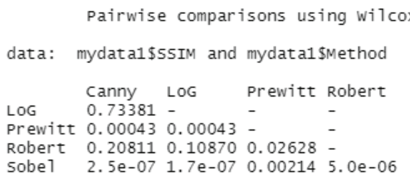
```
              Kruskal-Wallis rank sum test

data:  SSIM by Method
Kruskal-Wallis chi-squared = 48.129, df = 4, p-value = 8.87e-10
```

Figure 17: Results from RStudio for Kruskal-Wallis test

Then pairwise comparison tests are run to check which pairs are significantly different from each other.

The results in Figure 18 show that the results of SSIM for Canny are significantly different ($p < 0.05$) with Prewitt and Sobel. The results of the Prewitt method are also significantly different from Canny and LoG. The same goes for Roberts and Prewitt and Sobel. Since the median for Canny (median = 0.7840) and LoG (median = 0.7839) are approximately the same, then SSIM values do not significantly differ between Canny and LoG.



```
        Pairwise comparisons using Wilcoxon rank sum exact test

data:  mydata1$SSIM and mydata1$Method

        Canny   LoG     Prewitt Robert
LoG     0.73381 -       -       -
Prewitt 0.00043 0.00043 -       -
Robert  0.20811 0.10870 0.02628 -
Sobel   2.5e-07 1.7e-07 0.00214 5.0e-06
```

Figure 18: Results from RStudio for pairwise comparison tests

5.0 CONCLUSION

This study compares different established edge detection algorithms to detect edges of interest in OMR answer sheets such as bubbles or rectangles containing the bubbles. The compared algorithms are Sobel, Roberts, Prewitt, Canny, and Laplacian of Gaussian (LoG). The experimental results show that LoG has accurately detected the edges in OMR images, while Sobel is the least effective. Prewitt and Canny deliver almost similar performances, and Roberts falls second worst to Sobel. These findings might help future studies determine the most appropriate edge detection algorithm in developing a more effective automated OMR grading system.

ACKNOWLEDGMENTS

This work is supported by the research grant PJP/2020/FTMK/PP/S01772 which is granted by Universiti Teknikal Malaysia Melaka.

REFERENCES

[1] A. M. Smith, “Making OMR Easy for Users,” pp. 257–263, 1981.

[2] J. L. Pérez-Benedito, E. Q. Aragón, J. A. Alriols, and L. Medic, “Optical mark recognition in student continuous assessment,” *Rev. Iberoam. Tecnol. del Aprendiz.*, vol. 9, no. 4, pp. 133–138, 2014.

[3] S. C. Loke, K. A. Kasmiran, and S. A. Haron, “A new method of mark detection for software-based optical mark recognition,” *PLoS One*, vol. 13, no. 11, pp. 1–15, 2018.

[4] I. Exams, "E-Assessment Using Image Processing*" Complexity International Journal (CIJ)*," vol. 24, no. 1, pp. 254–262, 2020.

[5] V. Ware, "Cost-effective optical mark recognition software for educational institutions," vol. 5, no. 2, pp. 1874–1877, 2019.

[6] M. Alomran and D. Chai, “Automated Scoring System for Multiple Choice Test with Quick Feedback,” *Int. J. Inf. Educ. Technol.*, vol. 8, no. 8, pp. 538–545, 2018.

[7] G. M. R. I. Rasiq, A. Al Sefat, and M. M. F. Hasnain, "Mobile-based MCQ Answer Sheet Analysis and Evaluation Application," *Proc. 2019 8th Int. Conf. Syst. Model. Adv. Res. Trends, SMART 2019*, pp. 144–147, 2020.

[8] S. Şahin and S. İlkin, “Flexible and efficient mobile optical mark recognition,” *J. Electron. Imaging*, vol. 27, no. 03, p. 1, 2018.

[9] M. Nasim, T. Raj Shekhar, O. Singh Gautam, and Y. Gholap, “OMR Auto Grading System,” *IJISET-International J. Innov. Sci. Eng. Technol.*, vol. 2, no. 5, pp. 522–526, 2015, [Online]. Available: www.ijiset.com.

[10] R. Patel, S. Sanghavi, D. Gupta, and M. S. Raval, "CheckIt - A low-cost mobile OMR system," pp. 3–7.

[11] L.C. Roberts, “Machine perception of three-dimensional solids.,” *Opt. Electra-optical Inf. Process.*, pp. 159–197.

[12] M. R. B. Clarke, R. O. Duda, and P. E. Hart, “Pattern Classification and Scene Analysis.,” *J. R. Stat. Soc. Ser. A*, vol. 137, no. 3, p. 442, 1974.

[13] J. Canny, “A Computational Approach to Edge Detection,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-8, no. 6, pp. 679–698, 1986.

[14] J.M.S Prewitt, “Object Enhancement and Extraction,” *B.S. Lipkin, A. Rosenfeld (Eds.), Pict. Process. Psychopictorics, Acad. Press.*, pp. 75–149, 1970.

[15] D. Marr and E. Hildreth, “Theory of edge detection,” *Proc. R. Soc. London. Ser. B. Biol. Sci.*, vol. 207, no. 1167, pp. 187–217, 1980.

[16] G. S. Robinson, “Color edge detection,” *Opt. Eng.*, vol. 16, pp. 479–484, 1977.

[17] R. A. Kirsch, "Experiments in Processing life motion with a Digital Computer," *Proc. East. Jt. Comput. Conf.*, pp. 221–229, 1957.

[18] K. Atasoy, H., Yildirim, E., Kutlu, Y., & Tohma, “Webcam based real-time robust optical mark recognition,” *Lect. Notes Comput. Sci. (including Subsea. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 9490, no. June 2016, pp. 449–456, 2015.

[19] P. Sanguansat, “Robust and low-cost Optical Mark Recognition for automated data entry,” *ECTI-CON 2015 - 2015 12th Int. Conf. Electr. Eng. Comput. Telecommun. Inf. Technol.*, 2015.

[20] B. K. Kaur and A. Garg, “Comparative Study of Different Edge Detection,” *Int. J. Eng. Sci. Technol. 3*, vol. 3, no. April 2011, pp. 1927-1935., 2011.

[21] H. Aggarwal, “Study and comparison of various image edge detection techniques.” *International Journal of Image Processing (IJIP)*, vol. 3, pp. 1-12, 2009.

[22] B. Akdemir, “Comparison of Edge Detection Algorithms for Texture Analysis Glass Production,” *Procedia - Social and Behavioral Sciences*, vol. 195, pp. 2675–2682, 2015.

[23] B. C. C. Meng, D. S. A. Damit, and N. S. Damanhuri, “Comparative studies of multiscale edge detection using different edge detectors for MRI thigh,” *Bull. Electr. Eng. Informatics*, vol. 10, no. 4, pp. 1979–1986, 2021.

[24] H. A. W. Jasim Albayati and S. A. Ali, “A Comparative Study of Image Steganography Based on Edge Detection,” *J. Phys. Conf. Ser.*, vol. 1818, no. 1, pp. 012032, 2021.

[25] A. S. Mickey Kumar Shah, Vansh Kedia, Rohan Raut, Sakil Ansari, "Evaluation and Comparative Study of Edge Detection Techniques," *IOSR J. Comput. Eng. e*, vol. 22, no. 5, pp. 06–15, 2020.

[26] A. S. Ahmed, "Comparative study among Sobel, Prewitt and canny edge detection operators used in image processing," *J. Theor. Appl. Inf. Technol.*, vol. 96, no. 19, pp. 6517–6525, 2018.

[27] A. Podder and P. B. Chanda, "Comparative Performance Study and Analysis on Different Edge-based Image Segmentation Techniques of Thermal Images," *Int. J. Recent Innov. Trends Comput. Commun.*, vol. 5(5), no. May, pp. 632–636, 2017.

[28] F. Bachofer, G. Quénéhervé, T. Zwiener, M. Maerker, and V. Hochschild, “Comparative analysis of edge detection techniques for SAR images,” *Eur. J. Remote Sens.*, vol. 49, pp. 205–224, 2016.

[29] M. A. Ansari, D. Kurchaniya, and M.Dixit, “A Comprehensive Analysis of Image Edge Detection Techniques,” *Int. J. Multimed. Ubiquitous Eng.*, vol. 12, no. 11, pp. 1–12, 2017, DOI: 10.14257/ijmue.2017.12.11.01.

[30] M. Sharifi, M. Fathy, and M. T. Mahmoudi, “A classified and comparative study of edge detection algorithms,” *Proc. - Int. Conf. Inf. Technol. Coding Comput. ITCC 2002*, pp. 117–120, 2002, DOI: 10.1109/ITCC.2002.1000371.