A New Image Edge Detection Method using Quality-based Clustering

Bijay Neupane
Zeyar Aung
Wei Lee Woon

Technical Report DNA #2012-01

April 2012

Data & Network Analytics Research Group (DNA)
Computing and Information Science Program,
Masdar Institute of Science and Technology,
PO Box 54224, Abu Dhabi, UAE.

A New Image Edge Detection Method using Quality-based Clustering

Abstract

Due to the various limitations of existing edge detection methods, finding better algorithms for edge detection is still an active area of research. Many edge detection approaches have were proposed in the literature but in most cases, the basic approach is to search for abrupt change in color, intensity or other properties. Unfortunately, in many cases, images are corrupted with different types of noise which might cause sharp changes in some of these properties. In this paper, we propose a new method for edge detection which uses k-means clustering, and where different properties of image pixels were used as features. We analyze the quality of the different clusterings obtained using different k values (i.e., the predefined number of clusters) in order to choose the best number of clusters. The advantage of this approach is that it shows higher noise resistance compared to existing approaches. The performance of our method is compared against those of other methods by using images corrupted with various levels of "salt and pepper" and Gaussian noise. It is observed that the proposed method displayed superior noise resilience.

1 Introduction

Image edges contain useful information, which is very important in image processing, machine vision, and pattern recognition mainly in the areas of feature extraction and detection. An edge is the region in the image where there is a sharp change in color intensity, discontinuities in depth and other properties. An edge may represent two different surfaces of the object or a boundary between light and shadow falling on a surface. Since none of the existing methods can can produce the best results for all types of images for all types/levels of noises, finding better methods for edge detection is still an active area of research.

Natural images are prone to noise and artifacts. Salt and pepper noise is a form of noise typically seen on images. It is typically manifested as randomly occurring white and black pixels. Salt and pepper noise creeps into images in situations where quick transients, such as faulty switching, take place. On the other hand, White noise is additive in nature where the each pixel in the image is modified via the addition of a value drawn from a Gaussian distribution. To test the generality of the results, the proposed edge detection algorithm was tested on images containing both these types of noise.

A large number of studies have been published in the field of image edge detection, which attests to its importance within the field of image processing. Many edge detection algorithms have been proposed, each of which has its own strengths and weaknesses; for this reason, hitherto there does not appear to be a single "best" edge detector. A good edge detector should be able to detect the edge for any type of image and should show higher resistance to noise.

Examples of approaches to edge detection include algorithms such as the Sobel, Prewitt and Roberts edge detectors which are based on the first order derivative of the pixel intensities. The Laplacian-of-Gaussian (LoG) edge detector is another popular technique, using instead the second order differential operators to detect the location of edges [3]. However, all of these algorithms tend to be sensitive to noise, which is an intrinsically high frequency phenomenon. To solve this problem the Canny edge detector was proposed, which combines a smoothing function with zero crossing based edge detection [2]. Although it is more resilient to noise than the previously mentioned algorithms, its performance is still not satisfactory when the noise level is high. There are many situations where sharp changes in color intensity do not correspond to object boundaries like surface marking, recording noise and uneven lighting conditions [10].

In this paper we propose a clustering based technique for enhancing the performance of edge detection algorithms and providing better resistance to noise. This is achieved by filtering out outliers in the images and only identifying real object boundaries as edges. The proposed algorithm uses the variance, entropy, gradient, and busyness of each image pixel as a feature vector and employs the widely used k-means clustering algorithm on the pixel feature vectors in order to detect edge pixels. A further challenge is the determination of a suitable value of k (i.e., the number of clusters). Many of the previous papers on edge detection do not explain how this choice was made. In brief, this paper seeks to propose a novel edge detection algorithm with the following key properties:

- Pixels are encoded using feature vectors containing the variance, entropy, gradient and busyness.
- Use of the k-means clustering algorithm to extract pertinent patterns in the feature vectors, which are in turn labelled as either edge or non-edge.
- Application of silhouette analysis to determine the optimal value of k.

Experiments were conducted on images corrupted with up to 30% level of Gaussian and salt and pepper noise. The performance of our method is compared with the Sobel and Canny edge detectors for normal and noisy images. Our approach achieves good results with grayscale images and demonstrates a higher resistance to noise.

2 Related Work

Many of the previous algorithms like Sobel, Prewitt, Robert, Laplace etc. belong to the family of gradient-based edge detection techniques. The basic principle behind gradient-based edge detection is that edge lie along these large gradients with maximum amplitude [3]. Canny [2] introduced multi-stage algorithm to detect a wide range of edge in image. This algorithm show good result with low noise level but its performance degrades with increase in the level of noise.

Previously only a few research works using clustering technique for automatic edge detection were done. Most of them are based on k-means and fuzzy-c clustering technique. Becerikli *et al.* (2006) [1] purposed the alternative neural network based edge detection technique. It used Laplacian method to produce the edge of the image and neural network uses this edge to learn edge of all images.

Isa (2005) [6] proposed a modified seed based region growing (SBRG) algorithm with moving k-means clustering technique for edge detection. This technique overcomes the problem with SRGB proposed by Romberg *et al.* (1997). This new approach overcomes the problem of manual determination of threshold value and initial seed location, which was time consuming and dependent on user.

Zhai and Liu (2006) [11] proposed multi-stage edge detection based on fuzzy c-means clustering. They purposed multiscale wavelet transform for extraction of classification features and used fuzzy c-means clustering algorithm for automatic classification.

Li and Lei (2011)[7] proposed an improved method based on wavelet modulus maximum edge detection algorithm. They proposed a technique for automatic determination of function for eliminating noise threshold using clustering technique. In their experiment, they utilized B-spline wavelet and improved k-means clustering algorithm.

Ganguly *et al.* (2009) [5] proposed another approach for edge detection using artificial features of the image as the feature set and using k-means clustering algorithm for clustering to detect clearly the edges of the objects present in the image in question. It uses busyness, mean, variance and entropy as artificial features for clustering algorithm.

Although most of the above mentioned approaches claimed to have better performance with pure image or image with low level of noises, none of them showed the accuracy and sensitivity of their algorithms with respect to high levels of noises. They also did not propose how the choose the optimal number of clusters for each image.

Following the approach proposed by Ganguly *et al.*, we propose a new method using four image pixel properties as features. In additional to the three feature (busyness, variance, and entropy) proposed by them, we added gradient as the forth feature. Ganguly *et al.* did not show their performance with corrupted images and also did not present a clear idea about the selection of appropriate k value for k-means clustering.

So in this paper, we proposed a new method of edge-detection using augmented clustering approach with an automated selection of the optimal number of clusters based on clustering quality. This proves to be an effective technique for edge detection with higher tolerance for noises.

3 Our proposed method

In this section, we will describe the four different features for image pixels that we use, and k-means clustering and silhouette analysis that are used to cluster the pixels and to measure the quality of resultant clusters respectively and our algorithm for edge detection.

3.1 Pixel Features

The information of an image pixel can be best obtained by comparing the pixel's features with those of its neighboring pixels. This can be done by extracting 3×3 matrix of the neighboring pixels surrounding the pixel in question. (Other odd-number-sized matrices like 5×5 or 7×7 are also possible.) For any pixel $[x, y]$, its neighborhood matrix contains 9 pixels: $[x-1, y-1], \ldots, [x+1, y+1]$. We use a 3×3 neighborhood matrix for extracting of the features of variance, entropy, gradient, and busyness for each pixel in the image. These attributes hold special properties to determine edge and non-edge pixels of the image. For a grayscale image, the color intensity or grayness of a pixel $[x, y]$ will be denoted as $f(x, y)$. The value of $f(x, y)$ ranges from 0 to 255.

3.1.1 Variance

Statical properties like mean and variance contain important information about pixels. Variance is a common measure of how far the numbers lie from the mean. Low variance indicates small variation in grayness and high variance means large variation in grayness. So pixel with high variance is candidate to be an edge pixel. The mean $\mu(x, y)$ of grayness for 3×3 neighborhood matrix centered at the pixel $[x, y]$ is computed as:

$$\mu(x, y) = (1/9) \sum_{i,j=-1}^{1} f(x + i, y + i) \tag{1}$$

Then, the variance $var(x, y)$ of the grayscale values is calculated as follows:

$$var(x, y) = (1/9) \sum_{i,j=-1}^{1} (f(x + i, y + i) - \mu(x, y))^2 \tag{2}$$

3.1.2 Entropy

From information theory, we know that that the smaller the local entropy is, the bigger the information gain and the rate of change in the intensity is. Thus, we can conjecture that the smaller the local entropy is, the bigger the dispersion is. So, the pixel with a big local entropy is more likely to be an edge pixel [3].

For a given pixel $[x, y]$, we take a 3×3 neighborhood matrix with that pixel at the center. The entropy for the pixel $[x, y]$ is calculated as:

$$entropy(x, y) = -\sum_{i=x-1}^{x+1} \sum_{j=y-1}^{y+1} p_{ij} \log p_{ij} \tag{3}$$

$$\text{where } p_{xy} = f(x, y) / \sum_{i=x-1}^{x+1} \sum_{j=y-1}^{y+1} f(i, j)$$

3.1.3 Gradient

The gradient is the directional change in grayness of an image. The magnitude of the gradient tells us how quickly the image is changing, while the direction of the gradient tells us the direction in which the image is changing most rapidly. We use the similar gradient measure as the one used in the Sobel method [9]. The gradient $G(x, y)$ for a 3 neighborhood matrix centered at the pixel $[x, y]$ is computed as:

$$gradient(x, y) = \sqrt{G_X(x, y)^2 + G_Y(x, y)^2} \tag{4}$$

$G_X(x, y)$ is the mask in X direction and $G_Y(x, y)$ is the mask in X direction respectively.

3.1.4 Busyness

Business is a measure of statical dispersion, measuring how a pixel deviates locally from its neighboring pixels. It helps to find the difference between regions in an image. Busyness of a pixel $[x, y]$ in a 3×3 neighborhood matrix is the average of absolute grayness differences of all pairs of twelve adjacent pixels in the neighborhood [4]. The average difference is high in the busy neighborhood where many adjacent pair differ, but it should be low in neighborhood containing vertical or horizontal edges. Mathematically, the busyness of a pixel $[x, y]$ is defined as:

$$\begin{aligned} busyness(x, y) = 1/12 \times \Bigg(& \sum_{i=x-1}^{x+1} |f(i, y-1) - f(i, y)| + |f(i, y+1) - f(i, y)| \\ & + \sum_{j=y-1}^{y+1} |f(x-1, j) - f(x, j)| + |f(x+1, j) - f(x, j)| \Bigg) \end{aligned} \tag{5}$$

3.2 Clustering and Cluster Quality Analysis

3.2.1 K-means Clustering

K-means is a simple yet powerful clustering algorithm. The procedure groups a given set of data points into k clusters, where k is the number of desired clusters which is fixed a priori. The algorithm finds k cluster centroids and assigns each points to its nearest cluster. For example, assume that we have a set of all pixels in an image and we want to group them into two clusters. In an ideal case, all the edge pixels are assigned to one cluster and the non-edge pixels are assigned to another cluster.

3.2.2 Silhouette Analysis

Silhouette width [8] is used to evaluate the quality of a clustering result. This give us an idea of how well-separated the resultants clusters are. For a particular data point i in a cluster, let $a(i)$ be the average distance of the point i to all others points in the same cluster. Let $b(i)$ be the average distance of the point i to all the points in another cluster that is closest to i. Then, silhouette width $s(i)$ for that point i is calculated as:

$$s(i) = \frac{(b(i) - a(i))}{\min(b(i), a(i))} \tag{6}$$

The value of silhouette width ranges from -1 to 1. We calculate the overall average silhouette width $\overline{s}$ of all the points in the data set. The higher the value of $\overline{s}$, the better the quality of clustering.

3.3 Algorithm

Suppose the size of the input image is $m \times n$. First, the input image is padded with a single pixel in each direction with grayness value equal to its adjacent pixel. This is done to preserve the information of image at the margins. This results in a new $(m + 1) \times (n + 1)$ padded image. (If the input image is an RGB color image, we convert it into grayscale.) Then, we extract a 3×3 neighborhood matrix for each of $m \times n$ pixels in the original image. Next, we calculate the variance, entropy, gradient, and busyness values for each pixel.

	Variance	Entropy	Gradient	Busyness
pix1	P_{11}	P_{12}	P_{13}	P_{14}
pix2	P_{21}	P_{12}	P_{13}	P_{14}
...	...	...	...	...
...	...	...	...	...
pix(mxn)	P(mxn)1	P(mxn)2	P(mxn)3	P(mxn)4

Figure 1: Feature vectors, each containing 4 attributes, for $(m \times n)$ pixels in the image.

Then, we extract a feature vector containing those 4 values for each pixel. In this way, we obtain $(m \times n)$ feature vectors each containing 4 attributes as illustrated in Figure 1.

After that, we apply k-means clustering on the $(m \times n)$ feature vectors in our data set(which can also be viewed as data points in a 4-dimensional space). Selecting an appropriate value of k (i.e., the pre-determined number of clusters) is a major challenge in k-means clustering. We try to overcome this by carrying out k-means clustering iteratively with different values of k starting from 2, performing the silhouette analysis for each clustering result, and select the one that gives the highest overall average silhouette width. Due to the time complexity involved in silhouette analysis, we follow a greedy approach in which we stop when the new average silhouette width is less than the old one.

/* let $\mathbf{D}$ be the set of $(m \times n)$ feature vectors (data points). */
$\mathbf{C} \leftarrow \phi$; $\overline{s} \leftarrow -1$; $k \leftarrow 1$;
while (TRUE)
  $k' \leftarrow k + 1$;
  $\mathbf{C}' \leftarrow KMeansClustering(\mathbf{D}, k')$;
  $\overline{s}' \leftarrow AverageSilhouetteWidth(\mathbf{C}')$;
  if $\overline{s}' > \overline{s}$ **then**
    $\mathbf{C} \leftarrow \mathbf{C}'$; $\overline{s} \leftarrow \overline{s}'$; $k \leftarrow k'$;
  else
    exit while loop;
  end if
end while
return $\mathbf{C}, k$; /* optimal clustering result */

The clusters are then sorted by their average silhouette widths. The first $\lceil k/2 \rceil$ clusters are then taken as those of the non-edge pixels and the remaining ones as those of the edge pixels. We construct a new $m \times n$ matrix and assign the value of 255 to a pixel if it belongs to an edge cluster and assign the value 0 otherwise.

/* now $\mathbf{C} = \{C_1, \ldots, C_k\}$ is the optimal set of clusters, where the clusters $C_1, \ldots, C_k$
  are sorted by their average silhouette widths. */
/* let M be a matrix of size $m \times n$ */
for $l = 1$ to $\lceil k/2 \rceil$ /* to label non-edges */
  for each pixel $[i, j]$ in cluster C_l
    $M[i, j] \leftarrow 0$;
  end for
end for
for $l = \lceil k/2 \rceil + 1$ to k /* to label edges */
  for each pixel $[i, j]$ in cluster C_l
    $M[i, j] \leftarrow 255$;
  end for
end for
return M;

An additional filtering step is performed to remove singleton edges. If a particular pixel is labeled as an edge, then we check all of its neighboring pixels. If all of them turn out to be non-edge pixels, then we now

relabel that pixel itself as an non-edge. Finally, we plot the output matrix from the procedure, which gives us the image with the detected edges.

for each cell $M[i, j]$ in matrix M
  if $M[i, j] = 255$ **then**
    $value \leftarrow 0$; /* assumes a new value first */
    for $p, q = -1$ to 1
      if $(p \neq 0$ **or** $q \neq 0)$ **and** $M[i + p, j + q] = 255$ **then**
        $value \leftarrow 255$; /* retains the original value */;
        exit for loop;
      end if
    end for
    $M[i, j] \leftarrow value$;
  end if
end for
return M;

4 Experimental Results

Now, we will report the performance of our proposed method on the test image given in Figure 2(a). The main purpose of choosing this image is to test our method for different types of edge and ranges of edge complexity. We compare the performance of our method with those of Canny [2] and Sobel [9] edge detectors. We use Matlab's built-in functions for those two algorithms. For Canny, the lower threshold was set manually at 0.3, as this seemed to produce the best results. For Sobel, there is not parameter to tune.

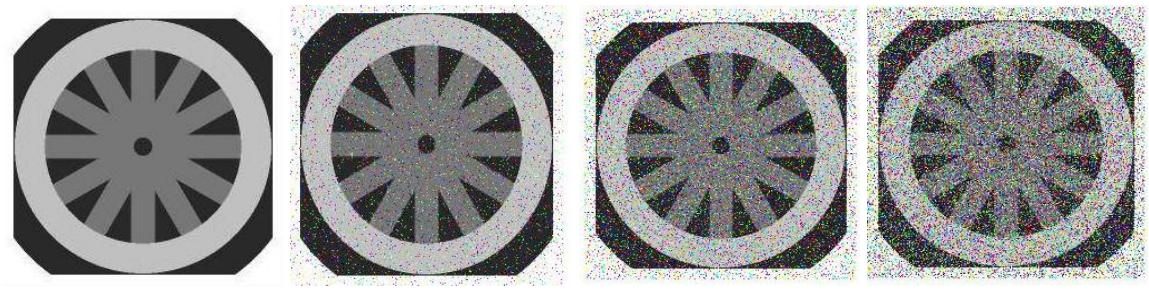
The image is corrupted with 0% to 30% of salt and pepper noises as shown in Figure 2. The results by the three methods (ours, Canny, and Sobel) for salt and pepper noise are depicted in Figures 3 to 5. Again, the image is corrupted with 0% to 30% of Gaussian noises as shown in Figure 6. The results by the three methods for Gaussian noise are depicted in Figures 7 to 9. (The results by our method on more test images with both salt and pepper and Gaussian noises can be found in the website **http://www.dnagroup.org/edge**.)

For slat and pepper noises, we have learned that Canny edge detector shows good result with lower level of noises but deteriorates faster when the noise level in image is increased. Sobel detector performs badly even with the noise level of 5%. When the level of noise is very high its results are the worst. Our proposed clustering-based edge detector produces as good result as Canny at lower level of noises. But it shows more resilience to very high level of noise. From the figures, we can observe that result by our method at 30% of noise level is better than that by Canny. However, we acknowledge that our method incurs some small islands of false edges that Canny does not.

For the images corrupted with Gaussian noises, Sobel fairs better but Canny performs worse than before. However, the performances of both methods are worse than our proposed method as can be clearly observed in the figures.

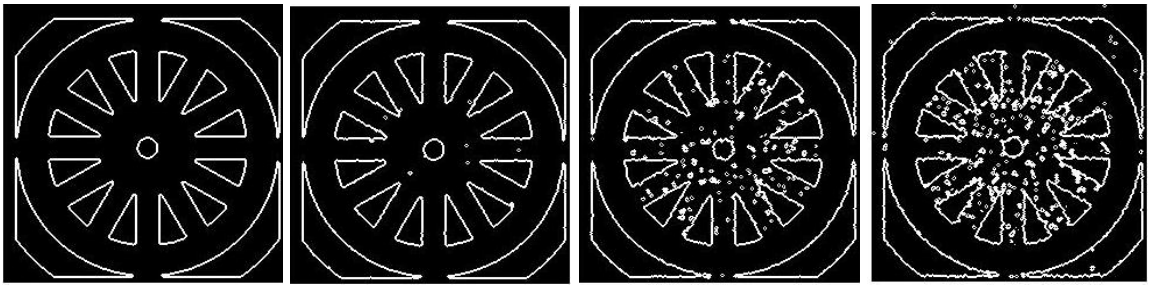
5 Conclusion

The results presented in this paper are preliminary but are sufficient enough to show that using variance, entropy, gradient, and busyness values of the image pixel as features for k-means clustering coupled with silhouette analysis is a good method for detecting edge for noisy images. As for future work, we will try to find out better filtering techniques to remove the small islands of false edges from our results. We also plan to modify our algorithm to detect edges from color images directly without converting it to grayscale.



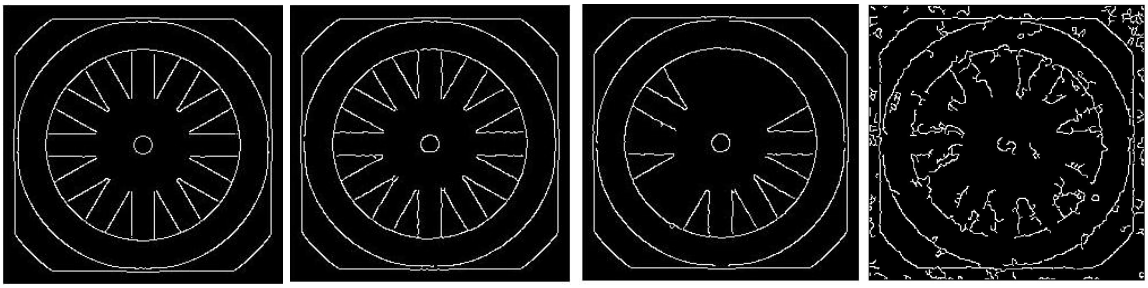
(a) 0% noise (b) 5% noise (c) 15% noise (d) 30% noise

Figure 2: Test images with different level of salt and pepper noises.



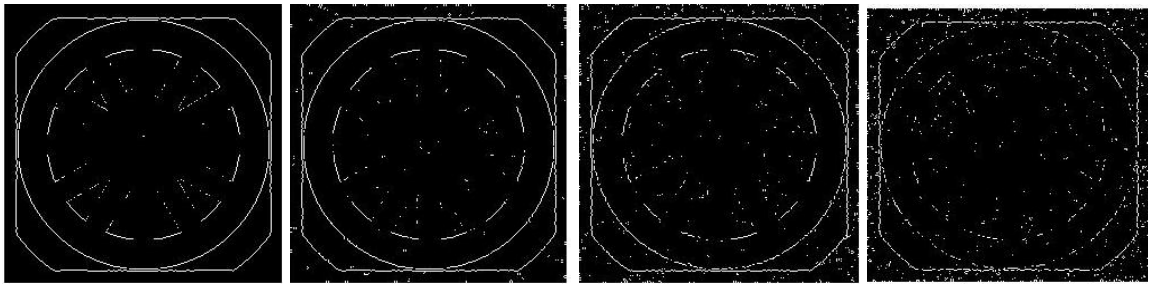
(a) 0% noise (b) 5% noise (c) 15% noise (d) 30% noise

Figure 3: Our method applied to the test images with salt and pepper noises.



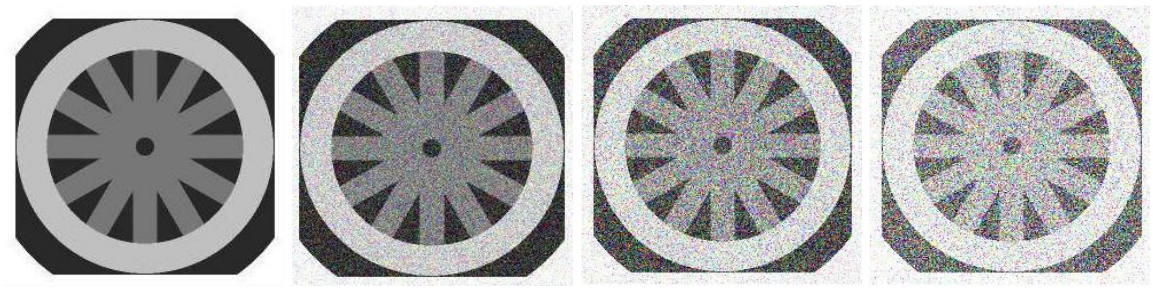
(a) 0% noise (b) 5% noise (c) 15% noise (d) 30% noise

Figure 4: Canny edge detector [2] applied to the test images with salt and pepper noises.



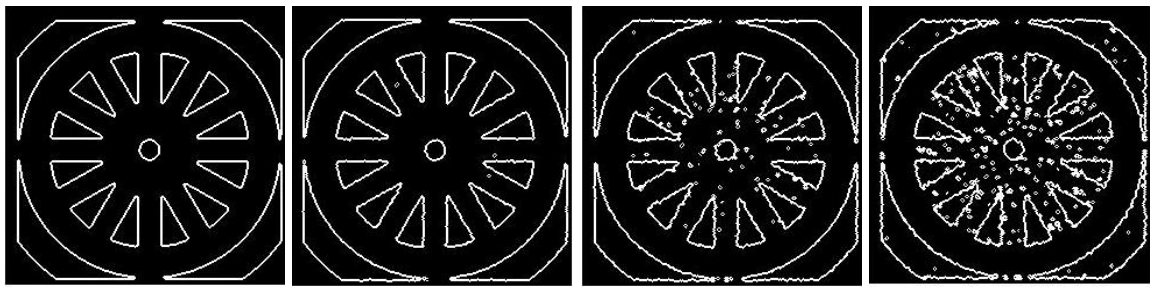
(a) 0% noise (b) 5% noise (c) 15% noise (d) 30% noise

Figure 5: Sobel edge detector [9] applied to the test images with salt and pepper noises.



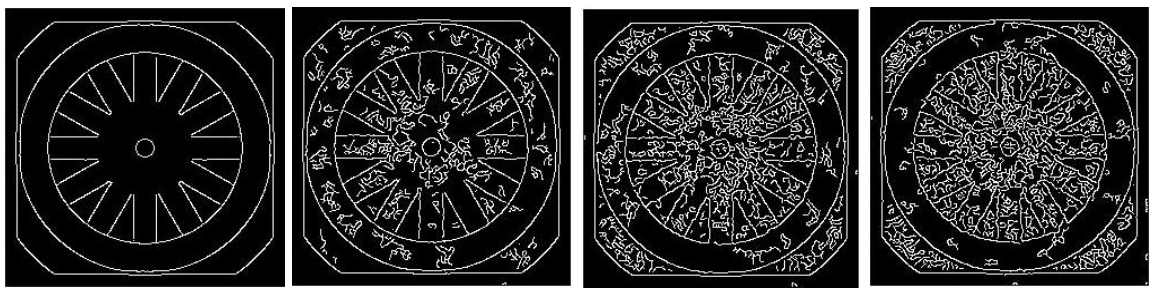
(a) 0% noise (b) 5% noise (c) 15% noise (d) 30% noise

Figure 6: Test images with different level of Gaussian noises.



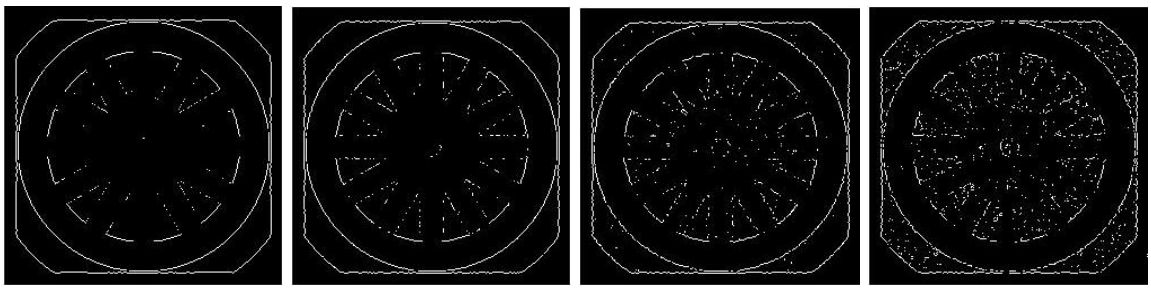
(a) 0% noise (b) 5% noise (c) 15% noise (d) 30% noise

Figure 7: Our method applied to the test images with Gaussian noises.



(a) 0% noise (b) 5% noise (c) 15% noise (d) 30% noise

Figure 8: Canny edge detector [2] applied to the test images with Gaussian noises.



(a) 0% noise (b) 5% noise (c) 15% noise (d) 30% noise

Figure 9: Sobel edge detector [9] applied to the test images with Gaussian noises.

References

[1] Y. Becerikli, H. E. Demiray, M. Ayhan, and K. Aktas, "Alternative neural network based edge detection," *Neural Information Processing – Letters and Reviews*, vol. 10, pp. 193–199, 2006.

[2] J. F. Canny, "Computational approach to edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, pp. 679–698, 1986.

[3] W. Dai and K. Wang, "An image edge detection algorithm based on local entropy," in *Proceedings of the IEEE International Conference on Integration Technology (ICIT'07)*, pp. 418–420, 2007.

[4] P. A. Dondes and A. Rosenfeld, "Pixel classification based on gray level and local "Busyness"," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 4, pp. 79–84, 1982.

[5] D. Ganguly, S. Mukherjee, K. Mitra, and P. Mukherjee, "A novel approach for edge detection of images," in *Proceedings of the International Conference on Computer and Automation Engineering (ICCAE'09)*, pp. 49–53, 2009.

[6] N. A. M. Isa, "Automated edge detection technique for pap smear images using moving k-means clustering and modified seed based region growing algorithm," *International Journal of the Computer, the Internet and Management*, vol. 13, pp. 45–59, 2005.

[7] J. Li and Z. Lei, "Adaptive thresholds edge detection for defective parts images based on wavelet transform," in *Proceedings of the 2011 International Conference on Electric Information and Control Engineering (ICEICE'11)*, pp. 1134–1137, 2011.

[8] P. J. Rousseeuw, "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis," *Computational and Applied Mathematics*, vol. 20. pp. 53–65, 1987.

[9] I. E. Sobel, *Camera Models and Machine Perception*, Ph.D. Thesis, Electrical Engineering Department, Stanford University, California, United States, 1970.

[10] W. L. Woon, P. Liatsis, and K. D. Wong, "Fusion Of multiple edge maps for improved noise resistance," in *Proceedings of MMU International Symposium of Information and Communication Technologies (M2USIC'06)*, pp. 1–8, 2006.

[11] Y. Zhai and X. Liu, "Multiscale edge detection based on fuzzy c-means clustering," in *Proceedings of the 1st International Symposium on Systems and Control in Aerospace and Astronautics (ISSCAA'06)*, pp. 1201–1204, 2006.