5th National Electrical Engineering Conference
I.A.U.N Islamic Azad University Najafabad Branch

Cellular edge detection for color images

Somayeh Golmohamadi, Elnaz Ahmadi Shad, Mohammadreza Asghari Oskoei*
Department of computer science
Allameh Tabatabai university
Tehran, Iran
e-mail: oskoei@atu.ac.ir

Abstract— **Edge is considered an important feature for image processing in computer science. However, to day this technology has mainly be developed for grey scale images as opposed to colored images. Converting to greyscale images involves information loss and is often time consuming. We therefore see a need to develop edge detection for color images as well.**

In this study we propose a Cellular Edge Detection (CED) algorithm based on Cellular automata (CA) and Cellular Learning Automata (CLA) which is a very accurate and reliable way of edge detecting. Every color edge detection procedure is done via the grayscale rules however there are some noticeable difference. The input image is different when a conversion from RGB to HSV occurs while, in RGB images we propose an average or other way of merging results. Results shows that this new way to find edges work accurately and more efficient. The evaluation of this proposed model calculated by MSE and PSNR.

***Keywords*—** CED (Cellular Edge Detection), color image edge detection, CA (Cellular automata), CLA (Cellular Learning Automata), MSE, PSNR.

I. INTRODUCTION

One of the most crucial tasks in image processing which include feature extraction and pattern recognition, is edge detection. This has its own advantages in many image analysis fields like image segmentation, object recognition, target tracking etc. [1]. As an image is considered a 2-D function of f (x, y) where x and y are dimensional mates, and f identifies the image intensity or grayscale level for (x, y) twin, edge detection decreases computation time and storage space of image. Besides, we benefit from more valuable information than less important one about image boundaries [1] [2]. This filters out irrelevant information thus the size of image cut down and structural properties of an image remains for further processing procedures [3]. Therefore, we can name edge detection as a way of data reduction in a wide range of study in image processing field [1]. In other words, we can identify edge detection as a process of simplifying image properties into a set of extracted objects boundaries [1]. Images are divided into color and grayscale ones thus, according to the type of the image edge detection algorithms and their implementations are different from each other [4]. Grayscale images are obviously carrying less information than color images also, human eyes considered sensitive to brightness so it can identify thousands of colors in a complex image rather than grayscale levels [1]. This leads us to start this study about color edge detection which is a more complex task than grayscale edge detection.

At this point we should first answer this question "what is considered as an edge?". In an image, boundaries between regions are categorized as edge [5]. Edge in an image is where there is a jump in a pixel's intensity or color in comparison to its adjacent pixel. In fact, areas in an image with strong intensity or color contrast are defined as edge [6] [7]. Edges should be precise in a way that one edge for a distinct brightness in grayscale images and color in color images recognized by the edge detector algorithm [1]. It is necessary for an edge to be thick enough to identifies patterns and divide regions correctly [1]. Finding a good edge depends on several factors like lighting condition, edge density in the given image, presence of same intensity objects, and noise [5].

In this paper we first focused on background works on CED, CA, and CLA. The next part is the results of our proposed model.

II. RELATED BACKGROUND

Automaton theory introduces a field named learning automata which is explained as a machine with the ability of doing finite number of actions and an evaluation in a probabilistic environment. The next step is that this evaluation response is used to learn automaton whether with positive or negative reinforcement signals which can lead to a different action taking by learning automaton. Eventually what the automaton should reach is learn to do the best action it can do among all possible actions. The best action is defined as follow: whatever can maximize the probability of getting positive signal from the

environment. Here there is a need to define several models for environment, learning automata, cellular automata, and cellular learning automata.

A model is considered for environment which consist of a three part: E =$\{\alpha, \beta, c\}$. In this triplet the input set is$\alpha(n) = \{\alpha_1, \alpha_2, \dots, \alpha_r\}$, the output set is $\beta = \{\beta_1, \beta_2, \dots, \beta_m\}$, and the set of penalty properties is $c = \{c_1, c_2, \dots, c_r\}$. This shows that α_i has a penalty signal with the probability of c_i. There is a different in c_i values between environment during the process of learning of automata, in static environment these values remain steady while in non-static environment they change [2]. In other words if c_i depend on *n* then the environment is non-static otherwise it is static. According to the nature of learning automata it can divide environment into three different models which are called P-model, Q-model, and S-model [8].

Learning automata is consist of five factor $\{\alpha, \beta, F, G, \varphi\}$ the first one is action set $\alpha = \{\alpha_1, \dots, \alpha_r\}$, the second one is input set $\beta = \{\beta_1, \dots, \beta_m\}$, the last one is internal states set $\varphi = \{\varphi_1, \dots, \varphi_s\}$, F and G are functions. $F: \varphi \times \beta \to \varphi$ function generates the new state of learning automaton and g function which is defined like $G: \varphi \to \alpha$ maps the current state to the next input of learning automaton and considered as the output function [2].

Cellular automata (CA) are mathematical models to discover how to act in a complex system which has many simple identical components plus local rules [9]. CA are considered discrete nonlinear dynamic systems which a network of cells and discrete time advance are defined as space. Each cellular and automata has their own meaning where cellular means consisting of cells and automata means obeying a simple rule. In CA at any time the next state of each cell can be predicted by its adjacent cell's state. Besides, rules are considered globally and apply through the whole CA. These rules show us how the main cell affected by its adjacent cells and how they provide complex behavior out of simple rules. But what cells are assumed neighbor and can have effects on each other? According to CA rules, cells which can influence the other cells are considered as the neighbors to the main cell.

To determine a D-dimensional CA we should consider it a quadruple of $\{Z^D, \varphi, N, F\}$. A network of D-tuple ordered integers is shown as Z^D. This set can be finite, semi-finite or infinite, where $\varphi = \{1, \dots, m\}$ is a finite set of action. A finite subset of Z^D is essential for neighborhood network as it is defined like $N = \{\bar{X}_1, \dots, \bar{X}_{\bar{m}}\}$ where $\bar{X}_{\bar{i}} \in Z^D$. At last the local rule of CA is shown as $F: \underline{\varphi}^{\bar{m}} \to \underline{\varphi}$.

In the network cell there is a neighborhood vector N(u) which defines the relative position of neighbors for each cell u. There are two main constraints which should be payed attention while calculating N(u) through the following equation:

$$N(u) = \{u + \bar{X}_i | \mathrm{i} = 1, \dots, \bar{m}\} \qquad (1)$$

These constraints are defined as follow:

$$\begin{cases} \forall u \in Z^D \Rightarrow u \in N(u) \\ \forall u, v \in Z^D \Rightarrow u \in N(u) \wedge v \in N(u) \end{cases} \qquad (2)$$

There are three local rule for F which calculate the next value of the main cell:

1.General: the neighboring cells in the current time step define the next value of main cell.

2.Totalistic: various states of neighboring cells characterized this value.

3.Outer totalistic: the current state plus the various state of neighboring cells determine the value.

A combination of LA and CA leads to a new model called CLA which combines learning ability from LA and collaboration neighborhood of CA. CLA can be used to model the real-world problems because the CLA is a model which considered as a tool to model the simple component systems. Also, each component behaves in a way that its neighbor and its previous experiment leads it to. In fact, simple interactions of single components eventuate a complex behavior.

Overall we can consider a big structure which consist of CA while there are one or more LA in each cell of this structure. LA has the responsibility to determines the state of the corresponding cell [2]. The procedure can be described as follow:

1.Each cell's LA's action probability vector specifies the state of that cell while the initial value may be considered random or from the past experience.

2.LA in each cell receive reinforcement signal from the rule of CLA.

3.This step is updating step. It means action probability vector of each LA updates based on supplied reinforcement signal plus the action that the cell choses.

This three step process continues until acceptable results will be acquired [9].

The D-dimensional CLA are defined as a quintuple of $\{Z^D, \varphi, A, N, F\}$. A network of D-tuple ordered integers is shown as Z^D which could be finite, semi-finite or infinite set. A finite set of actions is defined as $\varphi = \{1, \dots, m\}$. A set of LA is named as "A" each of which corresponds to a specific CLA cell. A finite subset of Z^D is defined as $N = \{\bar{x}_1, \dots, \bar{x}_{\bar{m}}\}$ which is the neighborhood vector. The last one is the local rule of CA which is shown as $F: \underline{\varphi}^{\bar{m}} \to \underline{\beta}$ where β is the reinforcement signal.

Hasanzadeh Mofrad and et al. proposed a cellular model to detect edges of binary and gray scale images

using a nested chain of CA and CLA which is called cellular edge detection [2]. As pointed before one of the most important ways of feature extraction is edge detection therefore, it is essential to be accurate in finding edges in images.

Each pixel of an image relates to a CA thus, the cellular edge detection (CED) utilize this assignment. Another point which should be considered here is that neighborhood type of CA determines with CLA. Combining these two techniques lead us to a better edge map with high accuracy [2]. CA and CLA both are used as a 2-D symmetric and nondeterministic matrix in cellular edge detection model. Variations in brightness, color or details in images are detected based on CA rules to find edges.

Cellular automata framework defines as follow: CA is a quadruple of $\{Z^2, \varphi, N, F\}$ where Z^2 is a 2-D CA which is identical to the input image $I_{m\times n}$, a set of finite state of 256 intensity levels is defined as $\varphi = \{0, \dots, 255\}$ for a grayscale image and each channel of color image, a set of available neighborhood types of CA is shown as $N = \{Moore_{3\times3}, von\ Neuman_{3\times3}\}$, and the local rule of CA is defined as F which controls the relationship between the central cell and its neighbors.

Cellular learning automata framework is defined as follow: CLA is a quintuple of $\{Z^2, \varphi, A, N, F\}$ where Z^2 is 2-D CLA which is identical to the input image $I_{m\times n}$, a set of finite state is shown as $\varphi = \{Moore_{3\times3}, von\ Neuman_{3\times3}\}$, A is the set of LA which is identical to the input image $I_{m\times n}$, a set of available neighborhood of CLA is defined as $N = \{Moore_{3\times3}\}$,

And the local rule of CLA is defined as:

$$\beta_{i,j} = \begin{cases} 1\ if\ A_{i,j} = A(N) \forall\ A \in N \\ 0 \qquad\qquad Otherwise \end{cases} \qquad (3)$$

This calculates the reinforcement signal through local interactions of LA of each learning automaton. In a neighborhood choosing the same action by all LAs leads to a reward otherwise a punishment will be received. Thus, in analogous segments of images similar neighborhood types will be chosen [2].

The central cell considered as edge (J=1) if the absolute difference between the intensity of central pixel and the maximum of its neighborhood violates the threshold θ, otherwise it is not considered as an edge (J=0).

$$J_{i,j} = F(N) = \begin{cases} 1\ \ if\ \left|I_{i,j} - \max(N)\right| \geq \theta \\ 0 \qquad\qquad otherwise \end{cases} \qquad (4)$$

III. METHOD

Our source code is freely available at https://github.com/somayeh71/Cellular-edge-detection-for-color-images. In the present study cellular edge detector is used as monochrome edge detector which develops for color images according to following ways:

1. After extracting average and luminance from each Red, Green, and Blue (RGB) color space channels, CED is applied on its result.
2. After mapping RGB into Hue, Saturation, and Value (HSV) color space, CED can be applied on each channel of new color space.
3. A combination of edges detected by CED in each channel of RGB produces a tensor which has three channels thus, the result is in color.
4. After applying CED on each channel of color images the maximum value of each pixel from three channels produces the new result [10].

IV. RESULTS

This algorithm uses CA and CLA to find edges of an image. The responsibility of dynamicity in boundary growth and object learning is on CA and CLA, respectively [2]. All in all, combining these two reinforcement learning techniques leads to finding edges of images with different intensities.

Two typical images are used in our study (coins, peppers color) to evaluate this algorithm and comparing it with color canny and image Vector Gradient edge detector (VG-edge detector) for color images [5]. The maximum number of generations is 50. α=β are set as 0.5 which are reward and penalty parameters in learning algorithm. The threshold value θ set in the range of [15,30] manually [2].

Fig.1. and Fig.3. show how good was the algorithm to find edges in color images according to four ways mentioned above.

Fig.2. and Fig.4. show color Canny edge detector and VG-edge detector for color images as it can be seen in addition to the coins and bell peppers there are several edges detected in the image which are not edges of objects and contain useless information. This may cause difficulties due to the fact that feature extraction is a very important task and it should be accurate enough to be reliable.

Edge detection of noisy images

We extended this study for noisy images to see if the proposed CED algorithm is sensitive to noise or not. Therefore, we applied salt and pepper impulse noise with probability of p=0.005 [2].

Fig.5. and Fig.7. show how resistant the CED algorithm is against noise. Results are approximately the same as image without noise just few dots can be mentioned as the difference between two images (noisy and without noise).

Fig.6. and Fig.8. show the results of noisy images edge detection by Canny color and VG-edge detection.

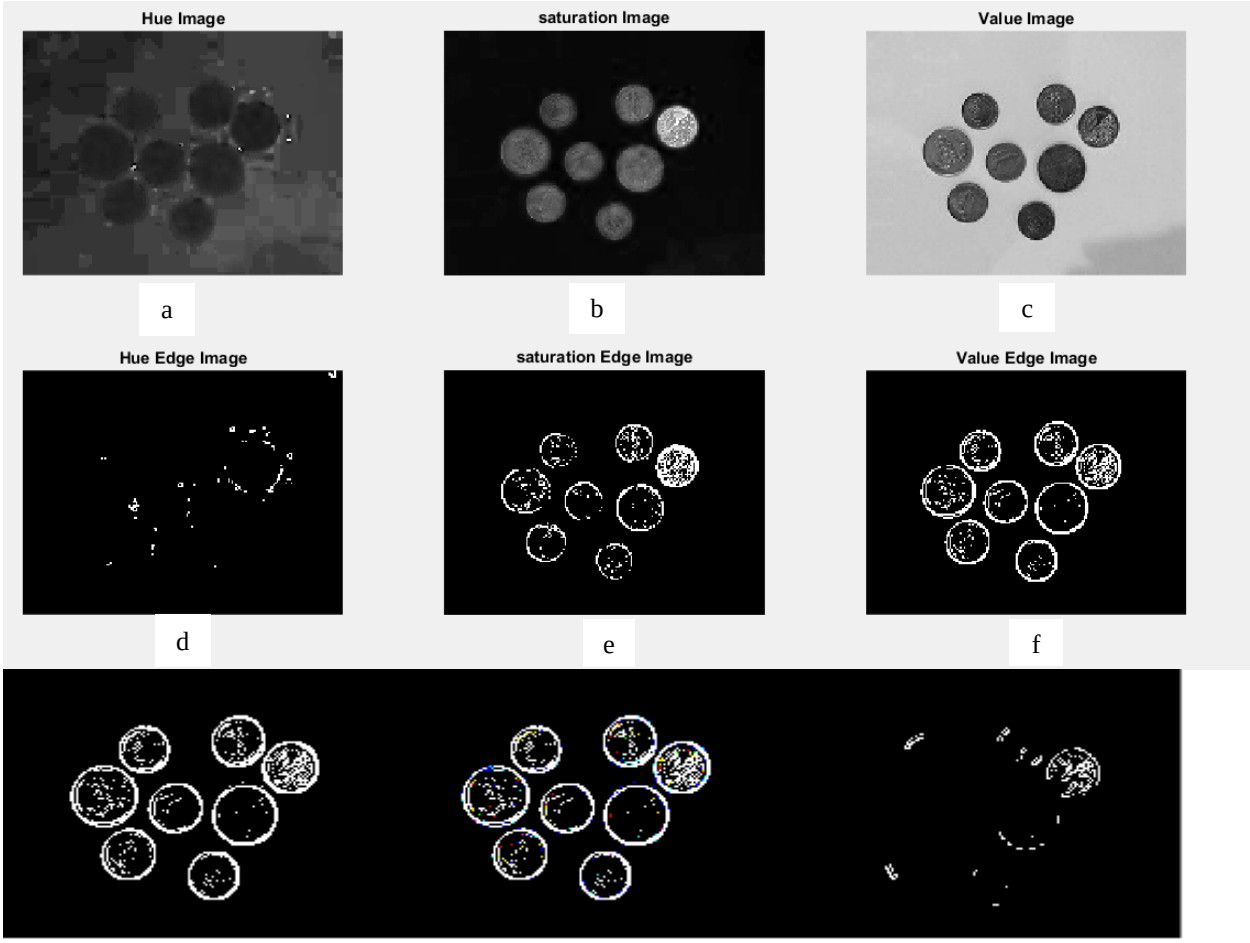
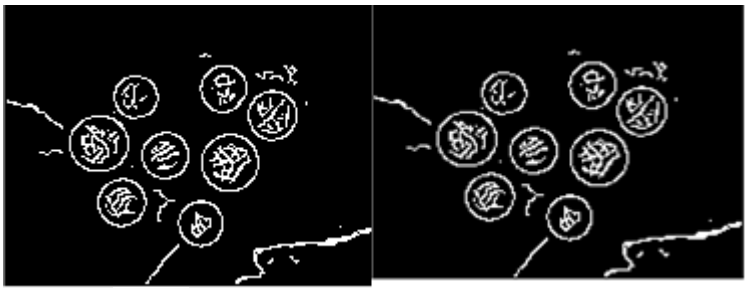


Fig.1. a) hue part of HSV, b) saturation part of HSV, c) value part of HSV, d) output of proposed edge detector for hue image, e) output of proposed edge detector for saturation image, f) output of proposed edge detector for value image, g) edge detected with the maximum of each channel, h) combining of RGB, i) the average of channels RGB.



a b

Fig.2. a) color canny edge detector, b) VG-edge detector

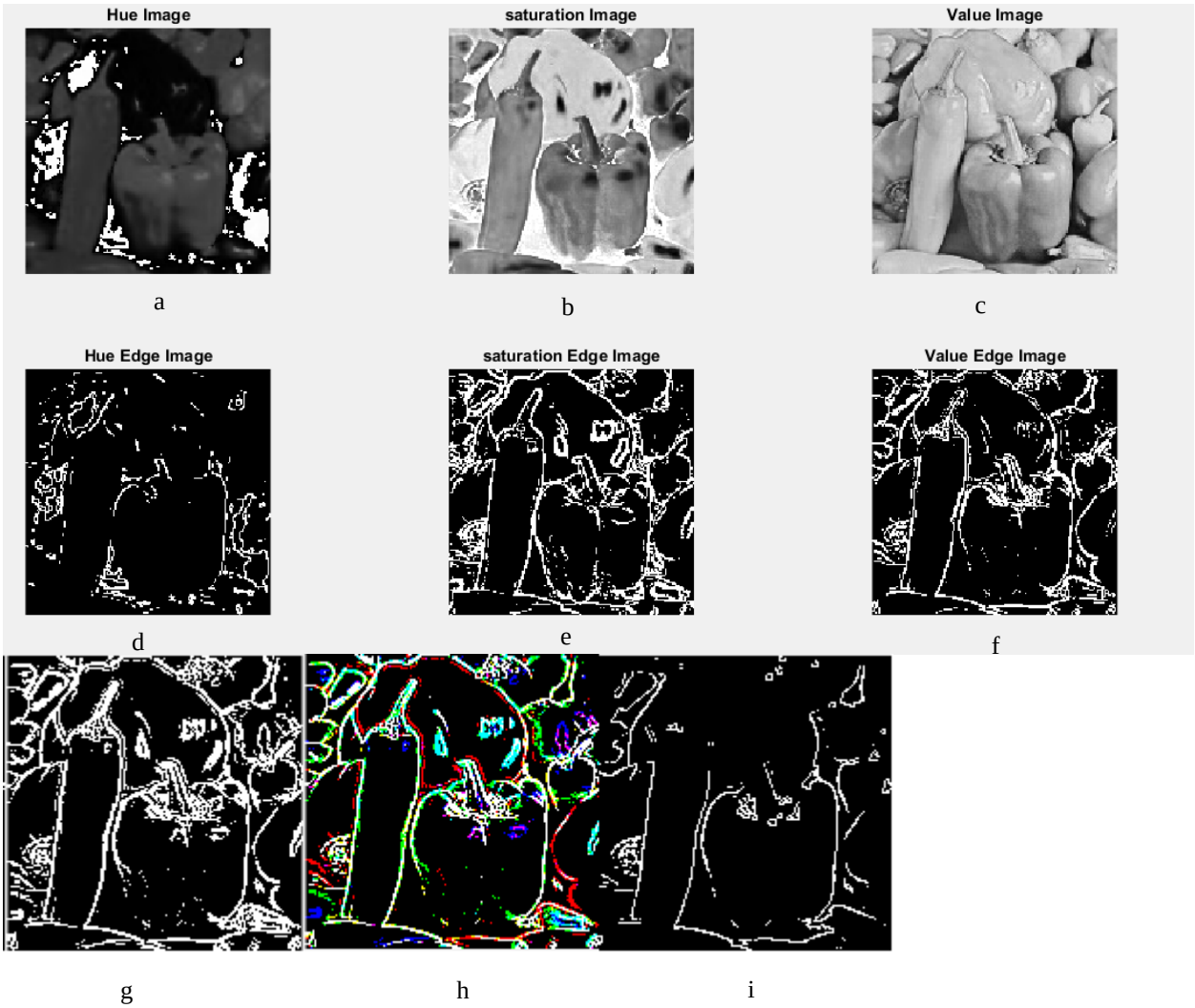


Fig.3. a) hue part of HSV, b) saturation part of HSV, c) value part of HSV, d) output of proposed edge detector for hue image, e) output of proposed edge detector for saturation image, f) output of proposed edge detector for value image, g) edge detected with the maximum of each channel, h) combining of RGB, i) the average of channels RGB.

Fig.4. a) color canny edge detector, b) VG-edge detector

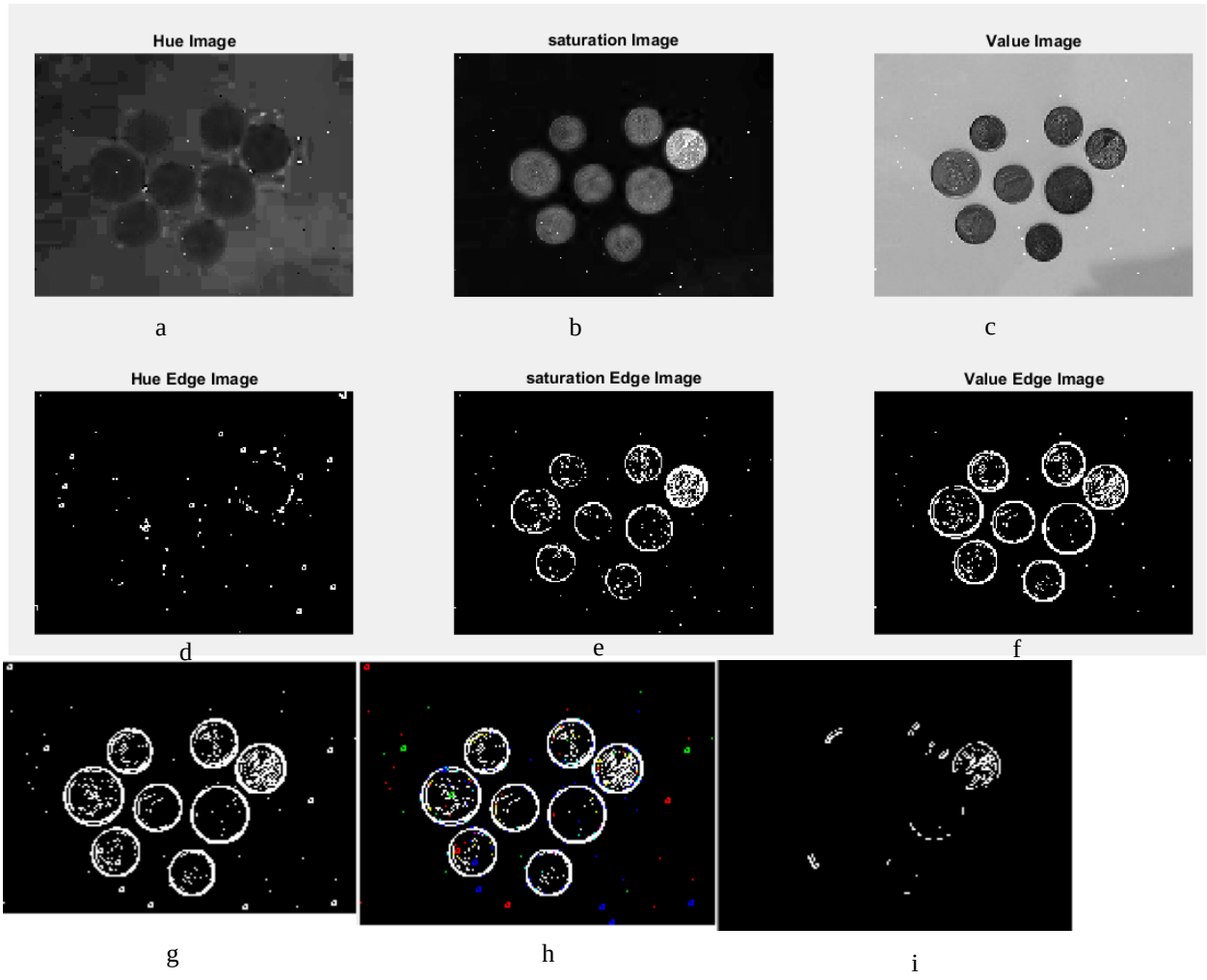


Fig.5. a) hue part of HSV with salt and pepper noise, b) saturation part of HSV with salt and pepper noise, c) value part of HSV with salt and pepper noise, d) output of proposed edge detector for hue image with salt and pepper noise, e) output of proposed edge detector for saturation image with salt and pepper noise, f) output of proposed edge detector for image with salt and pepper noise, g) edge detected with the maximum of each channel with salt and pepper noise, h) combining of RGB with salt and pepper noise, i) the average of channels RGB with salt and pepper noise.

a b

Fig.6. a) color canny edge detector with salt and pepper noise, b) VG-edge detector with salt and pepper noise.

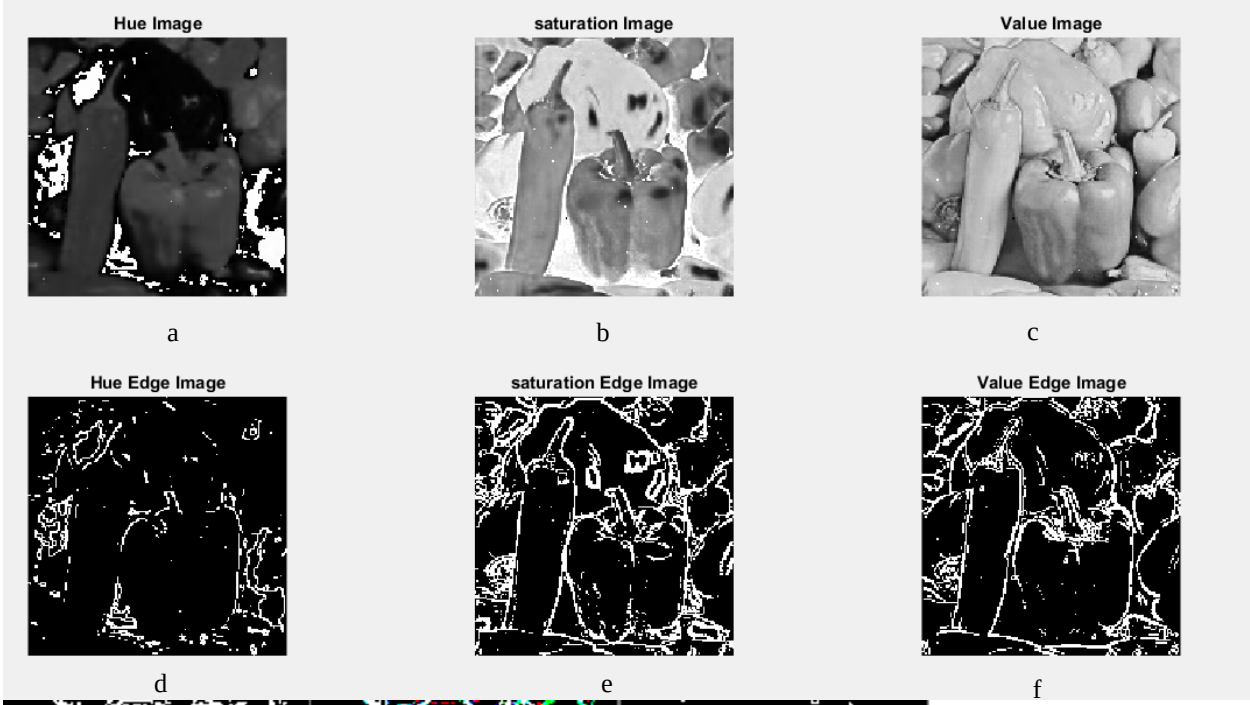


Fig.7. a) hue part of HSV with salt and pepper noise, b) saturation part of HSV with salt and pepper noise, c) value part of HSV with salt and pepper noise, d) output of proposed edge detector for hue image with salt and pepper noise, e) output of proposed edge detector for saturation image with salt and pepper noise, f) output of proposed edge detector for image with salt and pepper noise, g) edge detected with the maximum of each channel with salt and pepper noise, h) combining of RGB with salt and pepper noise, i) the average of channels RGB with salt and pepper noise.

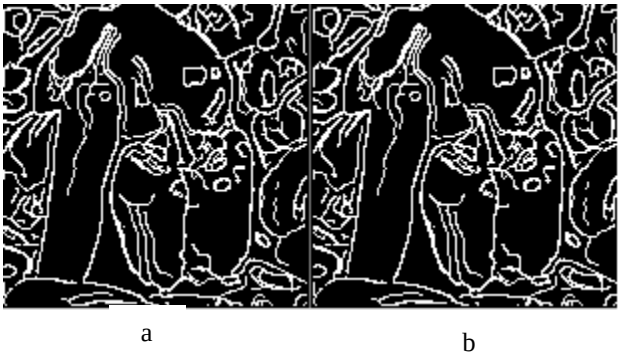


Fig.8. a) color canny edge detector with salt and pepper noise, b) VG-edge detector with salt and pepper noise.

V. EVALUATION CRITERION

Two important criterion to evaluate performance of edge detectors are mean square error and peak signal to noise ratio.

Mean square error (MSE) is used to calculate the average difference between original and edge detected image. Thus, lower MSE shows a very slight error between these two pictures. If we consider original image as I and edge detected image as J, then we can use the following formula to calculate MSE while m is height and n is width of images:

$$MSE = \frac{1}{m \times n} \sum_{i=1}^{m} \sum_{j=1}^{n} (I_{i,j} - J_{i,j})^2 \qquad (5)$$

Peak signal to noise ratio

Peak signal to noise ratio (PSNR) is measured in decibel (dB) scale. This ratio is between the original image and the distortion signal in an image. The higher PSNR will be, the better quality of image would achieve. PSNR is calculated by MSE:

$$PSNR = 10\, log_{10}(\frac{R^2}{MSE}) \qquad (6)$$

In this formula R=255 is the maximum variation of an 8-bit input image.

In CED lower MSE and higher PSNR shows how good is this model to detect edges in images.

In both tables 1 and 2 the highest PSNR and lowest MSE is marked by '**' and it is belonged to hue edge image. However, the color CED combine and color CED maximum which are marked by '*' are nearly the same and obviously acts better both from the evaluation criterion and edge images in comparison to other edge detectors.

VI. CONCLUSION

One of the most important task in image processing is edge detection. As most images are color, edge detection for these kind of images seams necessary. Due to the fact that a conversion of color image into grayscale or binary was obligatory for the previous edge detectors 'grayscale and binary edge detectors' computing time and procedures may cause difficulties. Benefiting from CA and CLA a CED model proposed for grayscale and binary images which we proposed it for the color images in this paper. We tried to investigate different kind of input color images to evaluate which one is the best color edge detector.

Experiments results indicated that CED is a very robust and accurate way to find edges in color images. In comparison with other color edge detection technique this one is required less time and produces better results.

Table 1

MSE and PSNR in decibel of VG-edge detection, color Canny, color average of channels, luminance of channels, hue edge, saturation edge, value edge, color CED combine, and color CED maximum of coins and pepper.

Algorithm	Image			
	Coins		Pepper	
	Mean MSE	Mean PSNR	Mean MSE	Mean PSNR
VG-edge detector	10411.04	7.9909	8076.58	9.1092
Canny color	10353.58	8.0174	5095.11	11.7336
Color average of channels	10358.50	8.0154	5104.84	11.7240
Luminance of channels	10353.66	8.0174	5095.83	11.7334
Hue edge image	5314.88**	10.9163 **	3670.08**	12.8512**
Saturation edge image	10358.57	8.0154	5105.76	11.7230
Value edge image	10357.26	8.0159	5099.23	11.7288
Color CED combine	10353.67*	8.0147*	5096.45*	11.7325*
Color CED maximum	10353.57*	8.0175*	5095.02*	11.7346*

Table 2
MSE and PSNR in decibel of VG-edge detection, color Canny, color average of channels, luminance of channels, hue edge, saturation edge, value edge, color CED combine, and color CED maximum of coins and pepper by salt and pepper impulse noise with probability of p=0.005.

Algorithm	Image			
	Coins with noise		Pepper with noise	
	Mean MSE	Mean PSNR	Mean MSE	Mean PSNR
VG-edge detector	10301.53	8.0368	8080.38	9.1074
Canny color	10355.95	8.0164	5097.40	11.7303
Color average of channels	10361.32	8.0142	5107.20	11.7207
Luminance of channels	10356.19	8.0163	5097.98	11.7303
Hue edge image	5316.68**	10.9149**	3670.70**	12.8480**
Saturation edge image	10360.99	8.0143	5108.13	11.7293
Value edge image	10359.88	8.0148	5101.61	11.7255
Color CED combine	10356.34*	8.0163*	5098.73*	11.7293*
Color CED maximum	10356.20*	8.0163*	5097.27*	11.7313*

References

[1] D. J. Bora, "A Novel Approach for Color Image Edge Detection Using Multidirectional Sobel Filter on HSV Color Space," *International Journal of Computer Sciences and Engineering*, pp. 154-159, 2017.

[2] M. Hasanzadeh Mofrad, S. Sadeghi, A. Rezvanian and M. R. Meybodi, "Cellular edge detection: Combining cellular automata and cellular learning automata," *International Journal of Electronics and Communications (AEÜ)*, p. 1282–1290, 2015.

[3] D. Ray, "Edge Detection in Digital Image Processing," 2013.

[4] J. B. Dibya and K. G. Anil, "A New Approach towards Clustering based Color Image Segmentation," *International Journal of Computer Applications*, pp. 23-30, 2014.

[5] E. Nadernejad, S. Sharifzadeh and H. Hassanpour, "Edge Detection Techniques: Evaluations and Comparisons," *Applied Mathematical Sciences*, pp. 1507 - 1520, 2008.

[6] S. Savant and R. Manza, "Color Image Edge Detection using Gradient Operator," *International Journal of Emerging Trends & Technology in Computer Science (IJETTCS)*, pp. 55-57, 2015.

[7] M. Ruzon and c. Tomasi, "Color Edge Detection with the Compass Operator," in *IEEE Conference on Computer Vision and Pattern Recognition*, Stanford, CA 94305, 1999.

[8] K. S. Narendra and M. A. L. Thathachar, "Learning Automata - A Survey," *IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS*, pp. 323-334, 1974.

[9] H. Beigy and M. R. Meybodi, "A Mathematical Framework for Cellular Learning Automata," *Advances in Complex Systems a Multidisciplinary Journal*, pp. 295-319, 2004.

[10] Umbaugh and E. Scott, Computer imaging: digital image analysis and processing, CRC press, 2005.