Original Research Article

Automatic Detection of Edges in Handmade Embroidery Patterns

Abstract

Aim / objectives: The study examined the specific characteristics responsible for the recognition of edges in handmade embroidery patterns, designed a computational model for the process, implemented the model and evaluated its performance. This is with the view to detecting the edges of handmade embroidery patterns in the context of computational modeling.

Study design: Computational modeling.

Place and Duration of Study: Department of Computer Science and Engineering and Department of Fine and Applied Art, between February 2016 and May 2017.

Methodology:Samples of hand embroidery patterns were collected through embroiderer shop in Ìbàdàn, Òsogbo, and Ilé-Ifẹ̀ and the collected samples were pre-processed and the edges of the patterns were detected using cellular automata (CA) and cellular learning automata (CLA). The performance of the system was evaluated in terms of computing time, Mean Square Error (MSE) and Peak Signal-to-Noise Ratio (PSNR).

Results:The result obtained from all the experiments carried out showed that the Cellular Edge Detection (CED) algorithm has lower value in terms of MSE, a higher value in terms of PSNR and lesser computational time as compared to the standard edge detection algorithm. The automatic detection of edges showed that the complex stitches of handmade embroidery patterns are amenable to computational rendering through efficient and effective techniques.

Conclusion: This study will enhance the performance of the edge detection techniques employed in pattern recognition and computer vision applications.

Keywords: Cellular Learning Automata; Cellular Automata; Handmade Embroidery Patterns; Edge detection; Image Processing; Pattern Recognition.

1 Introduction

The literature on the automatic detection of edges in regular and well-formed patterns is rich in solution methods. The question as to whether these solution methods are amenable to the automatic detection of edges in irregular patterns such as those exhibited by handmade embroideries on clothes remains an open research problem. This study investigated the development of a computational model to address the question. However, indigenous handmade embroidery exhibits irregular and inconsistent structured patterns due to the creativity involved in their production. The automatic detection of edges in such patterns present challenges to the existing methods. A number of methods have been applied to the problem of edge detection but all were based on regular and consistent structured patterns. Handmade embroidery is considered in this study because it is a traditional way of fabric embellishment which promotes creativity and allows imagination to run wild.

Embroidery is a classic form of art that requires great creativity and skill by the artist who use different geometric shapes and pattern. It can be mainly grouped into two categories depending on the creation technique used which are hand embroidery and machine embroidery, of which the latter can be sub-divided into computerized or digital embroidery. All kinds of decorative threads, materials, and stitches can be incorporated in hand embroidery, the range is enormous and the imaginations are wild. Edges are significant local intensity changes in the image and they are important features to analyse in an image, they serve as clues to separate regions within an object or to identify changes in illumination or colour and also are important feature in the early vision stages of the human eye [(Gonzalez, 2010)].

The process of edge detection is a basic tool in image processing, machine vision and specifically in the areas of feature detection and feature extraction [(Umbaugh, 2010)]. The purpose of the edge detection process is to reduce the mathematical expression of the image in order to minimise the amount of data to be processed so as to preserve some important information about the image. It is a technique that has many applications in computer vision algorithms with the desire to provide machines and robots with visual capabilities. Applying edge detection to hand embroidery patterns will enhance the performance of the techniques due to the irregularities nature of the pattern. The design of the automatic edge detection system for handmade embroidery patterns will identify and recognize edges of hand embroidery. Hand embroidery is a craft that has been practiced for centuries, however, since the emergence of machine embroidery, its use has reduced a bit because people think it is faster, cost-effective and more accurate, this however, underscores that the real essence and beauty of embroidery comes out when it is done by hand [(Arsalan, 2011)].

Hand Embroidery is a craft that has a lot of complexities involved in it as it requires a lot of time labour and patience, the amount of time it takes to create one hand embroidered sample, a number of samples of the same design could be produced by a machine in the same time span [(Couture, 2017)]. If the edges of hand embroidery patterns are precisely identified, the whole pattern can be recognized automatically. A number of methods have been applied to the problem of edge detection varying from traditional methods which include Sobel, Prewitt, Robert, Canny, Hilbert Transform and Slow Response Hilbert Transform to Cellular Automata. In image processing, an edge is a group of the pixels which has the gray value of step or roof changes, and it also refers the area of an image where the brightness changes significantly [(Gao, 2010)]. In typical images, edges characterize object boundaries and are therefore useful for segmentation, registration, and identification of objects in a scene [(Neelam, 2015)]. The application of Gaussian filter [Lakhania (2016)] provides the identification of detective edge when Sobel and Prewitt edge detector was employed. The efficiency of the first and second derivatives, Hilbert Transform, Slow Response Hilbert Transform algorithm was evaluated [Suman (2014)] in which SRHLT showed robustness to noise. The study by [Qiao (2012)] used Canny edge detector to get the edge information of the Region of Interest (ROI). The glass texture analysis was carried out with some edge detection algorithm (Sobel, Prewitt, Robert, Canny, and LoG) [Alkhateeb (2015)]. The performed controlled experiments reveal that LoG edge detection algorithm is better than others in determining texture analysis.

Cellular automata (CA) are discrete dynamical systems [(Kari, 1990)] with simple mathematical constructs and distinguished features and they have been used for simulating various complex systems in the real world [(Toffoli, 2003; Werfel, 2000)] and for modeling advanced computation such as massively parallel computers and evolutionary computation [(Sharma, 2013)]. Two-dimensional CA with Moore neighbourhood and Totalistic rule [Tamanaha (2003); Sharma (2013); Moham4med (2014)] has been used by the most author. CA Game of Life (GoL) of the two-dimensional square lattice which uses Moore neighbourhood was studied [Peer (2012)] with radius equals 1. Periodic boundary condition on all sides was used to avoid boundary problem that may lead to complexities of rule generation.

The rule generation problem for the CA-based image processing and computer vision system have been addressed by a number of literatures. Most of the authors have used the evolutionary algorithms, e.g. in [Slatnia (2013)], this approach was applied for the identification of CA that perform edge detection operations so as to reduce the number of rule generation. A deterministic Sequential Floating Forward Search (SFFS) method was used to select rules of CA [Rosin (2006)]. Another approach is based on parameter estimation methods from the field of system identification [(Werfel, 2000)]. A framework for solving the identification task for both deterministic and probabilistic CA was presented in [Sun (2011)]. The fuzzy logic theory has been combined with CA [Sahin (2014); Patel (2015)] for better performance of the detected edges and noise filtering.

In a related work, a linear CA rule with the fuzzy heuristic membership function [Uguz (2014)] optimised with Particle Swarm Optimisation (PSO) was proposed and the result gives a better performance for some linear rules. The method detects texture of images clearly without distortion and after applying PSO, the result was much better when compared with the result of Canny and Sobel. The combination of CA and CLA for adaptivity of the neighbourhood type [Mofrad (2015)] enhances the performance of CA for edge detection. The rule space of the CA was computationally complex because no evolutionary algorithm was used to select the best rule for edge detection.

The existing machine embroidery is yet to produce complex stitches of hand embroidery, however, the edges of hand embroidery patterns can be represented using edge detection techniques thereby producing a system that can reduce the complexities involved in hand embroidery designs. The approach considered in this research is different from the above-reviewed work because it investigated the behaviour of artistic work (embroidery) that is yet to be addressed by any of the existing approaches in edge detection. In this study, the irregularities exhibited by handmade embroidery patterns were investigated using Cellular Automata (CA) and Cellular Learning Automata (CLA) as edge detection techniques. This is with the view to detecting the edges of complex stitches of handmade embroidery patterns which has the properties different from the patterns considered in the reviewed literatures.

The rest of this paper is organised as follows. Sections 2 describes the materials and methods used to formulate the work. Model simulations were discussed in Sections 3. Results and discussion were analysed in Section 4. Section 5 concludes the paper.

1.1 Cellular Automata

In this section, two-dimensional Cellular automata (CA) are presented as a mathematical model for image processing and feature detection in which each cell of CA corresponds to each pixel of an image. The 2 Dimensional CA S^2 made up of large numbers of similar components with local interactions and this same components act together to produce a new complex global behaviour. Each component can assume a state from the set finite of states. The components then update their states simultaneously on a discrete time step according to a simple update rule. The new state of each component depends on the previous states of a set of cells, including the central cell itself and its surrounding neighbourhood [(Kari, 1990)]. A configuration described the state of all the cells in the cell structure. The evolution of the CA specified by the CA rule and the initial configuration indicates how each configuration is changed in one-time step.

A 2-dimensional CA is a quadruple of $\langle S^2, \phi, N, F \rangle$ where S^2 is a network of 2-tuple ordered

i-1, j-1	I, j-1	I+1, j-1
i-1, j	I, j	I+1, j
i-1, j+1	I,j+1	I+1, j+1

(a) Moore Neighbourhood

	I, j-1	
i-1, j	I, j	I+1, j
	I, j+1	

(b) Von Newmann Neighbourhood

Figure 1: Neighbourhood Type

integers that could be a finite, semi-finite or infinite set, $\phi = 1, ..., m$ is a finite set of actions, $N = \langle \bar{x_1}, ..., \bar{x_m} \rangle$, $\bar{x_i} \in S^2$ is the neighbourhood vector that is a finite subset of S^2 and $F : \phi^m \longrightarrow \phi$ is the local rule of CA. The neighbourhood vector $N(u)$ defines the relative position of neighbours for each cell u in the cell network. $N(u)$ is calculated by Equation 1.1 in which the neighbour cells meet the two constraints of Equation 1.2. The two neighbourhood used in this study are Moore and Von Newmann as shown in Figure 1.

$$N(u) = \{u + \bar{x_i} \mid i = 1, ...\bar{m}\} \tag{1.1}$$

$$\begin{cases} \forall \quad u \in S^D \Longrightarrow u \in N(u) \\ \forall \quad u, v \in S^D \Longrightarrow u \in N(v) \wedge v \in N(u) \end{cases} \tag{1.2}$$

The local rule F used in this study is characterised as totalistic, outer totalistic and general rules:

1. In general rule, the next value of a cell depends on the value of its neighbouring cells in the current time step.
2. In totalistic rule, the next value of a cell depends on the various states of its neighbouring cells.
3. In outer totalistic rule, the next value of a cell depends on its current state and the various states of its neighbouring cells.

Given a configuration C of the cells in the CA at a certain time t, the configuration C' at time $t + 1$ for each cell c is calculated using Equation 1.3.

$$C'(c) = F(C(c_1...c_n)) \tag{1.3}$$

For this 2D CA considered in this research, the specific neighbourhoods used is defined. Von Neumann neighbourhood (4- neighbourhood) and Moore neighbourhood (8- neighborhood) for a cell $c_{i,j}$ is defined as: $c_{i-1,j}, c_{i,j-1}, c_{i,j+1}, c_{i+1,j}$ and $c_{i-1,j}, c_{i,j-1}, c_{i,j+1}, c_{i+1,j}, c_{i-1,j+1}, c_{i+1,j-1}, c_{i+1,j+1}, c_{i-1,j-1}$ respectively. The state of the core cell $c_{i,j}$ at time $t+1$ depends on the states of itself and the cells in the neighbourhood at time t . The Von Neumann and Moore neighbourhood transition functions are given through Equation 1.4 and 1.5 respectively.

$$c_{i,j}(t+1) = \delta(c_{i,j}(t), c_{i,j-1}(t), c_{i,j+1}(t)), c_{i-1,j}(t), c_{i+1,j}(t) \tag{1.4}$$

$$c_{i,j}(t+1) = \delta(c_{i,j}(t), c_{i,j+1}(t), c_{i+1,j+1}(t)), c_{i+1,j}(t), c_{i+1,j-1}(t), c_{i,j-1}(t), c_{i-1,j-1}(t), c_{i-1,j}(t), c_{i-1,j+1}(t) \tag{1.5}$$

1.2 Learning Automaton

A learning automaton used in this study performs a number of finite action in which each action selected was examined in a probabilistic environment. The response obtained from the examination process was applied to a learning automaton with either positive or negative reinforcement signal. The next action taken by the learning automaton depends on the kind of signals obtained during the evaluation process. In this process, the main goal is to learn the best action among all the available actions. The best action is the one that has a high chance of getting a positive signal from the environment. Figure 2 depicts the learning automaton interactions toward the environment. The environment is modelled as a triplet of $E = \langle \alpha, \beta, c \rangle$ where $\alpha = \alpha_1, \alpha_2, ..., \alpha_r$ is the input set, $\beta = \beta_1, \beta_2, ..., \beta_m$ is the output set and $c = \langle c_1, c_2, ..., c_r \rangle$ is the set of penalty probabilities where c_i is the probability that α_i gets the penalty signal. In static environments, the c_i values remain unchanged while in non-static environments these values change during the progress of learning automaton.

2 Materials and Methods

The materials and methods used for development of edge detection model are discussed in this section.

2.1 Cellular Edge Detection Model

This study adopted the structured chain of CA and CLA for edge detection of images. The central cell $C_{i,j}$ in the two neighbourhood types used and its surrounding cells represent an image in a way that each cell corresponds to the pixel of an image. The CLA was used to measure the degree of intensity of each pixel and also to determine the type of neighbourhood that corresponds to the embroidery image. The CED algorithm composes of three key components including (1) CA that store the boundary information of edges; (2) CLA that learn the shape, texture, size and boundary distribution of edges (3) CLA that act as an optimisation algorithm. The combination of these three components allows the CED algorithm to accurately detect edge information from images. Figure 3 depicts the simplified representation of the Cellular Edge Detection model.

Using the window pattern of neighbourhood type, the square-like, 9-cell Moore neighbourhood was used to detect the horizontal edges while the plus-like, 5-cell Von Neumann neighbourhood fits most for vertical edges. In CED algorithm, a CLA is responsible for selecting the neighbourhood type of CA while the gradient of image intensity may vary in different parts of the image. After selecting the neighbourhood type of each cell, the CA rules was deployed on the input image and consequently, the output image was formed. Finally, the reinforcement signal was calculated and deployed in a way that members of LA of the same neighbourhood will gradually learn a unified local neighbourhood type. The framework and formulation of CED algorithm are presented as follows:

A. Cellular Automata Framework

A 2D-dimensional CA is a 4-tuple defined as:

(a) CA = $\langle S^2, \phi, N, F \rangle$

(b) S^2 - 2D CA identical to input image $I_{m \times n}$

(c) ϕ = $\langle 0....255 \rangle$ is the set of finite states for grayscale image with 256 intensity levels and ϕ = $\langle 0, 1 \rangle$ is the set of finite states for binary image with 2 intensity levels.

(d) N - $(Moore_{3\times3}, VonNewmann_{3\times3})$ is the set of available neighbourhood types for CA.

(e) F - Local rule of CA which is calculated through Equation 2.1 to 2.3, respectively. F acts as the relationship criterion of central cells and its neighbours.

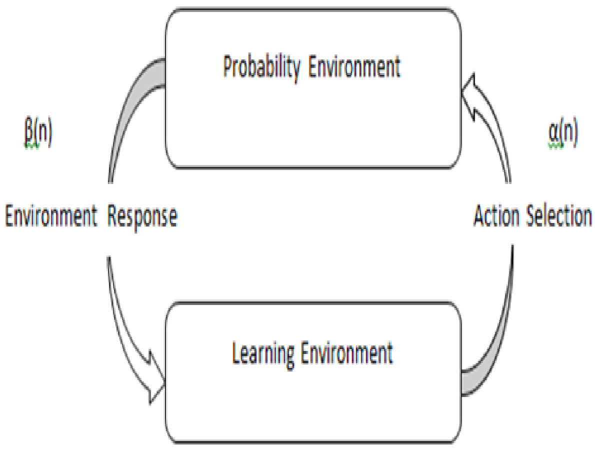


Figure 2: The Interaction Sequence between Learning Automaton and Environment

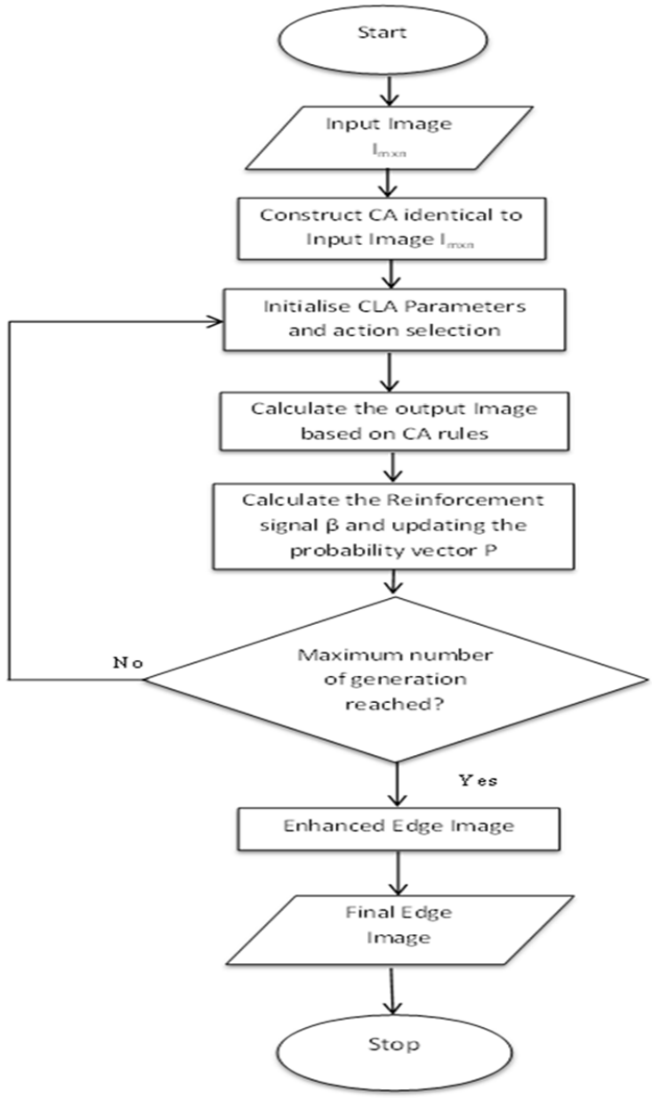


Figure 3: Cellular Edge Detection Algorithm

B. Cellular Learning Automata Framework

A 2-Dimensional CLA is quintuple defined as:

1. CA = $\langle S^2, \phi, A, N, F \rangle$
2. S^2 - 2D CA identical to input image $I_{m \times n}$
3. ϕ = $\langle 0....255 \rangle$ is the set of finite states.
4. A - is the set of LA identical to the input image $I_{m \times n}$
5. N - $(Moore_{3\times3}, VonNewmann_{3\times3})$ is the set of available neighbourhood types for CA
6. F - is the local rule of CLA which is define by Equation 2.4:

2.2 Binary Edge Detection Rule

The input image I consists of a set of pixels which values are zero or one. If the pixel value of I is one $(I = 1)$, the corresponding cell will evaluate with the following constraints through Equation 2.1:

1. If at least one of the cells surrounding the central cell equals to zero, the corresponding pixel will identify as an edge in the output image $J(J = 1)$.
2. If all of the cells surrounding the central cell equal to one, the corresponding pixel will not identify as an edge in the output image $J(J = 0)$.

Otherwise, if the pixel value of I is zero $(I = 0)$, it will evaluate through Equation 2.2. Based on Equation 2.2, if the central cell equals to zero, the corresponding pixel of output image J will set to zero $(J = 0)$.

$$J_{ij} = F(N|I_{ij} = 1) = \begin{cases} 1 & if\ \exists I \ \in \ N|I_N = 0 \\ 0 & if\ \forall I \ \in \ N|I_N = 1 \end{cases} \tag{2.1}$$

$$J_{ij} = F\ (N|I_{ij} = 0) \tag{2.2}$$

2.3 Grayscale Edge Detection Rule

In Equation 2.3, the local rule is derived from the states of the central cell and its neighbouring cells. If the total difference between the central pixel intensity and the maximum pixel intensity of neighbourhood violates the threshold value θ, then the central cell will be considered as edge $(J = 1)$, and otherwise, the central cell will not be considered as edge $(J = 0)$.

$$J_{ij} = F(N) = \begin{cases} 1 & if\ |I_{ij} - \max(N)| \geq \theta \\ 0 & otherwise \end{cases} \tag{2.3}$$

2.4 Learning Automata Rule

In Equation 2.4 the reinforcement signal of each learning automaton through local interactions of LA was calculated. If all LA in the same neighbourhood choose the same action, they will be rewarded, and otherwise, they will be punished. This technique allows LA to choose similar neighbourhood type of independent images. The probability vector of LA used is given in Equation 2.5 and synchronically all LA of CLA select action based on the probability vector. The reinforcement signal of each LA was calculated based on the local interactions of LA among the cells of cellular automata represented in Equation 2.6. When an action gets a reward, its probability P_i increases, while the probability of all other actions decreases.

Based on the action probability set P, automaton randomly selects an action $\alpha_{i,j}$, and performs it on the environment. After receiving the environment's reinforcement signal $\beta_{i,j}$, automaton updates its action probability set based on the following reinforcement scheme; the Equations 2.7 for positive response, and Equations 2.8 for negative response.

$$\beta_{ij} = F(N) = \begin{cases} 1 & if|A_{ij} = A(N) \forall A \in N| \\ \\ 0 & otherwise \end{cases} \tag{2.4}$$

$$P(r) = \frac{1}{|r|}, r \in \{1, 2\} \tag{2.5}$$

where $|r|$ is the number of actions, $r = 1$ is the probability of selecting Moore neighbourhood and $r = 2$ is the probability of selecting Von Newmann neighbourhood.

$$A_{i,j} = CLA((i-1, j-1), (i-1, j)(i-1, j+1), (i, j-1), (i, j), (i, j+1)(i+1, j-1), (i+1, j), (i+1, j+1)) \tag{2.6}$$

If β = 0

$$\begin{cases} p_i(n+1) = p_i(n) + a[1 - p_i(n)] \\ p_j(n+1) = (1-a)p_j(n) \forall j \neq i \end{cases} \tag{2.7}$$

If β = 1

$$\begin{cases} p_i(n+1) = (1-b)p_i(n) \\ p_j(n+1) = (\frac{b}{b-1}) + (1-b)p_j(n) \quad \forall j \neq i \end{cases} \tag{2.8}$$

3 Model Simulation

The selected standard edge detection methods such as Prewitt, Robert, Sobel, Canny, and the CED method were simulated in MATLAB R2013a environment. Samples of handmade embroidery data (rescaled to size 512 by 512) were processed on an Intel(R) 1.90 GHz and 4GB RAM machine using the model developed in this study. To evaluate the effectiveness of the method to detect edges of embroidery pattern used in this study, an experiment was made in which random samples of pattern images were presented to the model as input. The detection efficiency of the model using performance metrics was recorded. The performance metrics used are Peak Signal-to-Noise Ratio (PSNR) and Mean Square Error (MSE). The general settings of CED algorithm used are as follows:

1. The maximum number of generations was set to 8.
2. The learning algorithm is L_{RP} with $\alpha = \beta = 0.5$ as reward and penalty parameters respectively.
3. The threshold value of grayscale images θ was selected from the range of $[15, 30]$.

MSE calculates the average difference between original and edge detected image and the lower MSE indicates lesser error between the original image and edge detected one. For original image I and edge detected image J, MSE is calculated through Equation 3.1.

$$MSE = \frac{1}{m \times n} \sum_{j=0}^{m} \sum_{i=0}^{n} (I_{i,j} - J_{i,j})^2 \tag{3.1}$$

where in Equation 3.1, m and n are height and width of the images respectively. The PSNR is calculated based on MSE through Equation 3.2.

$$PSNR = 10 \log(\frac{P^2}{MSE}) \tag{3.2}$$

where in 3.2, $P = 255$ is the maximum variation for an 8-bit grayscale input image and MSE is calculated by Equation 3.1.

3.1 Edge Detection of Grayscale Images

The experiments were carried out over the selected 512×512 size of handmade embroidery pattern images. The images with linear and geometric; low and high intensity; plain and not plain fabric were selected for the simulation. Figures 4 and 5 show the original image and edge detected result of *Dídà Oníbọ̀* and *Alábẹ Méta* grayscale image for different algorithms. The test image has a linear and geometric shape with a number of vertical and horizontal lines. Figures 4(b)-(f) shows the result of different algorithms in which the output of Roberts misses out almost all the important edges of the pattern, both the thick and the inner edges were missed out. The output of Sobel and Prewitt algorithm is almost the same, but Sobel presented a brighter edge. Prewitt and Sobel lose a portion of all edges especially the inner edges of the pattern. Canny's output preserves more information about the inner edges with more contours in the background of the image due to its sensitivity to noise with some traces of false edges. This algorithm not only could preserve the inner edges, but it also detects both horizontal and vertical edges as it was presented in the original image. Figure 5(a)-(f) shows the original test image and detected edges of different edge detection algorithms. The embroidery pattern is geometric with a number of triangle and rectangle shapes. The output of Prewitt and Sobel are extremely pale and plain, they also missed most of the important edges in the rectangle on top of the three triangles and also all the edges in the triangles were missed out. This shows that Sobel and Prewitt could not handle edges with low intensity. The output of Roberts presents brighter edges as compared to Sobel and Prewitt. Canny's output is also pale but the edges are brighter than the result produced by Prewitt and Sobel. It detects more edges with no real connectivity especially the inner edges of the triangle and rectangle shapes. False edges are found to be detected all over the background, a small edge detected just above the right angle triangle at the left side was not found in the original test image. Figure 5(f) shows the result produced by the CED algorithm. The CED algorithm generated connected, clean and meaningful edges with more details and clarity. It responds well to low-intensity pixels as most of the inner edges were well represented.

3.2 Edge Detection of Binary Images

The algorithm was tested over a selected number of handmade embroidery pattern images. The image was resized to 512×512 and converted to the binary image prior to the execution of the edge detection process. The binary images were obtained using Otsu's method. Figure 6(a) to 6(e) show the original test image and output after applying Sobel, Prewitt, Canny and CED algorithm respectively. Prewitt and Sobel output are weak and not clear. Canny's output is much clearer than Prewitt and Sobel but Sobel and Prewitt were able to detect the low-intensity pixel at the lower part of the left arm better than Canny. The CED algorithm produced much better and clearer edges.

4 Results and Discussion

The MSE and PSNR of handmade embroidery images of selected pattern are reported for Prewitt, Roberts, Sobel, Canny and CED algorithms in Table 1 and 2. From Table 2, it was revealed that the Prewitt and Sobel method generated the same numerical value for MSE. Prewitt and Sobel value for PSNR are the same but Robert generated lesser value. Canny's output produced less error and a higher ratio of signal (original image) and noise (edge map) while comparing to Prewitt and Sobel algorithm. The CED algorithm had a lesser error and much better higher value in PSNR. Table 2 shows the performance evaluation result of *Dídà Olójúmẹ́ta* binary image. In the table, Robert and Sobel had a lesser error and a higher PSNR as compared to Prewitt and Canny. The CED method

Table 1: Performance Evaluation of *Dídà Oníbọ̀* and *Dídà Alábẹ Mẹ́ta* Grayscale Image using MSE and PSNR

Algorithm	Image			
	Dídà Oníbọ̀		*Dídà Alábẹ Mẹ́ta*	
	MSE	PSNR (dB)	MSE	PSNR (dB)
Robert	2562	4.0443	2321	4.4744
Prewitt	2562	4.0445	2321	4.4744
Sobel	2562	4.0445	2321	4.4744
Canny	2559	4.0510	2320	4.4766
CED	2539	4.0841	2301	4.5113

Table 2: Performance Evaluation of *Dídà Olójúmẹ́ta* Binary Image using MSE and PSNR

Algorithm	Image	
	Dídà Olójúmẹ́ta	
	MSE	PSNR (dB)
Robert	0.8243	48.4010
Prewitt	0.8571	48.8005
Sobel	0.8553	48.8097
Canny	0.8583	48.7943
CED	0.7715	49.2573

had a better numerical value in terms of MSE and PSNR. In the edge detection process, a lower MSE indicates that the obtained edge map contains strong in addition with weak edge points. furthermore, a higher PSNR indicates a higher ratio of the original image to produced edge map. However, CED algorithm with lower MSE and higher PSNR is an efficient and reliable choice. The inverse relation of MSE and PSNR in Equations 3.1 and 3.2 proves this relationship. The result outputs of all the images considered were represented in a graphical form as shown in Figures 7, 8, 9 respectively. The computational time of the standard edge detection and Cellular Edge Detection method were compared as shown in Table 3. In all the experiments performed, the CED method takes a significant lesser time for computational process.

5 Conclusion and Future Work

The standard edge detection methods such as Sobel, Prewitt look for big absolute value in the first derivative of an image and Canny which is considered as the optimal edge detection look for sign changes (Zero crossing) in the second derivative of an image. In this study, a cellular edge detection algorithm was proposed and it worked based on parallel cells of CA combined with CLA which speed up the edge detection process. CA is a discrete dynamic system which treated each pixel of an image with a simple update rule. This behaviour enables CA to efficiently detect edges of HEP than the standard edge detection algorithm which only looks for the maximum intensity changes and zero crossing in the gradient of an image. From this prototype, an automatic edge detection of HEP was developed. In all the experiments carried out, the CED algorithm accurately detects

Table 3: Computational Time of Various Edge Detection Methods

Computing Time (s)						
Test Image (s)	Image Size	Edge Detection Algorithms				
		Robert	Prewitt	Sobel	Canny	CED
Dídà Oníbọ̀	512 $\times$512	0.2145	0.2143	0.2156	0.2345	0.1845
Dídà Olójúmẹ́ta	512 $\times$512	0.2245	0.2346	0.2348	0.2543	0.1902
Dídà Alábẹ Mẹ́ta	512 $\times$512	0.2175	0.2467	0.2468	0.2623	0.1279

edges and does not extract unnecessary details as the edges are clearly visible. Furthermore, the conclusion can be made that the cellular edge detection algorithm turns out to be the most efficient among the standard edge detection considered as it is more accurate and effective because it has superior numerical value in terms of MSE and PSNR performance metrics. This will, however, make the system developed when incorporated into hardware be very useful as a model for art institutions, local embroidery factories, tailors and other such industries where embroidery is constantly in use and can also enhance their entrepreneurial skills and business. In future, the model developed in this study can also be incorporated into the hardware as its vision system to produce patterns of hand embroidery which can be used in place of human labour to address the problem associated with the manual process. Also, the designing of embroidery classification model will be considered as a future work.

References

Akdemir, B. and Ozturk. S. (2015). Comparison of Edge Detection Algorithms for Texture Analysis on Glass Production. Social and Behavioural Sciences, 195(2015), 2675–2682.

Arsalan, S. (2011). Is Machine Embroidery a Step Forward or Downfall? Department of Texitile Design,*Unpublished B. A. thesis.*, Indus Valley School of Art, Karachi, Pakistan.

Couture P. H. (2017).Difference between Computerised and Hand Embroidery. *(Last accessed on 4 June 2017 at 18:02)*.

Gao, W., Yang, L., Xiaoguang, Z. and Hulzhong, L. (2010). An Improved Sobel Edge Detection. Computing Control and Industrial Engineering 9 (97): 67-71.

Gonzalez, R. and Woods, R. E. (1993). Digital Image Processing. 5^th Edition, Addison Wesley.

Kari, J. (1990). Reversibility of 2D Cellular Automata is Undecidable. Physical Discrete 453: 379-385.

Lakhania, K., Minochaa, B. and Gugnani, N. (2016). Analyzing Edge Detection Techniques for Feature Extraction in Dental Radiographs. Perspectives in Science 86: 395-398.

Mofrad, M. H., Sadeghi, S., Rezeanian, A. and Meybodi, R. (2015). Cellular Edge Detection: Combining Cellular Automata and Cellular Learning Automata. International Journal of Electronics and Communication 699: 1282-1290.

Mohammed, J. and Nayak, D. R. (2014). An Efficient Edge Detection Techniques by two Dimensional Rectangular Cellular Automata. Computing Research Repositories, IEEE abs/1312.6370 1-4.

Neelam, T. (2016). A Review of literature Review on Edge Detection Techniques. International Journal of Electronics Networks, Devices and Fields 81: 1-6.

Patel, D. K. (2013). Edge Detection Techniques by Fuzzy Logic and Cellular Learning Automata using Fuzzy Image Processing. IEEE International Conference on Computer Communication and Informatics 1-6.

Peer, M. A., Fasel, Q. and Khan, K. A. (2012). Investigation of Cellular Automata Game of Life Rule for Noise Reduction and Edge Detection. International Journal of Information Engineering and Electronics Business 2: 22-28.

Qiao, Y. and Shi, Z. (2012). Traffic Parameter detection using Edge and Texture. Precedia Engineering 292: 3858-3862.

Rosin, P. (2006). Training Cellular Automata for Image Processing. IEEE Transaction on Image Processing 157: 2076-2087.

Sahin, U., Uguz, S. and Sahin, F. (2014). Salt and Pepper Noise Filtering with Fuzzy Cellular Automata. Computer and Electrical Engineering Journal 401: 59-69.

Sharma, P., Diwakar M. and Lal, N. (2015). Edge Detection using Moore Neighbourhood. International Journal of Computer Applications 613: 26-30.

Sipper, M. (1997). Evolution of Parallel Machines: the Cellular Programming Approach. In: applications of Fuzzy Sets Theory, Springer, Berlin Heidelberg 404-411.

Slatnia, S., Botouche M. and Melkemi, K. E. (2007). Evolutionary Cellular Automata based Approach for Edge Detection. Lecture Notes in Computer Science, Springer-verlag.

Suman and Pawan (2015). A Survey on various Methods of Edge Detection. International Journal of Advanced Research in Computer Science and Software Engineering 45: 888-895.

Sun, X., Rosin, P. L. and Martin, R. R. (2011). Fast Rule identification and Neighbourhood Selection for Cellular Automata. IEEE Transactions on Systems, Man and Cybernetics 413: 749-760.

Tamanaha, G. H. (2003). An Original Method of Edge Detection based on Cellular Automata. Tech. Rep. Department of Electrical Engineering and Computer Science, Korea Advanced Institute of Science and Technology.

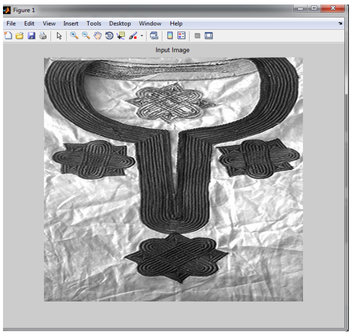
Toffoli, T. and Margolous, W. (1987). Cellular Automata Machines. MIT Press.

Uguz, S., Sahin, U. and Sahin, F. (2015). Edge Detection with Fuzzy Cellular Automata Transition Function Optimized by PSO. Computer and Electrical Engineering Journal 43C: 180-192.

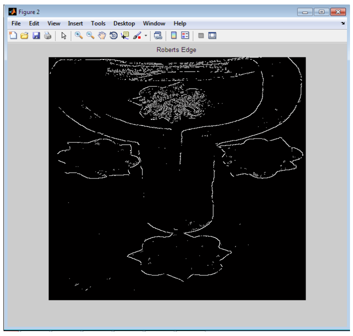
Umbaugh, S. (2010). Digital Image Processing and Analysis. Human and Computer Vision with CVIP Tools, FL: CRC Press.

Werfel, J., Mitchell, M. and Crutchfield, J. (2000). Resource Sharing and Co-evolution in Evolving Cellular Automata. IEEE Transactions on Evolutionary Computation 4 388-393.

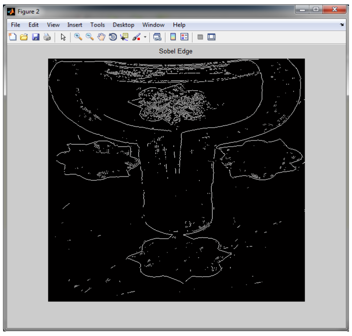
Zhao, Y. and Billings, S. (2007). The Identification of Cellular Automata. Journal of Cellular Automata 2 1:47-65.



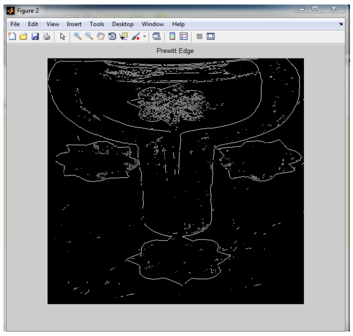
(a)



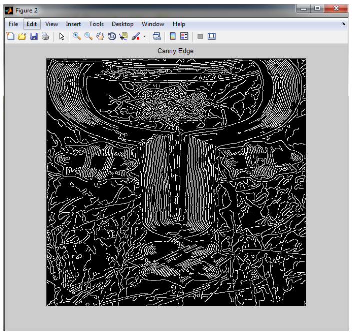
(b)



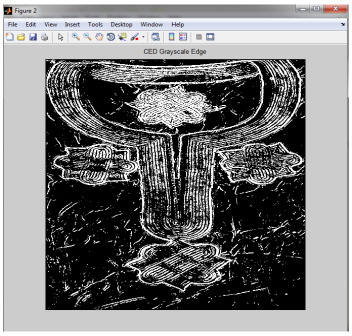
(c)



(d)

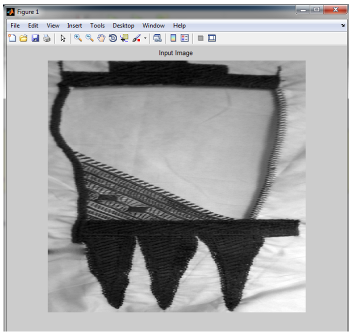


(e)

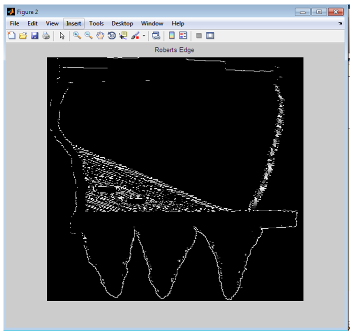


(f)

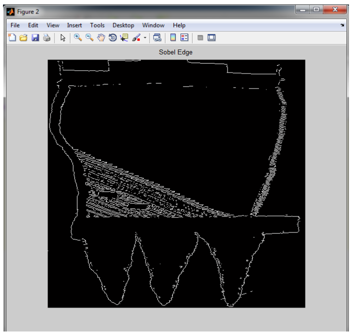
Figure 4: (a)*Dídà Oníbọ̀* Test Image (grayscale) (b) Roberts Algorithm (c) Sobel Algorithm (d) Prewitt Algorithm (e) Canny Algorithm (f) CED Algorithm



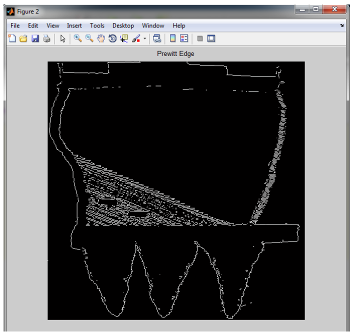
(a)



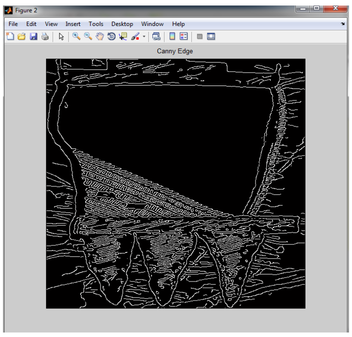
(b)



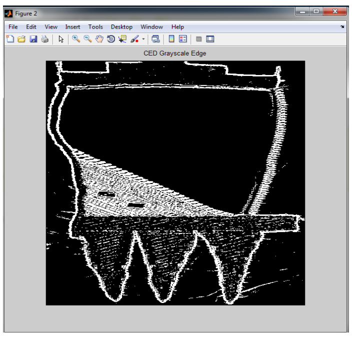
(c)



(d)



(e)



(f)

Figure 5: (a)*Alábẹ Méta* Test image (grayscale) (b) Roberts Algorithm (c) Sobel Algorithm(d) Prewitt Algorithm (e) Canny Algorithm (f) CED Algorithm

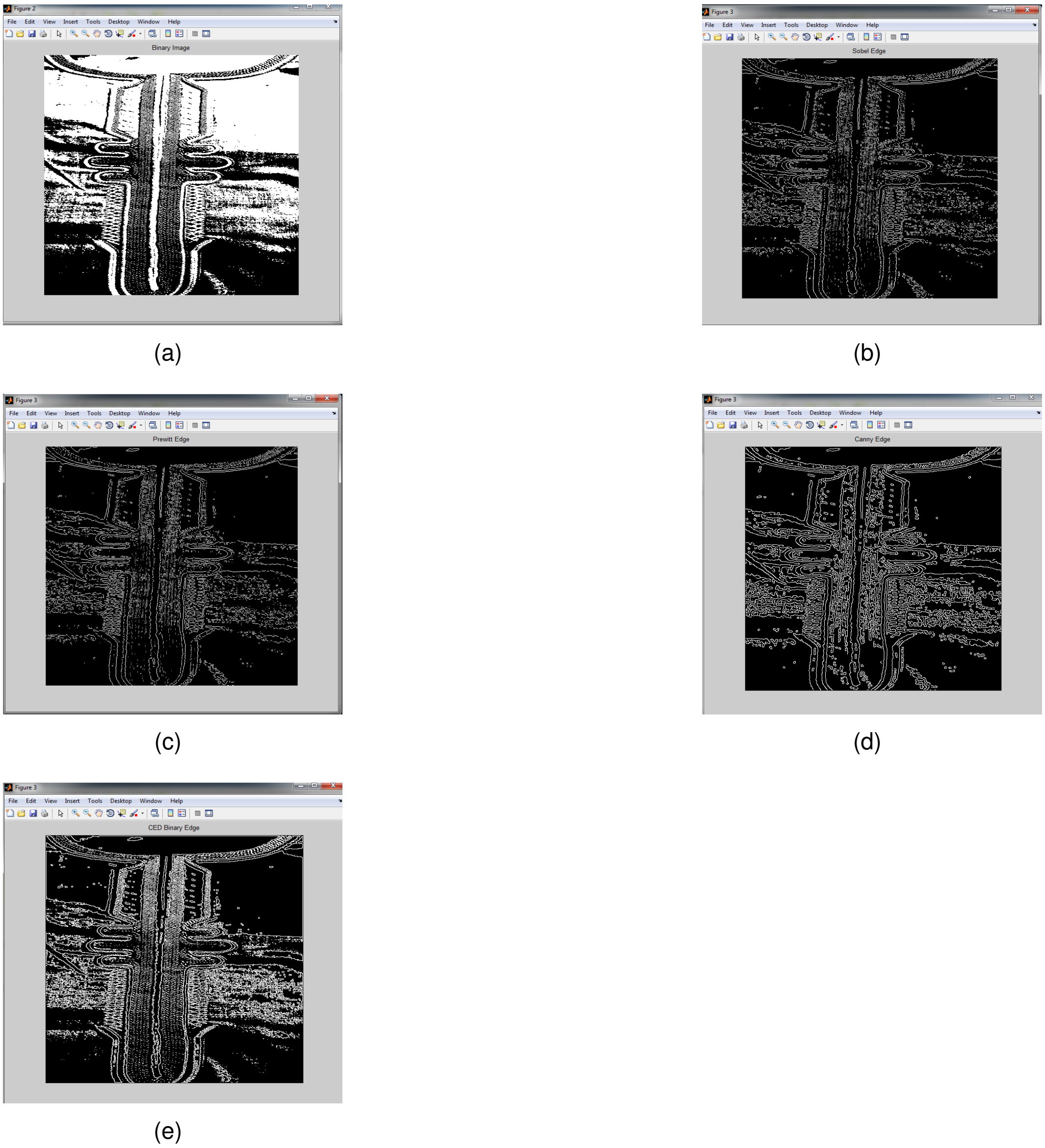
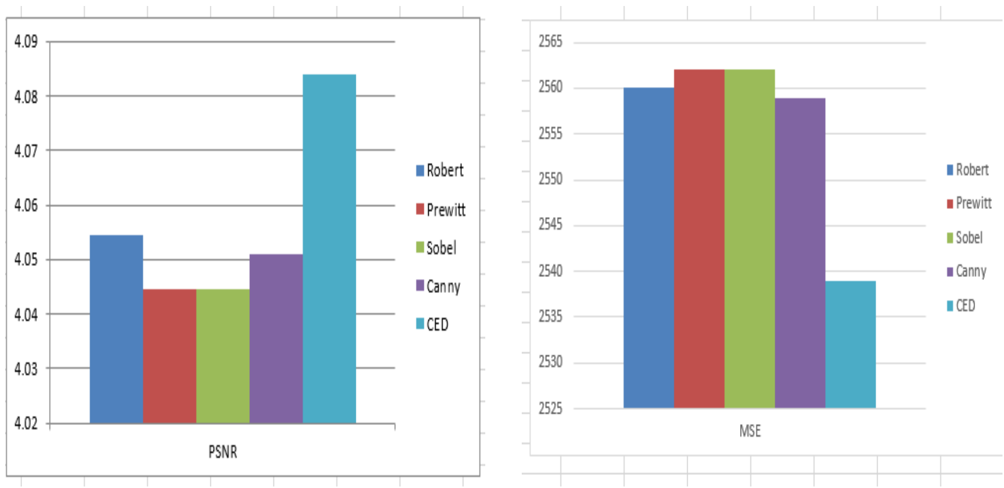
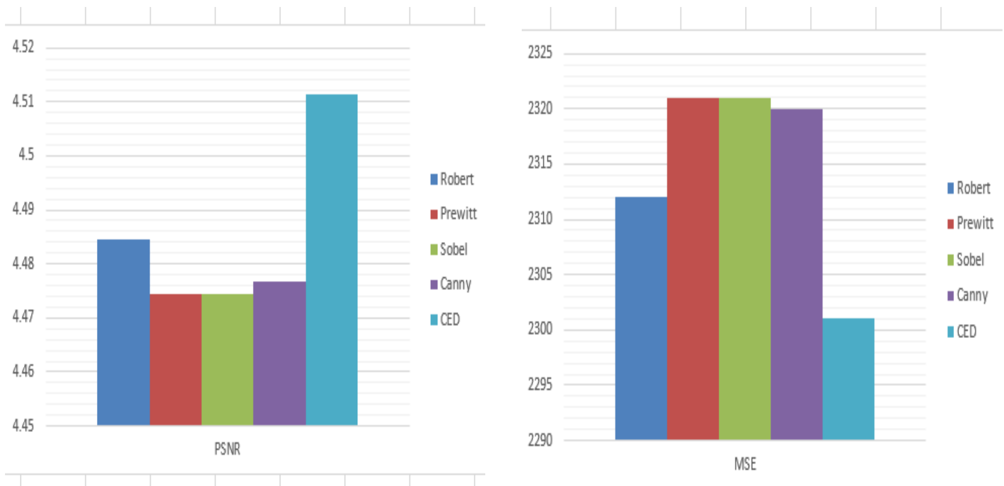


Figure 6: (a) *Dídà Olójúmẹ́ta* Test Image (Binary) (b) Sobel Algorithm(c) Prewitt Algorithm (d) Canny Algorithm (e) CED Algorithm



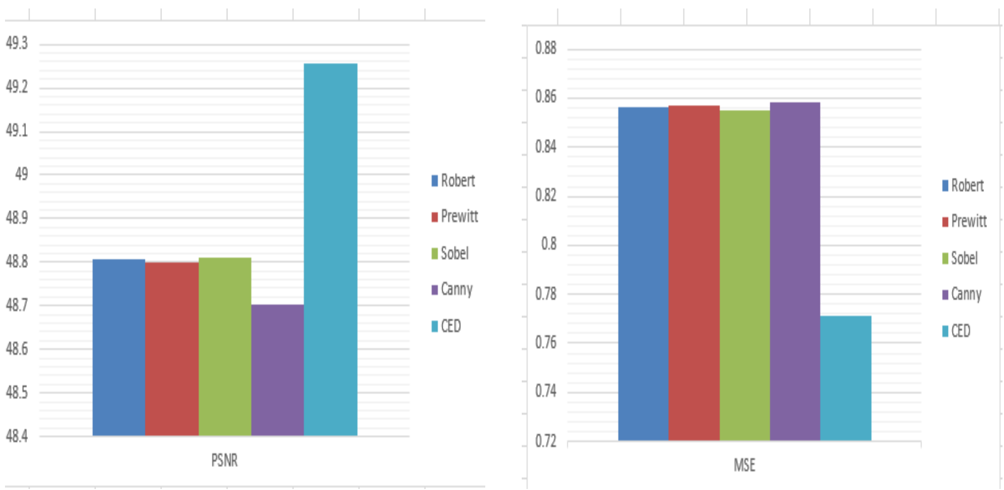
(a) (b)

Figure 7: Result Analysis of Dídà Oníbọ̀



(a) (b)

Figure 8: Result Analysis of Dídà Alábẹ Mẹ́ta



(a) (b)

Figure 9: Result Analysis of Dídà Olójúmẹ́ta