ISSN 1746-7659, England, UK
Journal of Information and Computing Science
Vol. 5, No. 1, 2010, *pp*. 003-010

A Cellular Neural Network- Based Model for Edge Detection

Hezekiah Babatunde, Olusegun Folorunso[1] and Adio Akinwale

Department of Computer Science, University of Agriculture, Ogun State, Nigeria

(*Received September 21, 2009, accepted October 8, 2009*)

Abstract. This study employed the use of Cellular Neural Networks (CNN) for Edge Detection in images due to its high operational speed. The process of edge detection is unavoidable in many image processing tasks such as obstacle detection and satellite picture processing .The conventional edge detector models such as Sobel Operator, Robert Cross, among which Canny is the best, have high computational time. The CNN Model is a class of Differential Equation that has been known to have many application areas and high operational speed. The work investigated four parameters: resolutions, processing time, false alarm rate, and usability for performance evaluation. The CNN Model was modified by using hyperbolic tangent (tanh x) and Von Neumann Neighborhood. The modified CNN Model and enhanced Canny Model were implemented using MATLAB 7.0 running on Pentium III and 128 MB RAM Personal Computer. A series of images served as input for both Canny and modified CNN Model. With several images tested, the overall results indicated that the two models have similar resolutions with average computational time of 1.1078 seconds and 2.293 seconds for CNN-based and Enhanced Canny Model respectively. The hyperbolic entry a_{22} of the cloning template A made our work fully controllable since max(tanh x) = +1 and min(tanh x) = -1. A consideration of the set of digital images showed that edge maps which result from Canny Model have adjacent boundaries that tend to merge. The CNN model obviously produced the optimum edge map with edges that are one-pixel wide and unbroken. The false alarm rate of noise variance probability $\left(P_F\left(\sigma^2\right)\right)$ for which an edge detector can easily declare an edge pixel given that there is no edge, showed the value 0.8525 for CNN Model and 0.4595 for Canny Model. The CNN parameters can be adjusted and modeled to solve Partial Differential Equations and Maximum likelihood problems for edge detection while Canny Model cannot be easily adjusted for any other functionality. The CNN-based edge detector performed better than the popular canny operator in terms of the computational time required, usability, and false alarm rate.

Keywords: Cellular Neural Networks**,** Canny Model**,** Edge Detection, hyperbolic tangent, Von Neumann Neighborhood

1. Introduction

In image processing, one of the most effective procedures used in sharpening the images is to improve the contrast. The contrast is improved by increasing the difference across discontinuities of the image components in order to improve the differences. Edge detection algorithms are designed to detect and highlight these continuities. It was first developed for processing satellites pictures and later become well known and widely used in digital image processing [5]. Detecting the edges of an image reduces the amount of data and filters out useless information, while preserving the important structural properties in an image. The two methods of edge detection are **gradient** and **Laplacian** [4]. The gradient method detects the edges by looking for the maximum and minimum in the first derivative of the image while the **Laplacian method** searches for zero crossings in the second derivative of the image to find edges. The common examples of edge detection algorithms are: Sobel, Canny, Laplace, Robert Cross, Prewitt and Sussan Edge Detectors. Roberts Cross, Prewitt and Sobel operators are gradient based edge detectors and they all have kernel operators [2] that calculate the strength of the slope in directories which are orthogonal to each other, commonly **vertical** and **horizontal**. Later, the contributions of the different components of the slopes are combined to give the total value of the edge strength. The Prewitt operator [3] measures two components. The vertical edge components is calculated with kernel k_x and the horizontal edge component is calculated

[1] Corresponding author Tel: +234-8035640707
Email address : folorunsolusegun@yahoo.com

with kernel k_y. The sum $|K_x| + |K_y|$ gives an indication of the intensity of the gradient in the current pixel.

In the image processing literature, all of the methods dealing with the detection of discontinuities in an image are normally classified under the general edge detection where the traditional techniques are based on the computation of local derivative. These traditional filters used for edge detection have to be optimized for different kind of edges. In this work, our proposed approach exploits the use of CNN with hyperbolic cloning templates.

2. Background

An edge is defined as a pixel at which the image values undergo a sharp variation – pixels with large element while edge detection or extraction is the act or process of finding pixels that belong to the borders of the objects. The detection of edges is useful in the following areas: reduction of data dimension, preservation of content information, inspection for missing parts and measurement of critical part dimensions using gauging. Other areas include identification and verification of electronic user interface display, object detection and tracking, structure from motion and distance calculation.

CNN stands for Cellular Neural networks and was invented in 1988 by Chua and Yang [1] at the University of California regular (rectangular, hexagonal, etc) array of mainly identical dynamical systems called cells which satisfy two properties: most interactions are local within a finite radius and all states values are continuous valued signals.

Following the Chua-Yang definition [1]:

(1) A CNN is an N-dimensional regular array of elements (cells) as shown in figure 1.

(2) The cell grid can be for example a planar array with rectangular, triangular or hexagonal geometry, a 2-D or 3-D torus, a 3-D finite array or a 3-D sequence of 2-D arrays (layers)

(3) Cells are multiple input-single output processors that describe by one or just few **parametric functions**

(4) A cell is characterized by an internal state variable, but sometimes not directly observable from outside the cell itself;

(5) More than one connection network can be present, with different neighborhood sizes;

(6) A CNN dynamical system can operate both in continuous (CT-CNN) or discrete time ((DT-CNN);

(7) CNN data and parameters are typically continuous values;

(8) CNN operate typically with more than one interaction i.e. they are recurrent networks.

The circuit theory and modeling is illustrated in figure 2. In most cases a first order is used consisting of a linear capacitor, a linear resistor, a constant current sources and additional voltage-controlled sources (i.e. the interconnections from the neighboring cells). The **voltage** measured on the linear capacitor is the state of the base cell. Each cell has its own **input** and **output** (the latter is calculated through the output characteristics from the state value).

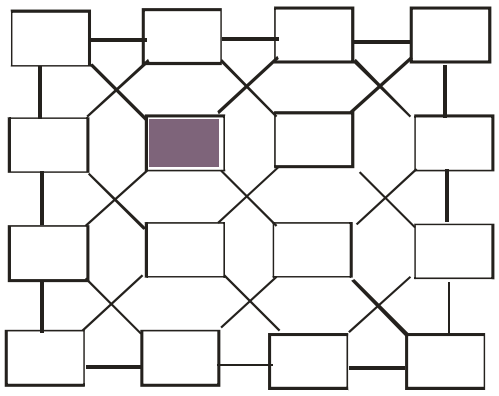

Fig 1: A 2-Dimesional CNN array on a square grid

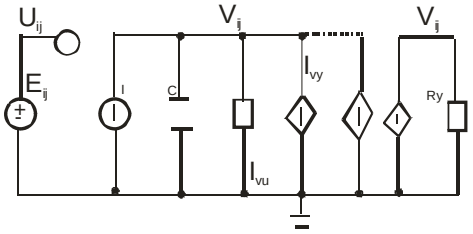

Fig. 2 : Typical Circuit of CNN ij-position.

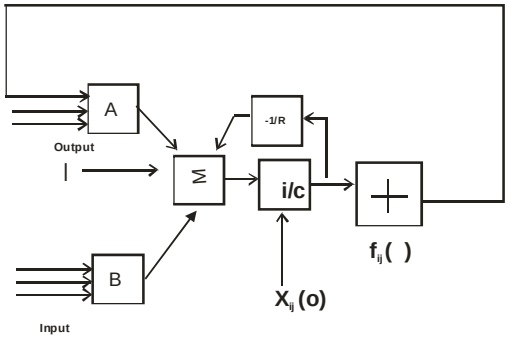

Fig 3 : Diagram of one continuous time cell.

In the original Chau and Yang model, each cell is a 1-D dynamical system and is the basic unit of CNN. Any cell is connected to its neighbor cells, i.e. adjacent cells interact directly with each other. Cells not in the immediate neighborhood have indirect effect because of the propagation effects of the dynamics in the network.

In an M x N CNN having MN cells arranged in M rows and N columns, the cell in position ij will be denoted by (ij) and its r-neighborhood $N^r ij$ is defined by $N^r ij = \{k_l : \max \}$

Where the size of the neighborhood r is a positive integer number. Each cell has a state **x**, a constant input **u** and an output **y**. The equivalent block diagram of a continuous time cell is shown in figure 3 above.

3. Enhanced Canny Edge Detection Algorithm

The Canny algorithm (known as Optimal Edge Detector) [4] first requires that the image be smoothed with a Gaussian mask, which cuts down significantly on the noise within the image. Then the image is run through the Sobel algorithm. This process is hardly affected by noise. Lastly, the pixel values are chosen base on the angle of the magnitude of that pixel and its neighbouring pixels.

In the format of algorithm Canny edge Detection is described as follows:

Given an Image I

- Apply algorithm of CANNY ENHANCER to I output I_E
- Denoise and gradient computation to enhance edges
- Apply algorithm of NONMAX-SUPPRESSION to I_E – Output I_m
- Find local maximum in I_E along edge normal "Suppress" all the other pixels.
- Apply algorithm of HYSTERESIS THRESH to I_m, Output I_H
- Remove edges pixels caused by noise

Algorithm of CANNY ENHANCER

- Apply Gaussian smoother to i, obtaining J = G * I
- For pixel (i, j)
- Compute gradient component Jx Jy
- Estimate the angles strength as

$$\ell_s(i, j) = \sqrt{J_x^{\,2}(i, j) + J_y^{\,2}(i, j)} \tag{1}$$

- Estimate the edge normal as

$$\ell_s(i, j) = \arctan\left(\frac{J_x}{J_y}\right)$$

- The output is the strength image $\ell_s(i, j)$ and an orientation image $\ell_0(i, j)$

Algorithm of NONMAX-SUPPRESSION

Quantize $\ell_0(i, j)$ into four directions 0^0, 45^0, 90^0, 135^0

For each pixel (i, j)

If $\ell_s(i, j)$ is smaller than at least one of its two neighbours along the normal direction set I_N (i, j) = 0 (Suppression)

Otherwise assign I_n (i, j) = $\ell_s(i, j)$

- The output is an image I_N (i, j)

Algorithm of HYSTERESIS THRESH

- For all the edge points in I_N, and scanning I_n in a fixed order
- Locate the next unvisited edge pixel I_N (i, j) such that I_N (i, J) > T_h
- Starting from I_N (i, j), follow the chains of connected local maxima, in both direction perpendicular to the edge normal, as long as $I_N > T_1$

Mark all visited points and save a list of all the locations of all points in the connected contours found

- the output is a set of lists

4. Modified CNN Edge Detection

The first order non-linear differential equation defining the dynamics of a CNN (using-evolution law and circuit theory) can be written as

$$C\frac{dx_{ij}(t)}{dt} = -R^{-1}x_{ij}(t) + \sum_{c_{ij}\in N_{ij}} A_{ij,kl,}Y_{k1}(t) + \sum_{c_{kl}\in N_{ij}} B_{ij,kl,}U_{k1} + I \quad (2)$$

$$Y_{ij}(t) = \frac{1}{2}\left(\left|x_{ij}(t)+1\right| - \left|x_{ij}(t)-1\right|\right) \quad (3)$$

Where x_{ij} is the state of a cell C_{ij} C and R conform to the **integration time constant** of the system and I is an independent bias constant.

The output equation $Y_{ij}(t) = f(X_{ij}(t))$, where f can be any convenient non-linear function. The matrices A(.) and B(.) are known as cloning templates and are also, respectively called **feedback functional** and **control functional**. The template A(.) acts on the output of neighbouring cells while B(.) in turn effects the **input-control**. The two templates are application-dependent and together with a constant I, determine the transcient behaviour of the CNN (i.e. triple {A, B, I})

In general, the cloning templates do not have to be space invariant, they can but it is not necessary. A significant feature of CNN is that it has two **independent input capabilities:** the generic input and the initial state of the cells.

Normally they are bounded by

$$\left|U_{ij}(t)\right| \le 1 \text{ and } \left|X_{ij}(t)\right| \le 1 \quad (4)$$

$$\text{Similarly, if } \left|f(.)\right| \le 1, \textit{ then } \left|y_{ij}(t)\right| \le 1 \quad (5)$$

The only nonlinear element in each cell is a piece wise linear voltage controlled voltage source with characteristic $y(i, j) = f(x(i, j))$

A widely used nonlinearity is the PWL function as given by

$$y(i, j) = f(x(i, j)) = 0.5^*\left(\left|x+1\right| - \left|x-1\right|\right) \quad (6)$$

4.1. CNN Interface with MATLAB

Using GUI facilities from MATLAB Enviroment as depicted in figure 4 was designed for capturing digital images both coloured and gray-scale image respectively.

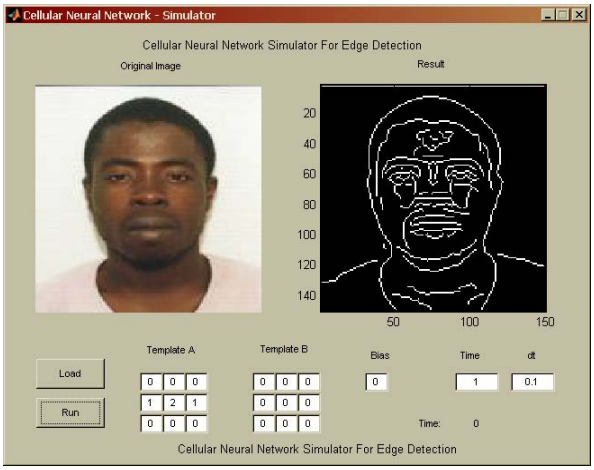

Fig 4 : CNN Graphical User Interface

The Runge-Kutta Method ODE 23 of MATLAB is the employed numerical methods. It is a 4th Order Method and guarantees fast convergence and little error.

CNN-based Edge Detector is simulated and the results (Edge Images) are compared with that of Enhanced Canny edge detector for performance evaluation. The use of symmetric and hyperbolic cloning templates guarantees stability, controllability and continuity of the solution to the CNN model. It's totally shown that CNN has edge results comparable to Canny Edge Detector. The hyperbolic cloning templates whose major entry is tanh x given as

$$\tanh x = \frac{\sinh x}{\cosh x}, \quad \tanh x = \frac{\left(e^{x} - e^{-x}\right) \div 2}{\left(e^{x} + e^{-x}\right) \div 2}, \quad \tanh x = \frac{e^{x} - e^{-x}}{2} x \frac{2}{e^{x} + e^{-x}}, \quad \tanh x = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$$

inf (tanhx) = -1, sup (tanhx) = +1 implies full controllability and continuity of the model.

The five outputs from the two model are hardly differentiable. The functionality of the CNN model is determined by the triple {A, B, I} where {A, B} stands for the two templates that specify the interaction between each cell and all its neighboring cells in terms of their input state and output variables. The interaction could be linear, nonlinear or delay type.

A typical example of the triple {A, B, I} is

$$\tilde{A} = \begin{pmatrix} \cos(t) & \sin(t) & \cos(t) \\ \sin(t) & -20\sin(t) & \sin(t) \\ \cos(t) & \sin(t) & \cos(t) \end{pmatrix}, \quad \tilde{\tilde{B}} = \begin{pmatrix} \sin(t) & \cos(t) & \sin(t) \\ \cos(t) & -20\sin(t) & \cos(t) \\ \sin(t) & \cos(t) & \sin(t) \end{pmatrix}, \text{ and I= } \frac{-3}{1000 \quad + \cos(t)},$$

where $t \in [0\ 2\pi]$

I = Independent bias constant

By imposing the **Von-Neumann boundary** condition and introducing **hyperbolic tangent** we have:

(1) $V_{ik} = V_{ik+1}$ and $V_{ik-1} = V_{ik+2}$, i = -1(1)n+2, k = 0(1) m

(2) $V_{kj} = V_{k+1j}$ and $V_{k-1j} = V_{k+2j}$, j = -1(1)m+2, k = 0(1) n

With (G(V)) being globally Lipschitz i.e. for every **x , y** ϵ **R** **||G(t,x)-G(t,y)|| <=||k (x-y)** || where k is a constant . This condition guarantees **existence**, **uniqueness** and **continuity**

$$we\,use\ A = \begin{pmatrix} 0 & 0 & 0 \\ 0 & k\tanh x & 0 \\ 0 & 0 & 0 \end{pmatrix} B = \begin{pmatrix} -0.25 & -0.25 & -0.25 \\ -0.25 & k\tanh x & -0.25 \\ -0.25 & -0.25 & -0.25 \end{pmatrix} and\ I = -1.5$$

The above templates become more analytical and stable due to its symmetric property and hyperbolic

entry.

5. Comparison between Enhanced Canny and Modified CNN Edge Detectors

The results of the model via simulation is given as follows

ORIGINAL IMAGE **ENHANCED CANNY** **MODIFIED CNN**

Image 1

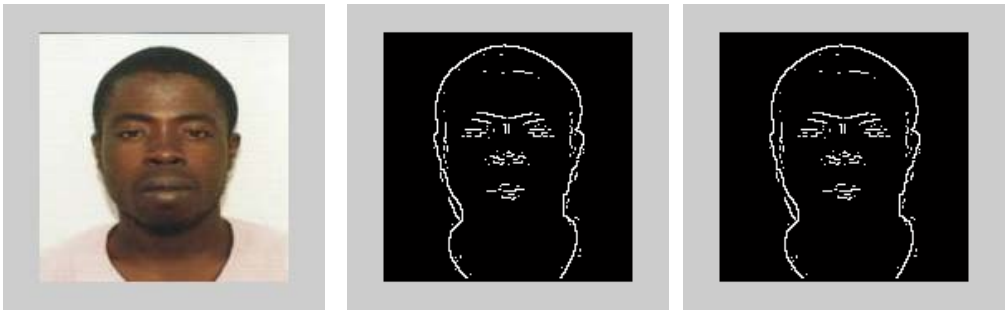

Image 2

Image 3

Image 4

Image 5

The computational time in seconds for the two models are given in table 1 which speaks of our model of Edge Detector as being timely and economical.

Table 1: Computational time in seconds

Images	Canny Model	CNN-Based Model
1	2.2527	1.0450
2	2.2232	1.0370
3	2.2128	1.1025
4	2.3198	1.1122
5	2.4180	1.2424

Table 2: False Alarm rate for the two Models $[\left(P_F\left(\sigma^2\right)\right)]$

Images	Canny Model	CNN-Based Model
1	0.4650	0.8554
2	0.4659	0.8681
3	0.4325	0.7857
4	0.4552	0.8910
5	0.4789	0.8623

The false alarm rate of noise variance probability $\left(P_F\left(\sigma^2\right)\right)$ for which an edge detector can easily declare an edge pixel is shown in table 2. A good false alarm rate should approach a value 1 which indicates that the strength of the detection is high.

We again consider the Differential Equation governing our CNN Model :

$$C\frac{dx_{ij}(t)}{dt} = R^{-1}\,X_{ij}(t_1) + \sum A_{ij,kl}\,Y_{kl}(t_2) + \sum B_{ij,kl}\,U_{kl} \qquad t_1,\, t_2 \in N_{ij} \tag{7}$$

$$Y_{ij}(t) = \frac{1}{2}\left(\,|\,x_{ij}(t)+1\,| - |\,x_{ij}(t)-1\,|\,\right)$$

The state equations of our CNN Model , a system of locally connected ordinary differential equations, can be considered as a spatially discretized representation of a given Partial Differential Equation (PDE) where the functions $A_{ij,kl}$ and $B_{ij,kl}$ have to be chosen appropriately. Then the cell states $x_{ij}(t)$ are approximations of the PDE's solution $x_{ij}(t)$ at a particular point in space and time.

Let the cell outputs be identical with the cells states, which can be achieved by choosing a sufficiently large parameter value c of the piecewise linear output function. Furthermore we set g($x_{ij}(t)$) ≡ 0, and as the PDE considered here are homogenous as the network has no inputs. Then for a single layer network with N(i) = {i-r,…,i+r}, the state equations are given by

$$\left[\frac{dx_{ij}(t)}{dt}\right] = \sum_{l=-r}^{r} A_{ij,kl}(x_{ij}(t+1), x_{ij}(t), I) \tag{8}$$

An approach for solving PDE with CNN is finite approximation of the spatial derivatives .We consider the homogenous Burgers equation

$$\frac{\partial u(x,t)}{\partial t} = \frac{1}{R}\frac{\partial^2 u(x,t)}{\partial x^2} - \frac{1}{2}\frac{\partial u(x,t)^2}{dx} \tag{9}$$

After spatial discretization of the above equation using three points to estimate the derivatives, the resulting set of equations can be represented by a single-layer CNN with state equations with r = 1.

$$\frac{du_i(t)}{dt} = \frac{1}{R}\frac{u_{i+1}(t)-2u_{x_i}(t)+u_{x_{i-1}}(t)}{(\Delta x)^2} - \frac{u^2_{x_{i+1}}(t)-u_{x_{i-1}2}(t)}{4\Delta x} \tag{10}$$

Thus this learning method can be used to determine the solutions of some PDE Models for Edge Detection. This shows that the CNN parameters can be adjusted and modeled to solve Partial Differential Equations and Maximum likelihood problems for edge detection while Canny Model cannot be adjusted for any other functionality. The comparison of the two models is illustrated in table 3.

Table 3: Comparison of The Two Methods

Modified CNN	Enhanced Canny Edge Detector
CNN application goes beyond only edge detection. It is used in other digital Image processing where Canny cannot be used	The smoothing of the image with a Gaussian mark cuts down significantly within the image
The use of symmetric cloning templates guarantees the stability of the CNN	The threshold cuts off points ti and tn makes it controllable
The use of hyperbolic periodic cloning templates guarantees the more stability, continuity and controllability of CNN	Canny Edge Detector takes longer time to get results due to its complexity.
The performance of CNN is determined by the cloning template or the triple {A, B, I}. The bias constant term I is a flexible	The beauty of Artificial Intelligence is not included in canny Edge Detector unlike CNN which takes on a hybrid nature from cellular Automation and Neural Networks
The CNN detect edges in fewer times	Takes longer time as shown in table 1

6. Conclusion

In this work, we have proposed a new approach for edge detection by using Modified Cellular Neural Network with the introduction of hyperbolic entry in the a_{22} of cloning template A. Some experimental results of the proposed algorithm were compared with Enhanced Canny Model. It can be seen that the results from Modified Cellular Neural Network are better than the results from Enhanced Canny Model. This new approach ensures higher degree of controllability and stability. It is timely and economical to use the new model for edge detection.

7. References

[1] Chua L. O. and Yang L. Cellular Neural Networks: theory and applications. *IEEE Transaction on circuit and systems.* 1988, **35**: 1257-1298.

[2] Fisher R, Perkins S, Walker A. and Udfat E. *Feature Detectors, hypermedia Image processing References.* http://homopages.inf.ed.ac.uk/rbf/HIPR2/featops.htm , 2003.

[3] Gonzalez R. and Woods R. *Digital Image Processing (Second Edition).* Prentice-Hall Inc. 2002, pp. 567 – 612.

[4] Green B. *Canny Edge Detection Tutorial*. Drexel University: PRISM, http://www.pages.drexel.edu/inmeg22/can_tuthtml , 2003.

[5] Rubino M. T.. *Edge Detection Algorithms.* http://homopages.inf.ed.ac.uk/rbf/HIPR2/featops.htm, 2005.