Genetically Evolved Cellular Automata for Image Edge Detection

Jebaraj Selvapeter[1] and Wim Hordijk[2]

[1]Texas Instruments (India) Pvt. Ltd, Bangalore, India, Jebaraj@ti.com
[2]SmartAnalytiX.com, wim@SmartAnalytiX.com

Abstract. This paper presents an alternative method for edge detection based on a Cellular Automata (CA) algorithm. The main task with such an algorithm is to find a suitable CA rule out of the 2^{2n} possible rules, where n refers to the number of neighboring cells considered in the update rule. Finding a good rule for the required image processing task by hand is difficult. Here, a genetic algorithm (GA) is used to find a good cellular automata update rule for the edge detection task, for both noise free and noisy images. The method used for edge detection has two distinct phases. In the training phase, the model is trained with simple example patterns. In the execution phase, the best rule found in the training phase is applied to several real images. The system is trained with both Von Neumann and Moore neighborhood configurations. The performance of this proposed approach is compared, using a subjective measure and the false alarm rate, with that of standard edge detection operators and different variations of the CA edge detection method. To further improve performance at higher noise levels, a CA filter is used as a preprocessing stage. Compared with other methods, the evolved CA performs better with less computational requirement.

Key words: Cellular Automata, Genetic Algorithm, Edge Detection

1 Introduction

A fundamental problem in image analysis is edge detection. In particular in the processing of medical or biological images, the study of edges has become an important task. There are numerous applications for edge detection, which is often used for various specific effects. Digital artists, for example, use it to create dazzling image outlines. The output of an edge detector can be added back to an original image to enhance its edges. Edge detection is often the first step in image segmentation, a method that is used to group pixels into regions to determine an image's composition.

The edges of an image hold much information about that image [1]. The edges tell where objects are located, what their shapes and sizes are, and they contain information about their texture. Edges in images are areas with strong intensity contrasts, a jump in intensity from one pixel to the next. Edge detecting an image can significantly reduce the amount of data, and filter out useless information, while preserving the important features.

The first and most obvious requirement for a successful edge detector is a low error rate. It is important that edges occurring in images are not missed, and that there are no responses to non-edges. The second criterion is that the edge points are well localized. In other words, the distance between the edge pixels as found by the detector and the actual edge is to be at a minimum. A third criterion is to have only one response to a single edge (the first two criteria alone are not necessarily enough to eliminate the possibility of multiple responses to an edge).

There are an infinite number of edge orientations, widths, and shapes. Some edges are straight, while others are curved with varying radii. There are many edge detection techniques to go with all these different types of edges, each having its own strengths [2]. Some edge detectors work well in one application and perform poorly in others. Sometimes it takes experimentation to determine what the best edge detection technique for an application is.

Generally, edge detection methods can be grouped into three categories:

1. First order or gradient edge detection operators.
2. Second order or Laplacian edge detection operators.
3. Cellular automata based edge detection operators.

The Sobel operator, Canny operator, Prewitt operator, Roberts's operator, and Isotropic operator are examples of gradient edge detection operators [3]. Gradient operators produce a large response across an area where an edge is present. Ideally, an edge detector should indicate any edges at the center of an edge, called "localization". First order edge detectors often produce multiple width edges. It then becomes necessary to employ a process called "thinning" to reduce the edge width to one pixel. Second order derivative edge detectors provide better edge localization. The Laplacian is a good example of a second order operator. The Laplacian method searches for zero crossings in the second derivative of the image to find edges.

Edge detection based on gradient operators and Laplace operators require much computing time. With an increasing demand for high speed real time image processing, the need for parallel algorithms instead of sequential ones is becoming more important. As an intrinsic parallel computational model, cellular automata (CA) can cater to this need. Previously, different cellular automata models were used for performing edge detection. For example, a simple CA rule is used in [4], an asynchronous CA is presented in [5], and a CA rule based on functional maximization is proposed in [6]. The main drawback with this functional maximization method, however, is the requirement of a threshold value. A new approach for edge recognition based on the combination of a CA and a traditional method of image processing is proposed in [7], in which the concept of a boundary operator is used to represent the state of a cell, and the local rule is defined based on prior knowledge. Here, we take an alternative approach by using a genetic algorithm to evolve a CA rule to perform the edge detection task.

Edge detection operators can be compared in a number of ways. First, the image gradient can be compared visually, since the eye itself performs some sort of edge detection. In the noiseless case, all the operators are roughly equivalent. Quantitatively, the performance under noise of an edge detection operator may be measured as follows. Let N_0 be the number of edge pixels declared, and N_1 be the number of missed or new edge pixels after adding noise. If N_0 is held fixed for the noiseless as well as noisy images, then the edge detection error rate is given in equation (1) below. False alarm rate can also be used to compare the performance of edge detection operators quantitatively.

$$P_e = N_1/N_0 \qquad (1)$$

This paper is organized as follows. Section 2 introduces the evolving cellular automata model for the edge detection task. Section 3 then describes an extension of this model by adding a CA-based noise filter. Section 4 presents the implementation results. Finally, section 5 summarizes the main conclusions.

2 Evolving Cellular Automata for Image Edge Detection

The evolving CA model used in this work operates in two distinct stages [9]: (1) a training phase, and (2) an execution phase. In the training phase, the model is trained with simple example patterns. In the execution phase, the best rule found in the training phase is evaluated on real images.

A. Training phase

In the training phase, the input images and target images are applied to the model as shown in Fig. 1. For this training phase, some simple patterns are used as the input images, which are shown in Fig. 2. The population of CA rules (transition functions) is applied to the input patterns one by one. Each CA rule is allowed to update the pattern for a fixed number of iterations. The output image obtained for each rule after the fixed number of CA iterations is then compared with the target image (i.e., the desired output image). The number of iterations is chosen as to allow adequate time for the cells to reach a stable state. Both Von Neumann and Moore neighborhood configurations are used.

One of the key issues in the success of the model lies in finding an appropriate fitness function, which quantitatively expresses the difference between the desired output pattern (target image) and the obtained output of a CA rule. For the edge detection task, a simple bit-by-bit comparison is adopted. This yields a bitwise match score for each image transformation. The overall fitness value F is then calculated as the root mean square of these matching scores produced on each image training pair.

These fitness values are calculated for each CA rule in the population, and are then used to decide which CA rules are selected for the next generation as parents. The selected parents are subjected to the genetic operators of crossover and mutation, creating “offspring” individuals. A new population of offspring individuals then replaces the previous population, and this process is repeated for a number of generations, until a satisfactory result is obtained [8].

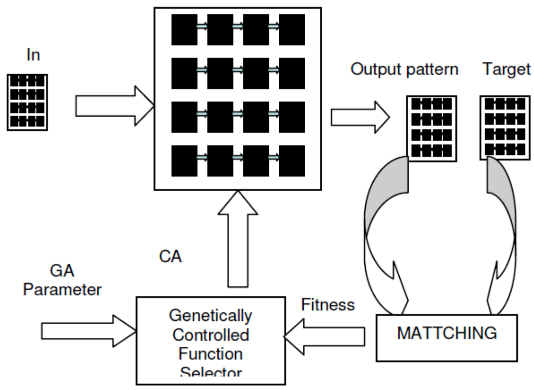

Fig. 1. Evolving CA model

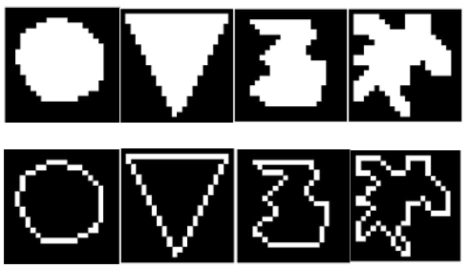

Fig. 2. Input (top row) and target (bottom row) images for the evolving CA model for the edge detection task.

B. Execution Phase

Once an optimal CA rule has been found for the edge detection task, or the maximum number of generations is reached, the system is trained and is ready to begin processing. In the execution phase, previously unseen images are presented to the evolved CA individually, which subsequently updates the cell states for the allowed number of iterations, thus transforming the raw input image to an edge-detected image.

3 Evolving CA with a Noise Filter for Edge Detection of Noisy Images

An extended approach is now introduced to further improve the performance of CA edge detection, in particular at higher noise levels. This new proposed method first uses a CA based noise filter [10,11], followed by the evolved CA for edge detection. Noise filtering is another fundamental problem in image processing. Often, noise filtering needs to be performed before any other image processing task can be done. In previous work, we described a CA based image noise filter [11]. Here, we combine the two methods in a two-step process, by first applying the CA noise filter to a noise corrupted image, and then the evolved CA edge detector to find the edges in an image.

4 Implementation and Results

The first part of this section analyzes the results for noise free images, and the second part analyzes the results for noisy images. The results obtained with the evolving CA model are compared with standard edge detection operators and variations of the CA edge detection methods. First, Table 1 provides an overview of the GA implementation and parameter values used in the training phase of the evolving cellular automata framework for the image edge detection task.

Figure 3 shows the convergence of the genetic algorithm with different population sizes, generations, and crossover probabilities. With a population size of 30 or 40, the convergence of the GA with crossover probability 0.8 is faster compared with a crossover probability of 0.7 and 0.9. When the population size increases to 50 or 60, the convergence of the GA is better with crossover probability 0.9.

Population Size	60
String Length	32
Fitness function	Image size – Error
Max. no. of CA iterations	3
Selection	Elitist selection E = 1
Crossover	One point cross over, Pc = 0.9
Mutation	Pm = 0.035
Generation	150

Table 1. The used GA parameters and values.

The best CA rule obtained in the training phase is applied to a cameraman image (not part of the training images). Fig. 4 shows the result of the edge detection. The CA edge detector finds the edges exactly. A comparison shows that the performance of the evolving CA framework is better (less fuzzy) than the standard first order and second order edge detection operators.

Next, the performance of evolving CA edge detection with noisy images is compared with other CA edge detection methods presented previously, in terms of false alarm rate. Noisy images with different noise ratios are considered. Salt & pepper noise is added to the images and the different CA algorithms are applied to the noisy and noise free images. Fig. 5

shows the comparative performance of the evolved CA edge detection method for noisy image with that of first and second order edge detection operators in terms of false alarm rate. The performance of the Canny operator is very poor. The performance of the evolved CA is comparable up to a 10% noise ratio. When the noise ratio increases, however, the performance starts to degrade, but remains comparable with Log, Canny, and Robert edge detection operators.

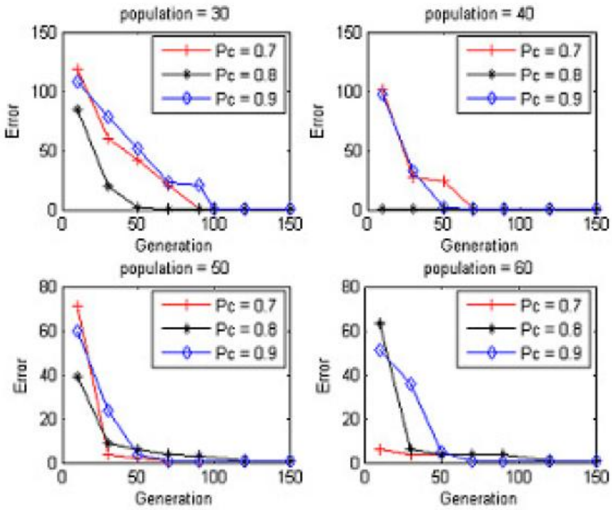

Fig.3. Convergence of the genetic algorithm for various parameter values.

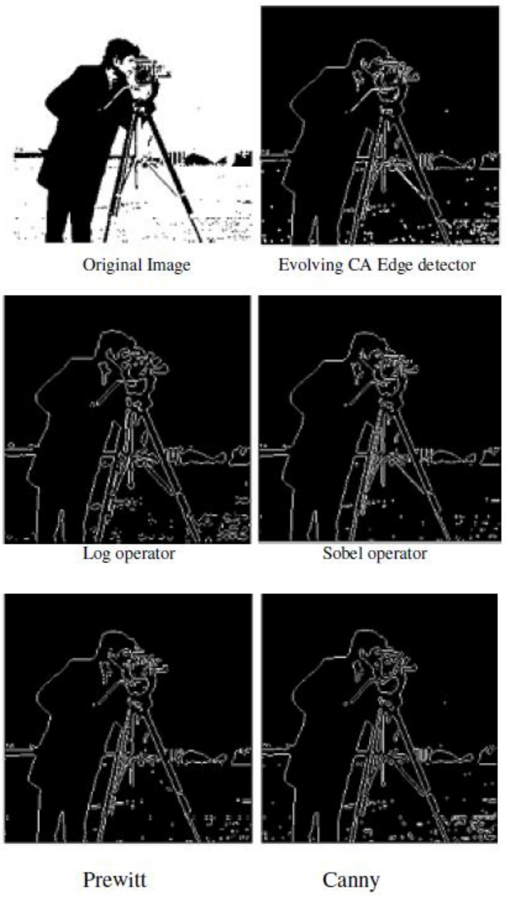

Fig.4. Evolved CA edge detection results and comparison.

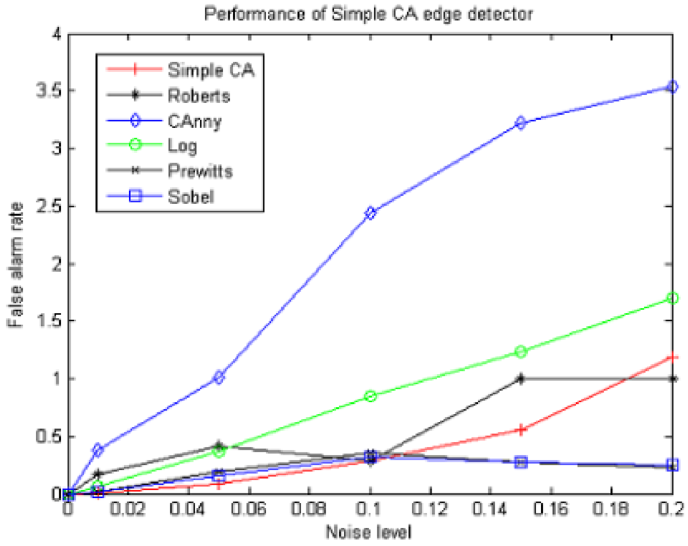

Fig.5. Comparison of the evolved CA edge detection in terms of false alarm rate.

To improve the performance of the evolved CA edge detection with higher noise levels, a Moore neighborhood configuration is used instead. With the Moore neighborhood configuration, the length of each chromosome is 2^9 = 512 bits. The Moore neighborhood configuration evolving CA model requires a larger number of generations to converge. The performance of the evolved CAs with both a Moore and a Von Neuman neighborhood is compared for a noisy image. Table 2 shows the performance in terms of false alarm rate. Compared with the Von Neumann neighborhood configuration, the Moore neighborhood performs well. But at a 20% noise level, performance of first and second order edge operator is superior to the evolved CA edge detection.

To further improve the performance of CA edge detection at higher noise levels, the proposed approach uses a CA based noise filter as a preprocessing stage to reduce the noise level of the input noisy image. Fig. 6 shows the performance of evolving CA edge detection with Moore and Von Neuman neighborhood configurations with a CA filter at 50% noise level. Based on subjective measures, the performance of evolving CA with Moore neighborhood is better than all other edge detection operators. But as discussed, the training of CA with a Moore neighborhood takes more generations to converge compared with a von Neuman neighborhood.

Noise Ratio	EVCA (Moore)	EVCA (Von Neuman)	Canny	Log	Prewitts	Sobel	Roberts
0.01	0.004	0.006	0.050	0.010	0.009	0.008	0.110
0.1	0.076	0.081	1.338	0.160	0.097	0.127	0.130
0.15	0.163	0.231	1.624	0.302	0.125	0.180	1.000
0.2	0.409	0.751	2.016	0.516	0.159	0.193	1.000

TABLE 2. False alarm rate on noisy images.

Finally, Table 3 shows the performance comparison in terms of a quantitative measurement. The false alarm rate parameter is used to evaluate the performance of different edge detection operators with that of the proposed CA edge detection methods. Even at a 20% noise level, the first and second order edge detection methods suffer a lot with noise. But the proposed evolving CA edge detection methods produces good results at a 50% noise level also.

Noise Ratio	CA (Von Neuman)	CA (Moore)	Roberts	Log	Prewitts	Sobel
0.01	0.0169	0.0757	0.0744	0.0152	0.0114	0.0122
0.05	0.0355	0.0866	0.4909	0.0527	0.0622	0.0657
0.1	0.0648	0.0988	0.4609	0.1355	0.093	0.0849
0.2	0.0493	0.0997	1.0000	0.4406	0.1477	0.1421
0.3	0.0191	0.1049	1.0000	0.7420	0.3432	0.4447
0.5	0.3965	0.2997	1.0000	1.428	1.0000	1.0000

TABLE 3. False alarm rate of evolving CA.

5 Conclusions

A simple genetic algorithm is used to train cellular automata to perform an edge detection task. A good CA rule for edge detection was found within 150 generations. The rule obtained was tested with a simple pattern and a real image with 10% salt & pepper noise. The results show that the system can be successfully trained with a genetic algorithm, and that the performance of evolving CA edge detection is better than standard edge detection operators like the Prewitt operator, Canny operator, Log operator, and Sobel operator. For real images, CA edge detection is better than Robert, Log, and Canny operators. At lower noise levels, the CA operator is comparable with the Sobel and Prewitt operators. With a Moore neighborhood configuration, the evolving CA model performance of edge detection improved significantly. The performance of edge detection is even further improved by the use of a CA filter as a preprocessing stage. At low noise levels, due to this preprocessing stage, the performance is similar to the Sobel and Prewitt methods. But even in the presence of 50% noise, the evolving CA model with a Moore neighborhood still gives good edge detection result. Future work may be carried out to find the CA rule for edge detection for even higher noise level images, by using a more complex genetic algorithm, for example by considering multi-point crossover and mutation operators. To enhance the performance at all noise levels, it is also possible to use a switching scheme. Depending on the noise level, it is possible to select a specific pre-processing stage. This evolving CA model can be extended to gray level images also.

References

1. M. Heath, S. Sarkar, T. Sanocki, and K. Bowyer. Comparison of edge detectors: A methodology of initial study. *Computer Vision and Pattern Recognition*, San Francisco, June 1996.
2. L. S. Davis. A survey of edge detection techniques. *Computer Graphics and Image Processing*, 12:248–270, 1975.
3. R. Boyle and R. Thomas. *Computer Vision: A First Course*. Blackwell Scientific Publications, 1988.
4. S. Wongthanavasu and R. Sadananda. A CA-based edge operator and its performance evaluation. *Journal of Visual Communication and Image Representation*, 14:83–96, 2003.
5. A. Scarioni and J. A. Moreno. Border detection in digital images with a simple cellular automata rule. In S. Bandini, R. Serra and F. S. Liverani (eds.), *Cellular Automata: Research towards Industry*, 1998.
6. A. Scarioni and J. A. Moreno. Border detection by the maximization of a functional of pixel difference.
7. Chen ang and hao e.G.Z.Wang. Cellular automata modeling in edge recognition.
8. D. E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison Wesley, 1989.
9. P. Sahota, M. F. Daemi and D. G. Elliman. Training genetically evolving cellular automata for image processing. *International Symposium on Speech, Image Processing and Neural Networks*, Hong Kong, 13-16 April 1994.

10. A. Popovici and D. Popovici. Cellular automata in image processing. In D. S. Gilliam and J. Rosenthal (eds.), *Proceedings of the 15th International Symposium on the Mathematical Theory of Networks and Systems*, 2000.
11. Jebaraj Selvapeter.P and W. Hordijk. Cellular automata for image noise filtering. In A. Abraham, A. Carvalho, F. Herrera and V. Pai (eds.), *Proceedings of the World Congress on Nature and Biologically Inspired Computing*, 193–197, 2009.

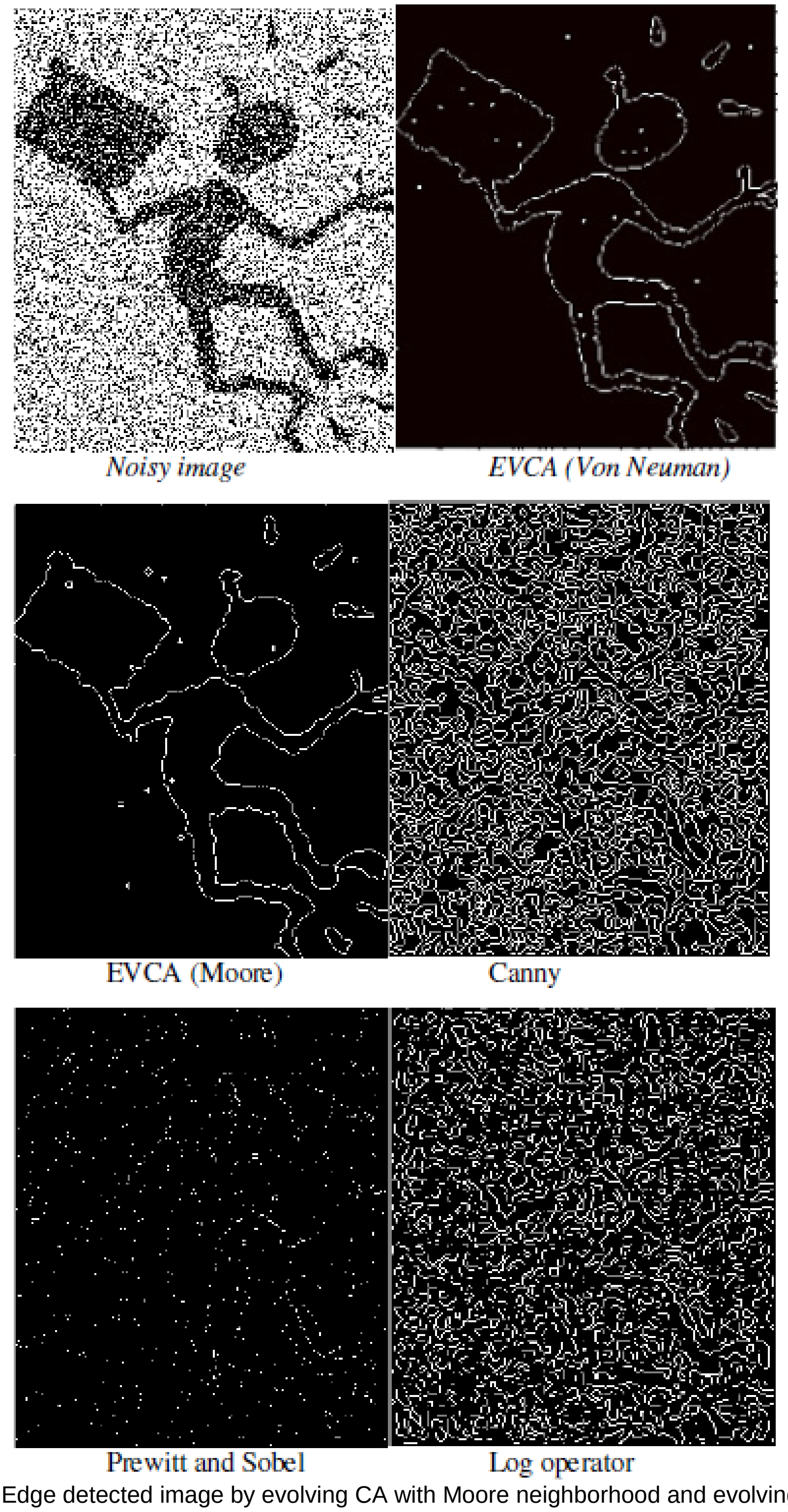

Fig. 6. Edge detected image by evolving CA with Moore neighborhood and evolving CA with Von Neuman neighborhood. Configurations with a 50% noise level.