Edge Detection using Moore Neighborhood

Pratibha Sharma
Student, Deptt. of Computer Science,
MITS, Lakshmangarh

Manoj Diwakar
Asstt. Professor, Deptt. of Computer Science,
MITS, Lakshmangarh

Niranjan Lal
Asstt. Professor, Deptt. of Computer Science,
MITS, Lakshmangarh

ABSTRACT
Edge detection is a fundamental tool in image processing. Several edge detectors have been proposed in literature for enhancing and detecting edges in images. Image Edge detection significantly reduces the amount of data and filters out useless information, while preserving the important structural properties in an image. In this paper, the application of two-dimensional cellular automata using Moore Neighborhood has been proposed for edge detection. The idea is simple but effective technique for edge detection. Edge basically occurs where there is significant change in intensity. The principle of the algorithm used is to increase the difference between those pixels where intensity values change significantly. So by using this concept, detected edges are wider and clear. The given algorithm can be applied to gray scale and monochrome images.

General Term
Edge Detection

Keywords
Cellular Automata, Moore Neighborhood

1. INTRODUCTION
Edge detection is the most fundamental and important tool for feature detection and extraction. Edge detectors aims at identifying points in digital image at which the image brightness changes abruptly or sharply. Edges occur when there is significant change in the local intensities of an image. Edges typically occur on the boundary between two different regions in an image. The goal of edge detection is to produce a line drawing which shows the edges in an image. By using edge detectors, important features can be extracted from images of an image like corners, line, curves etc. The detection results benefit applications such as image enhancement, recognition, morphing, restoration, registration, compression, retrieval, hiding etc.

Now the intensity changes are caused by geometrical or non geometrical events. Geometric events include the object boundary (discontinuity in depth and/or surface color and texture) and surface boundary (discontinuity in surface orientation and/or surface color and texture). Non Geometric events include the specularity (direct reflection of light, such as a mirror) and shadows (from other objects or from the same object).
Much of the work has been done for edge detection. Most of the algorithms used for edge detection have been broadly classified in 2 categories: Gradient based algorithms and Laplacian based algorithms. The gradient based algorithm detects the edges by looking for the maximum and minimum in the first derivative of the image. Roberts, Prewitt, Sobel are base on Gradient method. The Laplacian method searches for zero crossings in the second derivative of the image to find edges. A variety of algorithms have been proposed for analyzing image intensity variation, including statistical methods [1–5], difference methods [6–8] and curve fitting methods[9–13]. The early days of works on edge detection were done by Sobel and Roberts [14]. Their detection methods are based on simple intensity gradient operators. Later on, much of the research works have been devoted to the development of detectors with good detection performance as well as good localization performances.
Edge detection in noisy environment can be treated as an optimal linear filter design problem [15–19]. Canny [16] formulated edge detection as an optimization problem and defined an optimal filter, which can be efficiently approximated by the first derivative of Gaussian function in the one-dimensional case. Canny's filter was further extended to recursive filters [18], which provide a more efficient way for image noise filtering and edge detection.

Other edge detection methods include differentiation based edge detection using logarithmic image processing (LIP) models, contrast-based methods, relaxation labeling techniques and anisotropic diffusion. In fact, these methods can be combined to achieve better performance. For instance, the second directional derivative edge detector proposed by Haralick [10] can be regarded as a hybrid of the differentiation method and the statistical hypothesis testing method, which leads to better performance in a noisy environment.

Classical methods like Sobel, Prewitt, Robert, Prewitt, Marr-Hildreth and Canny some disadvantages also. For example, Sobel, Prewitt and Laplacian based methods are sensitive to noise and are less accurate. Marr-Hildreth algorithm malfunctions at corners and Canny algorithm takes more computation time and still results are not accurate.

So a method is required that provides better clarity, performs well in noisy images and also, it should not detect any false edges. In the proposed algorithm, simple mathematics has been applied for achieving the requirement of better clarity and good performance in noisy images. The principle used in the algorithm is based on finding mean and maximum again and again on every pixel and apply Moore neighborhood concept of Cellular Automata. The result of the algorithm will be compared with the existing common edge detectors and it will be shown that the results obtained by using Cellular Automata are much better.

2. CELLULAR AUTOMATA
A Cellular Automata is a discrete model studied in computability theory, mathematics, physics, complexity science, theoretical biology and microstructure modeling. It consists of a regular grid of cells, each one in one of a finite number of states, such as "On" and "Off. The grid can be in any finite number of dimensions. Generally one dimensional and two dimensional Cellular Automata are used. For each cell, a set of cells called its neighborhood (usually including the cell itself) is defined relative to the specified cell. For example, the neighborhood of a cell might be defined as the

set of cells a distance of 2 or less from the cell. An initial state (time t=0) is selected by assigning a state for each cell. A new generation is created (time t=1), according to some fixed rule (generally, a mathematical function) that determines the new state of each cell in terms of the current state of the cell and the states of the cells in its neighborhood. For example, the rule might be that the cell is "On" in the next generation if exactly two of the cells in the neighborhood are "On" in the current generation; otherwise the cell is "Off" in the next generation. Typically, the rule for updating the state of cells is the same for each cell and does not change over time, and is applied to the whole grid simultaneously. In 2-D CA space, the specified cell P with its North, South, West and East cells forms the Von Neumann neighborhood (see Fig. 1).

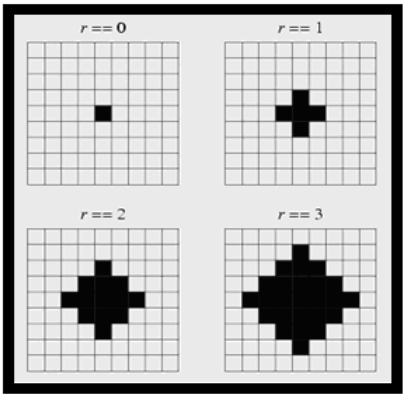

Fig 1: Von Neumann neighborhood (r denotes distance)

In 2D CA space, the specified cell P with 8 surrounding cell forms a Moore neighborhood (see Fig.2*)*.

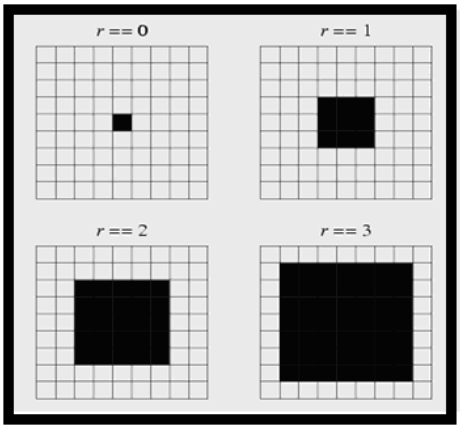

Fig 2: Moore neighborhood (r denotes distance)

The state of the given cell at time step (t+1) will be determined from the states of cells within its neighborhood at time step t. Using a specified rule, the states are updated synchronously in time steps for all cells. For a k-state CA, each cell can take any of the integer values between 0 and (k−1) then for 2-state CA the value of each cell can be either 0 or 1. In general, this relationship can be represented by equation 1.

$$S_i\ (t+1) = f\ (\ S_i(t) + \ S_{neigh(i)}(t))\quad \ldots\ldots\ldots\ldots\ldots\ldots (1)$$

where S_i denotes the state of the i^{th} cell, t denotes the number of generations that have evolved, $S_{neigh}(i)$ denotes the set of neighbors (according to the chosen neighborhood) of the i^{th} cell.

Much work has been carried out using Cellular Automata for edge detection. For instance, Shohei Sato et al. [19] proposed an improved method for designing the CA based edge detector. Different rules Cellular Automata provide different accuracy. For instance in the work carried out by Fasel Qadir et al. [20], they have investigated cellular automata linear rules for edge detection and based on this investigation they had classified the rules into no edge detection rules, strong edge detection rules and weak edge detection rules. Finally, they showed the comparative analysis of the proposed technique with already defined techniques for edge detection and the results showed desirable performance.

3. PROPOSED METHOD

The proposed algorithm is based on Moore Neighborhood concept with r=3. The algorithm starts with a single pixel and considers the Moore Neighborhood around that pixel i.e. 9 pixels are taken in account at a time. The value of the central cell is calculated by considering its 8 neighbors and the cell itself. This concept is represented in equation 2.

$$S_{i,j}(t+1) = f\ (\ S_{i-1,j-1}(t) + \ S_{i+1,j}(t) + \ S_{i-1,j+1}(t) + \ S_{i,j-1}(t) + \ S_{i,j}(t) + \ S_{i,j+1}(t) + \ S_{i+1,j-1}(t) + \ S_{i+1,j}(t) + \ S_{i+1,j+1}(t)\ )\quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2)$$

The above equation implies that the state of the target cell at time t+1 depends on the states of itself and the cells in the Moore neighborhood at time t.

3.1 Proposed Algorithm

As stated previously, cellular automata techniques appear as a natural tool for image processing due to their local nature and simple parallel computing implementation. In this section, the algorithm based on Moore neighborhood is proposed. The algorithm will correspond to edge detection for grayscale and monochrome images. The application of cellular automata concept to real images will be presented, which together with the results will show the performance characteristics and comparison. The Pseudo-code of the cellular automata algorithm for edge detection is as follows:

```
begin
[x y]=size(image)
for i=2 to x-1
  for j=2 to y-1
            img1 = 0;
            img2=image(i-1 to i+1 , j-1 to j+1);  //here
                  Moore neighborhood concept is used
            img2_mean=mean(mean(img2));
    for m=1 to3
      for n=1 to 3
            img1 = (img2(m,n)-img2_mean)^2+img1;
      end
    end
            a(i,j)=sqrt( ( img1/9));
  end
end
```

After this the maximum and mean of ‘a’ matrix is calculated and then apply the steps stated below:

```
for i=2 to x-1
   for j=2 to y-1
      img1 = (a(i,j)-a_mean)^2 + img1;
   end
end
b = sqrt( ( img1 / ((x-2) * (y-2))));
```

```
a = a - b;
for i=2 to x-1
   for j=2 to y-1
     if( a( i , j) < 0)    //removes very low intensity pixels
         a( i , j)=0;
     end
   end
end
for i=2 to x-1
   for j=2 to y-1
      a( i , j)=a(i , j)^(5);
   end
end
Max=max(max(a));
a = a*255/a;
Max=max (max(a));
```

After all these calculations, finally calculate the matrix containing detected edges.

```
for i=2 to x-1
   for j=2 to y-1
      output(i,j)=((a ( i , j) / b) / Max2);
      final_output = output*255;      //final output matrix
 end
end
```

In the above algorithm boundary pixels have not been taken in account. By considering Moore neighborhood around every pixel, simple mathematical calculations are done. This calculation is applied twice so as to make clear distinction between pixels where there is sharp change in intensity. Here the basic principle based on which the clarity is achieved is that the less intensity value pixels are made more negative and more intensity value pixels are made more positive. This helps in detecting edges wider and clear.

4. EXPERIMENTAL RESULTS

For comparison, the images will be compared with existing classical methods namely Sobel, Prewitt, Robert and Canny. Consider images shown in Figure 3. Figure 3(a) is the input image and figures from 3(b) to 3(f) shows result obtained after applying Sobel, Robert, Prewitt, Canny and Cellular Automata.

Figure 3(a): Input image

Figure 3(b): Sobel Operator

Figure 3(c): Robert Operator

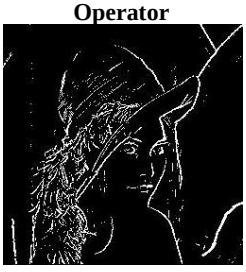

Figure 3(d): Prewitt Operator

Figure 3(e): Canny

Figure 3(f): Cellular Automata

Figure 1: Lena image

Figure 3 shows the output of Lena image after applying various edge detection methods. Figure 3(a) to figure 3(f) shows the result after applying Sobel, Robert, Prewitt, Canny and Cellular Automata respectively. Sobel operator provides thick edges that provide clarity. At the same time, it is not able to detect the left vertical lines and the cap edges are also not detected properly. Robert's operator result is similar to Sobel with similar problems. Prewitt and Canny does not provide continuous edges. The output achieved after applying Cellular Automata concept has much clarity i.e. detected edges are more clear.

Consider another example shown in figure 4. Figure 4(a) shows the input image.

Figure 4(a): Input image

Figure 4(b): Sobel Operator

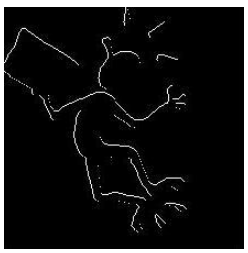

Figure 4(c): Robert Operator

Figure 4(d): Prewitt Operator

Figure 4(e): Canny

Figure 4(f): Cellular Automata

Figure 4: Monochrome Image

In the results shown from figure 4(b) to figure 4(f), the output obtained after applying Robert and Canny are not clear and all

other methods work well with monochrome images. Next example considers simple text as shown in figure 5.

Figure 5(a): Input image

Figure 5(b): Sobel Operator

Figure 5(c): Robert Operator

Figure 5(d): Prewitt Operator

Figure 5(e): Canny

Figure 5(f): Cellular Automata

Figure 5: Monochrome Text Image

Here also, the output obtained after applying Robert and Canny are not clear and all other methods work well with monochrome text images. Consider another example shown in figure 6.

Figure 6(a): Input image

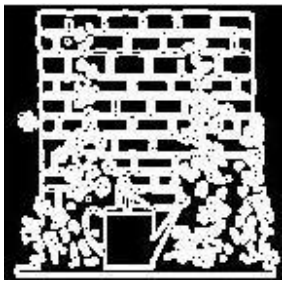

Figure 6(b): Sobel Operator

Figure 6(c): Robert Operator

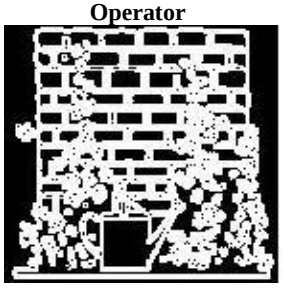

Figure 6(d): Prewitt Operator

Figure 6(e): Canny

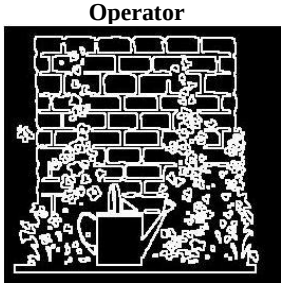

Figure 6(f): Cellular Automata

Figure 6: Gray Scale Image

As it can be seen from the outputs of figure 6, none of the methods provided clear results except Cellular Automata method.

5. CONCLUSION

A simple algorithm using Cellular Automata has been proposed and some mathematical calculations have been done in the algorithm. The result all these modifications are that the detected edges are more enhanced and clear. The disadvantage of the algorithm used is only that it is slightly time consuming. Time taken here basically depends on gray levels involved. More gray levels involved, more is the time taken. Monochrome images take lesser time than gray level images. But if the outputs achieved are better than time can be compromised up to some extent. Matlab 7.10.0 has been used for implementation of the algorithm. One can continue in future work to reduce the time taken for edge detection.

6. REFERENCES

[1] D. Stern, L. Kurz, Edge detection in correlated noise using Latin squares models, Pattern Recognition 21, Pages 119–129, 1988.

[2] J. Haberstroh, L. Kurz, Line detection in noisy and structured background using Graco-Latin squares, CVGIP: Graphical Models Image Process. 55, Pages 161–179, 1993.

[3] N.E. Nahi, T. Asse, Bayesian recursive image estimation, IEEE Trans. Comput. 7, Pages 734–738, 1972.

[4] F.R. Hansen, H. Elliot, Image segmentation using simple Markov models, Comput. Graphics Image Process. 20, Pages 101–132, 1982.

[5] J.S. Huang, D.H. Tseng, Statistical theory of edge detection, Comput. Vision Graphics Image Process. 43 , Pages 337–346, 1988.

[6] J.M.S. Prewitt, Object enhancement and extraction, Picture Processing and Psychopictorics, Academic Press, New York, 1970.

[7] R. Kirsh, Computer determination of the constituent structure of biological images, Comput. Biomed. Res. 4 , Pages 314–328, 1971.

[8] R.M. Haralick, Digital step edges from zero crossing second directional derivatives, IEEE Trans. Pattern Anal. Mach. Intell. PAMI-6 , Pages 58–68, 1984.

[9] M.H. Huechel, An operator which locates edges in digitized pictures, J. Assoc. Comput. Mach. 18, Pages 113–125, 1971.

[10] R.M. Haralick, L. Watson, A facet model for image data, Comput. Graphics Image Process. 15, Pages 113–129, 1981.

[11] V. Nalwa, T.O. Binford, On detecting edges, IEEE Trans. Pattern Anal. Mach. Intell. PAMI-6, Pages 58–68, 1984.

[12] V.S. Nalwa, T.O. Binford, On detecting edges, IEEE Trans. Pattern Anal. Mach. Intell. PAMI-8, Pages 699–714, 1986.

[13] V. Torre, T. Poggio, On edge detection, IEEE Trans. Pattern Anal. Mach. Intell. PAMI-2, Pages 147–163, 1986.

[14] Sobel E., 1970, Camera Models and Machine Perception. , PhD thesis. Stanford University, Stanford, California.

[15] J. Canny, A computational approach to edge detection, IEEE Trans. Pattern Anal. Mach. Intell. PAMI-8, Pages 679–698, 1986.

[16] T.D. Sanger, Optimal unsupervisedlearning in a single layer feedforward neural network, Neural Networks 2, Pages 459–473, 1989.

[17] B.S. Manjunath, R. Chellappa, A unified approach to boundary perception: edges, textures and illusory contours, IEEE Trans. Neural Networks 4, Pages 96–108, 1993.

[18] R. Deriche, Optimal edge detection using recursive filtering, Proceedings of the First International Conference on Computer Vision, London, 1987.

[19] Shohei Sato and Hitoshi Kanoh, Evolutionary Design of Edge Detector Using Rule -Changing Cellular Automata, Nature and Biologically Inspired Computing (NaBIC), Second World Congress, Page(s): 60- 65, 15-17 Dec. 2010.

[20] Fasel Qadir, Khan K. A, Investigations of Cellular Automata Linear Rules for Edge Detection, I. J. Computer Network and Information Security, Vol. 3, Pages 47-53, 2012.