ISSN: 2277-3754
ISO 9001:2008 Certified
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 3, Issue 9, March 2014

Region Based Image Compression and Feature Point Detection Using DCT and Cellular Automata

Garima Tiwari, Ajay Kumar, Sangam

***Abstract-* Biomedical data, such as MRI, MEG, EEG or optical imaging data, present a challenge to any data processing software. Feature point detection has many parameters such as edge detection, corner detection and blob detection (region of interest). Compression technique is used to reduce the size of image over selected region over medical image. So that it will helpful to transfer over WLAN. This paper has two parts first to remove the noise and enhance the medical images with compression and second part is the use of morphological operators to obtain region of interest, excluding the smaller and larger areas for obtaining region by Harris corner detection and cellular automata concept.**

***Keywords:* corner detection, cellular automata.**

I. INTRODUCTION

Digital image processing is an area characterized by the need for extensive experimental work to establish the viability of proposed solutions to a given problem [1]. An important characteristic underlying the design of image processing systems is the significant level of testing & experimentation that normally is required before arriving at an acceptable solution. This characteristic implies that the ability to formulate approaches &quickly prototype candidate solutions generally plays a major role in reducing the cost & time required to arrive at a viable system implementation. Compression technique reduces the size of corner and edges so that visibility of image is not so clear. Image compression addresses the problem of reducing the amount of data required to represent a digital image. Two fundamental components of compression are:

- Redundancy reduction.
- Irrelevancy reduction.

From a mathematical view point, this amount to transforming a 2d pixel array into statistically un-correlated data set. The transformation is applied to storage or transmission of the image. The compressed image can be decompressed to construct the original image or an approximation of it. There are number of compression techniques available like:

- Bit plane encoding.
- Predictive encoding.
- Loseless predictive encoding.
- Fractal based image and video compression.
- Cosine transforms.

II. REGION SPLITTING AND MERGING

In the split-and-merge technique, an image is first split into many small regions during the splitting stage according to a rule, and then the regions are merged if they are similar enough to produce the final segmentation.

Region Splitting:

a) Region growing starts from a set of seed points.
b) An alternative is to start with the whole image as a single region and subdivide the regions that do not satisfy a condition of homogeneity.

Region Merging:

a) Region merging is the opposite of region splitting.
b) Start with small regions (e.g. 2x2 or 4x4 regions) and merge the regions that have similar characteristics (such as gray level, variance).
c) Typically, splitting and merging approaches are used iteratively.

Splitting and merging attempts to divide an image into uniform regions. The basic representational structure is pyramidal, i.e. a square region of size m by m at one level of a pyramid has 4 sub-regions of size m/2 by m/2 below it in the pyramid. Usually the algorithm starts from the initial assumption that the entire image is a single region, and then computes the homogeneity criterion to see if it is TRUE. If FALSE, then the square region is split into the four smaller regions [2][3]. This process is then repeated on each of the sub-regions until no further splitting is necessary. These small square regions are then merged if they are similar to give larger irregular regions. The problem (at least from a programming point of view) is that any two regions may be merged if adjacent and if the larger region satisfies the homogeneity criteria, but regions which are adjacent in image space may have different parents or be at different levels (i.e. different in size) in the pyramidal structure. The process terminates when no further merges are possible.

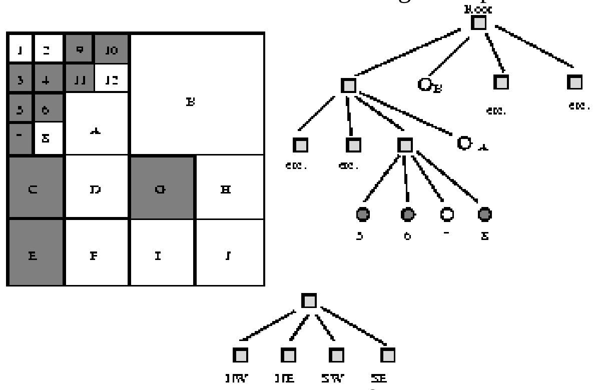

Fig 1: Quad splitting of an image

Although it is common to start with the single region assumption, it is possible to start at an intermediate level, e.g. 16 regions or whatever. In the latter case, it is possible

ISSN: 2277-3754
ISO 9001:2008 Certified
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 3, Issue 9, March 2014

that 4 regions may be merged to form a parent region. For simplicity, assume we start with a single region, i.e. the whole image. Then, the process of splitting is simple. A list of current regions to be processed, i.e. regions defined as not homogeneous is maintained [4] [8]. When a region is found to be homogeneous it is removed from the Process List and placed on a Region List. In its simplest form, region growing methods usually start by locating some seeds representing distinct regions in the image. The seeds are then grown until they eventually cover the entire image. The region growing process is therefore governed by a rule that describe the growth mechanism and a rule that check the homogeneity of the regions at each growth step. Region growing technique has been applied to MRI segmentation [9][14]. A semi-automatic, interactive MRI segmentation algorithm was developed that employ simple region growing technique for lesion segmentation. In, an automatic statistical region growing algorithm based on a robust estimation of local region mean and variance for every voxel on the image was proposed for MRI segmentation. The best region growing parameters are automatically found via the minimization of a cost functional. Furthermore, relaxation labeling, region splitting, and constrained region merging were used to improve the quality of the MRI segmentation. The determination of an appropriate region homogeneity criterion is an important factor in region growing segmentation methods. However, such homogeneity criterion may be difficult to obtain a priori. An adaptive region growing method is proposed where the homogeneity criterion is learned automatically from characteristics of the region to be segmented while searching for the region.

III. IMAGE COMPRESSION

The DCT is widely used transformation in transformation for data compression. It is an orthogonal transform, which has a fixed set of (image independent) basis functions, an efficient algorithm for computation, and good energy compaction and correlation reduction properties. Ahmed et al found that the Karhunen Loeve Transform (KLT) basis function of a first order Markov image closely resemble those of DCT. They become identical as the correlation between the adjacent pixel approaches to one. The DCT belongs t the family of discrete trigonometric transform, which has 16 members. The 1D DCT of a 1* N vector x(n) is defined as

$$Y[k] = C[k]\sum_{n=0}^{N-1} x[n]\cos\left[\frac{(2n+1)k\pi}{2N}\right]$$

… (1) Where k = 0,1,2......N- 1

and

$$C[k] = \begin{bmatrix} \sqrt{\frac{1}{N}} & \text{for } k = 0 \\ \sqrt{\frac{1}{N}} & \text{for } k = 1,2,\dots, N-1 \end{bmatrix}$$

The JPEG (Joint Photographic Experts Group) standard has been around for some time and is the only standard for lossy still image compression. There are quite a lot of interesting techniques used in the JPEG standard and it is important to give an overview of how JPEG works. There are several variations of JPEG, but only the ‘baseline’ method is discussed here.

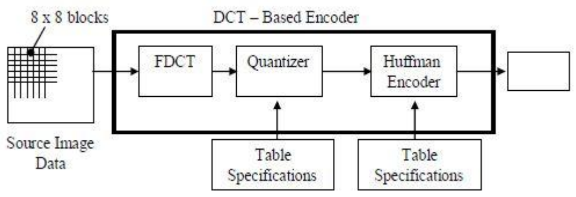

Fig 2 JPEG Encoder

As shown in figure, the image is first partitioned into non-overlapping 8*8 blocks. A Forward Discrete Cosine Transform (FDCT) is applied to each block to convert the spatial domain gray levels of pixels into coefficients in frequency domain. To improve the precision of DCT the image is ‘zero shifted’, before the DCT is applied. This converts a 0-255 image intensity range to a -128-127 range, which works more efficiently with DCT. One of these transformed values is referred to as the DC coefficient and the other 63 as the AC coefficients. After the computation of DCT coefficients, they are normalized with different scales according to a quantization table provided by the JPEG standard conducted by psycho visual evidence. The quantized coefficient are rearranged in a zigzag scan order for further compressed by an efficient lossless coding algorithm such as run length coding, arithmetic coding, Huffman coding.

IV. HARRIS CORNER DETECTION

Corner detection is used to know the shape as well as certain features of the image. Biomedical images also identified with help of corner detection. Conditions that need to be satisfied are as follows:

a) Detection of true corners.
b) Position of the corner should be accurately detected.
c) Detector should be stable and must have high repeatability rate.
d) It should be immune to noise.

Localization means that the position of the corner should be detected accurately. It is mainly used for medical purpose for e.g. Detection of Brain Tumor. If we consider an insect and its motion is recorded by the camera. The two consecutive frames will be similar but the difference lies in the viewpoint or position at that particular instant [14]17]. Corner detection fails to detect one corner after rotation. Stability or repeatability rate is defined as the ratio of the total numbers of corners that are repeated in the two images to the total number of corners. Harris corner detection algorithm is realized by calculating each pixel’s gradient

ISSN: 2277-3754
ISO 9001:2008 Certified
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 3, Issue 9, March 2014

[11]. If the absolute gradient values in two directions are both great, then judge the pixel as a corner.

V. CELLULAR AUTOMATA

A cellular automaton (abbrev. CA) 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 in one of a finite number of *states*, such as "On" and "Off" (in contrast to a coupled map lattice). The grid can be in any finite number of dimensions. For each cell, a set of cells called its *neighborhood* (usually including the cell itself) is defined relative to the specified cell[10][12]. 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 (advancing t by 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, though exceptions are known. In 2-D CA space, the specified cell *P* with its North, South, West and East cells forms the *von Neumann* neighborhood (see Fig. 3).

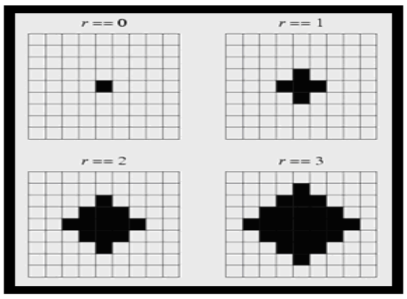

Fig 3: Von Neumann neighborhood

In 2D CA space, the specified cell P with 8 surrounding cell forms a *Moore neighborhood* (see Fig.4 *).*

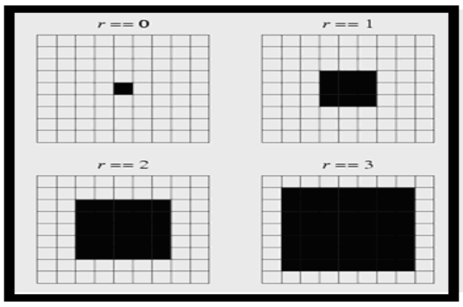

Fig 4: Moore neighborhood

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 is 0 or 1. In general, this relationship can be expressed [23][24] as:

$$S_i(t+1) = f(S_i(t) + S_{neigh}(i)(t)) \ldots\ldots\ldots.(2)$$

Where S_i denotes the state of the ith 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 ith cell. CA has been applied successfully to several physical systems, processes and scientific problems that involve local interactions, as in image processing [12, 20], data encryption [10, 21] and byte error correcting codes [19]; it has also been used in pseudorandom number generators for built-in VLSI self-tests [21]. In our project we have used Moore neighborhood concept i.e. we will process every pixel by considering its 8 neighbors.

VI. PROPOSED FEATURE POINT DETECTION METHOD

By using simple concept of Moore neighborhood i.e. calculating the value of central cell by considering its 8 neighbors and cell itself. This concept can be represented in equation form as:

$$\begin{aligned} Si,j\ (t+1) = f\ (\ & Si-1,j-1(t),\ Si+1,j(t),\ Si-1,j+1(t),\ Si,j-1, \\ & Si,j(t)\ ,Si,j+1(t),\ Si+1j-1(t),\ Si+1,j(t), \\ & Si+1,j+1(t)\ )\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(3) \end{aligned}$$

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.

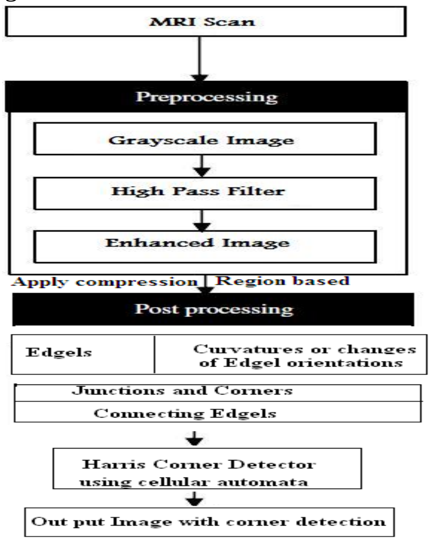

Fig 5: Proposed method

ISSN: 2277-3754
ISO 9001:2008 Certified
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 3, Issue 9, March 2014

A. *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, we present one algorithm and investigate its variation and effectiveness for processing of images. The algorithm will correspond to edge detection for grayscale and monochrome images then apply Harris corner detection method. 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 calculate the maximum and mean of ‘a’ matrix and then apply:

```
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 we have taken in account all pixels except boundary pixels. By considering Moore neighborhood of every pixel, we calculate mean and apply simple calculations. This calculation is applied twice so as to make clear distinction between pixels where there is sharp change in intensity. Here 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.

VII. EXPERIMENTAL RESULTS

Now we will compare our result with other methods so that the difference can be seen clearly. We will use different formats of images here.

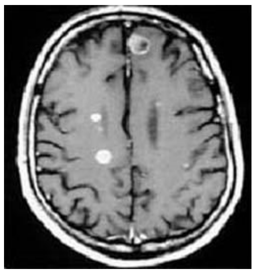
Fig 6: Input Image

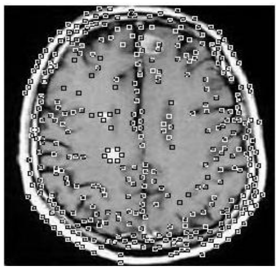
Fig 7: compressed image With corner detection

As it can be seen clearly from the above output that detected images are more clear and wider. Consider another example of png format.

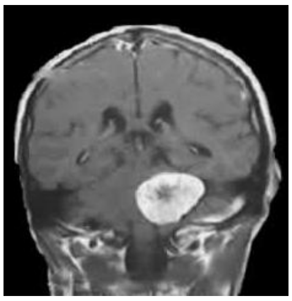
Fig 8: Input Image

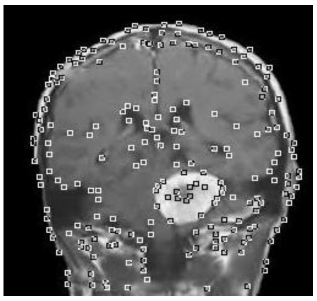
Fig 9: compressed image with conrner detection

VIII. CONCLUSION AND FUTURE WORK

We have proposed a simple algorithm and applied some mathematical calculations. The result all these modifications are that the detected edges and corners are enhanced and clear. Compression over image with region based give reduced size and corner detection is helpful to recover edges and corners for visual perception. One might think that the time taken is more by this algorithm but if the outputs are good, than it is not a problem. 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. We have used Matlab 7.10.0 for implementation of our algorithm. We are also trying to come up with iterative images as output where one output serves as second input to get more enhanced output. One can continue in future work to reduce the time taken for Feature point detection. We tested this work over bio-medical images. It can also work for other images.

REFERENCES

[1] Ziou, D. and Tabbone, S., 1998. Edge detection techniques—an overview, Pattern Recognition and Image Analysis 8 (4), pp. 537–559.

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

[3] J. Haberstroh, L. Kurz, Line detection in noisy and structured back ground using Graco-Latin squares, CVGIP: Graphical Models Image Process. 55 (1993) 161–179.

[4] N.E. Nahi, T. Asse4, Bayesian recursive image estimation, IEEE Trans. Comput. 7 (1972) 734–738.

[5] F.R. Hansen, H. Elliot, Image segmentation using simple Markov 4eldmod els, Comput. Graphics Image Process. 20 (1982) 101–132.

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

[7] J.M.S. Prewitt, Object enhancement and extraction, in: B.S. Lipkin, A. Rosenfeld(Ed s.), Picture Processing and Psychopictorics, Academic Press, New York, 1970.

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

[9] D. Marr, E. Hidreth, Theory of edge detection, Proc. Roy. Soc. London PAMI-6 (1984) 58.

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

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

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

[13] Harris, C., Stephens, M.: A combined corner and edge detector. Proc. 4th Alvey Vision Conference, pp. 147-151 (1988)

[14] Pei, S., J. Ding, J.: Improved Harris' Algorithm for Corner and Edge Detections. In: Proc. ICIP, pp. 57-60 (2007)

[15] Liu, Y, Hou, M., Rao, X., Zhang, Y.: A Steady Corner Detection of Gray Level Images

[16] Based on Improved Harris Algorithm. In: Proc. Int. Conf. on Networking, Sensing and Control, pp. 708-713 (2008)

[17] Kitchen, L., Rosenfeld, A.: Gray-level corner detection. Pattern Recognition Letters 1 (2), 95-102 (1982)

[18] Smith, S.M., Brady, J.M.: SUSAN—A New Approach to Low Level Image Processing. International Journal of Computer Vision 23 (1), 45-78 (1997)

[19] Wang, H., Brady, J.M.: Real-time corner detection algorithm for motion estimation. Image and Vision Computing 13 (9), 695-703 (1995)

[20] Mokhtarian, F., Suomela, R.: Robust image corner detection through curvature scale space .IEEE Trans. PAMI 20 (12), 1376-1381 (1998)

[21] Mokhtarian, F., Mohanna, F.: Enhancing the curvature scale space corner detector. In: Proc.

[22] Scandinavian Conf. on. Image Analysis, pp. 145-152 (2001)

[23] Zheng, Z., Wang, H., E. K. Teoh, E.K.: Analysis of gray level corner detection. Pattern Recognition Letters 20 (2), 149-162 (1999)

[24] Tomasi, C., Manduchi, R.: Bilateral Filtering for Gray and Color Images. In: Proc. ICCV, pp. 839-846 (1998)

[25] Brox, T., Weickert, J., Burgeth, B., Mrazek, P.: Nonlinear structure tensors. Image and Vision Computing 24 (1), 41-55 (2006)