Design of Graphical User Interface for different Image Representation Techniques in MATLAB

Bhaumik Vaidya[1] Parth Vaghasiya[2] Ankit Patel[3]
[1]Assistant Professor [2,3]Student
[1,2,3]Department of Electronics & Telecommunication Engineering
[1,2,3]SCET, Surat

Abstract— After an Image has been segmented into regions by some methods, the resulting segmented pixels usually is represented and described in a form suitable for further computer processing. Basically, representing a region involves two choices: It can be represented as the region in terms of its external characteristics (its boundary), or can be represented in terms of its internal characteristics (the pixel comprising the region). In this paper three techniques for Image Representation, boundary following, chain code and minimum-perimeter-polygon has been discussed and Graphical User Interface for these techniques has been developed in MATLAB software. The results of all these techniques are shown.

Key words: Image Representation, Boundary Following, Chain Code, Minimum Perimeter Polygon, Matlab, GUI

I. Introduction

After an Image has been segmented into regions by some methods, the resulting segmented pixels usually is represented and described in a form suitable for further computer processing. Basically, representing a region involves two choices: 1) Can be represented as the region in terms of its external characteristics (its boundary), or 2 Can be represented in terms of its internal characteristics (the pixel comprising the region). Choosing a representation scheme however is only part of the task of making the data useful for the computer. The next task is to describe is the region based on the chosen representation. For example, a region may be represented by its boundary, and the boundary described by features such as its length, the orientation of the straight line joining its extreme points, and the number of concavities in the boundary. Representing is task of making the data useful for the computer. An External Representation is chosen when primary focus is on the shape characteristics. An Internal Representation is chosen when the primary focus is on regional properties, such as color and texture. Features selected as descriptors should be as insensitive as possible to variation in size, translation and rotation. In this paper three image representation techniques are discussed and implemented in GUI. First is Boundary following whose output is an ordered sequence of points. This algorithm is sometimes referred to as Moore boundary tracking algorithm. Second, Chain codes are used to represent boundary by connected sequence of straight line segments of specified length and directions. Third Polygonal approximations using minimum-perimeter-polygon (MPP). It is one of the most powerful polygonal approximation techniques.

II. Boundary following Algorithm

The output of the algorithm is an ordered sequence of points. For this following assumptions are made:

1) Working with binary images in which object and background points are labeled as 1 and 0.
2) Images are padded with a border of 0s to eliminate the the possibility of an object merging with the image border.

Given a binary region R or its boundary, an algorithm for following the border of R, or the given boundary, consist of the following steps:

1) Let the starting point,b0 be the uppermost , leftmost point in the image that is labeled 1.Denote by c0 the west neighbor of b0.Clearly,c0 always is a starting at c0 and proceeding in a clockwise direction. Let b1 denote the first neighbor encountered whose value is 1, and let c1 be the point immediately b1 in the sequence. Store the location of b0 and b1 for use in step 5.
2) Let b=b1 and c=c1.
3) Let the 8-neighbors of b, starting at c and proceeding in a clockwise direction,be denoted by n1, n2,…, n8.Find the first nk labeled 1.
4) Let b==nk and c=nk-1.
5) Repeat steps 3 and step 4 until b=b0 and the next boundary point found is b1.The sequence of b points found when the algorithm stops constitutes the set of ordered boundary points.

The boundary following algorithm works equally well if a region, rather than its boundary is given. That is, the procedure extracts the outer boundary of a binary region. If the objective is to find the boundaries of holes in a region, a simple approach is to extract the holes and treat them as 1-valued regions on a background of 0s. Applying the boundary following algorithm to these regions will yield the inner boundaries of the original region. We could have stated the algorithm just as easily based on following a boundary in the counterclockwise direction. In fact, you will encounter algorithms formulated on the assumption that boundary points are ordered in that direction.

III. Chain code

It represents a boundary by a connected sequence of straight-line segments of specified length and direction.

1) 4-directional chain codes
2) 8-directional chain codes

The direction of each segment is coded by using a numbering scheme as shown in figure 1. A boundary code formed as a sequence of such directional numbers is referred to as a Freeman chain code.

Digital images usually are acquired and processed in a grid format with equal spacing in the x and y direction, so a chain code can be generated by following a boundary in say a clockwise direction and assigning a direction to segments connecting every pair of pixels. This method generally is unacceptable for two principle reasons:

1) The resulting chain tends to be quite long and
2) Any small disturbances along the boundary due to noise or imperfect segmentation cause changes in the code that may not be related to the principal shape features of the boundary.

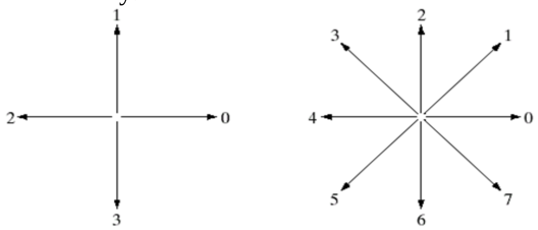


Fig. 1: 4 and 8- Connectivity

An approach frequently used to circumvent these problems is to resample the boundary by selecting a larger grid spacing. Then, as the boundary is traversed, a boundary point is assigned to each node of the large grid, depending on the proximity of the original boundary to that node, as shown. The resampled boundary obtained in this way then can be represented by a 4- or 8 code. It is a simple matter to convert from an 8- code to a 4-code,and vice versa. The starting point is at the topmost, leftmost point of the boundary. The accuracy of the resulting code representation depends on the spacing of the sampling grid. The chain code of a boundary depends on the starting point.

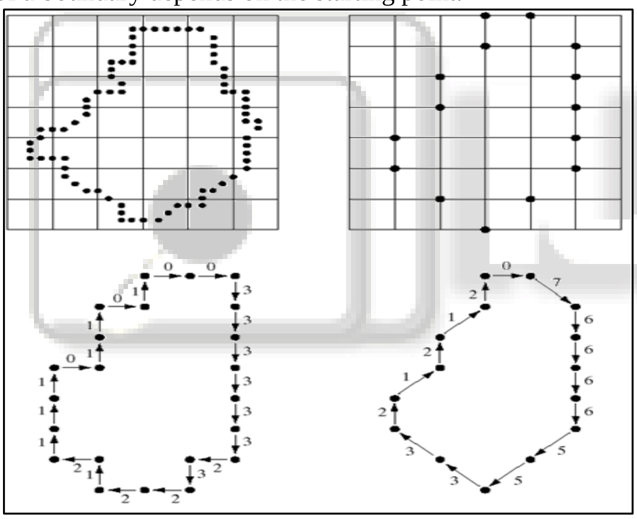


Fig. 2: Chain Code Example

However, the code can be normalized with respect to the starting point by simply treat the chain code as a circular sequence of direction of numbers and redefine the starting point so that the resulting sequence of numbers forms an integer of minimum magnitude. We can normalize also for rotation by using first difference of the chain code instead of the code itself. The first difference is obtained by counting number of direction changes that separate two adjacent elements of the code. These normalization are exact only if the boundaries themselves are invariant to rotation and scale change.. For instance, the same object digitized in two different orientations will have different boundary shapes in general, with the degree of dissimilarity being proportional to image resolution. This effect can be reduced by selecting chain elements that are long in proportion to the distant between pixels in the digitized image and/or by orienting the resampling grid along the principal axis of the object to be coded. Using any of these codes to represent the boundary results in a significant reduction in the amount of data needed to store the boundary. In addition, working with code numbers offers a unified way to analyze the shape of a boundary. Finally, keep in mind that the subsampled boundary can be recovered from any of the preceding codes.

IV. POLYGONAL APPROXIMATION USING MINIMUM-PERIMETER POLYGONS

A digital boundary can be approximated with arbitrary accuracy by a polygon. For a closed boundary, the approximation become exact when the number of segments of the polygon is equal to the number of points in the boundary. The goal of a polygonal approximation is to capture the essence of the shape in a given boundary using the fewest possible number of polygons. One of the most powerful polygonal approximation technique is minimum perimeter polygon. The size of the cells determines the accuracy of the polygonal approximation. In limit, if the size of each cell correspond to a pixel in the boundary ,the error in each cell between the boundary and the MPP approximation at most would be 1.414d, where d is the minimum possible distance between pixels. This error can be reduced in half by forcing each cell in the polygonal approximation to be centered on its corresponding pixel in the original boundary. The objective is to use the largest possible cell size acceptable in a given application, thus producing MPPs with the fewest number of vertices. The set of cells enclosing a digital boundary, is called a cellular complex. It is assumed that the boundaries under consideration are not self-intersecting, which leads to simply connected cellular complexes. Based on these assumptions, and letting black(B) and white(W) denote convex and mirrored concave vertices respectively, following observations are stated:

1) The MPP bounded by a simply connected cellular complex is not self-intersecting.
2) Every convex vertex of the MPP is a B vertex, but not every B vertex of a boundary is a vertex of the MPP.
3) Every mirrored concave vertex of the MPP is a W vertex, but not every W vertex of a boundary is a vertex of the MPP.
4) All W vertices are on outside the MPP ,and all B vertices are on or inside the MPP
5) The uppermost, leftmost vertex in a sequence of vertices contained in a cellular complex is always a B vertex of the MPP.

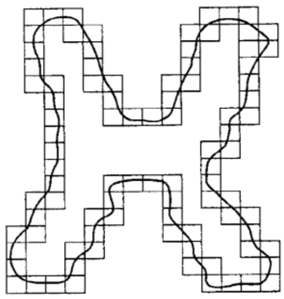


Fig. 3: Object boundary enclosed by cell

Convex (black dots) and mirrored concave points (white dots) of the above figure considering inner boundary generated by enclosed cells.

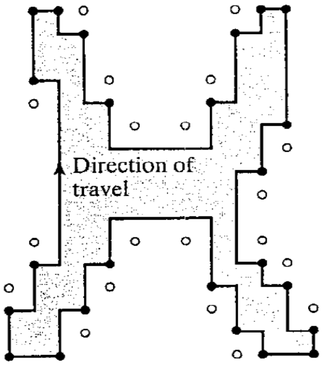


Fig. 4: Convex and Concave Points

In the discussion that follows consider a triplet of points (a,b,c),and let the coordinate of these points be a=(x1,y1),b=(x2,y2) and c=(x3,y3).

If these points are arranged as the row of matrix

A=[x1,y1,1;x2,y2,1;x3,y3,1] Then it follows from elementary matrix analysis that

det(A)>0 if (a,b,c) is a counterclockwise sequence

det(A)=0 if (a,b,c) points are collinear

det(A)<0 if (a,b,c) is a clockwise sequence .

To prepare the data for the MPP algorithm we form a list whose rows are the coordinate of each vertex and additional element denoting whether the vertex is B or W. The algorithm for finding MPP used two crawler points: White crawler (Wc) and Black Crawler (Bc). Wc crawls along the mirrored concave vertices and Bc crawles along convex vertices. These two crawler points, the last MPP vertex found and the vertex being examined are all that necessary. The algorithm starts by setting Wc = Bc = V0. Then at any step in the algorithm let Vl denote the last MPP vertex found and let Vk denote current vertex being examined. One of three conditions can exist between Vl, Vk and two crawler points.

1) Vk lies to the positive side of the line through pair (Vl, Wc) , that is sgn(Vl,Wc,Vk) > 0.
2) Vk lies on the negative side of the line through pair(Vl,Wc) or is collinear with it; that is sgn(Vl,Wc,Vk) <= 0. At the same time Vk lies to the positive side of the line through (Vl,Bc) or is collinear with it; that is sgn(Vl,Bc,Vk) > = 0.
3) Vk lies on the negative side of the line through pair (Vl,Bc) that is sgn(Vl,Bc,Vk) < 0.

If condition (a) holds , the next MPP vertex is Wc and let Vl = Wc then reinitialize the algorithm by setting Wc=Bc=Vl , and continue with the next vertex after Vl.

If condition (b) holds, Vk becomes candidate MPP vertex. In this case set Wc = Vk , if Vk is concave ; otherwise we let Bc = Vk. Then continue with the next vertex.

If condition (c) holds , the next MPP vertex is Bc and we let Vl = Bc , then reinitialize the algorithm by setting Wc = Bc = Vl and continue with the next vertex after Vl.

The algorithm terminates when it reaches the first vertex again. And thus as processed all the vertices in the polygon. The Vl vertices found by the algorithm are the vertices of the MPP. So, final MPP image for the above figure looks like the figure given below:

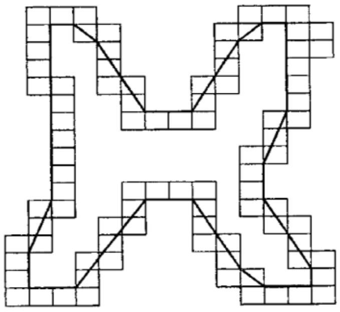


Fig. 5: Final MPP image

Thus it has been proved that this algorithm finds all the MPP vertices of a polygon by a simply connected cellular complex.

V. GUI

A graphical user interface (GUI) is a pictorial interface to a program. A good GUI can make programs easier to use by providing them with a consistent appearance and with intuitive controls like pushbuttons, list boxes, sliders, menus, and so forth. The GUI should behave in an understandable and predictable manner, so that a user knows what to expect when he or she performs an action.

A graphical user interface provides the user with a familiar environment in which to work. This environment contains pushbuttons, toggle buttons, lists, menus, text boxes, and so forth, all of which are already familiar to the user, so that he or she can concentrate on using the application rather than on the mechanics involved in doing things. However, GUIs are harder for the programmer because a GUI-based program must be prepared for mouse clicks (or possibly keyboard input) for any GUI element at any time. Such inputs are known as events, and a program that responds to events is said to be event driven.

VI. OUTPUT OF CHAIN CODE ALGORITHM

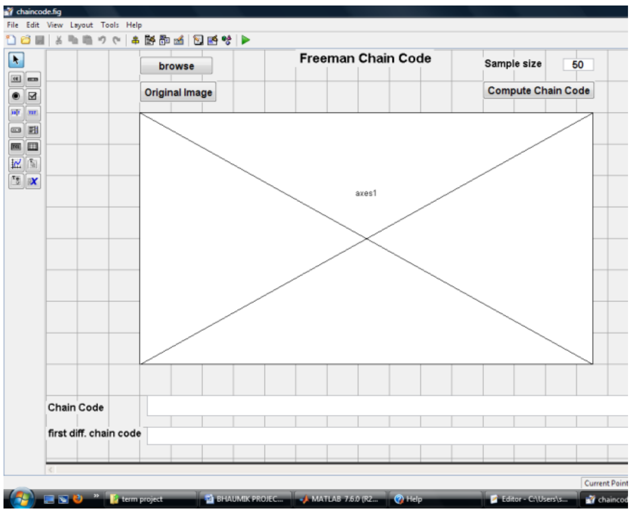


Fig. 6: Basic GUI

When we run the gui and click on browse to select an image it opens the following window:

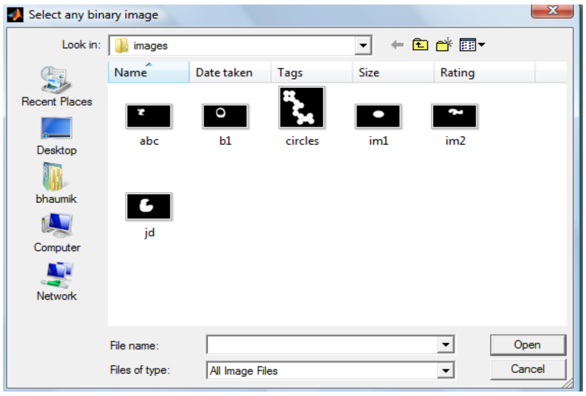


Fig. 7: Select an Image

From this if image circles.png is selected and click on open, then click on original image then it will display original image.

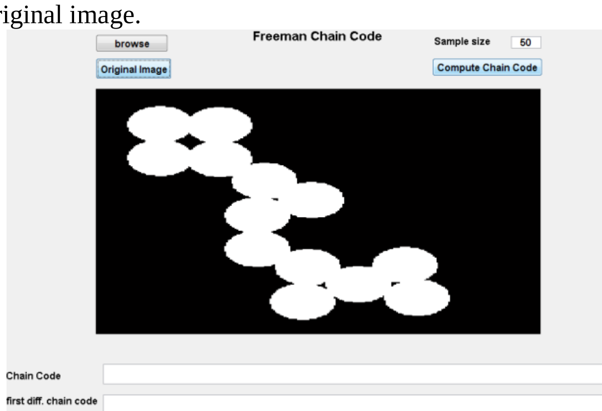
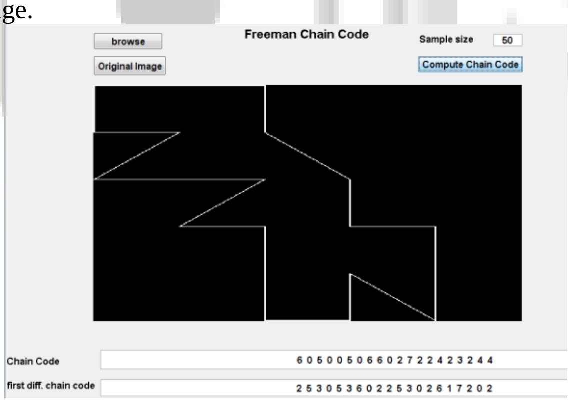


Fig. 8: GUI with original image displayed

Then click on compute chain code, then it will display the chain code and first difference chain code of the image.



If the sample size is decreased then the chain code approximation will be more close to original image. The number of codes will increase. In above sample size is changed to 40, then output will be,

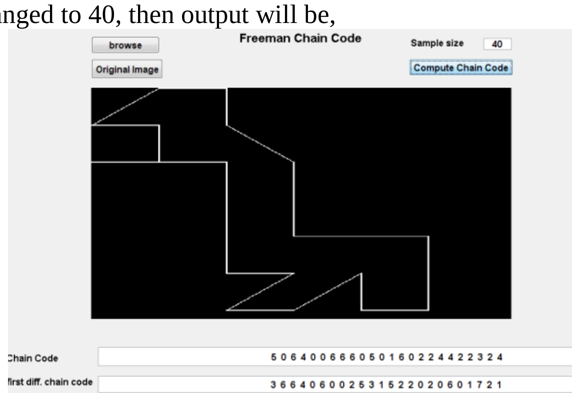


Fig. 9: Chain code with less sample size

VII. OUTPUT OF MPP ALGORITHM:

The GUI of chain code made using guide command:

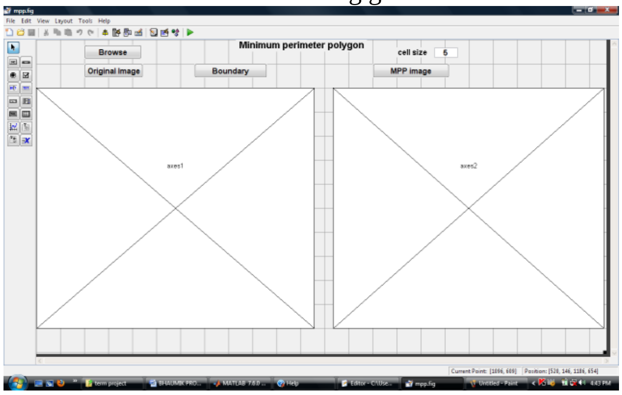


Fig. 10: Basic GUI of MPP algorithm

From this if image is selected as shown in chain code algorithm than original image will be displayed on Axis1.

Then if we click on boundary button, it will display boundary of the image.

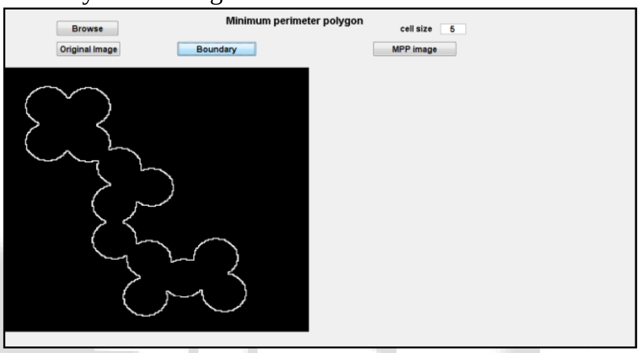


Fig. 11: Original Image

Then if we click on MPP image then it will display minimum perimeter polygone approximated image.

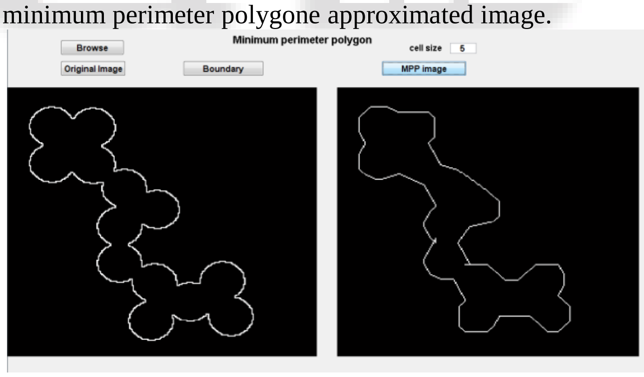


Fig. 12: Image and its boundary

If the cell size is decreased then the MPP approximation will be more close to original image. In above if we change cell size to 3, then output will be,

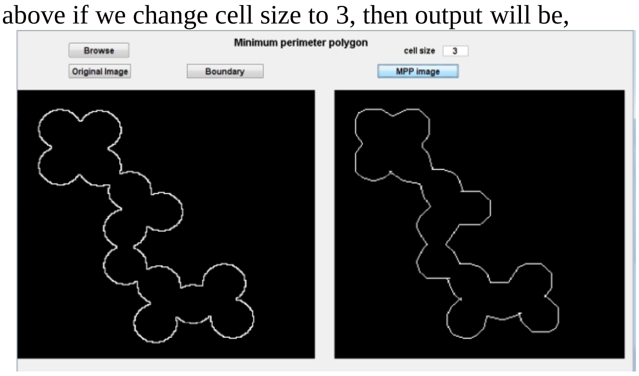


Fig. 12: MPP output with reduced step size

VIII. CONCLUSION

In this paper, three image representation techniques are implemented as a graphical user interface. From the result it can be observed that MPP algorithm provide accurate representation of original image. By decreasing step size accuracy can be increased. Chain code can be used for image transmission from one place to other. The accuracy of the resulting code representation depends on the spacing of the sampling grid. It can also be used to store images.

REFERENCES

[1] MATLAB Product Help
[2] Digital Image Processing –Third Edition by Rafael C. Gonzalez and Richard E. Woods
[3] Digital Image processing using MATLAB by Gonzalez ,Woods and Eddins
[4] http://www.mathworks.com/help/techdoc/creating_guis/bqz79mu.html
[5] http://www.mathhworks.com/help/techdoc/creating_guis/bqzzla9-2.html
[6] Power point presentation on Image Representation and Description – Boundary and Region Descriptors by Dr. Jiajun Wang School of Electronics and Information Engineering Soochow University