A Chain Code Approach for Recognizing Basic Shapes

Dr. Azzam Talal Sleit (Previously, Azzam Ibrahim)
azzam_sleit@yahoo.com

Rahmeh Omar Jabay
King Abdullah II for Information Technology College
University Of Jordan, Amman, Jordan
Rahmeh_jabay@yahoo.com

ABSTRACT

Shape recognition is a major problem in image understanding and computer vision. Applications of shape recognition can be found in many areas, such as, medicine, space exploration, manufacturing, defense and many others. Shape recognition involves three primary issues, shape representation, shape similarity measure and shape indexing. Among them, shape representation is the most important issue in shape retrieval. This paper deals with the representation and recognition of rectangle, square, triangle, hexagon, pentagon, circle and a line based on chain codes and other features. A system for educating children on basic shape understanding was developed using the proposed algorithms.

Key Words: Chain code, Corner, Compactness

1. Introduction

Two-dimensional shapes can be described into two different ways. The first method uses shape boundary and features such as boundary length, compactness. The other method is to describe shapes through the region they occupy such as areas, or skeletons. Shape features are classified into boundary features and region features. Boundary features are extracted from the boundary of the shape like perimeter and corners, while regional features are extracted from the region occupied by the shape such as the area.

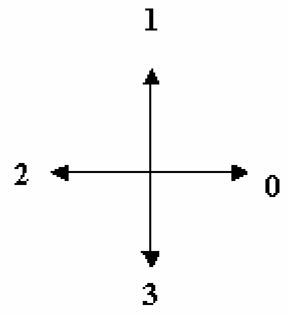
Basic shapes have certain features that make them recognized even if they share the same properties. Every shape has its own unique property. For example, rectangle and square have the same count of corners and identical angles.

2. Chain Codes and Compactness

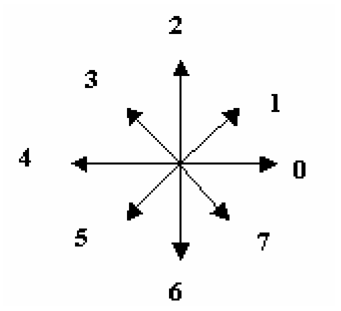
The first approach for representing digital boundary was introduced by Freeman in1961 using chain codes. Freeman states that in general any representing scheme must satisfy 3 objectives:

1. must faithfully preserve information of interest
2. permit compact storage
3. must facilitate any required process

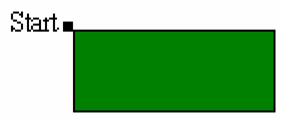
Chain codes are used to represent the boundary of an object composed of pixels of regular cells by connected sequence of straight-line segments of specified length and direction. The object is traversed in clockwise manner. As the boundary is traversed, the direction of each chain segments is specified using the following numbering scheme:



The above figure shows the direction of each segment if applied in 4-connected neighborhood, while 8-connected neighborhood can be numbered as follows:



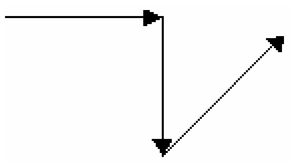
Many corner detection algorithms were developed. One approach was based on freeman chain codes which took into account the boundary of an object where direction is changed from one chain to another, where a corner is formed. Chain codes are widely used because they provide representation of objects that can be used in recognition. They also provide good compression because each segment has to be encoded with 2 or 3 bits. Some features can also be computed as area and perimeter.



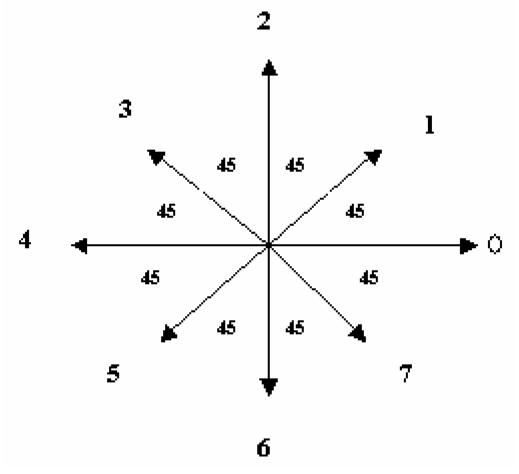
Tracing the above shape from the start point using 4-connected scheme moving in clockwise direction will produce the following code sequence: 000000,333,222222,111

Another advantage of chain code is that it is translation invariant but not as well for rotation and scaling. Rotation can be resolved by taking difference chain code while scaling by addressed by changing the size of the sampling grid which the shape overlays on. Each segment of the chain code has direction. If we want to move from one segment to another, the angle magnitude will change. For example, the following segment has the chain code sequence 00; no change in direction:

However, the following segment has the chain code sequence: 061 and direction of angles: 0, 90, 45 degree.



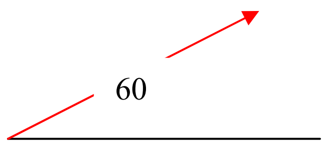
Between every two consecutive directions in the following numbering scheme, the angle magnitude is 45



The Angle = 90 if (A, C) belongs to {(0,2), (0,6), (2,0), (6,0), (4,2), (4,6), (2,4), (6,4)(7,1), (1,7), (7,5), (5,7) (3, 1), (1,3), (3,5), (5,3)}

Where A, C are direction of 2 segments producing a vertex.

Chain code is very sensitive to noise. Line segments that have directions in other than 0, 45, and 90,135 may experience corners.

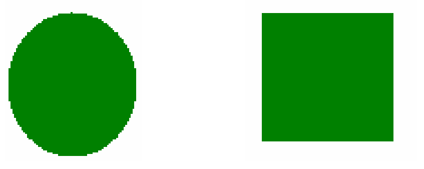


The red line of 60-degree direction has the chain code sequence: 10101010101. To resolve sensitivity problem, we may apply smoothing techniques on the chain code sequence to force line segments to have equal codes. Another problem with line segments occurs when they are more than 1 pixel thick or when shapes are not filled. Such problems can be resolved by

applying a thinning algorithm before computing the chain code sequence of a shape.
Another shape feature is Compactness, Complexity or Circularity. Compactness is defined by taking the perimeter, area of a shape and applying them to the following formula:

$$\frac{(\text{Perimeter})^2}{4\pi\ \text{Area}} \quad \ldots\ldots\ldots \quad (1)$$

Circle has a minimum compactness other shapes with complex boundaries have larger compactness. A Circle shape tends to have compactness equal to 1 while square has compactness of $4/\pi$.



Very Compact Not Compact

3. Algorithm and Experimental Results

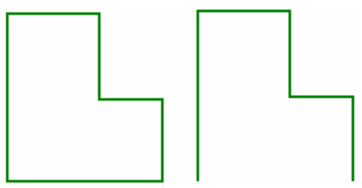
A system was built to be utilized for educating children about shapes taking into consideration the previous points. The following procedure is used to differentiate between basic shapes.

Step1: Create a binary image from the original image, possibly using some thresholding and edge connection techniques

Step 2: Save the image in matrix of 0's for back ground and 1's for the object. While this process is in progress, we compute the area of the object by adding for each iteration

Step3: Compute chain code by scanning the image to find the starting pixel of the object. From that pixel, we traverse the boundary and decide directions and save them as an array or list. This step is repeated until we reach the end pixel

```
If end pixel not equal start pixel then
        Shape is not closed
If end pixel =start pixel then
        We have a closed shape
Proceed to step 4
```



Closed Opened

Step4: Calculate the perimeter by summing up the elements of the chain code sequence

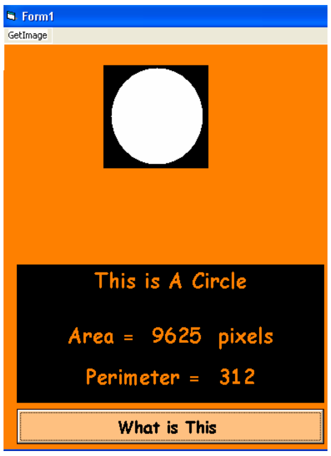
Step5: Compute the compactness by of the shape by applying equation (1). Check these values to find out if shape is circle otherwise go to step 6

Step 6: Count the corner based on direction change and save the elements for each corner. Elements are the 2-chain segments where direction changed. Check the angle in each corner. Depending on the number of corners and start and ending pixels, we can guess the number of sides that form the shape according to the following:

```
If corner count=0 then
        The object is a line

If corner count =1
        2-connected lines

If corner count=2
        3-connected lines

If corner count=3 and start pixel=endpixel
        Shape= Triangle

If corner count=4 and startpixel=endpixel
        Shape=Rectangle or square or any other 4-side shape
        To decide the shape we have to find if the angles are 90 degree.
```

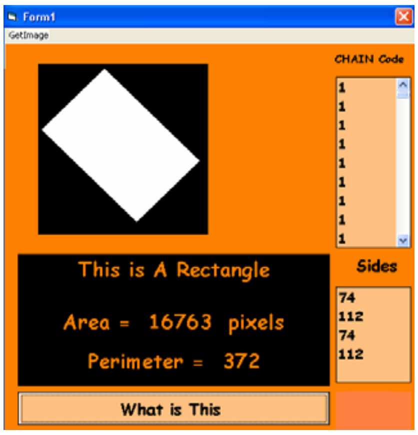
To differentiate between rectangle and square, check the length of each side per shape based on the chain code sequence.

A system was created based on the above algorithm for shape recognition and children education. The system checks the images and displays as text what kinds of shapes exist in the image. Also, it displays features such as area, perimeter, list of chain codes and count per code. In this experiment, a circle tends to have compactness between 0.80-0.82, while square and rectangle have approximately 1.25. The compactness for hexagon is close to circle. The following figures show the outcome of applying the algorithms for circle, rectangle, square pentagon, hexagon, triangle and line.



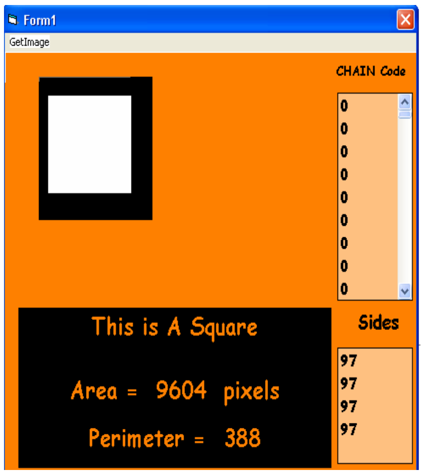
```
This is A Circle
Area = 9625 pixels
Perimeter = 312
```

Compactness =0.81



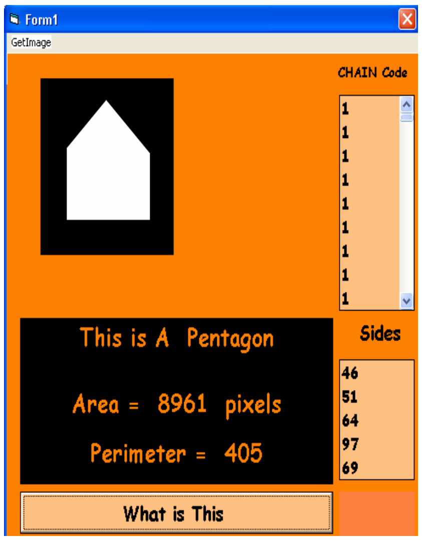
```
1
1
1
1
1
1
1
1
1
This is A Rectangle
Area = 16763 pixels
Perimeter = 372
Sides
74
112
74
112
```

Compactness =0.66



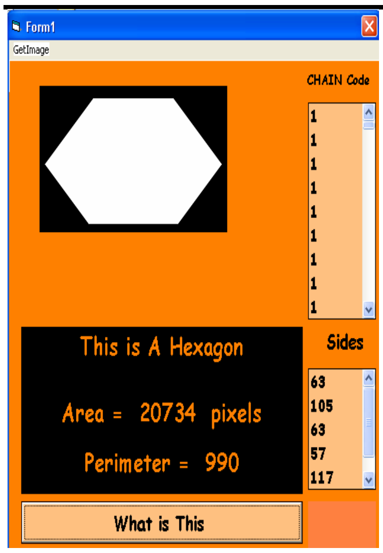
```
0
0
0
0
0
0
0
0
0
This is A Square
Area = 9604 pixels
Perimeter = 388
Sides
97
97
97
97
```

Compactness =1.25



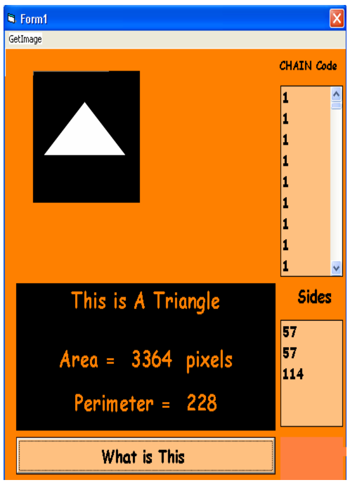
```
1
1
1
1
1
1
1
1
1
This is A Pentagon
Area = 8961 pixels
Perimeter = 405
Sides
46
51
64
97
69
```

Compactness =0.95



```
1
1
1
1
1
1
1
1
1
This is A Hexagon
Area = 20734 pixels
Perimeter = 990
Sides
63
105
63
57
117
```

Compactness =0.82



```
1
1
1
1
1
1
1
1
1
```

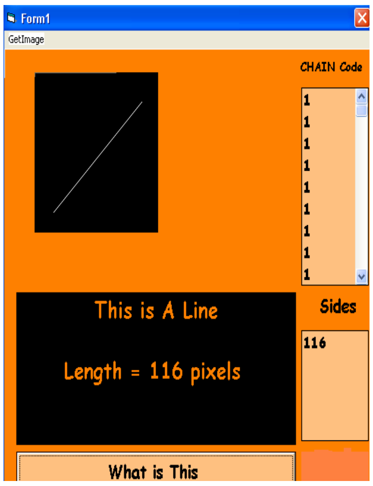
This is A Triangle

Area = 3364 pixels

Perimeter = 228

```
57
57
114
```

Compactness =1.23



```
1
1
1
1
1
1
1
1
1
```

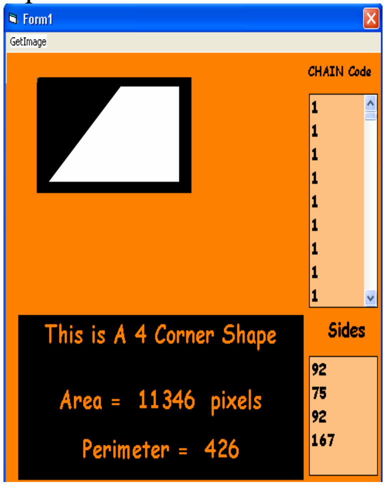
This is A Line

Length = 116 pixels

```
116
```

Compactness =9.16

The program can also check 4-side shapes and differentiate between rectangles and squares.



```
1
1
1
1
1
1
1
1
1
```

This is A 4 Corner Shape

Area = 11346 pixels

Perimeter = 426

```
92
75
92
167
```

Compactness =1.27

4. Conclusion

This paper demos a system intended for children education to recognize basic shapes based on chain codes. It is suitable for simple shapes composed of straight-line segments and specified direction. Rotation and scaling are not an issue because the proposed algorithm counts corners and checks angles. The algorithm and corresponding implementation demonstrated merits in recognizing a large number of shapes.

References

[1]" Digital Image Processing "2nd Edition by Gonzalez and Woods, Prentice Hal, 2002

[2]" Digital Image Processing Algorithms and Applications" by Ioannis Pitas, Wiley 2000

[3] Miroslav Trajkovic and Mark Hedley, "Fast Corner Detection_", Image and Vision Computing, 1998

[4] Herbert Freeman,"Computer Processing of Line-Drawing Images" March 1974

[5] ILYA LEVNER, Shape Detection Analysis and Recognition, 2000

[6] Abdel-Badeeh M. Salem, Adel A. Sewisy, Usama A. Elyan, "a vertex chain code for image processing, " 2005