HCI/ComS 575X: Computational Perception
Homework 2
Out: Wed Feb 4, 2009
Due: Wed Feb 18, 2009
Submission format: post code and images on your section of the class wiki page.

Problem 1 - Basic Mathematical Morphology

In this problem you are asked to replicate a result similar to the ones given in the "Mathematical Morphology" paper by Haralick and Shapiro (i.e., Chapter 5 in "Computer and Robot Vision", Addison-Wesley, 1993). Using the morphological operators provided by Matlab and OpenCV write a short program that reads the image and performs the desired effect on the image. Solve (A) and (B) in both Matlab and OpenCV and post your code to the course wiki.

(A) For the first image you're required to separate out the large and small circles.

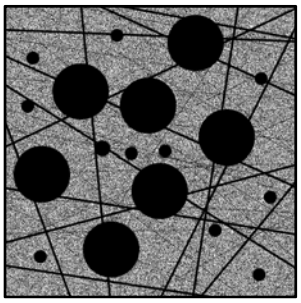
Figure 1: Original image

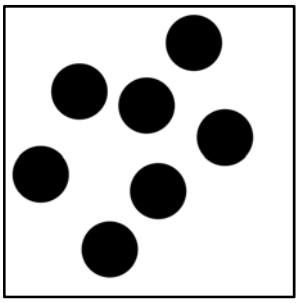
Figure 2: Segmented large circles

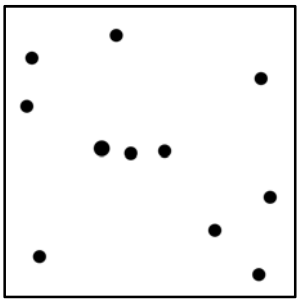
Figure 3: Segmented small circles

(B) For the image of the circle, pendulum, and ring: separate the shape into its three basic components.

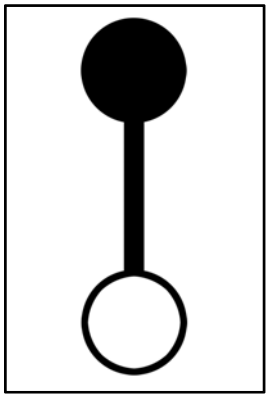
Figure 4: Original image

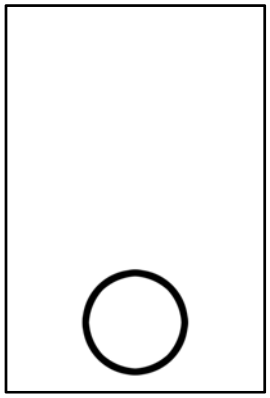
Figure 5: Ring segment

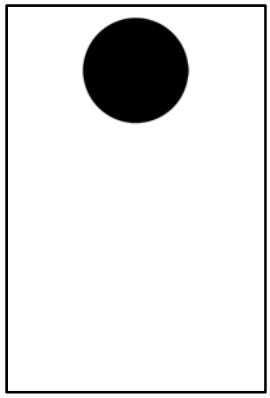
Figure 6: Circle segment

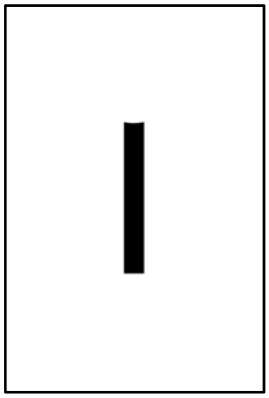
Figure 7: Pendulum segment

Problem 2 – PostNET Barcode Recognition

This problem can be completed in Matlab or OpenCV. Your grade does not depend on your choice.

In this problem you will be asked to detect and decode PostNET barcodes from a set of images. Images are available from the course wiki and range in difficulty from easy to hard. It is expected that your code will solve all 5 of the easy images, and at least 4 of the medium images. There is an optional extra credit option if you can scan or take a picture of some of your own mail and then run those images trough your program and post the correct results on the wiki. It is still encouraged to try and fine tune your code to solve all 10 of the ones given to you and try some of your own images.

Value	Encoding
0	
1	
2	
3	
4	
5	
6	
7	
8	
9	

Table 1: Encodings

PostNET codes can be found primarily on mail handled by the USPS (United States Postal Service). These barcodes are added to assist in the automated routing of mail by zip-code.

Figure 8: PostNet barcode

These barcodes are simple to detect and decode. They code a 5-digit zip-code and can support the additional sub zip-code routing numbers as well as a primitive form of error checking. Bars that makeup the barcode are either half or full bars.

Every PostNET barcode starts and ends with a full bar. This is commonly referred to as the guard rail. The remaining bars are grouped into sections of 5, each representing a single digit in the code. **For the purposes of this homework we will only use barcodes with 52 bars.**

The first 9 digits of the barcode make up the actual values we are looking for, and the final digit in the barcode is used for error correction. Summing all the digits within the barcode should result in a multiple of 10. This last digit is used bring the sum to this multiple.

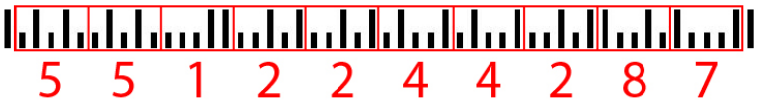

Figure 9: Decoded PostNET barcode

Example: if the encoded barcode was 55122-4428 (as in figure 9), then we could add 5+5+1+2+2+4+4+2+8=33 and the tenth digit of the barcode should be 7 as 33+7=40 and 40 mod 10 = 0. If we decoded some other barcode, and then summed all ten digits, we should be able to take it mod ten, and get zero. We can use this simple math as a means of error checking to ensure we decoded the barcode properly.

When decoding, simply highlight and label the barcode sections as show in figure 9.

For more information regarding PostNET barcodes and a sample generator, see the following links:
http://en.wikipedia.org/wiki/POSTNET
http://www.barcoding.com/upc/

Problem 3 – DataMatrix Barcode Recognition

This problem can be completed in Matlab or OpenCV. Your grade does not depend on your choice.

In this problem you will be required to find and decode two-dimensional DataMatrix barcodes. In a DataMatrix barcode, data is represented as a grid of blocks. Each block or cell represents a "1" or a "0".

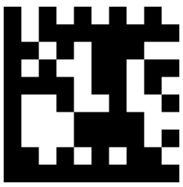

Figure 10: Sample DataMatrix

While the size of a DataMatrix barcode can be huge, for the purposes of this assignment, we will only ask that you deal with barcodes for size 10x10. Extra credit will be awarded for anyone who can decode a matrix of variable size.

Structure of DataMatrix barcodes

Each barcode will have a border around to assist in detection of the barcode. The border is the same for every DataMatrix barcode. The left and bottom squares are solid, and the top and right squares have a checker pattern.

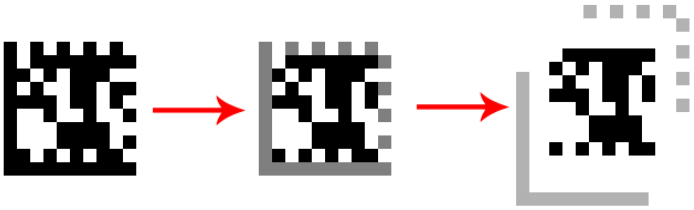

Figure 11: Breakdown of edges

Inside is the data. For the purposes of this assignment you will be asked to read the boarder and the inside data from left to right starting with the top leftmost square. The first square will always be filled in and thus represented as a 1. The top of your output must state the width/height of the matrix (the matrix must be square, so this they should be the same).

Black squares = 1
White squares = 0

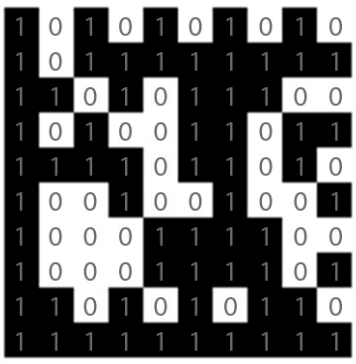

Figure 12: Decoding of DataMatrix (Student output not expected to look like this)

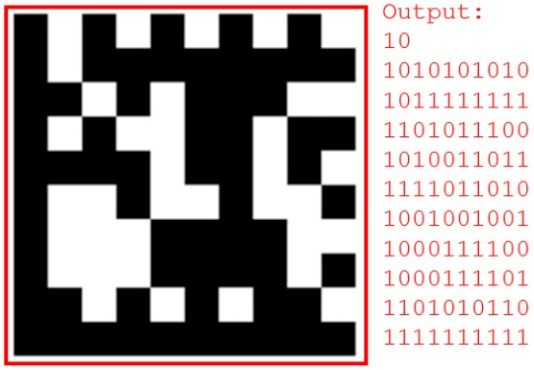

Figure 13: Decoded student output (This is what students are expected to output)

After decoding the barcode into an array, you can then feed it into the supplied program (DataMatrixDecoder.exe) that is available on the wiki homepage. Your program must generate the output file that this program will use.

The first line of the file must be an integer, n, where n is the height and width of the DataMatrix (For most everyone, this will be 10). After the integer, there will be n lines of strings, where each string in n characters long. Each character must be a 1 or a 0. See table 2 for an example of what the output file should be for the above DataMatrix.

Be sure the output is named "input.txt" as failing to do this will cause the decoder to fail. Please also note that version 0.1 of DataMatrixDecoder.exe will only support 10x10 matrixes. An updated version will follow soon that will allow for testing larger output files for the extra credit portion.

```
10
1010101010
1011111111
1101011100
1010011011
1111011010
1001001001
1000111100
1000111101
1101010110
1111111111
```

Table 2: Sample "input.txt" file

Post your code, your decoded DataMatrix barcodes, and decoded messages on the Wiki

For more information regarding DataMatrix barcodes and an online encoder, see these links:

Encoders:

http://datamatrix.kaywa.com/

http://invx.com/

Information:

http://en.wikipedia.org/wiki/Data_Matrix

http://grandzebu.net/index.php?page=/informatique/codbar-en/datamatrix.htm (no info you really need to know, but kind of interesting)

http://www.libdmtx.org/ (library I used to create the decoder. Again, you do not need to use or know, just interesting)