Homework #3
Due Th. 3/27

You must turn in your code as well as output files. Please generate a report that contains the code and ouput in a single readable format.

Getting Started

- You may want to download Irfanview image viewing software. It handles pretty much any image type, lets you convert, and provides batch processing.

  `http://www.irfanview.com/`

- Download the sample images from the class website.

  `http://www.ee.unlv.edu/~b1morris/ecg782/hw/hw03`

1. (Sonka P13.10)
2. Give the structuring element and morpohological operation(s) that produced each of the results shown in Fig1. Clearly mark the origin of your structing element. The dashed lines show the boundary of the original set and are included only for references. Note that in (d) all corners are rounded.

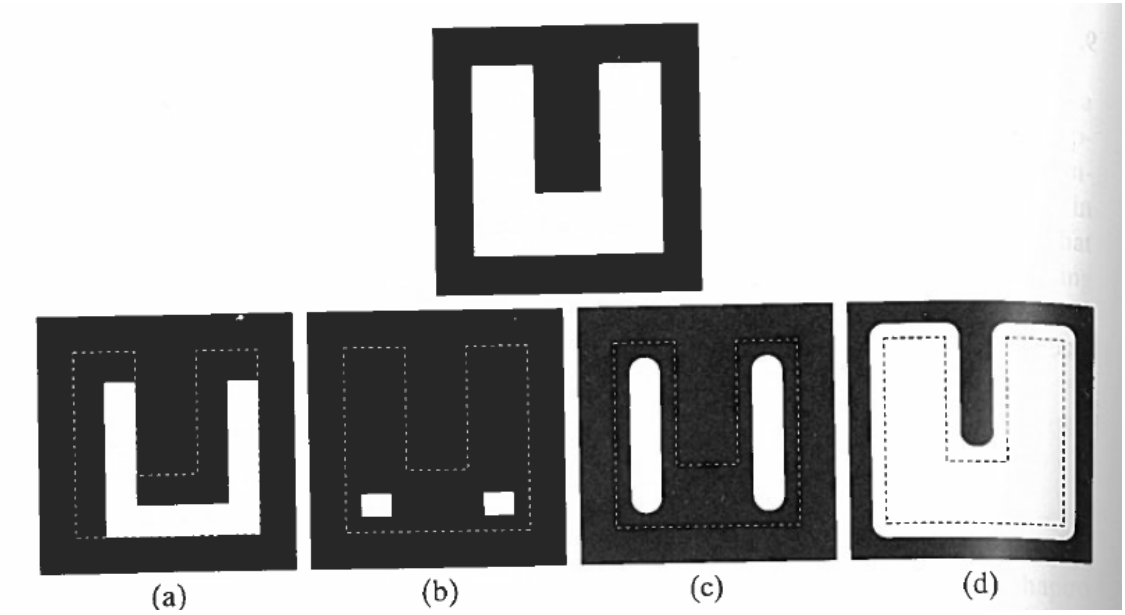

Figure 1: Problem 2

3. Sketch the result of the following morphological operations. Use the image sets in Fig. 2 where the black dot indicates the origin or the structuring element.

   (a) $(A \ominus B^4) \oplus B^2$
   (b) $(A \ominus B^1) \oplus B^3$
   (c) $(A \oplus B^4) \ominus B^3$
   (d) $(A \oplus B^3) \ominus B^2$

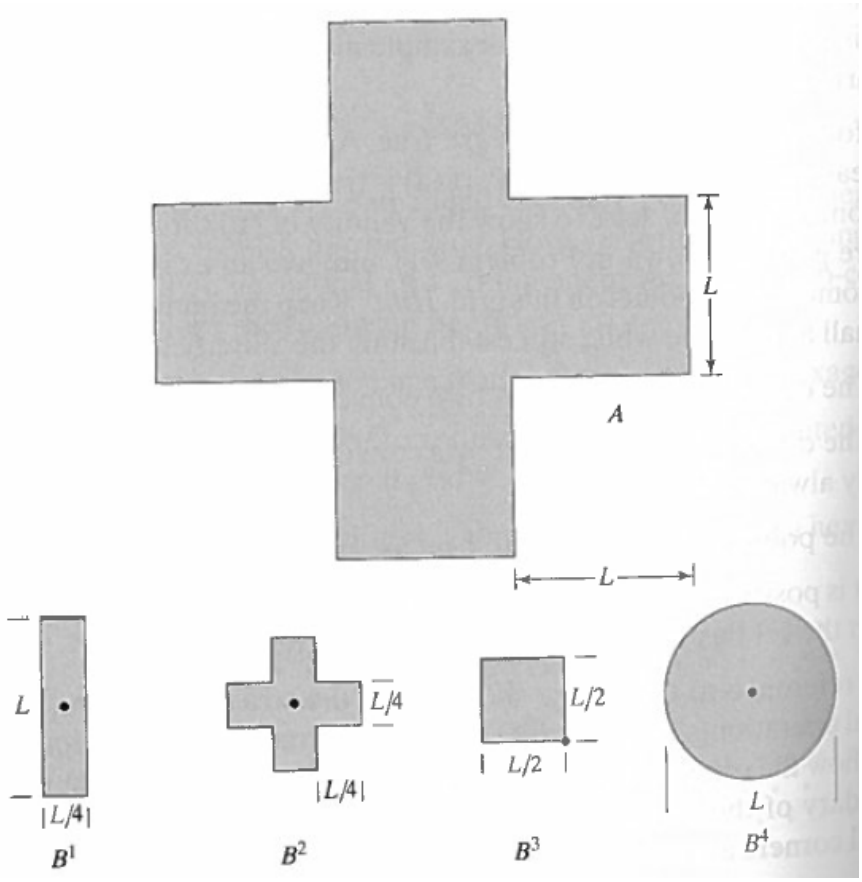

Figure 2: Problem 3

4. Correlation Detection

   This problem requires the UIUC Car Detection Database found at

   `http://cogcomp.cs.illinois.edu/Data/Car/`.

   Notice the images are in .pgm format and cannot be natively displayed on Windows.

   (a) Write a function `corr_detect.m` that will take an image template to compare with a test image and a $0 \leq \tau \leq 1$ threshold for object detection. The function should return a list of the bounding box $[x, y, w, h]$ locations that are above the match threshold τ. Notice the detector will have multiple responses in the same area. This is a common occurrence for object detection algorithms because of similarity of bounding boxes with small pixel displacements. Design a non-maximum supression (NMS) routine to reduce these effects by only keeping the largest response in a local area.

   (b) Test the performance of your detector on the images in the `\TestImages` directory and create an ROC curve using the `trueLocations.txt` file. You should compare the performance using image `pos-1.pgm` and `pos-125.pgm` as the templates. Which template performs better?

   Use the 50% overlap ratio to determine TP or FP as described for the Pascal VOC challenge

   `http://pascallin.ecs.soton.ac.uk/challenges/VOC/pubs/everingham10.pdf`

See "Framework for Performance Evaluation of Face, Text, and Vehicle Detection and Tracking in Video: Data, Metrics, and Protocol " by Kasturi et. al in 2009 for more information about methodolgies for measuring performance of detectors.

(c) Modify `corr_detect.m` to handle both directions of the template to handle orientations that are from left-right and right-left. Explain your technique to pool the results from each template orientation.

5. Hough Transform

(a) Study the Matlab function `hough.m`. Compute the Hough transform of the `city.jpg` image. Display the Hough accumulator image and the original image with the top 5 lines as an overlay. Each overlay line should be a different color and a legend should be included.

(b) Write your own Hough transform implementation for circle detection. The function should take an image and radius as input. Test your function on the `quarters.bmp` image. Display the accumulator image and the original image with the top 3 circles as an overlay.

Bonus: i) Use gradient magnitude accumulation instead individual count. ii) Upgrade the detector to find circles of different sizes. You may test results on `us_silver_coins.jpg`.