

EE 569: Homework #2

Issued: 2/3/2017 Due: 11:59PM, 2/26/2017

General Instructions:

1. Read *Homework Guidelines* and *MATLAB Function Guidelines* for the information about homework programming, write-up and submission. If you make any assumptions about a problem, please clearly state them in your report.
2. You need to understand the USC policy on academic integrity and penalties for cheating and plagiarism. These rules will be strictly enforced.

Problem 1: Geometric Image Modification (40%)

In this problem, you will need to apply geometric modification and spatial warping techniques to do some interesting image processing tricks. Please note that during these operations, you may need to solve some linear equations to get the matrix parameters. **ONLY** for this part, you can use some offline tools, such as Matlab or online equation solver.

(a) Geometrical Warping (Basic: 10%)

Design and implement a spatial warping technique that transforms an input square image into an output image of a diamond shape. An example is given in Figure 1.



Figure 1: Warp the original image to a diamond-shaped image



Figure 2: cup1.raw and cup2.raw images

Apply the same developed spatial warping algorithm to both 500x500 color images, *Cup1* and *Cup2* in Figure 2.

(b) Puzzle Matching (Basic: 15%)

You are given two images: *Hillary* and *Trump* with size 512x512, and two puzzle pieces in one image *Pieces*, each of which is shown in Figure 3 below. Apply geometric transformations to each puzzle piece and put them back to the correct positions. The holes in *Hillary* and *Trump* images are all of size 100x100.



Figure 3: Component images: (a) Pieces (b) Hillary (c) Trump

Write a program to implement the hole-filling algorithm. Print out the result for both images after hole-filling.

One possible way of doing this is:

1. Find the coordinates of corners in each puzzle piece image. This must be done by your program. You are NOT allowed to do this manually.
2. Design a generic geometric transform on each puzzle piece. Here you may need to combine multiple operations (e.g. translation, rotation, scaling, etc). After the transform, each of your puzzle pieces should be a square image with its sides aligned to the horizontal and vertical axes. For up-scaling, you may need to implement an interpolation function. Drop redundant pixels when down-scaling.
3. Find the coordinates of holes in *Hillary* and *Trump* images. This must be done by your program.
4. Put the transformed puzzle piece back to the original image. You should use your program to achieve this task.

Note: Bilinear interpolation could be needed to generate pixel values at fractional positions.

(c) Homographic Transformation and Image Overlay (Advanced: 15%)

One can use the homographic transformation and the image overlap techniques to synthesize interesting images. One example is shown in Figure. 4, where the left two images are seed images and the right one is the desired output. The field image is the host image and the *Tartans* image is the embedded text image. Note that the black region outside the yellow contour of the embedded image is removed.

The homographic transformation procedure is stated below. Images of points in a plane, from two different camera viewpoints, under perspective projection (pin hole camera models) are related by a homography:

$$P_2 = HP_1$$

where H is a 3x3 homographic transformation matrix, P_1 and P_2 denote the corresponding image points in homogeneous coordinates before and after the transform, respectively. Specifically, we have

$$\begin{bmatrix} x'_2 \\ y'_2 \\ w'_2 \end{bmatrix} = \begin{bmatrix} H_{11} & H_{12} & H_{13} \\ H_{21} & H_{22} & H_{23} \\ H_{31} & H_{32} & H_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} \text{ and } \begin{bmatrix} x_2 \\ y_2 \\ w_2 \end{bmatrix} = \begin{bmatrix} \frac{x'_2}{w'_2} \\ \frac{y'_2}{w'_2} \\ \frac{1}{w'_2} \end{bmatrix}$$

To estimate matrix H , you can proceed with the following steps:

- Fix $H_{33} = 1$ so that there are only 8 parameters to be determined.
- Select four point pairs in two images to build eight linear equations.
- Solve the equations to get the 8 parameters of matrix H .
- After you determine matrix H , you can project all points from one image to another by following the backward mapping procedure and applying the interpolation technique.

Implement above homographic transformation steps to generate the desired result as shown in Figure 4.

Now, let us keep the same host image and the same overlay region, apply the algorithm to the embedded text image *Trojans* in Figure 5 to generate a new image overlay. Show the result and make discussion on the performance.

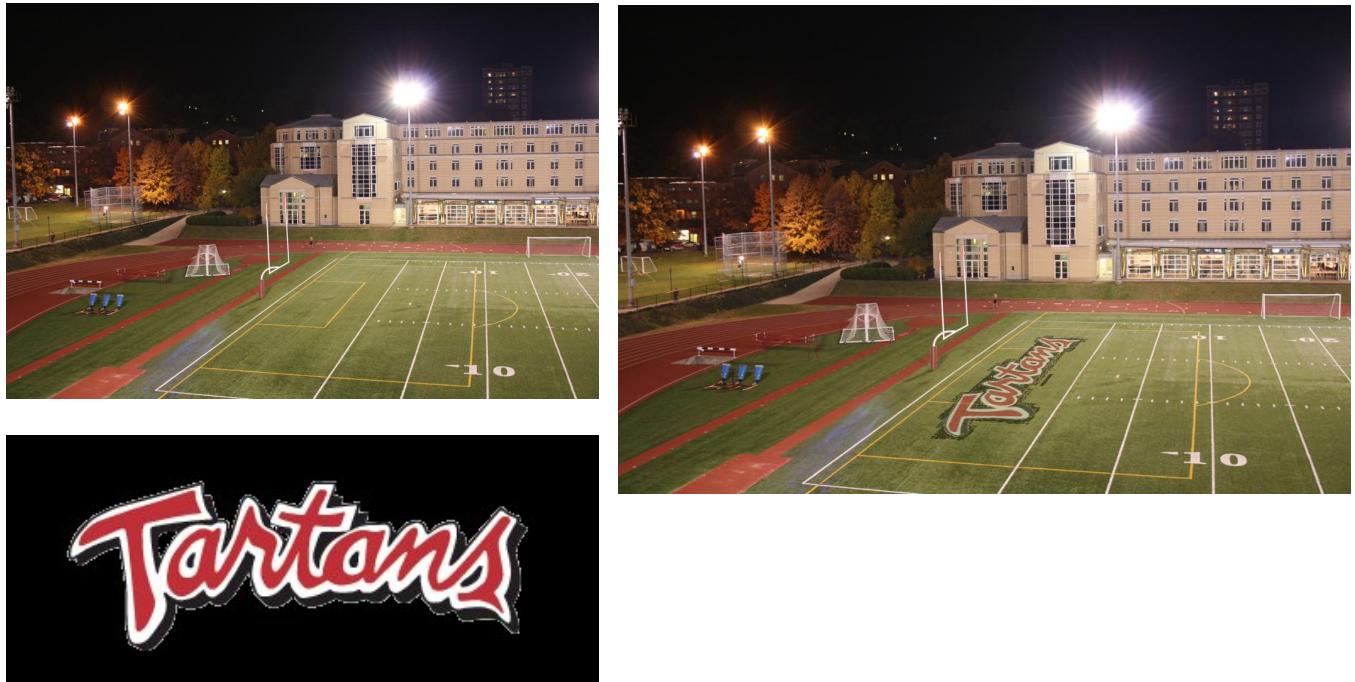


Figure 4: An example of homographic transformation and image overlay



Figure 5: The Trojans text image.

Problem 2: Digital Halftoning (30 %)

There are 256 gray levels for pixels in Figure 6. Please implement the following procedures (dithering matrices and error diffusion) to convert *Man* image to a binary image. In the following discussion, $F(i,j)$ and $G(i,j)$ denote the pixel of the input and the output images at position (i, j) , respectively. **Compare the results obtained by different algorithms in your report.**



Figure 6: Man.raw

(a) Dithering Matrix (Basic: 15%)

Convert the 8-bit *Man* image in Figure 6 to a half-toned image using the dithering method. Dithering parameters are specified by an index matrix. The values in an index matrix indicate how likely a dot will be turned on. For example, an index matrix is given by

$$I_2(i,j) = \begin{bmatrix} 0 & 2 \\ 3 & 1 \end{bmatrix}$$

where 0 indicates the pixel most likely to be turned on, and 3 is the least likely one. This index matrix is a special case of a family of dithering matrices first introduced by Bayer.

The Bayer index matrices are defined recursively using the formula:

$$I_{2n}(i,j) = \begin{bmatrix} 4 * I_n(x,y) & 4 * I_n(x,y) + 2 \\ 4 * I_n(x,y) + 3 & 4 * I_n(x,y) + 1 \end{bmatrix}$$

The index matrix can then be transformed into a threshold matrix T for an input gray-level image with normalized pixel values (*i.e.* with its dynamic range between 0 and 255) by the following formula:

$$T(x,y) = \frac{I(x,y) + 0.5}{N^2} \times 255$$

where N^2 denotes the number of pixels in the matrix. Since the image is usually much larger than the threshold matrix, the matrix is repeated periodically across the full image. This is done by using the following formula:

$$G(i,j) = \begin{cases} 1 & \text{if } F(i,j) > T(i \bmod N, j \bmod N) \\ 0 & \text{otherwise} \end{cases}$$

where $F(I,j)$ and $G(I,j)$ are the normalized input and output images.

There are different ways to construct a dithering matrix. Another example for a 4x4 matrix is:

$$A_4(i,j) = \begin{bmatrix} 14 & 10 & 11 & 15 \\ 9 & 3 & 0 & 4 \\ 8 & 2 & 1 & 5 \\ 13 & 7 & 6 & 12 \end{bmatrix}$$

Answer the following questions.

1. Construct $I_2(i,j)$ and $I_8(i,j)$ Bayer index matrices and apply them to the *Man* image.
2. Compare the above $A_4(i,j)$ matrix with $I_4(i,j)$ Bayer Matrix and discuss their differences. Apply both and compare the results.
3. If a screen can only display FOUR intensity levels, design a method to generate a display-ready *House* image. Show your best result in gray-scale with four gray-levels (0, 85, 170, 255) and explain your design idea and detailed algorithm.

(b) Error Diffusion (Basic: 15%)

Convert the 8-bit *Man* image to a half-toned one using the error diffusion method. Show the outputs of the following three variations, and discuss these obtained results. Compare these results with dithering matrix. Which method do you prefer? Why?

1. Floyd-Steinberg's error diffusion with the serpentine scanning, where the error diffusion matrix is:

$$\frac{1}{16} \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 7 \\ 3 & 5 & 1 \end{bmatrix}$$

2. Error diffusion proposed by Jarvis, Judice, and Ninke (JJN), where the error diffusion matrix is:

$$\frac{1}{48} \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 7 & 5 \\ 3 & 5 & 7 & 5 & 3 \\ 1 & 3 & 5 & 3 & 1 \end{bmatrix}$$

3. Error diffusion proposed by Stucki, where the error diffusion matrix is:

EE 569 Digital Image Processing: Homework #2

$$\frac{1}{42} \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 8 & 4 \\ 2 & 4 & 8 & 4 & 2 \\ 1 & 2 & 4 & 2 & 1 \end{bmatrix}$$

Describe your own idea to get better results. There is no need to implement it if you do not have time. However, please explain why your proposed method will lead to better results.

Problem 3: Morphological Processing (30%)

In this problem, you will implement three morphological processing operations: shrinking, thinning, and skeletonizing. A pattern table (patterntables.pdf) is attached for your reference. Please show outputs for all following parts in your report and discuss them thoroughly. Please state any assumptions you make in your solution.

(a) Shrinking (Basic: 7%)

Please apply the ‘shrinking’ filter to the squares image (squares.raw). Please implement the filter, and discuss your solution for the following questions:

- Count the total number of squares in the image.
- How many different square sizes are present in the image? What is the frequency of these square sizes? (Hint: Plot the histogram of the square size with respect to frequency.)

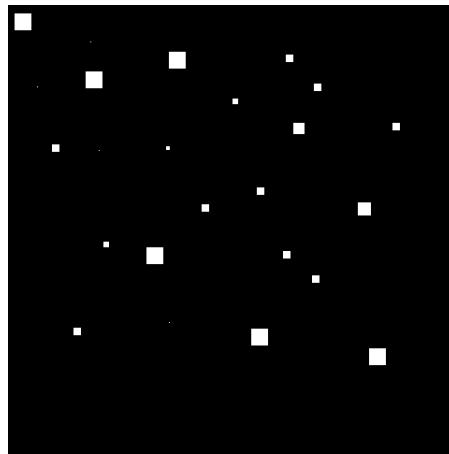


Figure 7: squares.raw

(b) Thinning (Basic: 7%)

Please apply the ‘thinning’ filter to the letterE image (letterE.raw) and show your result.

(c) Skeletonizing (Basic: 8%)

Please apply the ‘skeletonizing’ filter to the letterE image (letterE.raw) and show your result.

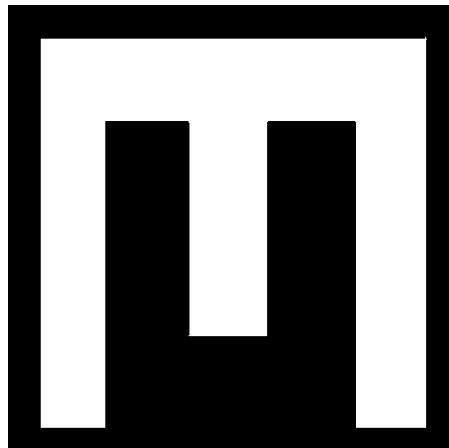


Figure 8: letterE.raw

(d) Counting game (Advanced: 8%)

Figure 3 (see below) is target image with circle and square objects (the background is black and the objects are white). Some of these objects have one or more holes in them. You need to design an algorithm that uses morphological and logical operations to solve and answer the questions below. You may use any of these operators that you learned in class or ones that you invent, but all of them must be detailed in your report with respect to how they operate and how their results help you solve the problem. In your report, discuss your algorithm, results, and analysis in detail. You should also submit your MATLAB or C/C++ programs along with your source code submission. State any assumptions you make for this problem.

Note: Do NOT use any built-in morphological processing functions or logical operators Note: Manual counting of objects is not permitted; your code must automatically do this.

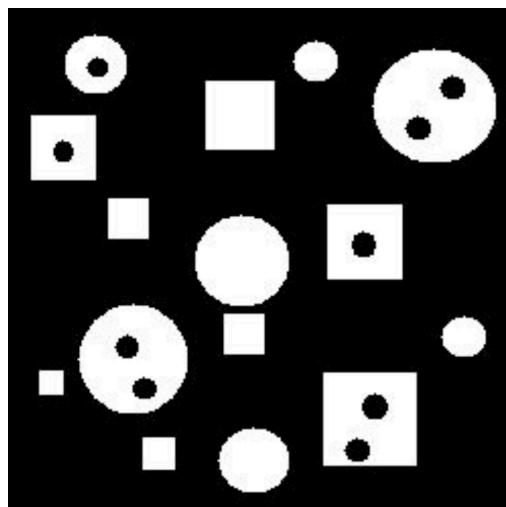


Figure 9: board.raw

- Find the total number of white objects in the image
- Find the total number of holes (black circular holes within white objects) in the image.
- Find the total number of white square objects (with or without holes) in the image.
- Find the total number of white circle objects (with or without holes) in the image.

Appendix:**Problem 1: Geometric image modification and image warping**

Field.raw	972x648	24-bit	color(RGB)
Tartans.raw	350x146	24-bit	color(RGB)
Trojans.raw	350x197	24-bit	color(RGB)
Hillary.raw	512x512	24-bit	color(RGB)
Trump.raw	512x512	24-bit	color(RGB)
Piece.raw	500x500	24-bit	color(RGB)
Cup1.raw	500x500	24-bit	color(RGB)
Cup2.raw	500x500	24-bit	color(RGB)

Problem 2: Digital halftoning

Man.raw	512x512	8-bit	gray
---------	---------	-------	------

Problem 3: Morphological Processing

Squares.raw	480x480	8-bit	grag
LetterE.raw	480x480	8-bit	grag
Board.raw	480x480	8-bit	grag

Reference Images

All images in this homework are from Google images [1] or the USC-SIPI [2] image database.

Reference

- [1] <http://images.google.com/>
- [2] <http://sipi.usc.edu/database/>