ECG 782 Homework #2

Minsung Cho

2021/02/1

1. (GW 6.17) Perform the operation in Matlab(Python) for parts (b) and (c).

    - (b) Suggest an automated procedure for coding the water in Fig. 6.27 in a bright blue color.

      Because the red value of water is very low, thresholding with a lower value of red as black and the rest as white can split the image into riverbanks and the rest.

Figure 1: 6.17 (b) output image

- (c) Suggest an automated procedure for coding the predominantly man-made components in a bright red color. [Hint: Work with Fig. 6.27(f)]

  This could be done the same way as part b but the opposite where the higher red values are thresholded as black and the rest is white.

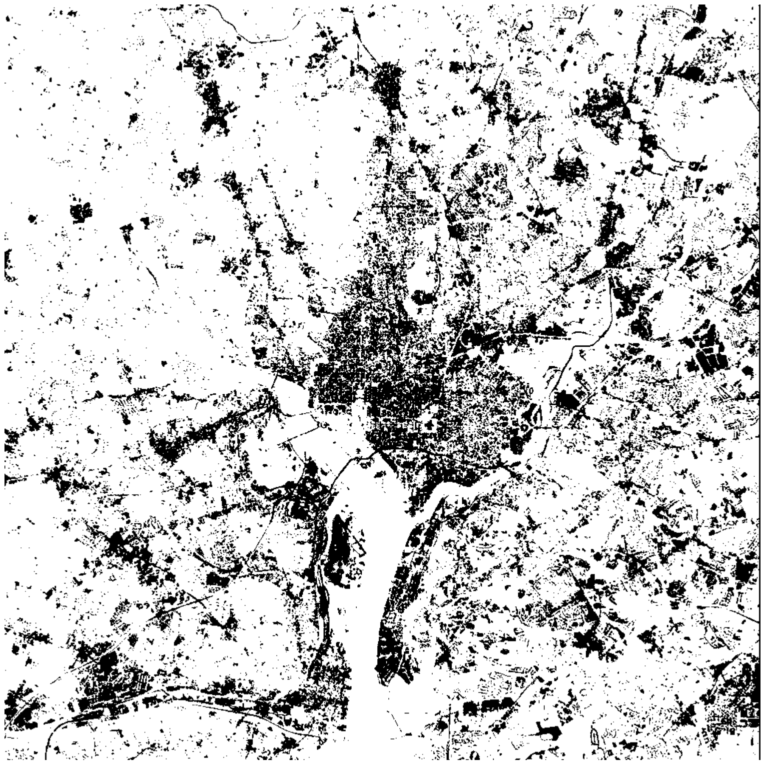


Figure 2: 6.17 (c) output image

2. (GW 6.28) Use Matlab(Python) to plot the surface. *Hint:* ellipsoid.m.

   - 6.28 Sketch the surface in RGB space for the points that satisfy the equation:

     $$D(z,a) = [(z-a)^T C^{-1}(z-a)]^{-1/2} = D_0$$

     where D_0 is a specified nonzero constant. Assume that a = 0 and that

     $$C = \begin{bmatrix} 8 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

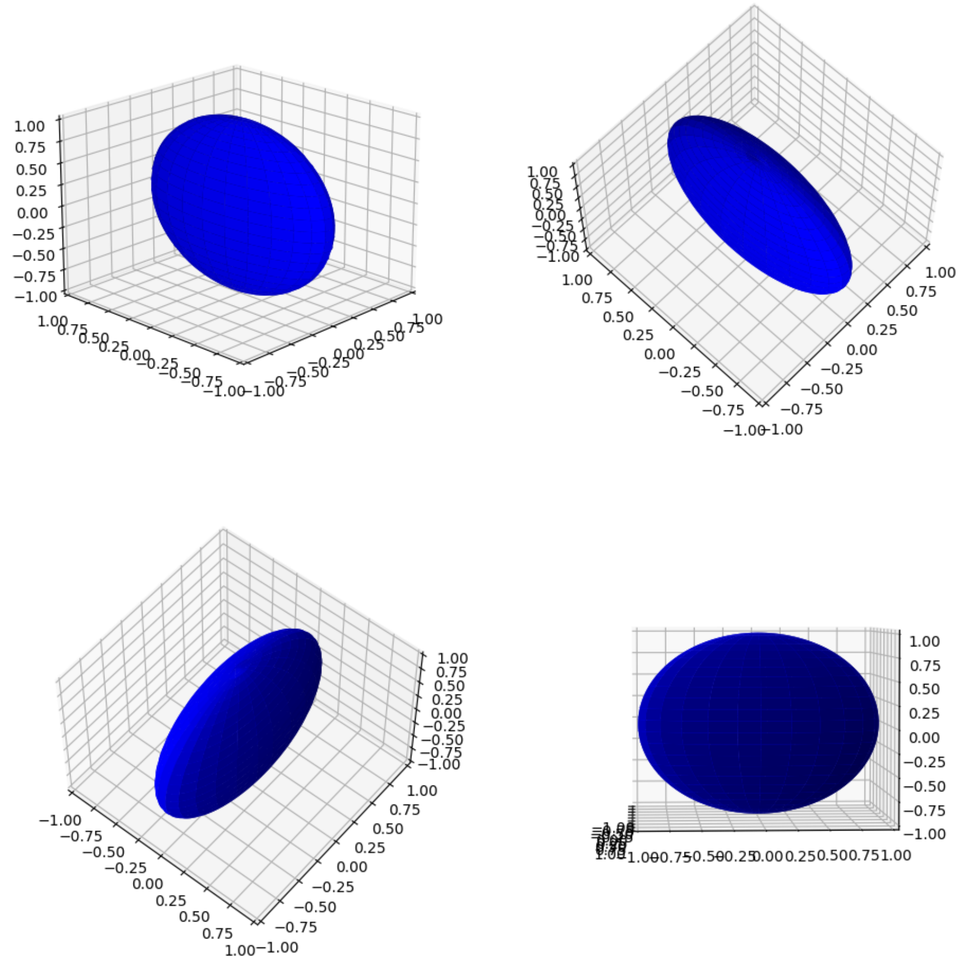


Figure 3: 6.28 output ellipsoid image

3. Prove the validity of the duality expressions:

   (a) $(A \oplus B)^C = (A^C \ominus \hat{B})$

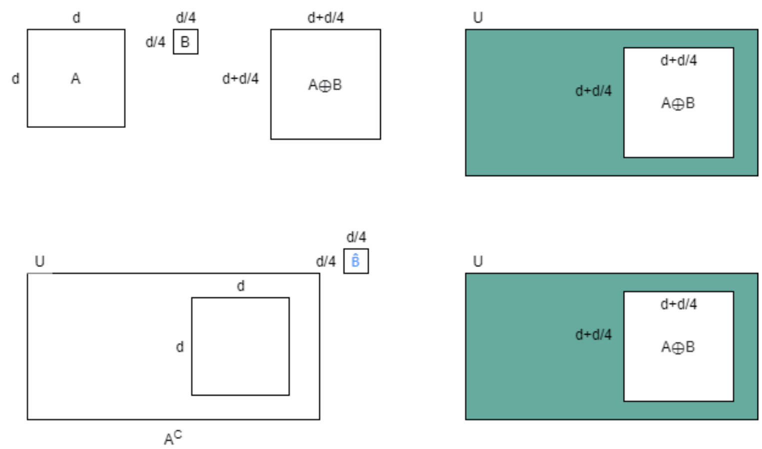


Figure 4: graphical proof

(b) $(A \cdot B)^C = (A^C \circ \hat{B})$

$$(A \cdot B)^c = ((A \oplus B) \ominus B)^c = (A \oplus B)^c \oplus \hat{B} = (A^c \ominus \hat{B}) \oplus \hat{B}$$
$$= A^c \circ \hat{B}$$

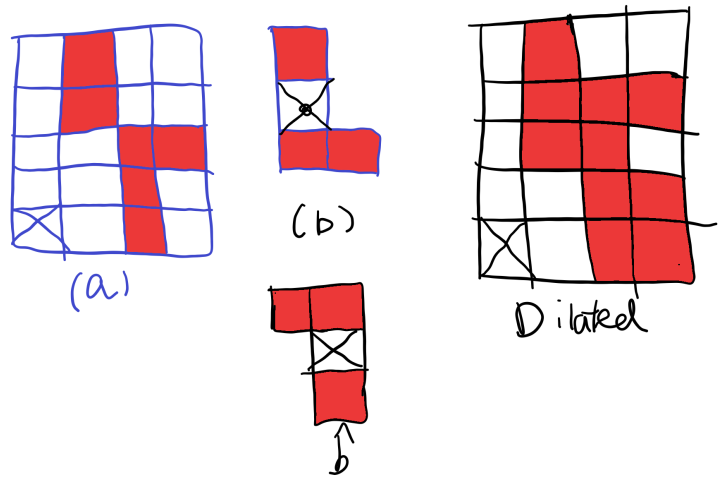
Figure 5: algebraic proof

(c) $(A \circ B)^C = (A^C \cdot \hat{B})$

$$
\begin{aligned}
(A \circ B)^c &= ((A \ominus B) \oplus B)^c \\
&= (A \ominus B)^c \ominus \hat{B} \\
&= (A^c \oplus \hat{B}) \ominus \hat{B} \\
&= A^c \bullet \hat{B}
\end{aligned}
$$

Figure 6: algebraic proof

4. Dilate the image given in Figure 13.50(a) with the structuring element in (b). Do this by hand and very using Matlab(python)(give code).




```
0 1 0 0
0 1 0 0
0 0 1 1
0 0 1 0
2 0 1 0
original image; 1 represents a filled block and 2 represents the x-point.
0 1 0 0
0 1 1 1
0 1 1 0
0 0 1 1
2 0 1 1
output image
```

Figure 7: Hand calculation programming output

5. (GW 2.23)

   (a) With reference to Fig.2.31, sketch the set $(A \cap B) \cup (A \cup B)^C$

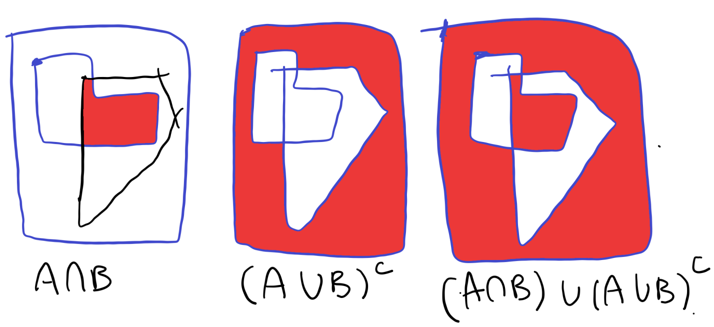


Figure 8: $(A \cap B) \cup (A \cup B)^C$

(b) Give expressions for the sets shaded in the following figure in terms of sets A, B, and C. The shaded areas in each figure constitute one set, so give one expression for each of the three figures.

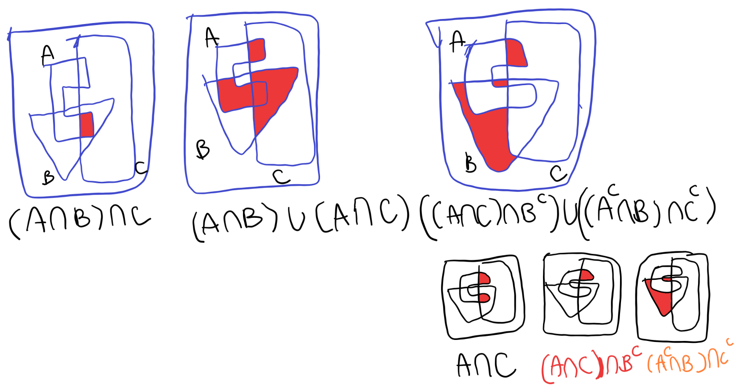


Figure 9: Sets A, B, and C expressions

6. (GW 9.5) With reference to the image shown, give the structuring element and morphological operation(s) that produced each of the results shown in images (a) through (d). Show the origin of each structuring element clearly. The dashed lines show the boundary of the original set and are included only for reference. Note that in (d) all corners are rounded.

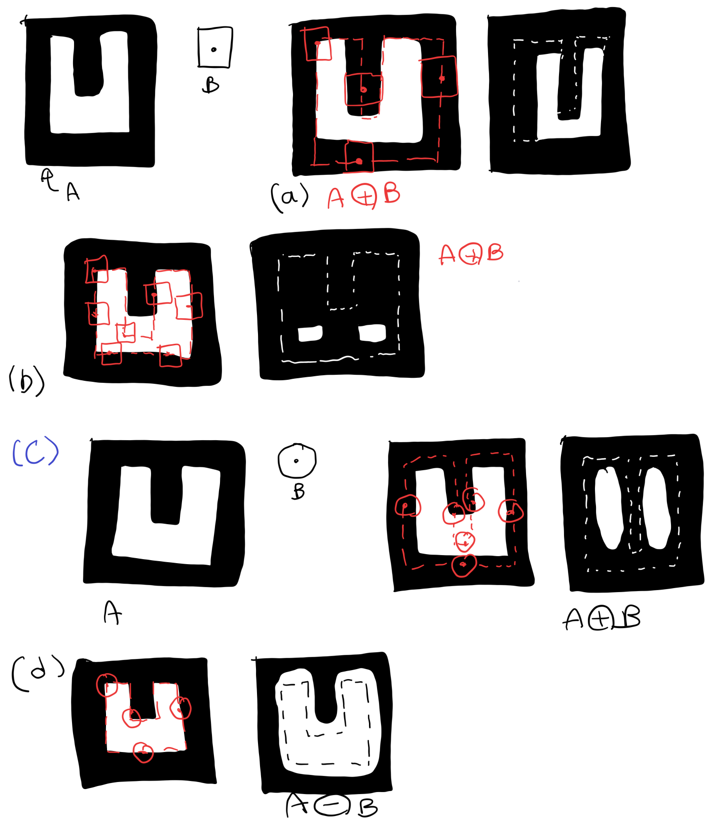


Figure 10: The structuring element and morphological operations

7. (GW 9.6) Let A denote the set shown shaded in the following figure. Refer to the structuring elements shown (the black dots denote the origin). Sketch the result of the following morphological operations:

   (a) $(A \ominus B^4) \oplus B^2$

   (b) $(A \ominus B^1) \oplus B^3$

   (c) $(A \oplus B^1) \oplus B^3$

   (d) $(A \oplus B^3) \ominus B^2$

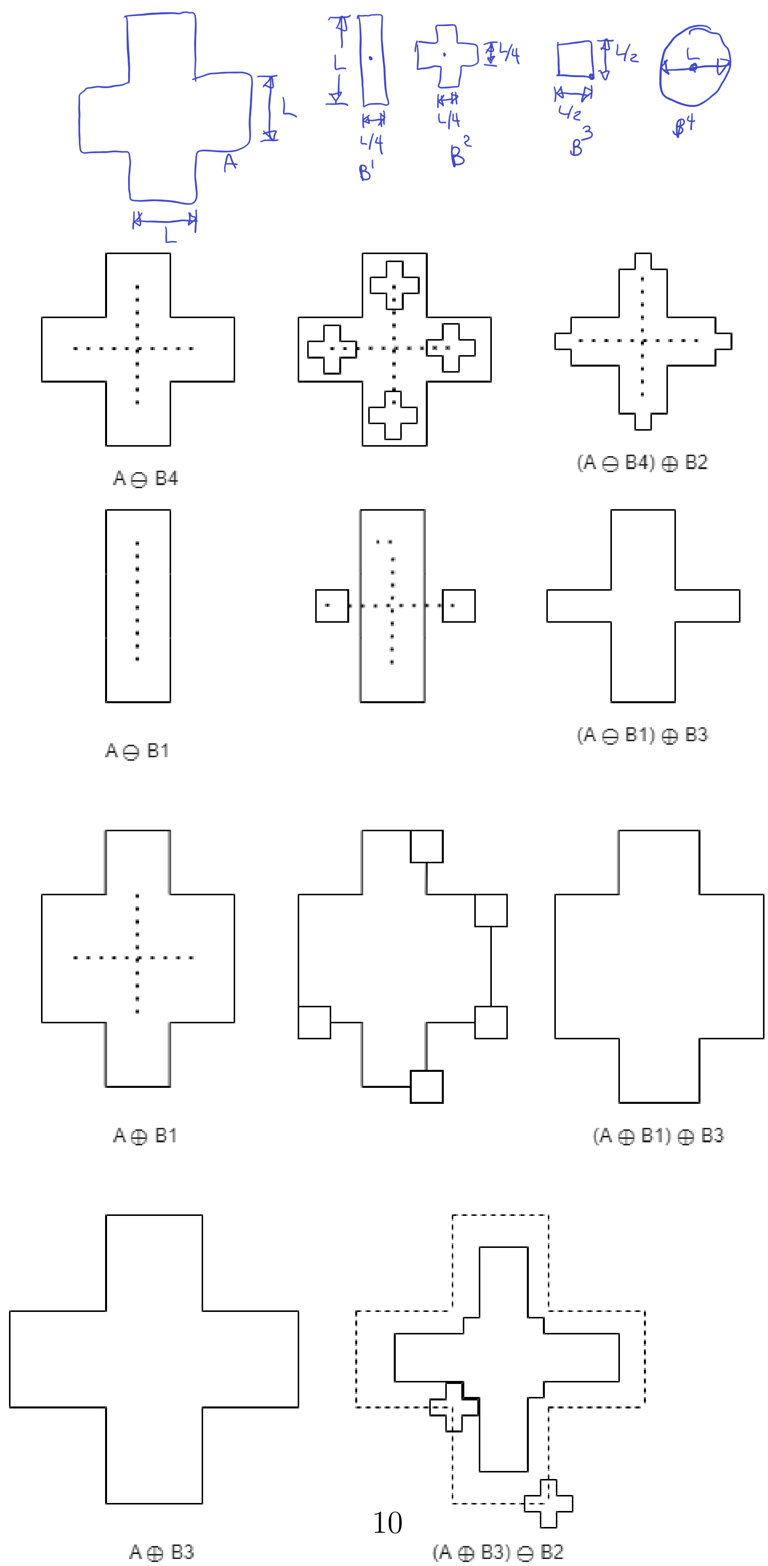
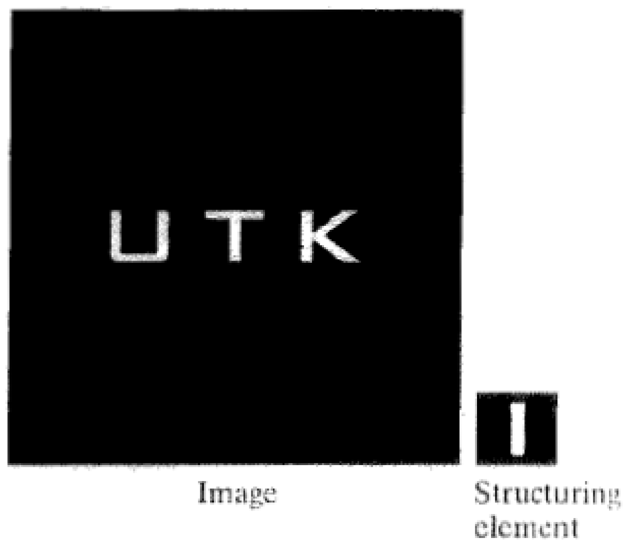


Figure 11: the result of the morphological operations

8. (GW 9.19) Sketch the result of applying the hit-or-miss transform to the image and structuring element shown. Indicate clearly the origin and border you selected for the structuring element. Then OR them both.

   To do this, first pick a very thin layer outside the shape T and call it W-T. Then find the midpoint both for horizontal and vertical. then get the complement of the whole image. Erode the whole image by T. Erode the compliment of the whole image by W-T.



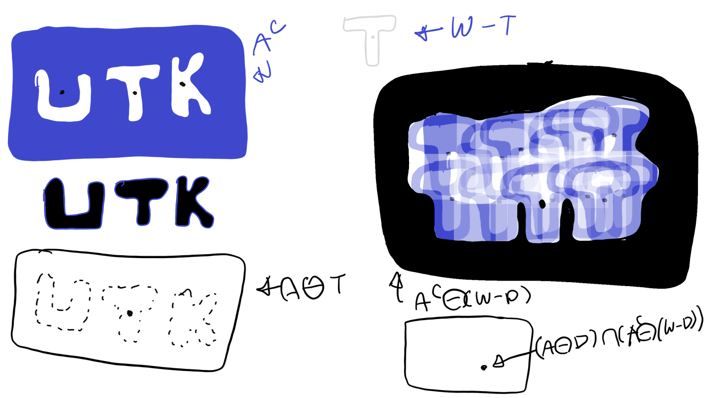


Figure 12: the result of the morphological operations

Codes

```python
from PIL import Image
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
import numpy as np

# Setting the input and output files.
IN_FILE = 'hw2/sample_images/Fig0627(d)(WashingtonDC Band4).
    tiff'
OUT_FILE = 'hw2/sample_images/Fig0627_out.png'

# Loading the image, storing width and height into constants
# Load image, store width and height into constants
pillow_img = Image.open(IN_FILE)
IMG_W, IMG_H = pillow_img.size
img = np.array(pillow_img)

threshold_img = np.zeros((IMG_H, IMG_W), dtype=int)
for i in range(IMG_H):
    for j in range(IMG_W):
        if (img[i,j] > 50):
            threshold_img[i, j] = 255
        else:
            threshold_img[i, j] = 0

# Saving the threshold image as .jpg
output_file = Image.fromarray(np.uint8(threshold_img.reshape
    ((IMG_H, IMG_W))))
output_file.save(OUT_FILE)
```

Listing 1: GW 6.17 (b) code

```python
from PIL import Image
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
import numpy as np

# Setting the input and output files.
IN_FILE = 'hw2/sample_images/Fig0627(a)(WashingtonDC Band3-
    RED).tiff'
OUT_FILE = 'hw2/sample_images/Fig0627_out_b.png'

# Loading the image, storing width and height into constants
# Load image, store width and height into constants
pillow_img = Image.open(IN_FILE)
IMG_W, IMG_H = pillow_img.size
```

```python
img = np.array(pillow_img)

threshold_img = np.zeros((IMG_H, IMG_W), dtype=int)
for i in range(IMG_H):
    for j in range(IMG_W):
        if (img[i,j] > 100):
            threshold_img[i, j] = 0
        else:
            threshold_img[i, j] = 255

# Saving the threshold image as .jpg
output_file = Image.fromarray(np.uint8(threshold_img.reshape((IMG_H, IMG_W))))
output_file.save(OUT_FILE)
```

Listing 2: GW 6.17 (c) code

```python
from mpl_toolkits.mplot3d import Axes3D
import matplotlib.pyplot as plt
import numpy as np

fig = plt.figure(figsize=plt.figaspect(1))
ax = fig.add_subplot(111, projection='3d')

c = (8, 1, 1)
rx, ry, rz = 1/np.sqrt(c)

u = np.linspace(0, 2 * np.pi, 100)
v = np.linspace(0, np.pi, 100)

x = rx * np.outer(np.cos(u), np.sin(v))
y = ry * np.outer(np.sin(u), np.sin(v))
z = rz * np.outer(np.ones_like(u), np.cos(v))

# Plot:
ax.plot_surface(x, y, z,  rstride=4, cstride=4, color='b')

max_radius = max(rx, ry, rz)
for axis in 'xyz':
    getattr(ax, 'set_{}lim'.format(axis))((-max_radius, max_radius))

plt.show()
```

Listing 3: GW 6.28 code

```
from PIL import Image
import matplotlib.image as mpimg
import numpy as np

original = np.zeros((7, 5), dtype=int)
original[1,2] = 1
original[2,2] = 1
original[3,3] = 1
original[3,4] = 1
original[4,3] = 1
original[5,3] = 1
original[5,1] = 2

for i in range(1,6):
    for j in range(1,5):
        if j < 4:
            print(original[i,j], end=" ")
        else:
            print(original[i,j])
print(’original image; 1 represents a filled block and 2 represents the x-point.’)

output = np.zeros((7, 5), dtype=int)
for i in range(1,6):
    for j in range(1, 5):
        if original[i,j]==2:
            output[i,j]=2
        if (original[i-1,j] == 1) or (original[i-1,j-1] == 1) or (original[i+1, j] == 1):
                output[i,j] = 1

for i in range(1,6):
    for j in range(1,5):
        if j < 4:
            print(output[i,j], end=" ")
        else:
            print(output[i,j])

print(’output image’)
```

Listing 4: number 4 code