Lab 2: Geometric Transformations and Binary Image Processing

(10% of final score, due by Feb. 12 11:59 PM)

In the following tasks, you can use your input images to test your codes or select images in the skimage.data module.

https://scikit-image.org/docs/stable/api/skimage.data.html

Upload your results and *.*py* files to the folder of **Lab 2** under **Assessments** > **Assignment** in the D2L system. Name the *.*py* files according to the series numbers of the questions. The *.*py* files should follow the PEP-8 Style Guide for Python Code.

https://www.python.org/dev/peps/pep-0008/

1. Read Scikit-image tutorial "*Using geometric transformations*" https://scikit-image.org/docs/dev/auto_examples/transform/plot_geometric.html#sphx-glr-auto-examples-transform-plot-geometric-py

   (a) Create a transformation using the following explicit parameters:

   ```
   scale = 0.5, rotation = 3*pi/8, translation = (20, 30)
   ```

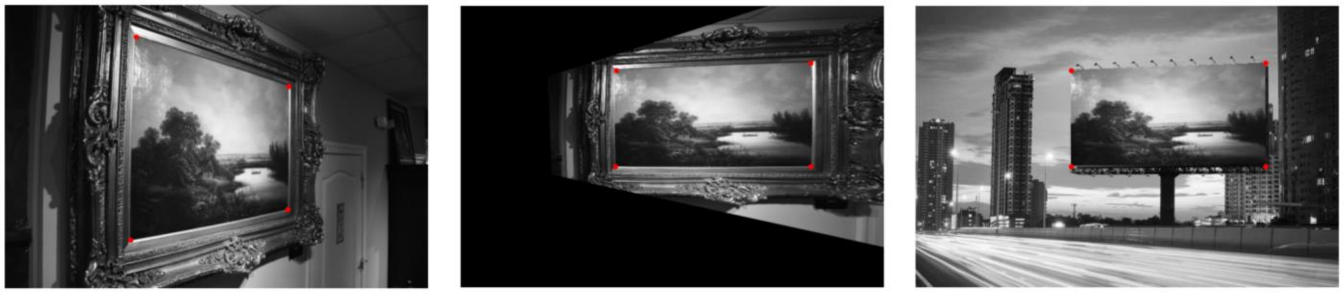
   Print out the transformation matrix in Python. Calculate the transformation matrix by hand and compare it with the printed one. Use an image to visualize the transformation. Show the original image and the transformed image in one window.

   (b) Apply the following projective transformation on the image skimage.data.text()

   $$H = \begin{bmatrix} 1 & -0.5 & 100 \\ 0.1 & 0.9 & 50 \\ 0.0015 & 0.0015 & 1 \end{bmatrix}$$

   Show the original image and the transformed image in one window.

2. Download two images from the folder of Lab 2, "Oil painting.jpg" and "Highway billboard.jpg." Project the oil painting to the highway billboard in Python. Print out the transformation matrix and show the transformed image. The output image would look like this:



Steps:

- Follow the example "*Parameter estimation*" in the link of Question #1.

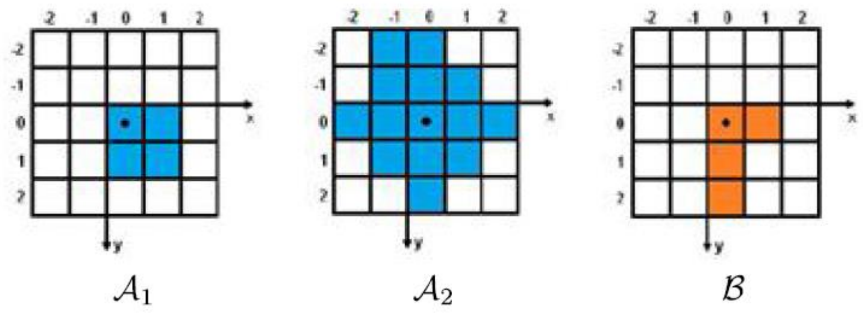
- Manually get the four coordinates from the “Oil painting.jpg” and “Highway billboard.jpg.” Mark them in red points. Set them as the “destination” and “source” for warping.
- Use ImageDraw.Draw(img).polygon(polygon, outline=1, fill=255) to define a binary mask in which the board region is white, and the background is black. Check the Pillow documentation to find out how to import ImageDraw and how to call it.
- Paste the warped image to the board region. Use img.paste(warped_img, (0, 0), mask=mask_img). Here is an example to do “paste.” https://note.nkmk.me/en/python-pillow-paste/
- Be aware that when using “paste,” all the images or image parts should be an image “object” instead of an image array. If it is an array, use Image.fromarray() to change it to an object. The datatype of warped image in scikit-image module is float. Need to switch float to uint8 using img_as_ubyte().
- Welcome to explore functions in other modules, *e.g*. OpenCV to implement the same transformation. cv2.getPerspectiveTransform(), cv2.warpPerspective(), cv2.add().

3. Given the images A and B:

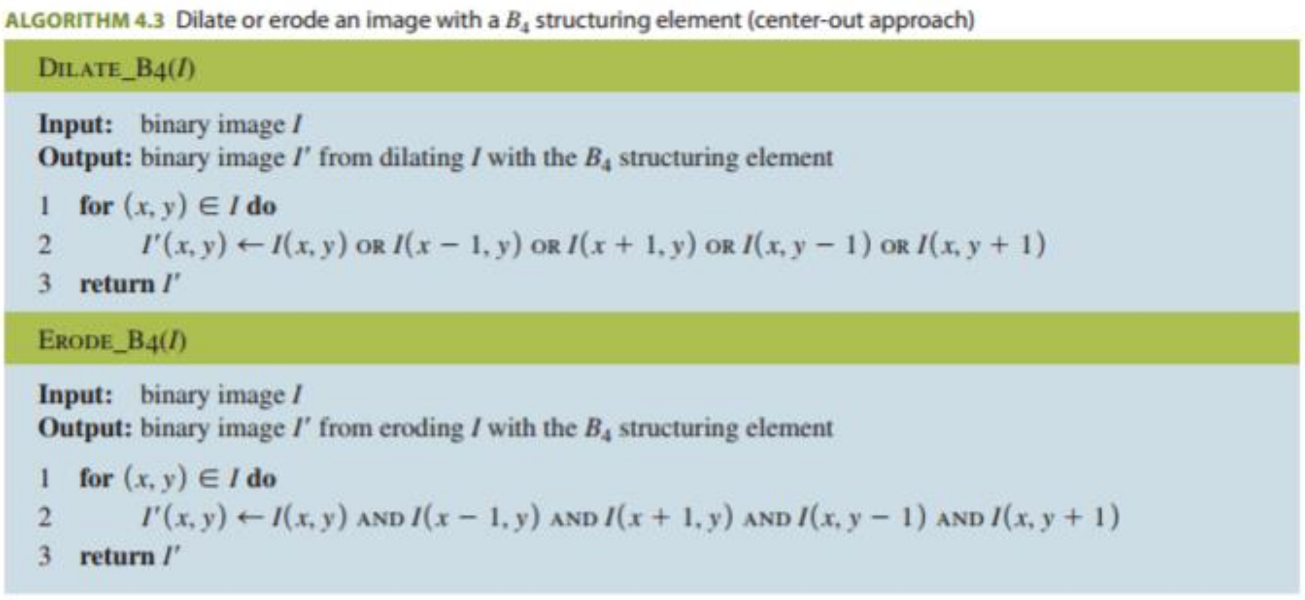
$$\mathcal{A} = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \qquad \mathcal{B} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 \end{bmatrix}$$

Compute **by hand** $A \cup B$ (union), $A \cap B$ (intersection), $\breve{B}$ (reflection about the origin), $\neg A$ (complement), and A\B (set difference). Write the results as arrays. Compute in Python. Use np.array([], dtype=np.bool) to define binary arrays. Use logical operators np.logical_and, np.logical_or, np.logical_not.

4. Given sets A1, A2, and B. Consider B as a structuring element. Compute dilation for sets A1 and B; Compute erosion for sets A2 and B. Compute **by hand**.



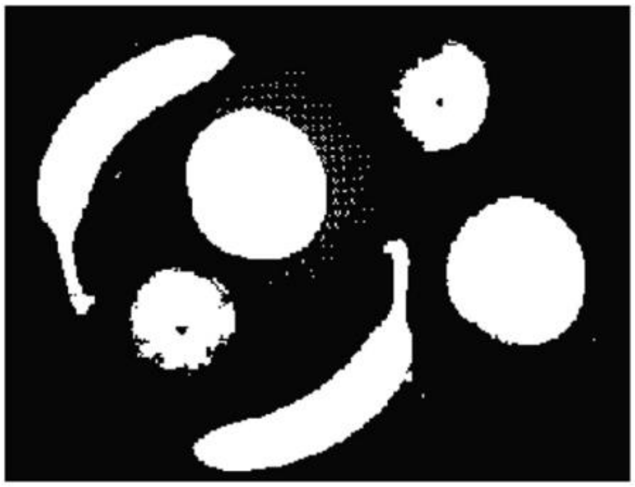
5. Follow the pseudocode provided below to dilate and erode an image with a B4 structuring element. Code in Python. Use your own binary image.



ALGORITHM 4.3 Dilate or erode an image with a B_4 structuring element (center-out approach)

```
DILATE_B4(I)

Input:  binary image I
Output: binary image I' from dilating I with the B_4 structuring element

1 for $(x, y) \in I$ do
2     $I'(x, y) \leftarrow I(x, y)$ OR $I(x-1, y)$ OR $I(x+1, y)$ OR $I(x, y-1)$ OR $I(x, y+1)$
3 return I'

ERODE_B4(I)

Input:  binary image I
Output: binary image I' from eroding I with the B_4 structuring element

1 for $(x, y) \in I$ do
2     $I'(x, y) \leftarrow I(x, y)$ AND $I(x-1, y)$ AND $I(x+1, y)$ AND $I(x, y-1)$ AND $I(x, y+1)$
3 return I'
```

6. Download the following image <fruit.PNG> from the folder Lab2.
   (a) Identify the type of noise (lake, bay, channel, cape, isthmus, or island?) Which morphological operator should be applied to the image to remove noise? Implement morphology operations using *scikimage.morphorlogy* module. [*Note: check the image mode and convert it to "binary" if it is not a binary image as it looks like.*]
   (b) After cleaning up the noise, use region properties *skimage.measure.regionprops* to classify the fruits.



7. Demonstrate by a simple example that repeated applications of opening or closing do nothing. Implement opening and closing operations using *scikimage.morphorlogy* module.

8. Compute the Euclidean, Manhattan, and chessboard distances from each pixel in a 5 by 5 image to the central pixel. What shape do the isocontours take in each case?

9. Download the image ‘blood-cells.jpg’
   Write code to **threshold** the image (with thresholding methods), **clean** up the noise (with morphology methods), **separate** the neutrophils from the red cells.
   This is an open question. You can use any methods you have learned.