3460:435/535 Algorithms

Project 3 --- Dynamic Programming

Images are often viewed on different devices with different resolutions and viewing space. To accommodate limited space, one can resize the image using techniques such as cropping. However, cropping alters the image and is not always desirable. Your third programming assignment is to implement a dynamic programming algorithm called seam craving for image resizing. This project is to be done in Python. Use A_P3_template.jpynb.

Here is a link to the paper on seam carving published on SIGGRAPH2007
http://www.faculty.idc.ac.il/arik/site/seam-carve.asp (paper and video).

Seam carving changes the size of an image by removing the least visible pixels in the image. The visibility of a pixel can be defined using an energy function. Seam carving can be done by finding a one-pixel wide path of lowest energy crossing the image from top to bottom (vertical path) or from left to right (horizontal path) and removing the path (seam).

The energy of each pixel can be described using the gradient magnitude:

$$e(\mathbf{I}) = |\frac{\partial}{\partial x}\mathbf{I}| + |\frac{\partial}{\partial y}\mathbf{I}|,$$

where I is a *nxm* image. For this project,

$$e(i,j) = |v(i,j)-v(i-1,j)| + |v(i,j)-v(i+1,j)| + |v(i,j)-v(i,j-1)| + |v(i,j)-v(i,j+1)|,$$

v(i,j) = pixel value at (i,j). For the boundary cases, the difference = 0 if one of the pixels is outside of the given image.

The cumulative minimum energy M for all possible connected vertical seams for each entry (i,j) can be calculated as the following (i = x-axis; j = y-axis):

$$M(i,j) = e(i,j) + \min\{M(i-1,j-1), M(i, j-1), M(i+1,j-1))$$

Similarly, the cumulative minimum energy for horizontal seams can be calculated also.

Part I. Vertical seam removal.
For this part, the image to be resized is in pgm (portable gray map) format. The pgm image file requires a header of 4 entries followed by the greyscale values (some files include comments lines starting with the character #). The four entries are: the literal "P2", an integer representing the x dimension, an integer representing the y dimension, and an integer representing the maximum greyscale value. There should be x times y number of grey-level values after these 4 numbers. Part of a sample plain pgm image (40 columns × 42 rows) bug.pgm is shown below.

```
P2
# Created by IrfanView
40 42
255
192 192 192 192 192 192 192 192 192 192 192 192 192 192 192 192 192
192 192 192 192 192 192 192 192 192 192 197 197 197 191 192 192
...
```

The image of bug.pgm looks like:

Your program should provide all the necessary functionalities, including:

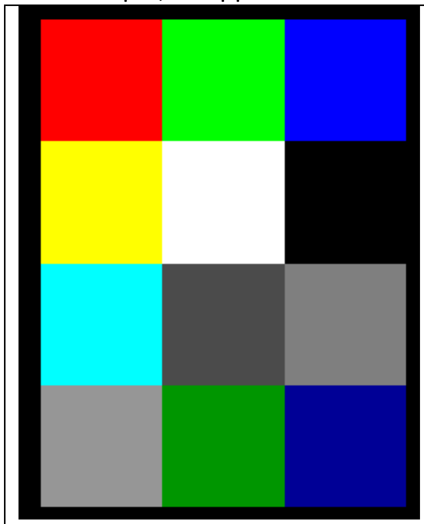
(1) Allow the user to provide the image file name through the 1st cell of the Jupyter notebook.
(2) Allow user to specify #s of vertical seams to be removed through the 1st cell of the Jupyter notebook.
(3) Save the processed image in a pgm file named: original_image_file_name_**processed_v_h**.pgm, where h=0.
(4) To facilitate grading, use the left-most minimum as your minimum when you trace the seam with lowest energy. i.e. if there is a tie, the **left-most minimum is the minimum**.

Part II. Both vertical and horizontal seams removal

Same as for part I, the image to be resized is in pgm format. Your program should be able to handle both vertical and horizontal seams. Note: When you process the image, remove the lowest-energy **vertical seams** first before you remove the horizontal ones. Your processed file: **image_processed_v_h.pgm**

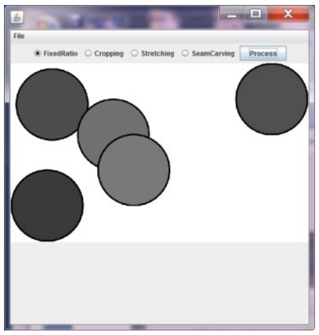
Part III. Graduate students/honors students

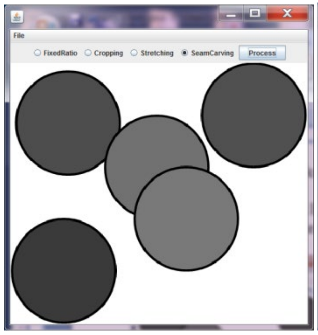
For Part III, you handle colored **ppm** (portable pix map) images with format P3. Like a pgm file, a **ppm** file consists of two parts, a header and the image data. The header includes four entries, the literal "P3", an integer representing the x dimension, an integer representing the y dimension, and an integer representing the maximum scale value. The image data includes 3 times x times y number of pixel values. For example, the ppm file for the image on the left:

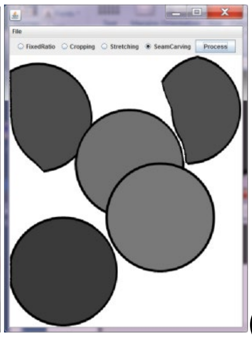
	P3 3 4 255 # P3 means colors are RGB triplets 255 0 0 0 255 0 0 0 255 255 255 0 255 255 255 0 0 0 0 255 255 75 75 75 127 127 127 150 150 150 0 150 0 0 0 150

For colored images, the energy of a pixel is the sum of the energy of the pixel over all 3 colors. I.e. treat each color (red, green, blue) separately, then add the energy of each channel. When you process the image, remove the lowest-energy **vertical seams** first before you remove the horizontal ones.

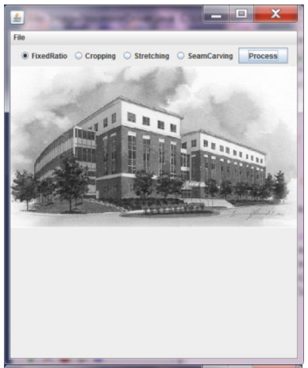
Examples:

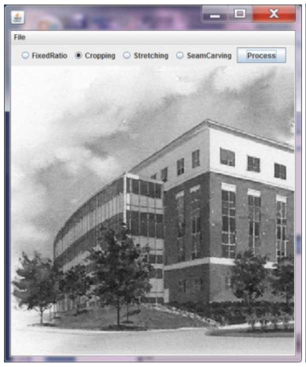


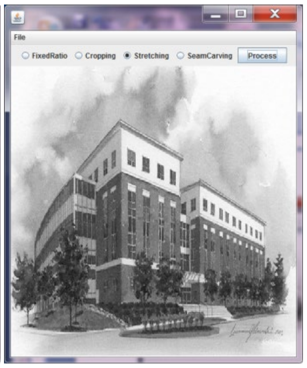


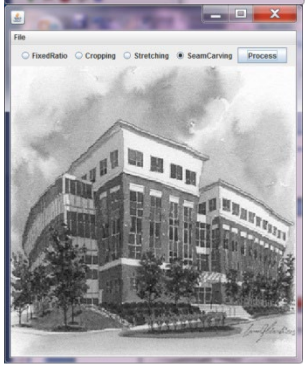


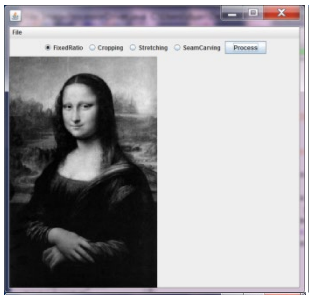
(Further resizing)

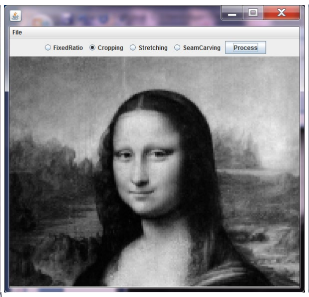


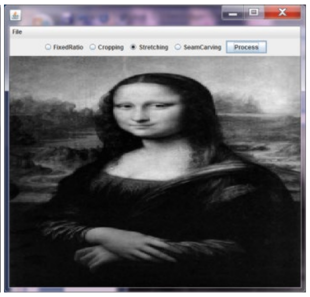


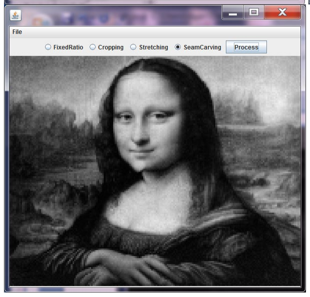












What to submit.

- One jupyter notebook: A_P3.jpynb
- Be sure to electronically submit your working solution before the due date! Do not submit non-working programs. The electronic submission time will be used to assess late penalties (if applicable).

Grading. Your code will be graded on **correctness**, efficiency, clarity, and elegance.