Due: Monday March 30 by 11pm, no late submissions after 1am on March 31 Worth: 7%

Background. For this project, you will implement *triangulation matting* and then experiment with the algorithm. For this project, you will use the following version of the matting equation:

$$C = F + (1-\alpha)B.$$

Here, C is a pixel in the composite image, F is a a foreground pixel, and B is a background pixel. The alpha channel α is 0 if the pixel is transparent (in that case, F must be 0 as well, though we do not enforce that requirement), and 1 if the pixel is opaque.

What to submit Submit your report as `pdf` and `tex` files, and submit your python code. Submit any images that you used (Tip: do not use images that cannot fit on a page in order to keep the file size down.)

Your `__main__` block of your `p4.py` should generate the images that you are using in the report.

Part 1. The Matting Equation in Matrix Form (2 pts.)

Assume that you have two composite images, C_1 and C_2, with two different known backgrounds, B^1 and B^2 respectively. Denote, e.g., the red channel of B_1 as B_1^r. By writing the matting equations in vector form and rearranging terms, prove that the two matting equations for the two backgrounds can be written in matrix form as follows:

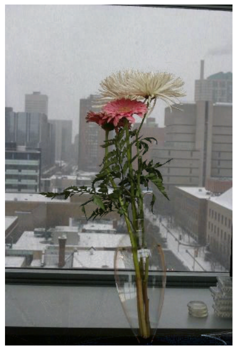
$$\begin{pmatrix} 1 & 0 & 0 & -B_1^r \\ 0 & 1 & 0 & -B_1^g \\ 0 & 0 & 1 & -B_1^b \\ 1 & 0 & 0 & -B_2^r \\ 0 & 1 & 0 & -B_2^g \\ 0 & 0 & 1 & -B_2^b \end{pmatrix} \begin{pmatrix} F_r \\ F_g \\ F_b \\ \alpha \end{pmatrix} = \begin{pmatrix} C_1^r - B_1^r \\ C_1^g - B_1^g \\ C_1^b - B_1^b \\ C_2^r - B_2^r \\ C_2^g - B_2^g \\ C_2^b - B_2^b \end{pmatrix}.$$

Part 2. Changing the Background using Matting (20 pts.)

For the sets of images of flowers and leaves, compute and display the alpha matte and the foreground colour by solving the equation in Part 1 for every pixel. Include the foreground and the alpha channel that you obtain for both sets of images in your report.

In order to solve a system of equations of the form $Ax = b$ where A is non-square, you can compute A^+, the pseudo-inverse of A. Then, $x \approx A^+b$. The NumPy function `linalg.pinv` computes the pseudo-inverse. Note that A^+b will not necessarily be in the range $(0, 1)$, and so it needs to post-processed before you use and/or display the results. You may find `np.clip` to be helpful for post-processing.

Then, for both sets of images, create a new composite image with the original foreground but with `window.jpg` as the background. Include both the composite images that you obtain in your report.



Sample alpha and composite images

Part 3. A New Composite Image (3 pts.)

Capture 5 JPEG images with your camera (Background 1, Background 2, Composite 1, Composite 2, New Background) and use them as input for your implementation of the matting algorithm. Produce the alpha matte, the foreground image, and the new composite image and include them in your report. Include the 5 images that you took in your report. In 2-3 sentences, describe your image acquistion procedure (e.g. "I put the camera on a table, then put the object in front of a piece of cloth, and took a picture; then ..."). If your experiment is unsuccessful, try to explain what went wrong with the image capture process.