1 Assignment 5 (100 points)

To submit your work, collect your inputs not provided as part of the assignment, the required outputs, and your answers to the written questions in pdf format (preferred) or Word, or .txt . Clearly label all of your work. For the math-oriented portion of the assignment, you may typeset in LaTeX or Word or provide a clear photograph of a hand-written solution. Work that is not legible will not be graded. Name a zip file `CS585_Assignment5_username.zip` and submit with web-submit. There is no code to submit for this assignment.

Collect your .cpp files and label code that you changed with comments containing your username like: `//Modified by deht`

1.1 Learning Objectives

- Reinforce the basic, primitive types of transformation matrices
- Understand how to compose more complex transformations using those primitives
- Get comfortable controlling image transformations

1.2 Written Questions (20 points)

Here are the primitive matrices we described in class:

Translation:

$$T(t_x, t_y) = \begin{bmatrix} \mathbf{I} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix}$$

Scaling:

$$S(\alpha) = \begin{bmatrix} \alpha\mathbf{I} & \mathbf{0} \\ \mathbf{0} & 1 \end{bmatrix} = \begin{bmatrix} \alpha & 0 & 0 \\ 0 & \alpha & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Rotation:

$$R(\theta) = \begin{bmatrix} \mathbf{R} & \mathbf{0} \\ \mathbf{0} & 1 \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Shearing:

$$Sh(\beta_x, \beta_y) = \begin{bmatrix} 1 & \beta_x & 0 \\ \beta_y & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

1. Using the primitives given above, write a formula describing how to transform points in an image in order to rotate an image about its center, (c_x, c_y).
2. Using the primitives above, write a formula describing how to transform points in an image in order to rotate *and scale* an image about its center

3. Using the primitives above, write a formula describing how to transform points in an image in order to rotate and scale an image about its center, then translate the rotated and scaled image to some other location (i_x, i_y).

4. Using the primitives above, write a formula for describing how to transform points in an image in order to shear an image about its center

1.3 Technical Task (80 points)

For this assignment, we will make a spring-time collage using image transformations / matrices to combine a set of source images into a result image. The steps towards this goal are outlined below.

You should choose a source image of your own that you like looking at to use for debugging. The image should be no larger than 640 x 480. Submit your input image with your submission.

Beware: The code to write is quite short, but tricky. Small mistakes can manifest themselves in dramatic ways, so be methodical if your results do not match your expectations.

1. (Given) In lab, we worked with a program to control rotation of the image using a slider. We compared the OpenCV function, which rotates the image about the center, with our hand-written version that rotates about the upper-left corner

2. Implement Scaling. Part 0 of the lab is to implement the function `getScaleMatrix` to create a matrix representing a scaling operation. The provided skeleton code has event handling that will call your function in order to produce a result. Drag the slider to produce a scaled version of your image and save your result as ”Assignment5_Part0_Output_Scale.png” (Do not submit the image for rotation)

3. Reproduce the OpenCV rotation function. Part 1 of the lab is to reproduce the OpenCV `getRotationMatrix2D` function. To do this, you will first implement the `getTranslationMatrix` function to create a matrix representing a basic translation. Then, using the scaling operation from Part 0, you will need to implement the function `myGetRotationMatrix2D` by correctly chaining together the matrix primitives. The skeleton code will display the result of your version along side the result of the OpenCV version. Save a rotated and scaled version of your image as ”Assignment5_Part1_Output_Mine.png”. Submit it along with the matching ”Assignment5_Part1_Output_OpenCV.png”.

4. Implement Shearing. Part 2 of the lab is to create functionality that is missing in OpenCV: create a transformation where the image is sheared about its center. You will start by implementing the primitive `getShearMatrix`. Then, in a style similar to Part 1, you will implement the function `getShearMatrix2D` that composes several primitive transformations together to shear the image about its center. Use the sliders to shear the image horzontally and vertically. Save a horizontally sheared version as ”Assignment5_Part2_Output_Horizontal.png”. Save a vertically sheared version as ”Assignment5_Part2_Output_Vertical.png”

5. Create a spring-time collage by assembling a collection of at least 5 source images no more than 350 x 350 pixels. Use any transformations you like to place the images with a 750 x 750 pixel canvas. Save your result as ”Assignment5_Output.png”.

   The source images should be placed at different scales and orientations. You can use shears too if you like. The source images should be scattered throughout the canvas (not clustered in the upper left). Each image may be used more than once if you would like. (If you don't want to make a spring-themed collage, you can use subject matter of your choosing)

The skeleton code for this part is minimal in order to give you freedom to develop your own solution from scratch. You can assemble your code using any pieces from any of the labs or homeworks that we have done. You can make an interface to support your composition if you want, but you don't have to. You can generate the configuration of the sub-images with mathematical formula, by hand, or at random. No matter how you choose to construct your placement, your source images should remain mostly within the canvas and be spread throughout the canvas. An example composition is provided in "Assignment5_Output_Diane.png".