Lab 03: Matrix inverse and Matrix Mappings

Due: September 5, 8:30am

Assignment 1

- **Write a Python program to invert matrices using the technique covered in the lecture**
  - Input the matrix to be inverted from a text file
  - Compute the inverese using np.float64 types
    - ✓ Identify singular matrices, print that fact to the screen, and exit
  - Print the inverse you compute to the screen
  - Verify you have found the inverse by multiplying it by the original and comparing it to the identity matrix. Use numpy
    - ✓ Print the maximum absolute error of any element in your AA^{-1} calculation
    - ✓ Print the MSE of the elements in your AA^{-1} calculation

Mapping of (x, y) to (x', y')

- **Matrix formulas can map points at one point on the XY plane to another point on the XY plane**

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

Visualizing where points go

- **We can see where points go with image processing**
  - Create an image with a known intensity profile, $I(\cdot, \cdot)$
  - Pick a point in this image, call it (x, y)
  - Compute the points new coordinates after transformation, (x', y')
  - Set $I(x', y') = I(x, y)$

Background

- **Multiplying by the following matrix scales a point in the x-direction by λ_1 and in the y-direction by λ_2**

$$S(\lambda_1, \lambda_2) = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$$

Background

- **Multiplying by the following matrix rotates a point on the XY plane clockwise around the origin (this assumes you are looking down the z-axis in the positive z-axis direction)**

$$R_Z(\theta) = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}$$

Background

- **Adding a vector is a translation**

Assignment 2

- **We want to see the visual effect of the following transformation on an image**

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

- **Write a Python method to create a 512 by 512 black image with a 129 by 129 red square in its center (RGB values [255, 0, 0])**
  - The origin [0, 0] of the *Pixel Image Coordinate System* (PICS) is the upper-left corner of the image (positive x is to the right and positive y is down)
  - Center the square around pixel [255, 255] in the PICS.
    - ✓ This pixel is the origin [0, 0] of the *Canonical Image Coordinate System* (CICS) (positive x is to the right and positive y is down)

Assignment 2 cont.

- **Write a Python method to bilinearly interpolate the red color plane of a bmp image's intensity for *fractional* pixel offsets**

- **Create a new blank image**
  - Loop through the PICS coordinates of this blank image, convert the coordinates from PICS to CICS to get $[x', y']^T$
  - Compute the corresponding pixel in the original image (i.e. the one with the red square) in CICS coordinates
    - ✓ You need to invert the desired transformation, i.e., solve for $[x, y]^T$ in terms of $[x', y']^T$
    - ✓ You will get fractional coordinates for $[x, y]^T$. Convert these cooridnates to PICS, then use your bilinear inerpolation method on the original image to compute the pixel's intensity
      - – You can skip the interpolation for the first and last row and first and last column of the new image to make your code simpler. In other words, exclude them from your for loops
    - ✓ Write the computed intensity value into the "blank" image at the appropriate CICS coordinates

Transformatons to investigate

- **Let**

$$A = R_2(\theta_2)S(\lambda_1, \lambda_2)R_1(\theta_1)$$

- **Include images for the following combinations in your lab report, feel free to try others**

Why do this?

- **All linear transformations can be decomposed into a combination of a rotation, followed by a scaling, followed by another rotation**

Angles are in degrees

theta_2	lambda_1	lambda_2	theta_1	tx	ty
0	1	1	45	0	0
0	2	1	30	0	0
0	1	2	30	0	0
15	2	1	30	0	0
25	2	0.5	50	0	0
25	2	0.5	50	80	-70