COMPUTING THE INVERSE TRANSFORM MATRIX

1. In class, we saw how to compute the inverse of the scaling matrix, not by a linear algebra approach but by intuiting what it means to scale downwards (shrinking is the multiplicative inverse of enlarging). Similarly, compute the inverse of a translation matrix. Hint: think about what it means to invert translation. We saw that inverting scaling means performing the opposite of the particular scaling transform: if you were enlarging the image to get the transformed image, then the image itself is a shrunk version of the transformed image, and vice versa. Apply a similar intuition for translation. (5 points)

A translation matrix is simply the identity matrix with the x and y translation values:

$$T = T(d_1\, d_2) = \begin{matrix} 1 & 0 & d_1 \\ 0 & 1 & d_2 \\ 0 & 0 & 1 \end{matrix}$$

To invert the translation, the values can simply be negated so it would return to the original coordinates:

$$\begin{matrix} 1 & 0 & -d_1 \\ 0 & 1 & -d_2 \\ 0 & 0 & 1 \end{matrix}$$

2. Using the same intuitions as what you used above, invert a rotation matrix. There is a fundamental relationship between a rotation matrix and its inverse. Can you identify this relationship? (5 points)

A rotation matrix of θ degrees is as follows:

$$\begin{matrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{matrix}$$

To have the inverse of the rotation, all that is needed is to rotate by -θ:

$$\begin{matrix} \cos(-\theta) & -\sin(-\theta) & 0 \\ \sin(-\theta) & \cos(-\theta) & 0 \\ 0 & 0 & 1 \end{matrix}$$

Which is the same as the transpose of the original:

$$\begin{matrix} \cos(\theta) & \sin(\theta) & 0 \\ -\sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{matrix}$$

3. What should the inverse of a reflection matrix be, and why? (5 points)

The inverse of a reflection matrix, due to its nature, would simply be itself. This is because in order to do the inverse of reflecting across the x axis, the only way to do so is by again reflecting across the x axis.

4. The inverse for the shear matrix is slightly trickier to intuit, so we will first calculate its inverse using a standard matrix-inversion formula. You will only need the inversion formula for a 2 ×2 matrix:

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}^{-1} = \frac{1}{ad-bc}\begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$$

Use the above formula to calculate the inverse of the matrix for the shear transform along the x-direction. Intuitively, what does this inverse mean? Can you obtain a similar formula for the inverse of the shear transform along the y-direction? (5 points)

The matrix for a horizontal shear is $\begin{matrix} 1 & d_x \\ 0 & 1 \end{matrix}$, and when put through the above equation the inverse can be determined to be $\begin{matrix} 1 & -d_x \\ 0 & 1 \end{matrix}$, using the same methodology the inverse vertical shear matrix is $\begin{matrix} 1 & 0 \\ -d_y & 1 \end{matrix}$.

2
IMPLEMENTING IMAGE TRANSFORMS

1. Change the size of the image to 1080 ×1920 (make sure the image you start with is not already 1080 ×1920). (8 points)

```
scale_meech = [1920/size(meech)[2] 0 0; 0 1080/size(meech)[1] 0; 0 0 1];
scale_plant = [1920/size(plant)[2] 0 0; 0 1080/size(plant)[1] 0; 0 0 1];
scale_room = [1920/size(room)[2] 0 0; 0 1080/size(room)[1] 0; 0 0 1];
save("meech1.png", transform_image(meech, scale_meech, "scale"));
save("plant1.png", transform_image(plant, scale_plant, "scale"));
save("room1.png", transform_image(room, scale_room, "scale"));
```

2. Reflect the image in the y direction. (8 points)

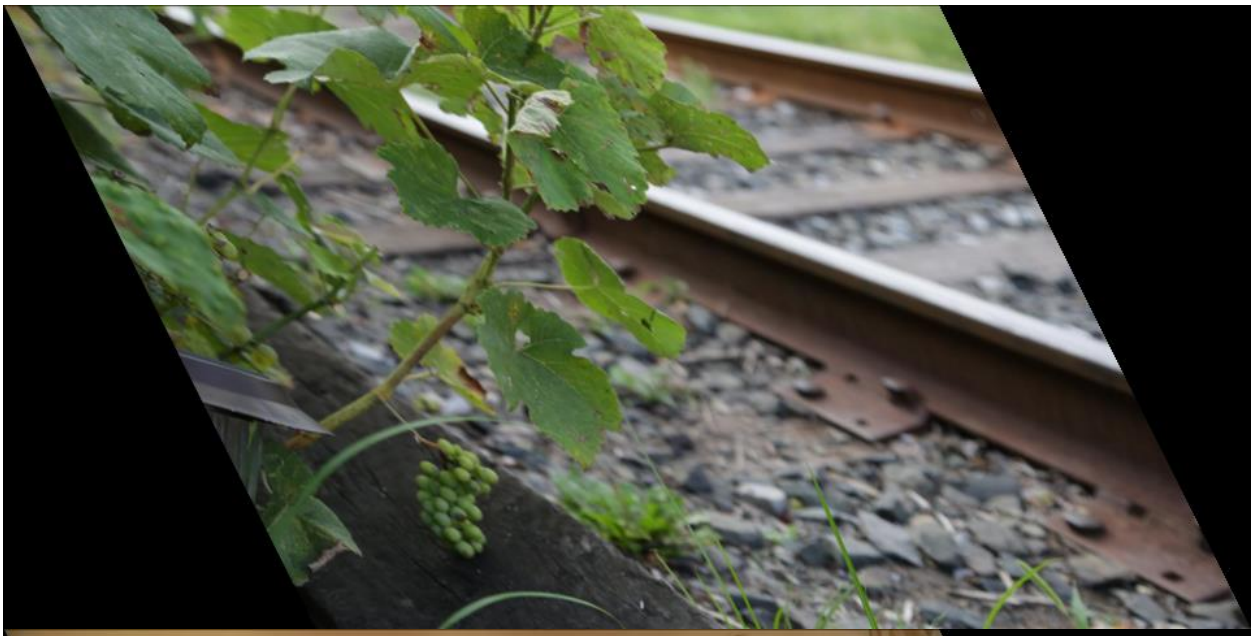
```
reflect = [1 0 0; 0 -1 0; 0 0 1];
save("meech2.png", transform_image(meech, reflect, "reflect"));
save("plant2.png", transform_image(plant, reflect, "reflect"));
save("room2.png", transform_image(room, reflect, "reflect"));
```

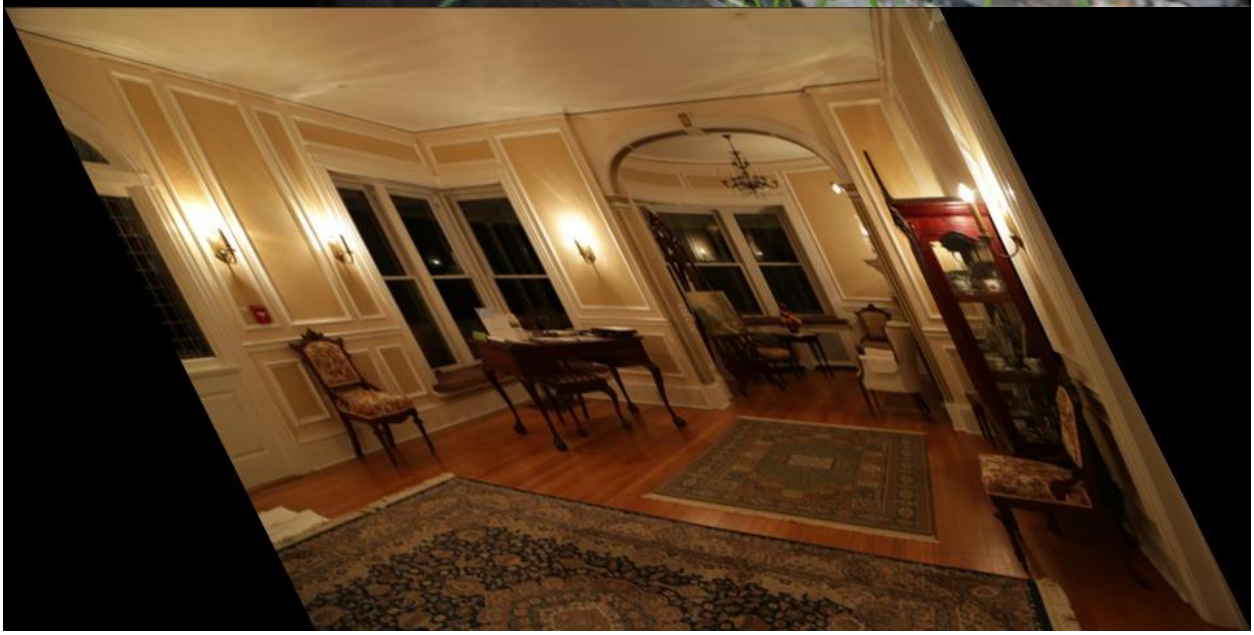
3. Rotate the image clockwise by 30 degrees. (8 points)

```
rotate30 = [cos(deg2rad(30)) -sin(deg2rad(30)) 0; sin(deg2rad(30))
cos(deg2rad(30)) 0; 0 0 1];
save("meech3.png", transform_image(meech, rotate30, "rotate"));
save("plant3.png", transform_image(plant, rotate30, "rotate"));
save("room3.png", transform_image(room, rotate30, "rotate"));
```

4. Shear the image in the x-direction so that the additional amount added to each x value is 0.5 times each y value. (8 points)

```
shear = [1 0.5 0; 0 1 0; 0 0 1];
save("meech4.png", transform_image(meech, shear, "shear"));
save("plant4.png", transform_image(plant, shear, "shear"));
save("room4.png", transform_image(room, shear, "shear"));
```





5. Translate the image by 300 in the x-direction and 500 in the y-direction, then rotate the resulting image counterclockwise by 20 degrees, then scale the resulting image down to one-half its size. You should apply the transformImage function only once to do this. (16 points)

```
scale_half = [0.5 0 0; 0 0.5 0; 0 0 1];
rotate20 = [cos(deg2rad(-20)) -sin(deg2rad(-20)) 0; sin(deg2rad(-20))
cos(deg2rad(-20)) 0; 0 0 1]
translate = [1 0 300; 0 1 500; 0 0 1];
save("meech5.png", transform_image(transform_image(transform_image(meech,
translate, "translate"), rotate20, "rotate"), scale_half, "scale"));
save("plant5.png", transform_image(transform_image(transform_image(plant,
translate, "translate"), rotate20, "rotate"), scale_half, "scale"));
save("room5.png", transform_image(transform_image(transform_image(room,
translate, "translate"), rotate20, "rotate"), scale_half, "scale"));
```

6. The following two affine transforms: (16 points)

$$\begin{bmatrix} 1 & .4 & .4 \\ .1 & 1 & .3 \\ 0 & 0 & 1 \end{bmatrix} \text{ and } \begin{bmatrix} 2.1 & -.35 & -.1 \\ -.3 & .7 & .3 \\ 0 & 0 & 1 \end{bmatrix}$$

```
affine_1 = [1 .4 .4; .1 1 .3; 0 0 1];
save("meech6-1.png", transform_image(meech, affine_1, "affine"));
save("plant6-1.png", transform_image(plant, affine_1, "affine"));
```

```
save("room6-1.png", transform_image(room, affine_1, "affine"));

affine_2 = [2.1 -.35 -.1; -.3 .7 .3; 0 0 1];
save("meech6-2.png", transform_image(meech, affine_2, "affine"));
save("plant6-2.png", transform_image(plant, affine_2, "affine"));
save("room6-2.png", transform_image(room, affine_2, "affine"));
```

7. The following two homographies: (16 points undergrad, 12 points grad)

$$\begin{bmatrix} .8 & .2 & .3 \\ -.1 & .9 & -.1 \\ .0005 & -.0005 & 1 \end{bmatrix} \text{ and } \begin{bmatrix} 29.25 & 13.95 & 20.25 \\ 4.95 & 35.55 & 9.45 \\ 0.045 & 0.09 & 45 \end{bmatrix}$$

```
homography_1 = [.8 .2 .3; -.1 .9 -.1; .0005 -.0005 1];
save("meech7-1.png", transform_image(meech, homography_1, "homography"));
save("plant7-1.png", transform_image(plant, homography_1, "homography"));
save("room7-1.png", transform_image(room, homography_1, "homography"));

homography_2 = [29.25 13.95 20.25; 4.95 35.55 9.45; 0.045 0.09 45.0];
save("meech7-2.png", transform_image(meech, homography_2, "homography"));
save("plant7-2.png", transform_image(plant, homography_2, "homography"));
save("room7-2.png", transform_image(room, homography_2, "homography"));
```