CSc433-Spring2023-hw1. Pixel-by-Pixel Due **2/10/24**

Theoretical and implementation assignment.
Basic Operations on Images, 2D transformations and basic JS.
Pair submission is allowed for the implementation part (but for not the theoretical part)

1 Implementation assignment - 75% of the final grade

Benjamin Meyer, CSC433 SP 25

Abstract

In this exercise, you will write a program that animates a picture rotating on the screen. The exercise is slightly more involved for graduate students, for which the image should also scale up and down. The purpose of this assignment is to gain some familiarity in working with JS and to apply operations to buffers of pixels representing images. It also introduces 2D linear transformations.

As you might know already, and as we will study later in the course, OpenGL, WebGL and other libraries could implement this assignment quite painless and fast, using texture and operations as rotation. You **cannot** use any such operations in this homework.

1. You are provided with a template which loads an image (bunny.ppm provided as an example) and draws it into the canvas. The center of the image should be aligned with the center of the display window at all time (rotating around the middle point of the image). Next, animate the image so that it rotates perpendicular to the view axis, either clockwise or counter-clockwise, as shown in the figure below. The image should finish a full cycle every few seconds. For simplicity, you can fix the picture aspect ratio at 1:1 with a resolution of 600×600 pixels.

2. **For grads and/or for bonus:** While rotating, the image should also shrink (scale down) and expand, so for any angle, its corners is on the edge of the displayed image.

   Clarification. This is mandatory for grads, and will buy you a bonus in the case your are a ugrad.

3. For Ugrads: The size of the displayed image could stay the same during the animation. This might mean that depending on the rotation angle θ, not all details of the input image are shown, and corners might be 'chopped'.

4. In this assignment, you should use only the 2D rendering capabilities of JS. In particular, you can initialize the drawing canvas using the function *canvas.getContext('2d')* and

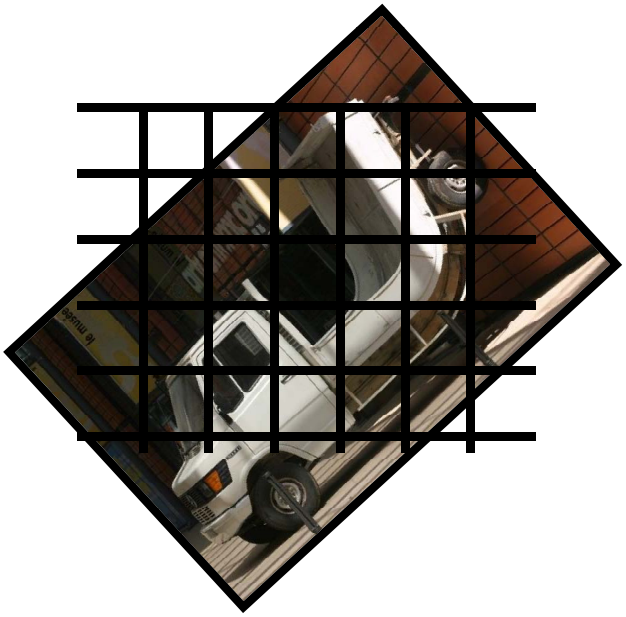

Figure 1: A grid and an arbitrary image.

then paint pixels into the window canvas using *putImageData*. Note that you cannot read from the canvas. Some math functions are provided for you in "Math.js" file. You can extend the math file for your own code.

(Hint: For animating the rotation you may need a function from WebGL API.)

5. You are provided with a template to start working with. The template reads a PPM image and draws it to a canvas. The HTML required for this assignment is provided to you. Feel free to edit both JavaScript and HTML file for this assignment.

6. Use individual matrices to represent the five or six transformations that takes place. For each θ_i (the angle θ and frame $i = 1, 2, 3...$). Calculate these matrices, and calculate their product. Then use this matrix to map output to input pixels. You might be asked to output these matrices.

   Do not recycle the image from frame $i - 1$ to generate the image for frame i.

7. Pay attention to how the colors of image pixels are mapped to the pixels in the window. This is where the 2D transformation will come into play. Do not leave pixels in the display window whose color is not assigned.

8. Your program should be reasonably efficient. So avoid unnecessary nested loops. For the sample image provided in the template your code should have at least around 2 frames per second.

9. Your project should contain the following items:

(a) Your code files (you can only add to the template files or add new JS files if you want). If you plan to submit the assignment using a different language or platform, discuss it first with the instructor.

(b) pic1.ppm, a picture chosen by you, which your program will read from the same directory.

(c) A Readme file, containing your name and email, known errata, and a summary of any additional features you may want to be considered for extra credit.

(d) If you are submitting as a pair of students, please include your partner's name and email. Please both students include their partner's name in their Readme file on Github repository. Both students need to submit their work.

10. Please use the following links to create your repository from the template provided on Github.

    If you a Graduate student:

    https://classroom.github.com/a/WhYqlU5N

    If you an Undergraduate student:

    https://classroom.github.com/a/pSPBy8hC

    Theoretical questions must be submitted to **Gradescope**.

1.1 Grading = point out of 100% of implementation portion of the homework

2 Theoretical Part - 25% of the final grade. This part should be submitted by each student individually.

Submit this part to gradescope - a link will be shared prior to the deadline.

1. Which geometric transformations (from the ones covered in class so far), transfer parallel lines to parallel lines? Which transformation maintains angles?

   Scale and Shear transfer parallel lines to parallel lines. Scale and Rotate maintain angles.

2. Assume the input is a given using nonhomogeneous coordination. What will be the effect of each one of the following matrices? That is every input point p is mapped to the point $M \cdot p$, where M is the matrix in the question[1]. Use geogabra to check your answer - link

   (a)
   $$M = \begin{bmatrix} 1 & 0 \\ 1.5 & 1 \end{bmatrix}$$
   If $p = (x, y)$ then $M \cdot p = (x, y + 1.5x)$ This is a sheering transformation in which the x values remain the same, and the y values increase the further away the point is from the Y-axix.
   Example of an answer. If $p = (x, y)$ then $M \cdot p = (x, y + 1.5x)$ This is a sheering transformation.

   (b)
   $$M = \begin{bmatrix} 1 & 0 \\ -1.5 & 1 \end{bmatrix}$$
   If $p = (x, y)$ then $M \cdot p = (x, y - 1.5x)$ This is a sheering transformation in which the x values remain the same, and the y values decrease the further away the point is from the Y-axix.

   (c)
   $$M_3 = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$
   If $p = (x, y)$ then $M \cdot p = (x, -y)$ This is a reflection transformation across the X-axis.

   (d)
   $$M = \begin{bmatrix} -2 & 0 \\ 0 & -2 \end{bmatrix}$$
   If $p = (x, y)$ then $M \cdot p = (-2x, -2y)$ This is a scaling transformation in which both x and y values are doubled combined with a reflection transformation across both axes.

[1] Since p in this note is a row vector, then formally we should have written $M \cdot p^T$, where p^T is the transpose of p, which is a column vector. $p^T = \begin{bmatrix} x \\ y \end{bmatrix}$. We will ignore this issue when there is no risk of confusion.

(e)

$$M = \begin{bmatrix} \cos\theta & -sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

This is a clockwise rotation around the origin by θ (or counterclockwise by $-\theta$).

(f)

$$M = \begin{bmatrix} \alpha & -\sqrt{1-\alpha^2} \\ \sqrt{1-\alpha^2} & \alpha \end{bmatrix}$$

where $0 \leq \alpha \leq 1$ is a given constant.

This is a matrix that scales by α and shears by a positive value in the y direction and a negative value in the x direction. The closer α is to 1, the less the amount of shear.

(g)

$$M = R(\theta) \cdot \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \cdot R(-\theta)$$

Where $R(\theta)$ is the rotation matrix. That is, $R(\theta) \cdot p$ rotates p by $\theta°$ counterclockwise.

Hint: Note that the "middle" matrix is the matrix M_3 from Question 2c above. No pick a point p and think

i. where is $R(-\theta) \cdot p$,
ii. $M_3 \cdot \big(R(-\theta) \cdot p\big)$
iii. where is $R(\theta) \cdot \Big(M_3 \cdot \big(R(-\theta) \cdot p\big)\Big)$

Once you have figured it out, you could use link to check your answers

This is a counterclockwise rotation by 2θ around the origin point combined with a reflection across the x-axis. $R(-\theta)$ rotates clockwise by theta, M_3 reflects across the axis (with the same effect as if we had reflected and then rotated by θ), and then $R(\theta)$ rotates counterclockwise by a further theta.

3. Similar to the previous question, but now the input given using **homogeneous coordination**.

(a)

$$M = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

If $p = (x, y)$ then $M \cdot p = (2x, 2y)$ This is a scaling transformation that doubles all x and y values.

(b)

$$M = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0.5 \end{bmatrix}$$

If $p = (x, y)$ then $M \cdot p = (4x, 4y)$ This is a scaling transformation that quadruples all x and y values.

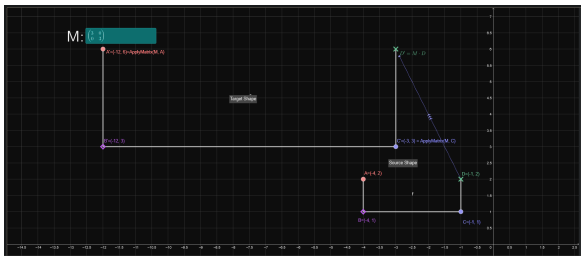

Figure 2: Nonhomogenous 1

(c)

$$M = \begin{bmatrix} 1 & 0 & 0 \\ 0.5 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

If $p = (x, y)$ then $M \cdot p = (x, y + 0.5x)$ This is a sheering transformation in which the x values remain the same, and the y values increase the further away the point is from the Y-axix.

(d)

$$M = \begin{bmatrix} \cos\theta & -sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

This is a clockwise rotation around the origin by θ (or counterclockwise by $-\theta$).

4. Let

$$M = \begin{bmatrix} \cos\theta & -sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

What is M^{-1}? What is M^{-5} ? Hint: use geometric arguments. Explain these arguments.

The inverse of the clockwise rotation matrix M is a counterclockwise rotation by θ. Since a matrix multiplied by its inverse is the identity matrix, and same-magnitude clockwise and counterclockwise rotations cancel out, this holds true.

M^{-5} is a rotation in the counterclockwise direction by 5θ Raising a rotation matrix to any power n is identical to rotating it n times.

5. For the following set of questions, your answers should be submitted as screenshots. Note that I sometimes use randomness to change the input of the question.

A set of questions for Nonhomophonous transformations - here, and another set of questions about homogenous transformations here

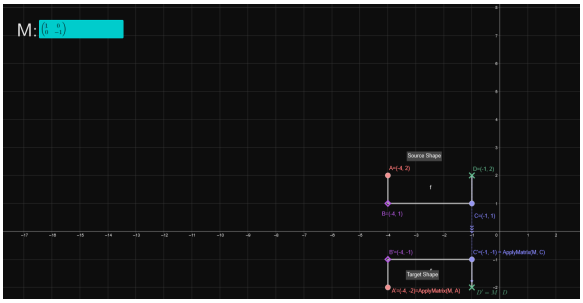

Figure 3: Nonhomogenous 2

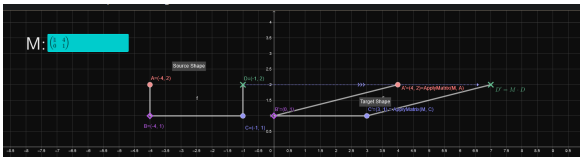

Figure 4: Nonhomogenous 3

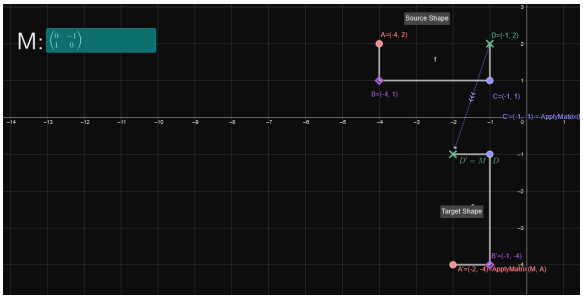

Figure 5: Nonhomogenous 4,

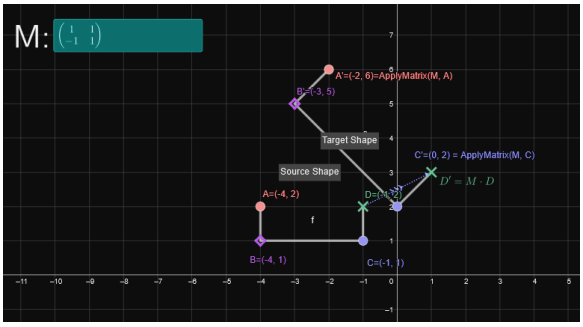

Figure 6: Enter Caption

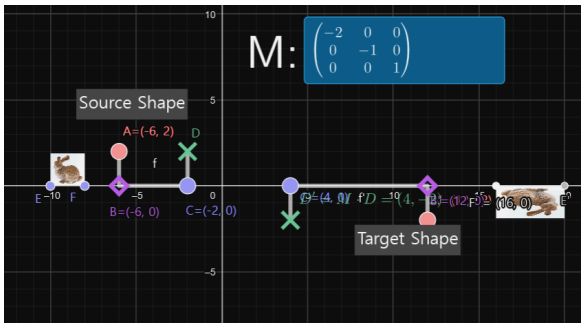

Figure 7: Homogenous 1

Figure 8: Homogenous 2

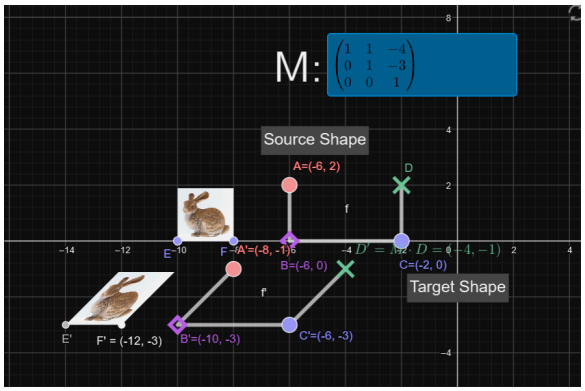

Figure 9: Homogenous 3

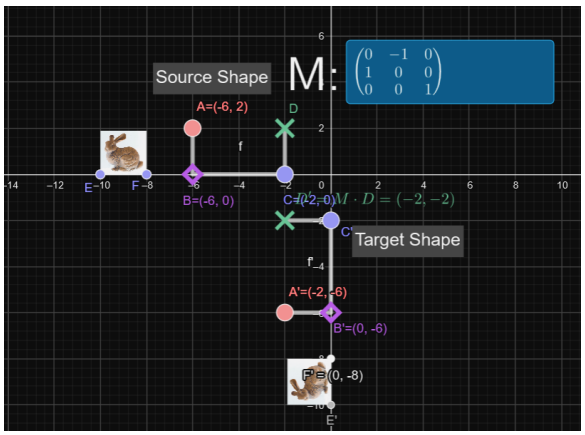

Figure 10: Homogenous 4

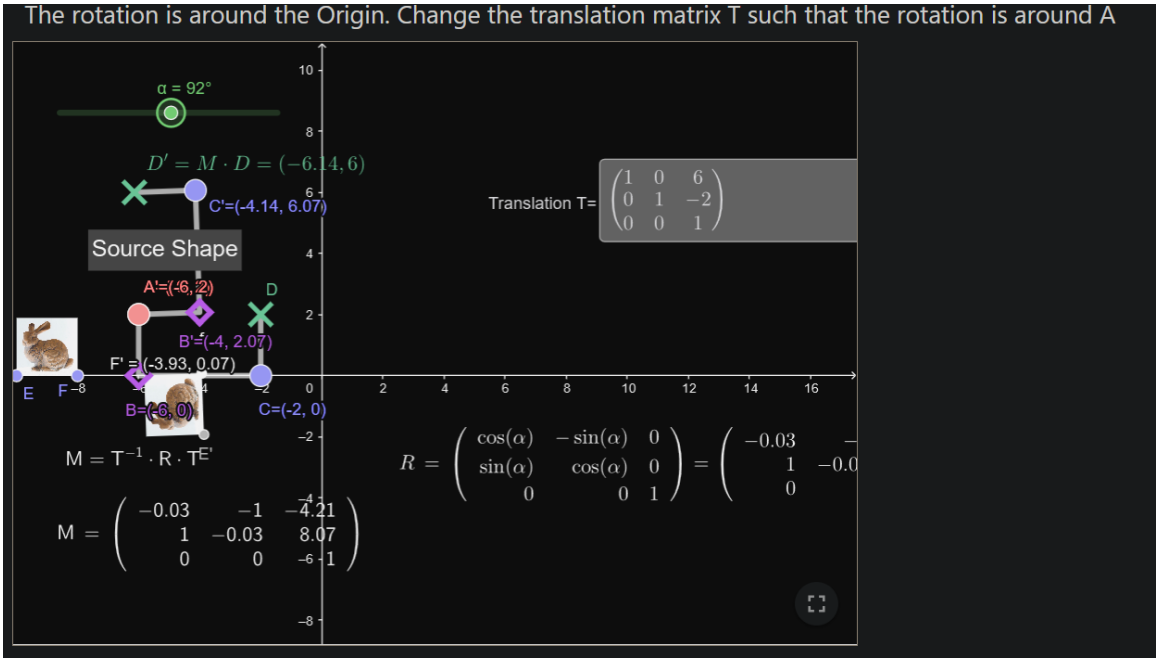

Figure 11: Rotation Change

Reason	Value
Program crashes due to bugs	-10 each bug
Requirements	
Consistent modular coding style, clear organization of files	10 points
External documentation (README.md)	5 points
Class documentation, Internal documentation (Block and Inline). Wherever applicable / for all files	15
Loading: Correctly parsing P6 PPM format and storing it internally.	10
Properly resizing the canvas on load and tracking the image dimensions throughout the program.	20
Displaying: Code correctly displays the PPM image in an HTML Canvas	20
Rotations (correctness of animation) and Scaling (for 533 Students)	20
Correct Usage of homogeneous coordinates and Matrix to represent the transformations	Bonus 5 points

Table 1: Penalties for applied part