EN842100, 2567/2 - Homework #2

Problem 1

(a) Construct a 2D transformation matrix (using homogeneous coordinates) that rotates an object 90 degree **clockwise** around the origin

$$\begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \xrightarrow{\text{rotates } -90^\circ} \begin{bmatrix} \cos(-90) & -\sin(-90) & 0 \\ \sin(-90) & \cos(-90) & 0 \\ 0 & 0 & 1 \end{bmatrix} \longrightarrow \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\therefore \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} y \\ -x \\ 1 \end{bmatrix}$$

(b) In 2D, write the sequence of matrix transformations that would need to be multiplied in order to achieve a rotation around the point (10, -20)

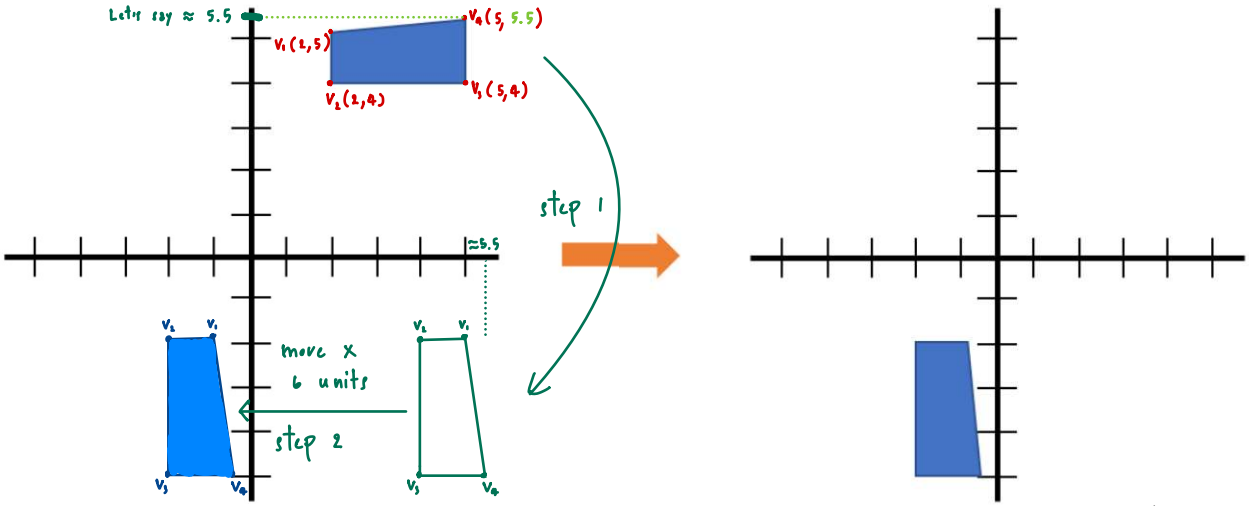
1 # $T_{\text{to origin}}$ translate the point (10, -20) to the origin (0,0) $\begin{bmatrix} 1 & 0 & -10 \\ 0 & 1 & 20 \\ 0 & 0 & 1 \end{bmatrix}$

2 # R rotate around the origin using the rotation matrix. $\begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$

3 # T_{back} translate the point back to its original position $\begin{bmatrix} 1 & 0 & 10 \\ 0 & 1 & -20 \\ 0 & 0 & 1 \end{bmatrix}$

$\therefore$ Final transformation matrix = $T_{\text{back}} \times R \times T_{\text{to origin}}$

(c) Construct a 2D transformation matrix (using homogeneous coordinates) that transform the object as shown below



step 1 rotate the object by 90° clockwise around the origin, matrix to rotate 90° clockwise $\begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

step 2 move the object 6 units to the left (move x 6 units, y 0 units), matrix to move x 6 units and y 0 unit to the left $\begin{bmatrix} 1 & 0 & -6 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

$$\therefore M = \begin{bmatrix} 1 & 0 & -6 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 1 & -6 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

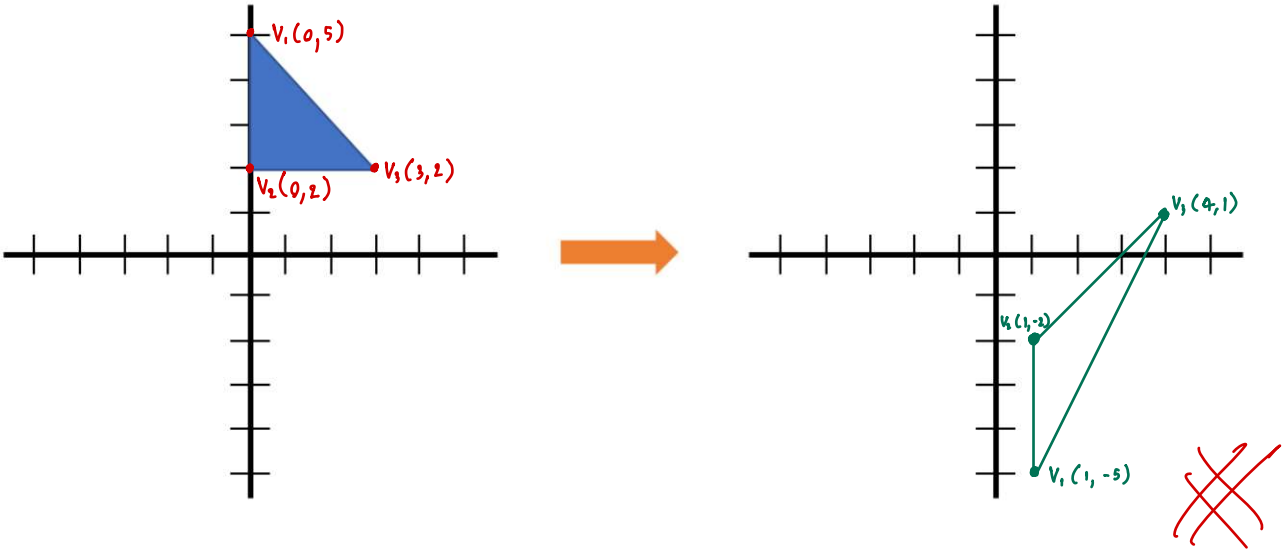
(d) Given a 2D transformation matrix (using homogeneous coordinates) and the original image as shown below, draw the transformed image in the plot on the right.

$$\begin{bmatrix} 1 & 0 & 1 \\ 1 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} x+1 \\ x-y \\ 1 \end{bmatrix}$$

$V_1(0,5) = \begin{bmatrix} 0+1 \\ 0-5 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ -5 \\ 1 \end{bmatrix}$ $V_1(1,-5)$

$V_2(0,2) = \begin{bmatrix} 0+1 \\ 0-2 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ -2 \\ 1 \end{bmatrix}$ $V_2(1,-2)$

$V_3(3,2) = \begin{bmatrix} 3+1 \\ 3-2 \\ 1 \end{bmatrix} = \begin{bmatrix} 4 \\ 1 \\ 1 \end{bmatrix}$ $V_3(4,1)$



Problem 2

(a) What are homogeneous coordinates? Why is the main reason it is used in Computer Graphics?

Homogeneous coordinates add an extra dimension to Euclidean coordinates, enabling transformations like translation, scaling, rotation and perspective projection to be performed using matrix multiplication. It makes combining transformations easier and faster.

(b) What is the equivalent Euclidean coordinate of the following homogeneous coordinate?

$$\begin{bmatrix} -4 \\ 1 \\ 8 \\ -2 \end{bmatrix} \longrightarrow \begin{bmatrix} \frac{-4}{-2} \\ \frac{1}{-2} \\ \frac{8}{-2} \end{bmatrix} \longrightarrow \begin{bmatrix} 2 \\ -0.5 \\ -4 \end{bmatrix}$$

∴ the equivalent Euclidean coordinates are $(2, -0.5, -4)$

(c) Given a 3D transformation matrix (using homogeneous coordinates) that reflects objects through the plane y = 0 below, show that this is equivalent to multiplying a rotation matrix around the z-axis (what is the angle?) and a reflection matrix through the plane x = 0 (what is the order of multiplication?)

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

① reflection matrix through the plane x = 0.

$$\begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

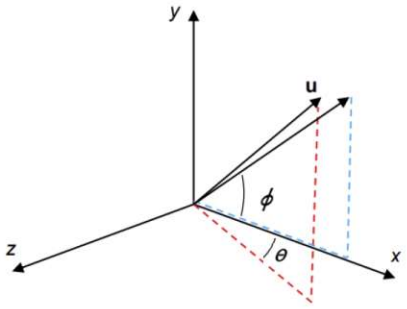
② rotation around the z-axis (to align with the reflection, we rotate by 180°)

③ combining those 2 matrixes above.

$$\begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

∴ Order of multiplication : Reflection comes first, followed by Rotation.

(d) In 3D, we discussed in class that we can rotate an object around arbitrary axis **u** by β degree counter-clockwise as follows: (1) rotating the axis **u** to align with x-axis, (2) rotating around **u** (which is now x-axis) by β degree, and (3) rotating the axis **u** back to its original



The final formula is

$$R_u(\beta) = R_y(-\theta)R_z(\phi)R_x(\beta)R_z(-\phi)R_y(\theta)$$

Alternatively, it is possible that we align axis **u** with y-axis (instead of x-axis) to perform the rotation. What would be the final formula if we do that?

The final formula is

$$R_u(\beta) = R_x(-\theta)R_z(-\phi)R_y(\beta)R_z(\phi)R_x(\theta)$$

Problem 3

(a) Explain what each of the following transformations is used for in Computer Graphics, and what is the order that they are applied.

1. Perspective Projection Transformation

   Adds the depth by making distant object appear smaller

2. Model Transformation

   Moves, rotates, or scales object in the scene.

3. Orthographic Projection Transformation

   Projects 3D objects onto 2D screen without depth effect.

4. Viewport Transformation

   Maps the final 2D image to the screen or window.

5. View Transformation

   Moves the camera to define what is seen in the scene.

Order: Model transformation → View transformation → Perspective Projection transformation → Orthographic Projection transformation → Viewport transformation

(b) Explain what the vectors **eye**, **n**, **u**, **v** vectors are, and how they are used in view transformation.

eye : The position of camera in the 3D world

n : A vector pointing in the direction the camera looks

u : A vector pointing to the right side of the camera

v : A vector pointing upward from the camera

How they're used in view transformation:

These vectors define the camera's position and orientation, It helps to convert world coordinates into the camera's view for rendering the scene.