Problem Statement

Write a program to illustrate composite transformations (translation, rotation, scaling) as well as shearing, in graphics window.

Background Theory

In computer graphics, geometric transformations are essential for manipulating shapes and objects within a scene. These transformations modify an object’s position, orientation, size, or shape. The most common transformations are translation, rotation, scaling, and shearing

1. TRANSLATION

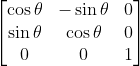
Translation shifts an object from one location to another in the 2D plane. It does not alter shape, size, or orientation, just position. Its example matrix is:



where, `tx` and `ty` are the distances to move in x and y directions.

2. ROTATION

Rotation turns an object around a fixed point, typically the origin. It preserves the shape and size but changes the direction it faces. It’s useful in animations and object orientation control. Its example matrix (for counterclockwise rotation by θ):



3. SCALING

Scaling resizes an object, i.e. enlarges or shrinks it along x and/or y axes. Uniform scaling keeps the proportions the same; non-uniform changes the aspect ratio. It’s example matrix is:



where `sx` and `sy` are the scale factors for width and height.

4. SHEARING

Shearing skews an object, i.e. it distorts the shape by shifting layers. It’s commonly used to simulate 3D perspectives or italic effects in fonts.

X-Shear Matrix:



Y-Shear Matrix:



where `shx` and `shy` determine how much to slant in respective directions.

Algorithm

1. Start
2. Input triangle coordinates (x1,y1), (x2,y2) and (x3,y3).
3. Calculate maximum and minimum height and width and draw axes respectively.
4. Draw the triangle based on input coordinates.
5. Translation

* Input translation vector (tx, ty)
* Draw triangle with updated coordinates as:
  + x1+tx, x2+tx, x3+tx, y1-ty, y2-ty, y3-ty for x1,x2,x3,y1,y2,y3

1. Rotation

* Input the angle by which triangle is to be rotated.
* Calculated new points as:

xr1 = centerX + (x1-centerX) \* cos(thetaRadian) - (y1-centerY) \* sin(thetaRadian);

xr2 = centerX + (x2-centerX) \* cos(thetaRadian) - (y2-centerY) \* sin(thetaRadian);

xr3 = centerX + (x3-centerX) \* cos(thetaRadian) - (y3-centerY) \* sin(thetaRadian);

yr1 = centerY + (x1-centerX) \* sin(thetaRadian) + (y1-centerY) \* cos(thetaRadian);

yr2 = centerY + (x2-centerX) \* sin(thetaRadian) + (y2-centerY) \* cos(thetaRadian);

yr3 = centerY + (x3-centerX) \* sin(thetaRadian) + (y3-centerY) \* cos(thetaRadian);

* Draw new triangle with updated coordinates

.

1. Scaling

* Input scale factors Sx and Sy.
* Calculate new coordinates as:

xs1=x1\*sx+x1 \* (1 - sx);

xs2=x2\*sx+x1 \* (1 - sx);

xs3=x3\*sx+x1 \* (1 - sx);

ys1=y1\*sy+y1 \* (1 - sy);

ys2=y2\*sy+y1 \* (1 - sy);

ys3=y3\*sy+y1 \* (1 - sy);

* Draw triangle with updated coordinates.

1. Shearing

* Input the values of shx and shy.
* Calculate new coordinates as:

xsh1 = x1 + shx \* (centerY - y1);

ysh1 = y1 + shy \* (x1 - centerX);

xsh2 = x2 + shx \* (centerY - y2);

ysh2 = y2 + shy \* (x2 - centerX);

xsh3 = x3 + shx \* (centerY - y3);

ysh3 = y3 + shy \* (x3 - centerX);

* Draw triangle with updated coordinates.

1. Stop.