Kathmandu University

Department of Computer Science and Engineering

Dhulikhel, Kavre

COMP 342 (Computer Graphics)

Lab 4 Report

Submitted To:

Mr. Dhiraj Shrestha

Department of Computer Science and Engineering

Submitted By:

Mani Dumaru

Roll no.: 15

CE-2019 3rd Year/ 2nd Semester

Submission Date: 1st May, 2023

Implementation of 2D Transformations using Homogenous Coordinates

Implementing 2D Translation

Algorithm:

Input: Coordinates of a triangle, translation factor

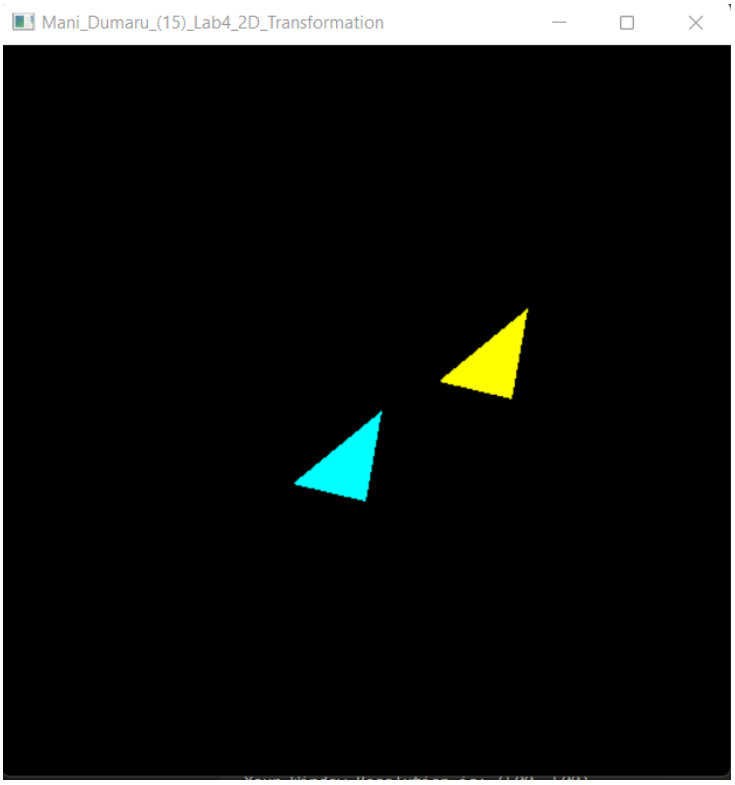
Output: Translated triangle

1. Given coordinates are plotted to form a triangle
2. Use the translation matrix on each coordinate to get its translated coordinate.

$$\text{Translation Matrix: } \begin{matrix} 1 & 0 & tx \\ 0 & 1 & ty \\ 0 & 0 & 1 \end{matrix} \cdot \begin{matrix} x \\ y \\ 1 \end{matrix}$$

3. Use the coordinates obtained from the above product to plot a new translated triangle

Output:



Implementing 2D scaling

Algorithm:

Input: Coordinates of a triangle, scaling factor

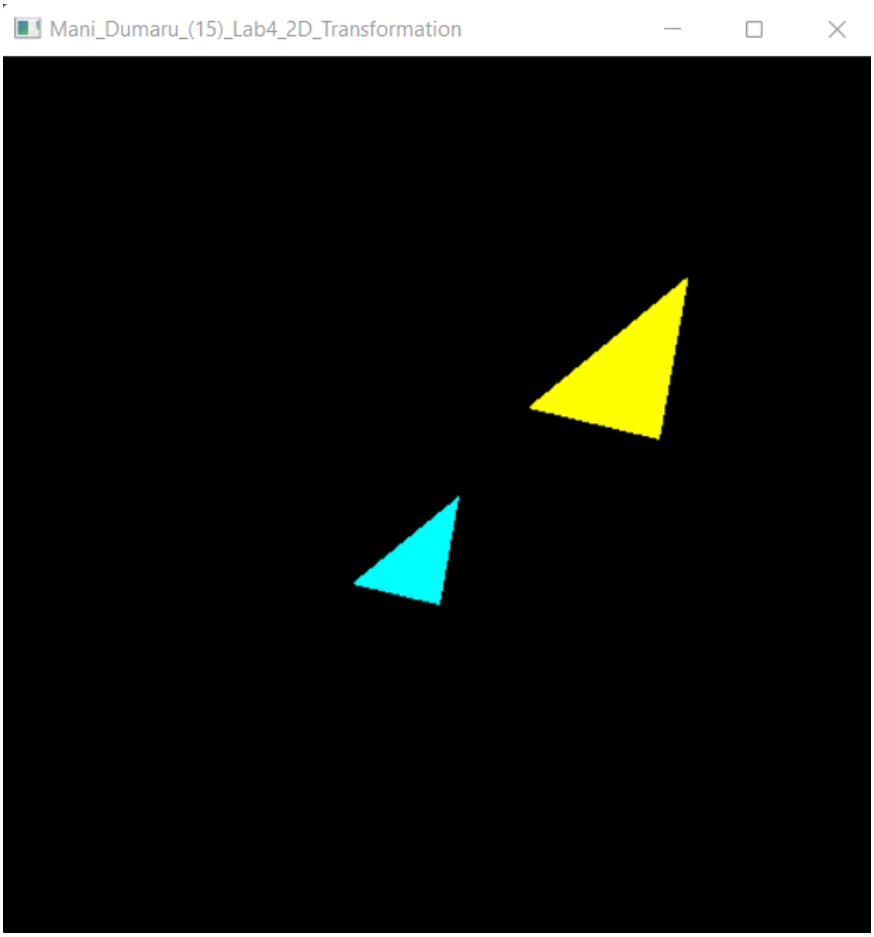
Output: Translated triangle

1. Given coordinates are plotted to form a triangle
2. Use the scaling matrix on each coordinate to get its scaled coordinate.

$$\text{Scaling Matrix: } \begin{matrix} Sx & 0 & 0 \\ 0 & Sy & 0 \\ 0 & 0 & 1 \end{matrix} \cdot \begin{matrix} x \\ y \\ 1 \end{matrix}$$

3. Use the coordinates obtained from the above product to plot a new scaled triangle

Output:



Implementing 2D rotation

Algorithm:

Input: Coordinates of a triangle, angle of rotation θ

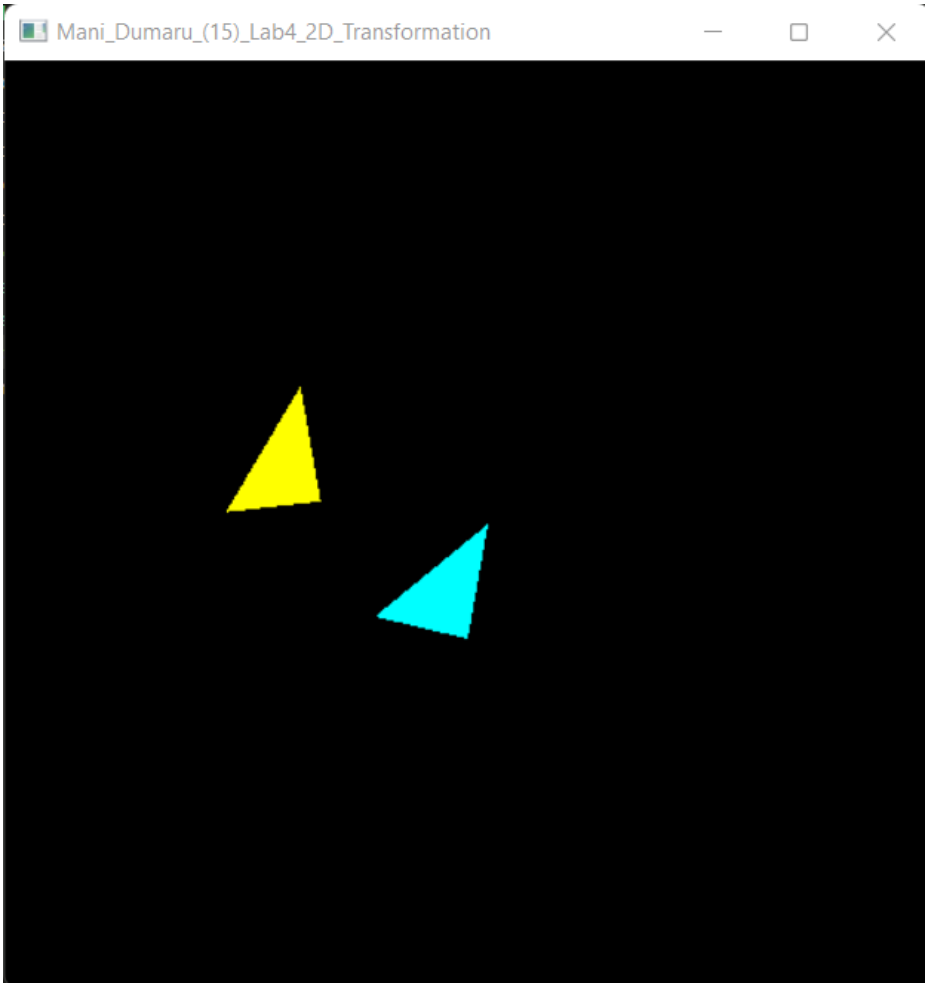
Output: Translated triangle

1. Given coordinates are plotted to form a triangle
2. Use the rotation matrix on each coordinate to get its rotated coordinate.

Rotation Matrix:
$$\begin{matrix} cos\theta & -sin\,\theta & 0 \\ sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{matrix} \cdot \begin{matrix} x \\ y \\ 1 \end{matrix}$$

3. Use the coordinates obtained from the above product to plot a new rotated triangle

Output:



Implementing 2D shearing:

Algorithm:

Input: Coordinates of a triangle, angle of rotation θ

Output: Translated triangle

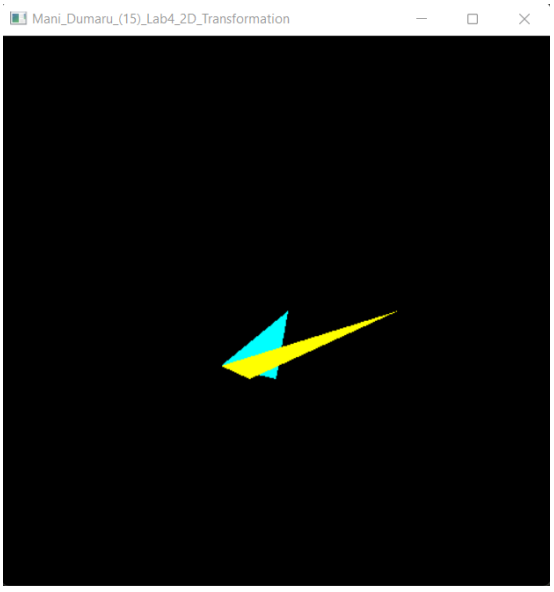
1. Given coordinates are plotted to form a triangle
2. Use the shearing matrix on each coordinate to get its sheared coordinate.

$$\text{Shearing Matrix: } \begin{matrix} 1 & shx & -shx * yref \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{matrix} \cdot \begin{matrix} x \\ y \\ 1 \end{matrix} \quad \text{(X shear with reference to y-axis)}$$

$$\text{Shearing Matrix: } \begin{matrix} 1 & 0 & 0 \\ shy & 1 & -shy * xref \\ 0 & 0 & 1 \end{matrix} \cdot \begin{matrix} x \\ y \\ 1 \end{matrix} \quad \text{(Y shear with reference to x-axis)}$$

3. Use the coordinates obtained from the above product to plot a new sheared triangle

Output:



X shear with reference to y-axis *Y shear with respect to x-axis*

Source Code for Translation, Scaling, Rotation and Shear: [transformations.py]

Implementing 2D reflection:

Algorithm:

Input: Coordinates of a triangle, angle of rotation θ

Output: Translated triangle

1. Given coordinates are plotted to form a triangle
2. Use the reflection matrix on each coordinate to get its reflected coordinate.

Reflection about the line Y=0 (x-axis):

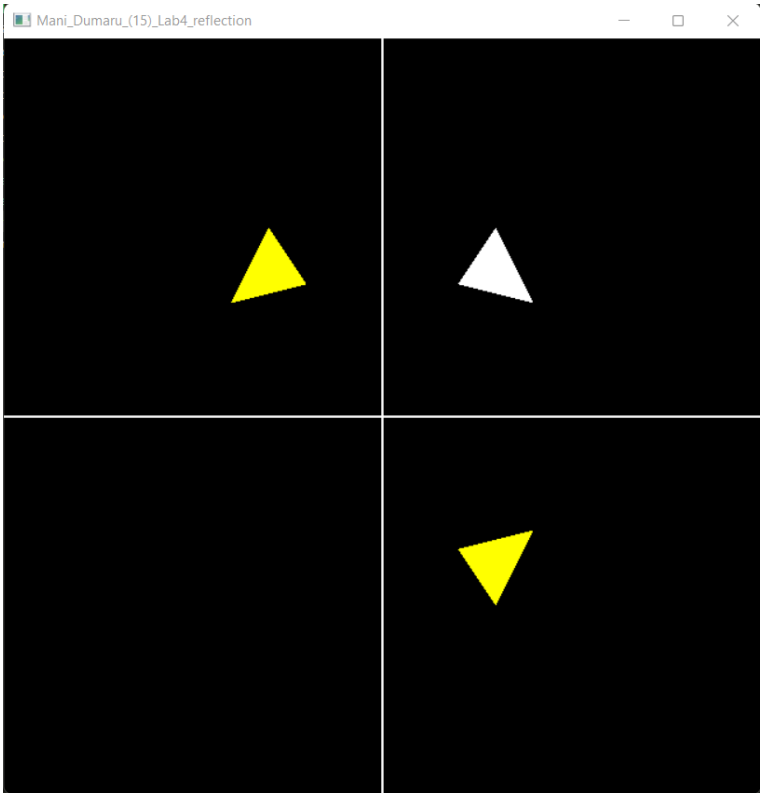
$$\begin{matrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{matrix} \cdot \begin{matrix} x \\ y \\ 1 \end{matrix}$$

Reflection about the line X=0 (Y-axis):

$$\begin{matrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{matrix} \cdot \begin{matrix} x \\ y \\ 1 \end{matrix}$$

3. Use the coordinates obtained from the above product to plot a new reflected triangle

Output



Source Code for reflection: reflection.py

Conclusion:

Hence, implementation of various basic 2D transformations were done using PyOpenGL library.