Laboratory work 4

1 Objectives

This laboratory presents the key notions on 2D and 3D transformations (translation, scale, rotation).

2 Defining 2D and 3D points

A 2D point is defined in a homogenous coordinate system by (x*w, y*w, w). For simplicity in bi-dimensional systems the w parameter is set to 1. Therefore, the point definition is (x, y, 1).

Another representation for the point is the following: $P = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$.

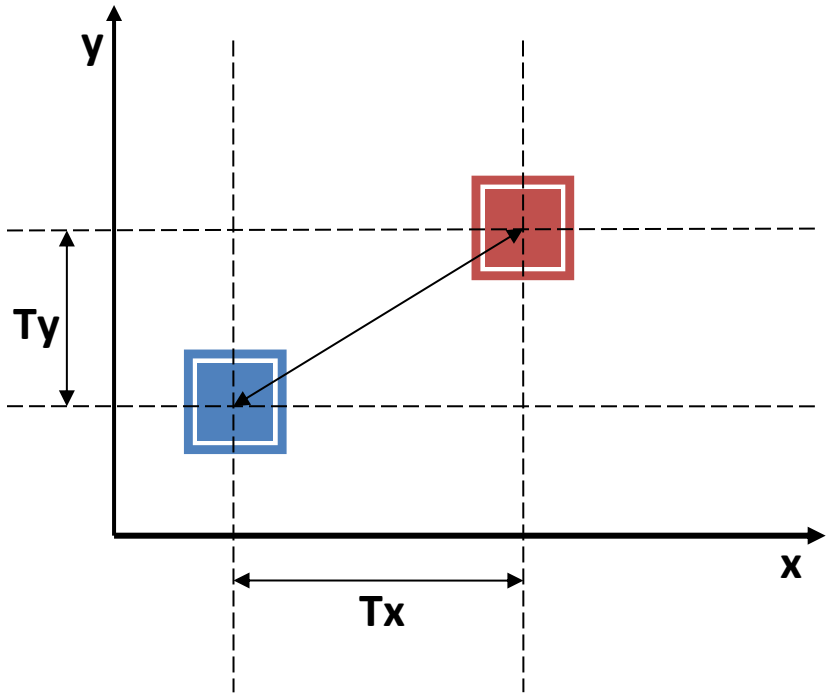
Similar in 3D we define points as (x*w, y*w, z*w, w) and we represent the point as a column vector:

$$P = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

3 Transformations

3.1 2D Translation

The translation transformation is used to move an object (point) by a given amount.



The matrix for the translation operation is the following:

$$T = \begin{bmatrix} 1 & 0 & Tx \\ 0 & 1 & Ty \\ 0 & 0 & 1 \end{bmatrix}$$

where Tx and Ty represent the translation factors on x and y axes. If we apply the transformation to the 2D point, $P' = T * P$, we obtain the new coordinates for that point:

$$x' = x + Tx$$

$$y' = y + Ty$$

The matrix for the inverse transformation is the following:

$$T = \begin{bmatrix} 1 & 0 & -Tx \\ 0 & 1 & -Ty \\ 0 & 0 & 1 \end{bmatrix}$$

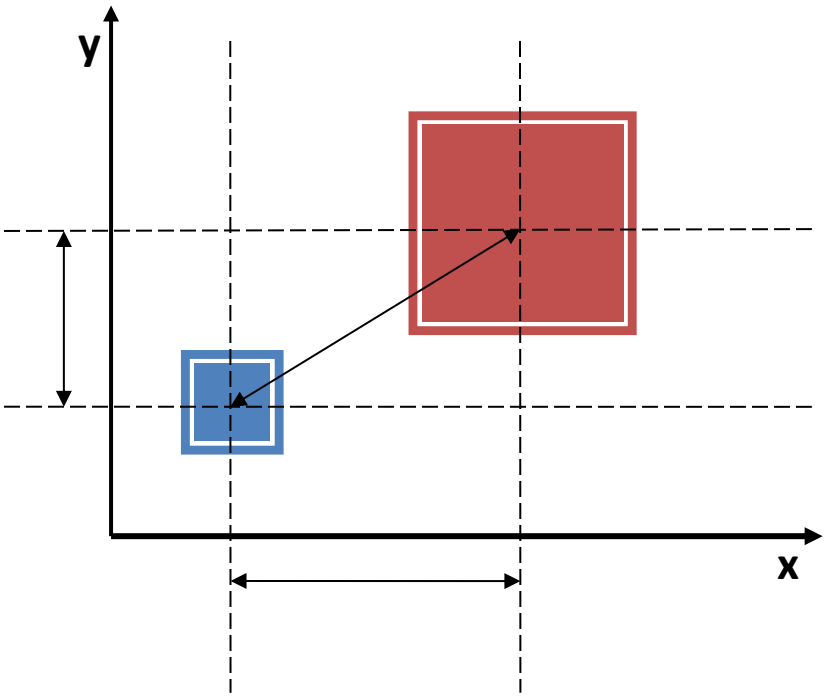
3.2 3D Translation

The only difference from the 2D case is that here we have one more coordinate and the transformation matrix will be 4x4.

$$T = \begin{bmatrix} 1 & 0 & 0 & Tx \\ 0 & 1 & 0 & Ty \\ 0 & 0 & 1 & Tz \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

3.3 2D Scale

The scale transformation enlarges or reduces an object. The transformation is *relative to the origin*.



The matrix for the scale operation is the following:

$$S = \begin{bmatrix} Sx & 0 & 0 \\ 0 & Sy & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

where Sx and Sy represent the scale factors on x and y axes. If the Sx and Sy factors are equal then the scaling transformation is uniform. If the Sx and Sy factors are not equal then the scaling transformation is non-uniform. If we apply the transformation to the 2D point, $P' = S * P$, we obtain the new coordinates for that point:

$$x' = x * Sx$$

$$y' = y * Sy$$

If you set the scaling factors to +/- 1 then you can reflect the original shape.

The matrix for the inverse transformation is the following:

$$S = \begin{bmatrix} 1/Sx & 0 & 0 \\ 0 & 1/Sy & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

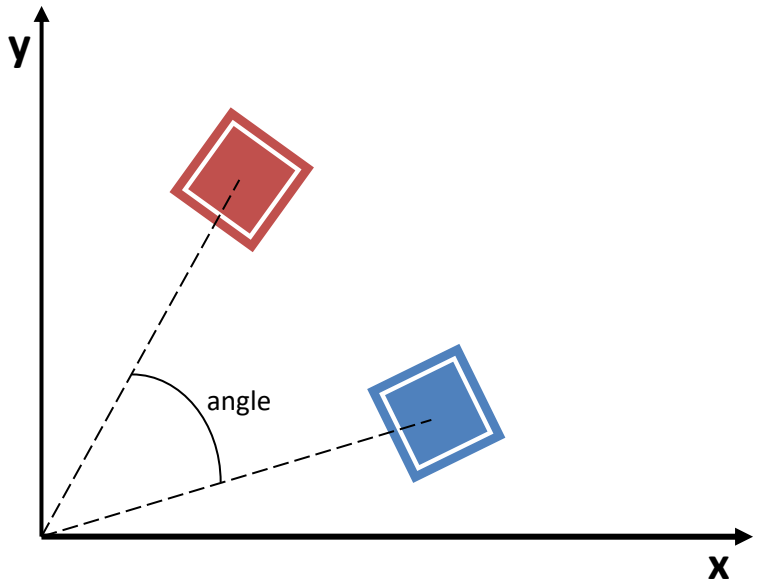
3.4 3D Scale

The 3D matrix representation of the scale transformation is the following:

$$S = \begin{bmatrix} Sx & 0 & 0 & 0 \\ 0 & Sy & 0 & 0 \\ 0 & 0 & Sz & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

3.5 2D Rotation

This transformation rotates an object with a given angle. This transformation is also relative to the origin.



The matrix for the rotation operation is the following:

$$R = \begin{bmatrix} \cos \propto & -\sin \propto & 0 \\ \sin \propto & \cos \propto & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

where$\propto$ represents the rotation angle. If we apply the transformation to the 2D point, $\mathrm{P}' = \mathrm{P} * \mathrm{R}$, we obtain the new coordinates for that point:

$$x' = x * \cos \propto \ - y * \sin \propto$$

$$y' = x * \sin \propto \ + y * \cos \propto$$

The matrix for the inverse transformation is the following:

$$R = \begin{bmatrix} \cos \propto & \sin \propto & 0 \\ -\sin \propto & \cos \propto & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

3.6 3D Rotation

We specify rotation in the 3D space independently on the x, y, and z axis. Rotation around the z axis is similar to the rotation in 2D (the z coordinate remains unchanged).

$$R_z = \begin{bmatrix} \cos \propto & -\sin \propto & 0 & 0 \\ \sin \propto & \cos \propto & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$R_x = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos \propto & -\sin \propto & 0 \\ 0 & \sin \propto & \cos \propto & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$R_y = \begin{bmatrix} \cos \propto & 0 & \sin \propto & 0 \\ 0 & 1 & 0 & 0 \\ -\sin \propto & 0 & \cos \propto & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

4 Assignment

Download the source code from the web repository. You have to implement the methods inside the source file (transform.cpp). The header file (transform.h) contains the definition of methods that should be implemented.