Lab Report

Week 3

Ritobroto Maitra (1301CS50)

19/01/2016

Title

- Applying transformations to basic primitives (a line, a circle and a polygon).

Procedure

We use a combination of in built matrix transformations along with some hard coded matrix multiplications to do the requisite modeling transformations.

Transformation Matrices

1. Scaling

$$\begin{bmatrix} 0.33 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

2. Reflection

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

3. Shear

$$\begin{bmatrix} 1 & 0 & 0 \\ 3 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

4. Rotation

$$\begin{bmatrix} cos(45) & -sin(45) & 0 \\ sin(45) & cos(45) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

5. Translation

$$\begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0.1 \\ 0 & 0 & 1 \end{bmatrix}$$

Drawing the Basic Shapes in OpenGL

The following are the functional parts of the code required to plot the shapes.

Line

We create two points and draw a line between them.

```
  glVertex3f(0.0, 0.0, 0.0);
  glVertex3f(0.5, -0.5, 0);
```

Circle

We create a 0.5 radius circle using sin(i) and cos(i) values as follows.

```
  for(i=0;i<10*3.14;i+=0.0001){
    glVertex3f(cos(i)/2, sin(i)/2, 0.0);
  }
```

Polygon

The polygon is created using the following lines.

```
    glBegin(GL_LINES);
    glVertex3f(0.5, 0.5, 0.0);
    glVertex3f(0.0, 0.0, 0.0);
    glVertex3f(0.0, 0.0, 0.0);
    glVertex3f(-0.5, 0.5, 0.0);
    glVertex3f(-0.5, 0.5, 0.0);
    glVertex3f(-0.7, -0.5, 0.0);
    glVertex3f(-0.7, -0.5, 0.0);
    glVertex3f(-0.2, -0.6, 0.0);
    glVertex3f(-0.2, -0.6, 0.0);
    glVertex3f(0.5, 0.5, 0.0);
```

Transformation Code in OpenGL

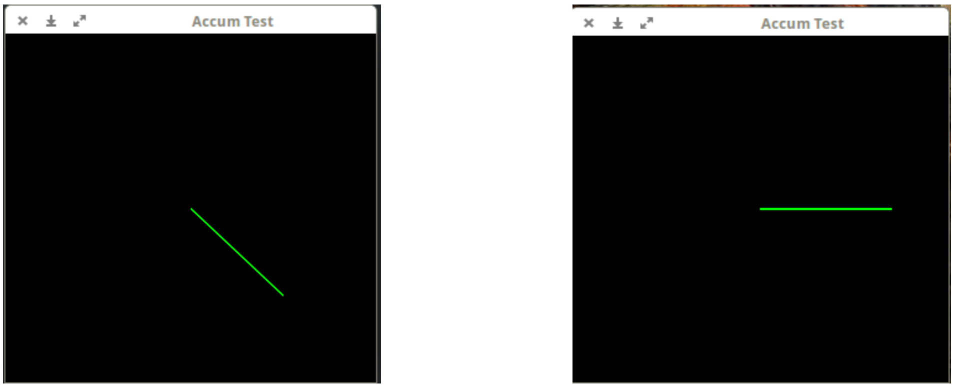
The transformation is fed by an input of options which is dealt by the following switch case mechanism.

```
static void Key(unsigned char key, int x, int y)
{
  GLfloat m[] = {1, 0, 0, 0,   3, 1, 0, 0,   0, 0, 1, 0,   0, 0, 0, 1};
  GLfloat n[] = {1, 0, 0, 0,  -3, 1, 0, 0,  0, 0, 1, 0,   0, 0, 0, 1};
  glMatrixMode(GL_MODELVIEW);
    switch (key) {
      case ’1’:
      glScalef(0.33, 1, 1);
      glutPostRedisplay();
      break;
        case ’2’:
      glScalef(3, 1, 1);
      glutPostRedisplay();
      break;
    case ’3’:
      glRotatef(45, 0, 0, 1);
      glutPostRedisplay();
      break;
    case ’4’:
      glMultMatrixf(m);
      glutPostRedisplay();
      break;
    case ’5’:
      glMultMatrixf(n);
      glutPostRedisplay();
      break;
    case ’6’:
      glScalef(-1, 1, 1);
      glutPostRedisplay();
      break;
    case ’7’:
      glTranslatef(0, 0.1, 0);
      glutPostRedisplay();
      break;
    case ’8’:
      glTranslatef(0, -0.1, 0);
      glutPostRedisplay();
      break;
        case 27:
    exit(0);
    }
}
```

Transformation Code in MATLAB

```
while in ~= 0
in = input(prompt);
    if in==1
        % scaling
        T = [2, 0, 0; 0, 1.5, 0; 0, 0, 1];
        pts = T*pts;
        plot(pts(1,:), pts(2,:), ’r*-’);
        axis([-10, 10, -10, 10]);
        display(pts)
    elseif in==-1
        % rev scaling
        T = [0.5, 0, 0; 0, 2/3, 0; 0, 0, 1];
        pts = T*pts;
        plot(pts(1,:), pts(2,:), ’r*-’);
        axis([-10, 10, -10, 10]);
        display(pts)
    elseif in==2
        % reflection
        T = [-1, 0, 0; 0, 1, 0; 0, 0, 1];
        pts = T*pts;
        plot(pts(1,:), pts(2,:), ’r*-’);
        axis([-10, 10, -10, 10]);
        display(pts)
    elseif in==3
        % shear
        T = [1, 1, 0; 0, 1, 0; 0, 0, 1];
        pts = T*pts;
        plot(pts(1,:), pts(2,:), ’r*-’);
        axis([-10, 10, -10, 10]);
        display(pts)
    elseif in==-3
        % rev shear
        T = [1, -1, 0; 0, 1, 0; 0, 0, 1];
        pts = T*pts;
        plot(pts(1,:), pts(2,:), ’r*-’);
        axis([-10, 10, -10, 10]);
        display(pts)
    elseif in==4
        % rotation
        T = [cosd(45), -sind(45), 0; sind(45), cosd(45), 0; 0, 0, 1];
        pts = T*pts;
        plot(pts(1,:), pts(2,:), ’r*-’);
        axis([-10, 10, -10, 10]);
        display(pts)
    elseif in==5
        % translation
        T = [1, 0, 1;0, 1, 0.5; 0, 0, 1];
        pts = T*pts;
        plot(pts(1,:), pts(2,:), ’r*-’);
        axis([-10, 10, -10, 10]);
        display(pts)
     elseif in==-5
        % rev translation
        T = [1, 0, -1;0, 1, -0.5; 0, 0, 1];
        pts = T*pts;
        plot(pts(1,:), pts(2,:), ’r*-’);
        axis([-10, 10, -10, 10]);
        display(pts)
    end
```

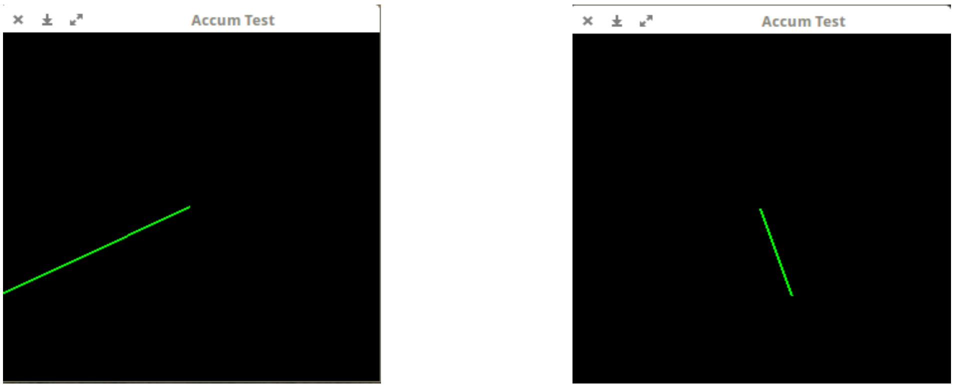
Examples



(a) Initial Line

(b) Line : Rotation

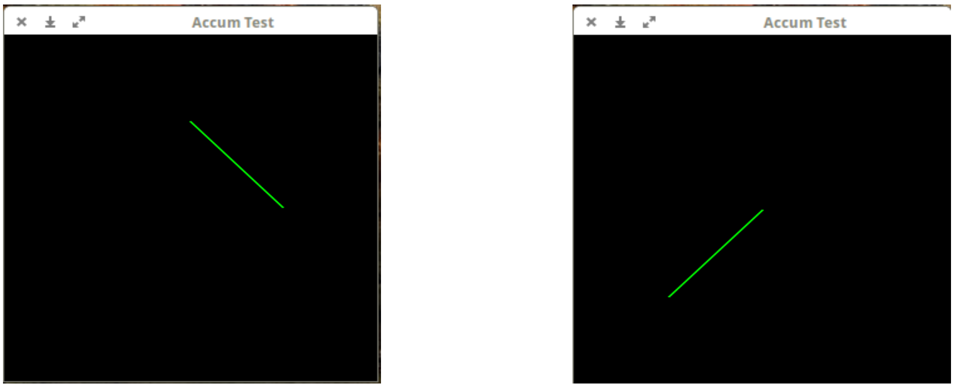
FIGURE 1 – Transformations on Line : Rotation



(a) Line : Shear

(b) Line Scale

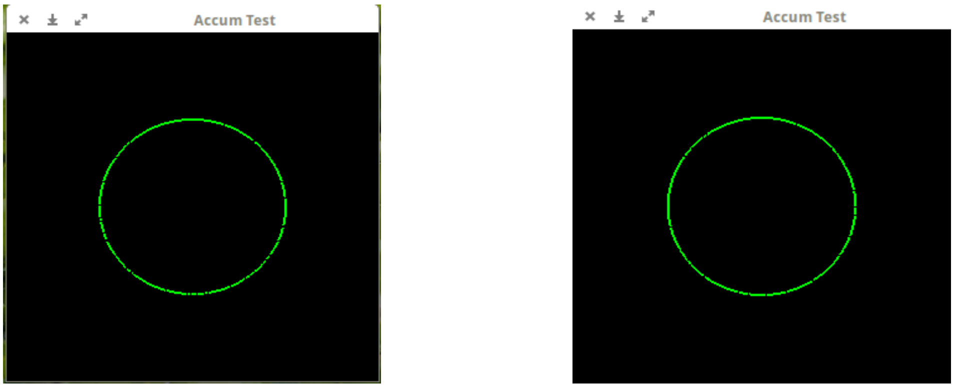
FIGURE 2 – Transformations on Line : Shear, Scale



(a) Line : Translate

(b) Line : Reflect

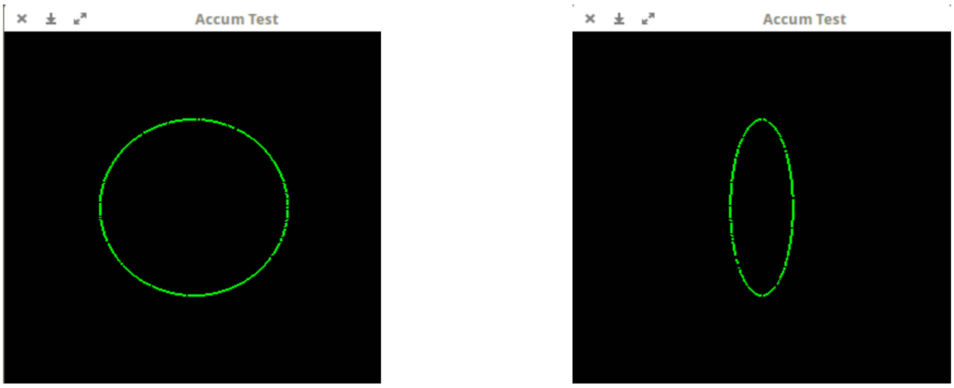
FIGURE 3 – Transformations on Line : Translate, Reflect



(a) Initial Circle

(b) Circle : Rotation

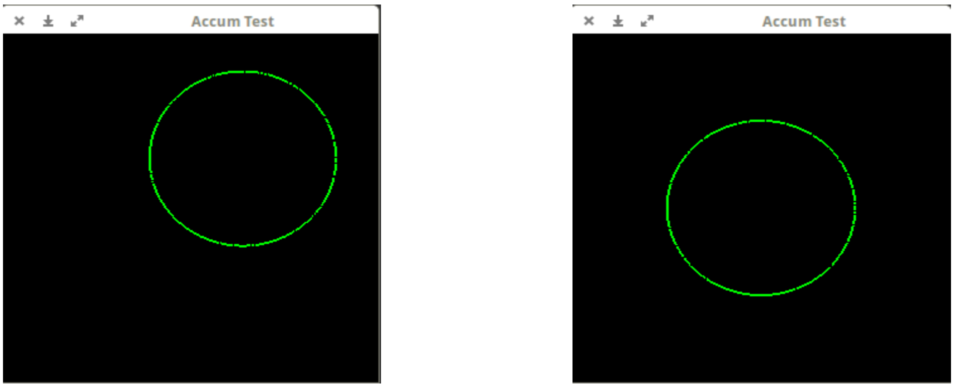
FIGURE 4 – Transformations on Circle : Rotation



(a) Circle : Shear

(b) Line Scale

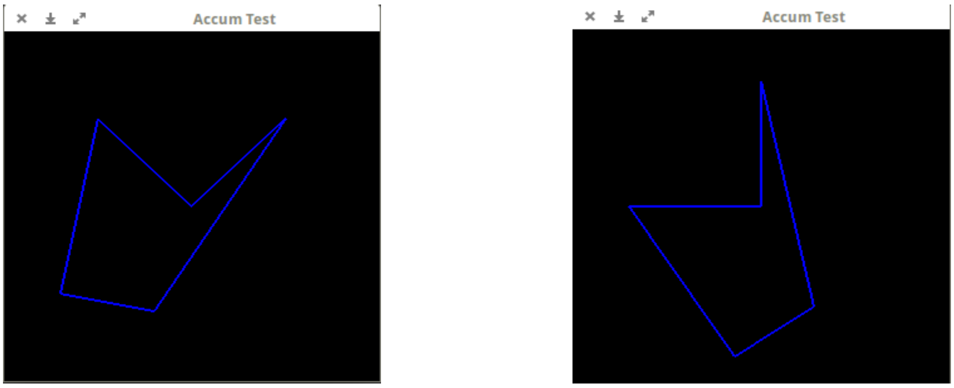
FIGURE 5 – Transformations on Circle : Shear, Scale



(a) Circle : Translate

(b) Circle : Reflect

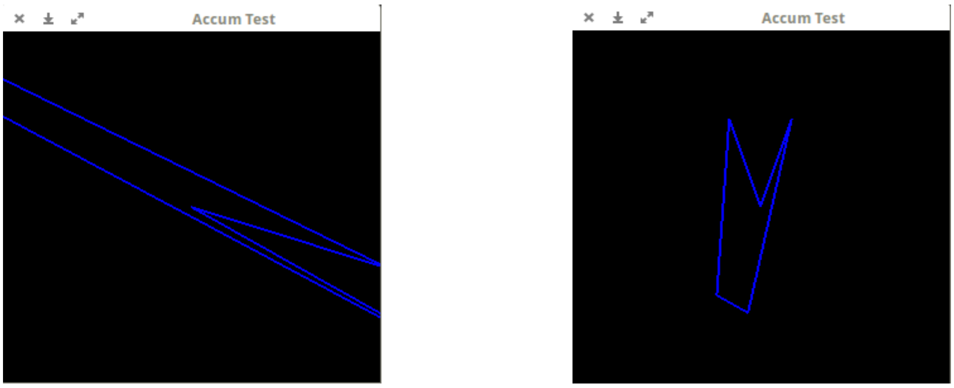
Figure 6 – Transformations on Circle : Translate, Reflect



(a) Initial polygon

(b) polygon : Rotation

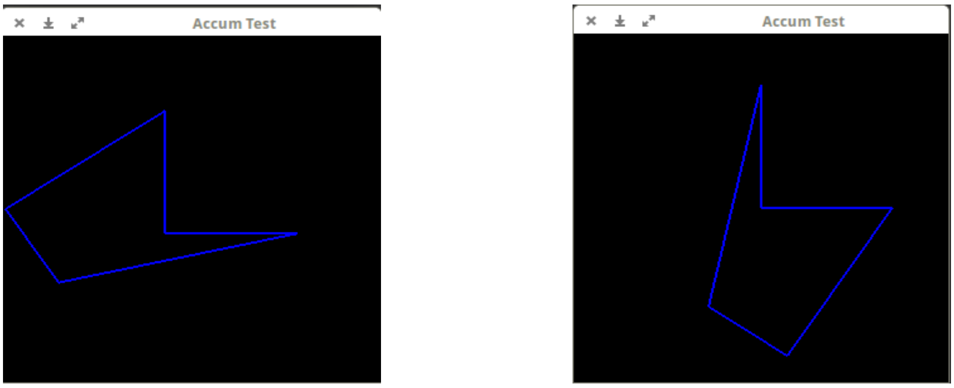
Figure 7 – Transformations on polygon : Rotation



(a) Polygon : Shear

(b) Polygon : Scale

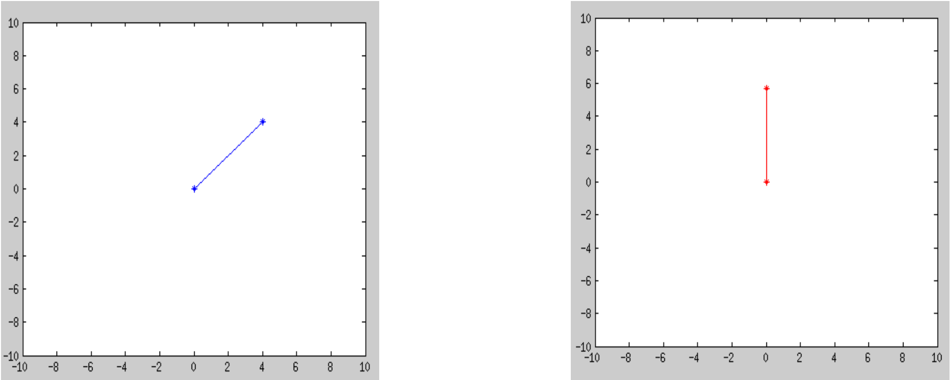
Figure 8 – Transformations on Polygon : Shear, Scale



(a) Polygon : Translate

(b) Polygon : Reflect

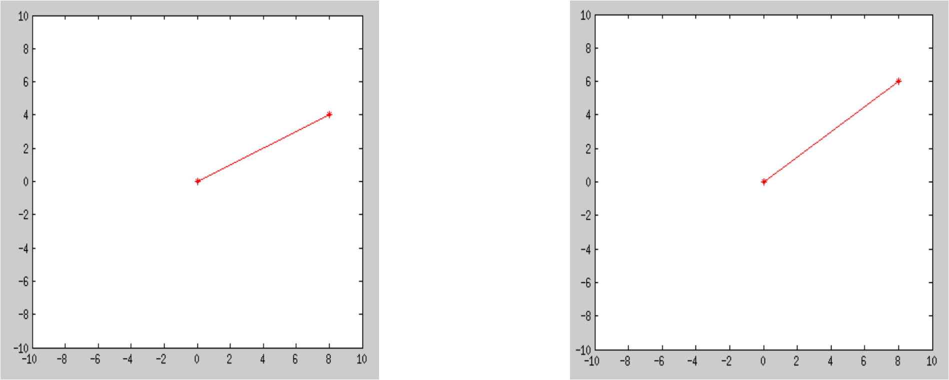
FIGURE 9 – Transformations on Polygon : Translate, Reflect



(a) Initial Line

(b) Line : Rotation

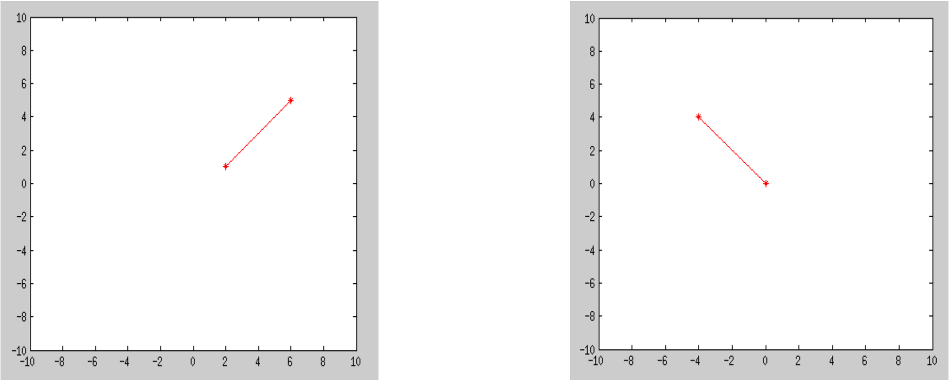
FIGURE 10 – Transformations on Line : Rotation



(a) Line : Shear

(b) Line Scale

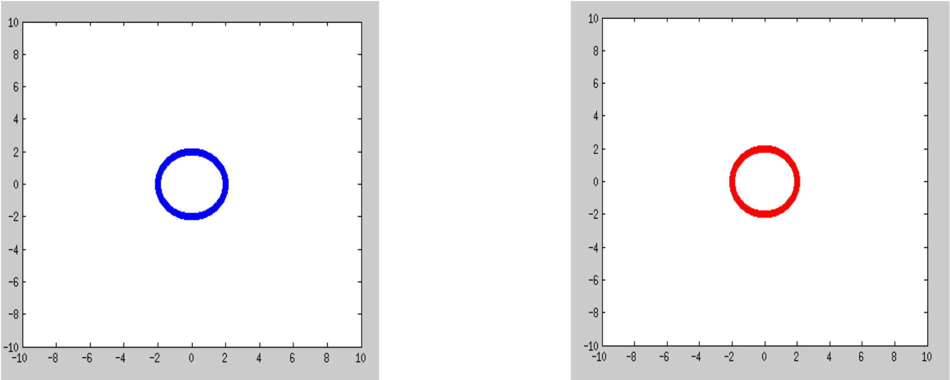
FIGURE 11 – Transformations on Line : Shear, Scale



(a) Line : Translate

(b) Line : Reflect

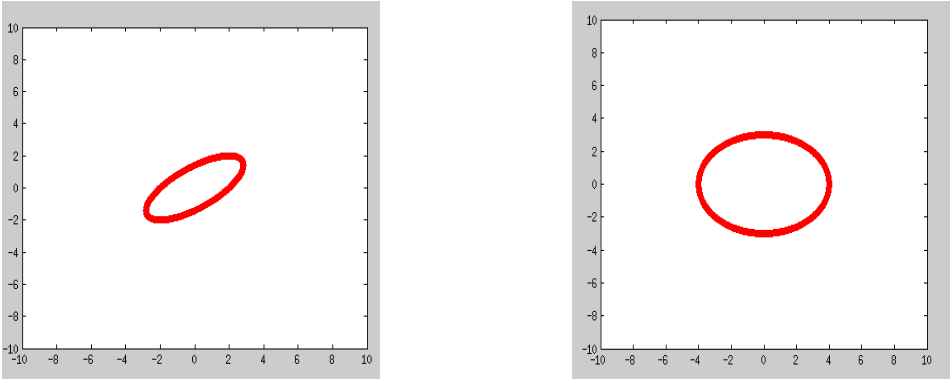
FIGURE 12 – Transformations on Line : Translate, Reflect



(a) Initial Circle

(b) Circle : Rotation

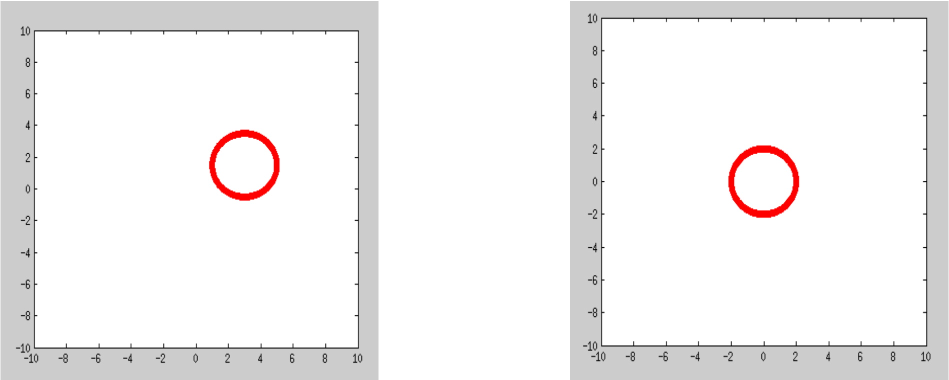
FIGURE 13 – Transformations on Circle : Rotation



(a) Circle : Shear

(b) Circle : Scale

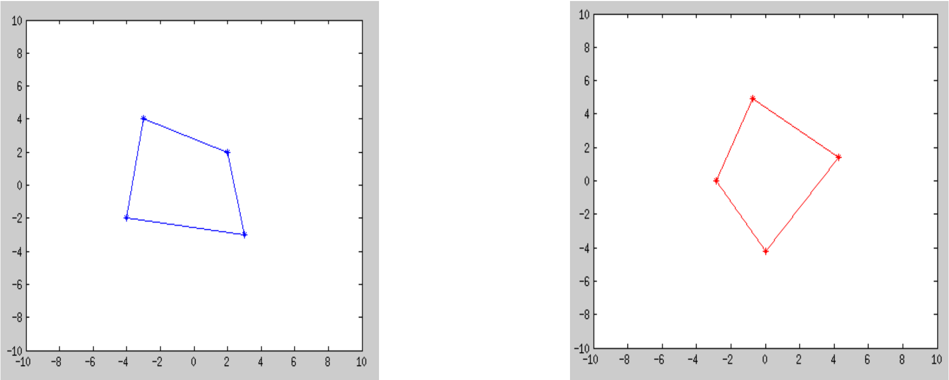
FIGURE 14 – Transformations on Circle : Shear, Scale



(a) Circle : Translate

(b) Circle : Reflect

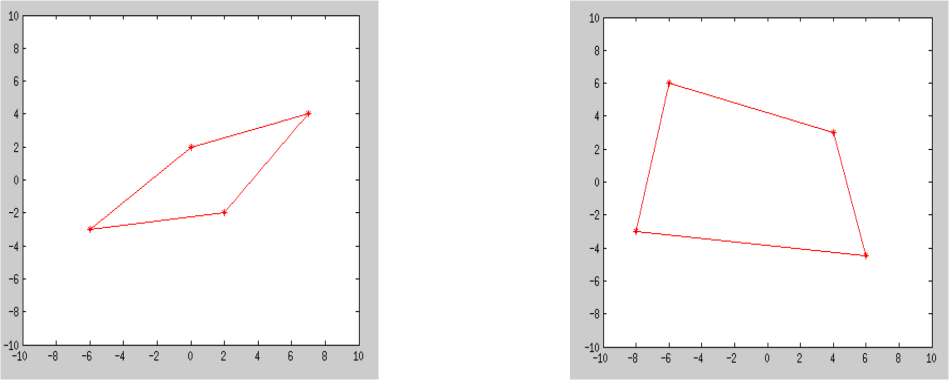
Figure 15 – Transformations on Circle : Translate, Reflect



(a) Initial polygon

(b) Polygon : Rotation

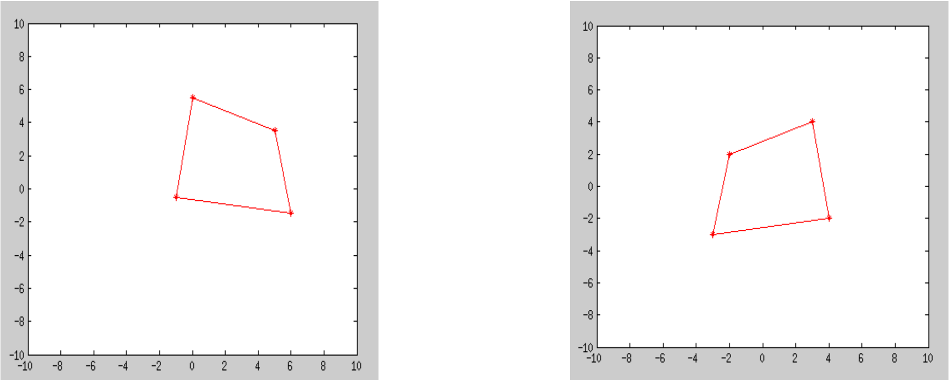
Figure 16 – Transformations on Polygon : Rotation



(a) Polygon : Shear

(b) Polygon : Scale

Figure 17 – Transformations on Polygon : Shear, Scale



(a) Polygon : Translate

(b) Polygon : Reflect

Figure 18 – Transformations on Polygon : Translate, Reflect