Application of Affine Transformations in a Mathematical Cartesian Coordinates for Java Students

Dr. Hieu Vu, and Dr. Francis Adepoju

Professors, School of Information Technology and Computing,
American University of Nigeria

Abstract
Affine transformation preserves the original shapeof an object, therefore it is a very important aspect in computer graphics. This paper presents a technique, how to apply reflections transformation in a mathematical Cartesian coordinates for Java students.

I. Introduction

In computer graphics, the official language is OpenGL (Open Graphics Library), and it is a component of C++ programming language for graphics. For the last 10 years, Java has replaced C++ as the dominant language for teaching computer programming in most universities and colleges in the United States and across the world. How can a student without background in C++ learn computer graphics?

There are two types of transformations of objects. Distortion that can be achieved by moving the anchor points and control points, distortion transformations will change the original shape of objects. The other is affine transformation which preserves the fundamental shape of objects. The most important of affine transformations are: translation, scaling, rotation about a point, reflection about a line and shearing (distortion of an angle).This paper will present a method how to apply affine transformations: reflections through x, y-coordinates and through the origin in a mathematical Cartesian coordinates for Java students.

II. Problem

The Java coordinate system places the origin (0, 0) at the upper left corner, stretches out to the right as x-coordinate and downward as the y-coordinate. The points (pixels) reside inside the quadrant (screen), all have positive coordinates (Figure 1).

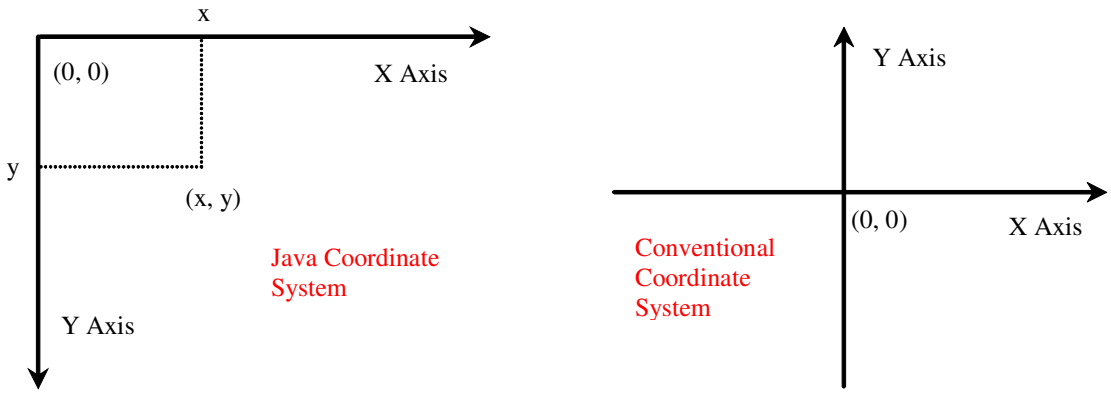

Figure 1: Java and conventional (Cartesian) coordinate systems (Liang, 2009)

In a conventional coordinate system, the origin (0, 0) should be at the center of the display area and the two axes (x-coordinate, y-coordinate) divide the display area into four quadrants (Figure 1).

III. Conversion of coordinate system and points

III.1. Conversion of Java coordinate system to conventional coordinate system

The origin of the conventional system is set at the center of the display area. For an example, the display area is typically 640 x 480 pixels.

```
    JFrame frame = new JFrame("TRANSLATION"); //Create frame
frame.getContentPane().add(new Translation());
frame.setSize(640, 480);              //Set frame size
frame.setVisible(true);
```

Therefore, the origin of the conventional coordinate system should have coordinates:

$$x = \frac{screenWidth}{2} = \frac{640}{2} = 320$$
$$y = \frac{screenHeight}{2} = \frac{480}{2} = 240 \qquad (1)$$

We now have the origin of the conventional coordinate system is set at O(320, 240).

III.2. Conversion of points onto conventional coordinate system

First, we already divided the x, and y-axis of the Java coordinate system into halves, therefore the x and y-coordinates of the points set on the Java display area should also be divided by 2 to fit on the new Cartesian conventional system. For example, point $P_0(x_0, y_0)$ will have coordinates $(x_0/2, y_0/2)$ on the conventional coordinate system.

Second, all points on the original Java display area have positive values for both x, and y-coordinates and they should appear on the first quadrant of the conventional coordinate system. We have to reset the points with respect to the new origin O(320, 240) of the new coordinate system.

III.3. Example illustration

First we select four points on the original screen (display area) then draw the pyramid with the base as a triangle (Figure 2).

```
        Point p1 = new Point(100, 100);     //100  100
    Point p2 = new Point(260, 200);     //260  200
            Point p3 = new Point(220, 300);     //220  320
            Point p4 = new Point(60, 260);      //60   260
```

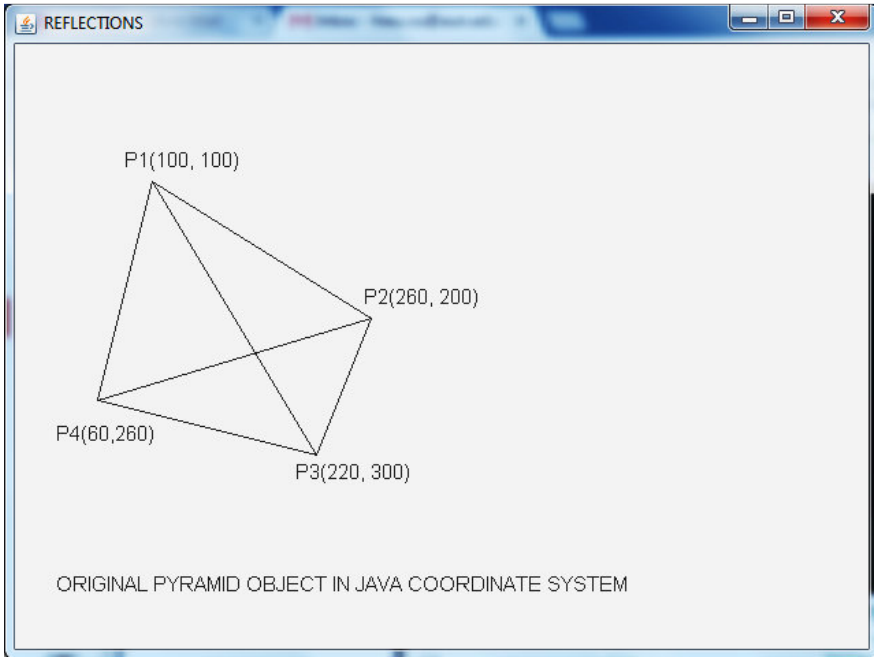

Figure 2: original shape on screen (Java coordinate system)

Next step, we find the origin for the conventional coordinate system (1), draw two axes for x and y-coordinates.

Let P'_ibe the new points of P_i,$(1 \leq i \leq 4)$ in the new conventional coordinate system. Since, we divided the x and y-coordinates of the screen by 2, the coordinates of the new pointsP'_iwould be halves of the original coordinates.

$$P'_i.x = P_i.x / 2 \quad \text{and} \quad P'_i.y = P_i.y / 2 \qquad (2)$$

andall the points should be on the first quadrant with respect to conventional origin O(x, y) (both x_i, $y_i > 0$), we have:

$$P'_i.x = P_i.x/2 + O.x$$
$$P'_i.y = P_i.y/2 + O.y \qquad (3)$$

The x-coordinates $P'_i.x$is in correct positions, to get the y-coordinates $P'_i.y$in the first quadrant, we need to reflect them through the x-axis of the conventional coordinate system (x = O.y), therefore:

$$(3) \Rightarrow P'_i.x = O.x + P_i.x/2$$
$$P'_i.y = O.y - P_i.x/2 \qquad (4)$$

The four new points will have coordinates in the conventional system as:

$$\begin{aligned} &\text{Point } p'_1(370,\ 190)\\ &\text{Point } p'_2(450,\ 140)\\ &\text{Point } p'_3(430,\ 90)\\ &\text{Point } p'_4(350,\ 110) \end{aligned} \quad (5)$$

Then draw the new pyramid on the conventional coordinate system (Figure 3).

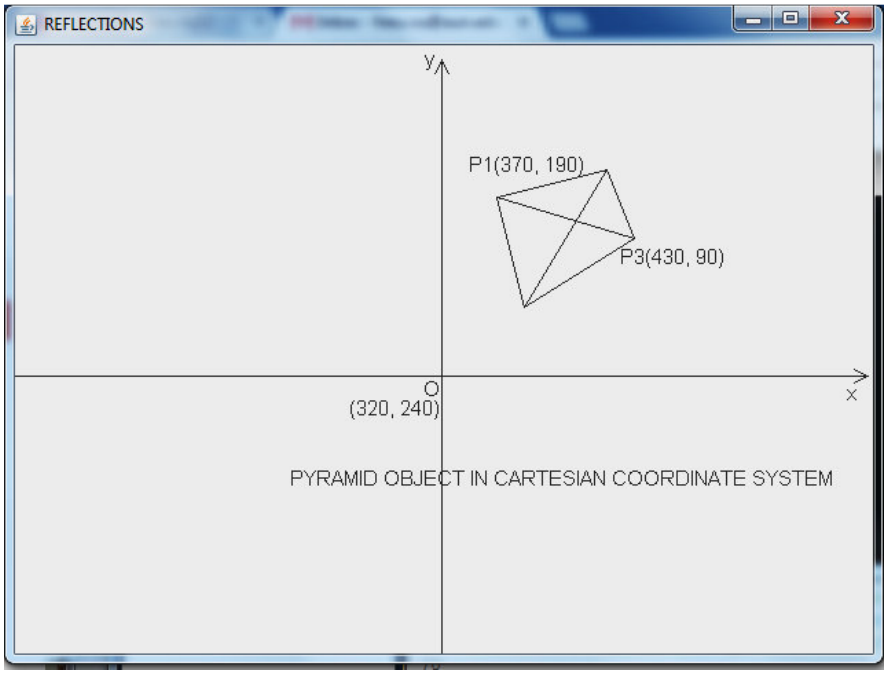

Figure 3: New shape on Cartesian conventional system

IV. Affine transformations

Affine transformation preserves original shape of objects, it is an important part in computer graphics and has many applications in movie industry, animation, CAD/CAAD, simulation, etc…

Affine transformations are linear transformations which included: translation, scaling, rotation, reflection, and shearing (Renka, 2013). This paper uses translation transformation with respect to the conventional coordinate system as an illustrate example.

IV.1. Reflections

Reflection is the simplest operation of transformation. Reflection will reflect the object through the x-axis, y-axis, the origin or a line.

IV.1.1. Reflection through x-axis

For example, supposed we want to reflect the object in figure 3 with respect to the x-axis of the conventional coordinate system. The x-coordinate of the image point Q would be the same, while the y-coordinate is the calculated by adding the distance between point P and the x-axis ($O_y - P_y$) to the y-coordinate of origin O, we have

$$Q_x = O_x + P_x$$
$$Q_y = O_y + O_y - P_y = 2Q_y - P_y \quad (6)$$

Four points in (5) would have images with coordinates:

$Q_1(370, 290)$
$Q_2(450, 340)$
$Q_3(430, 390)$
$Q_4(350, 370)$ (7)

Afterperforming the reflection through x-axis, draw new picture that contains both original and new shapes (Figure 4).

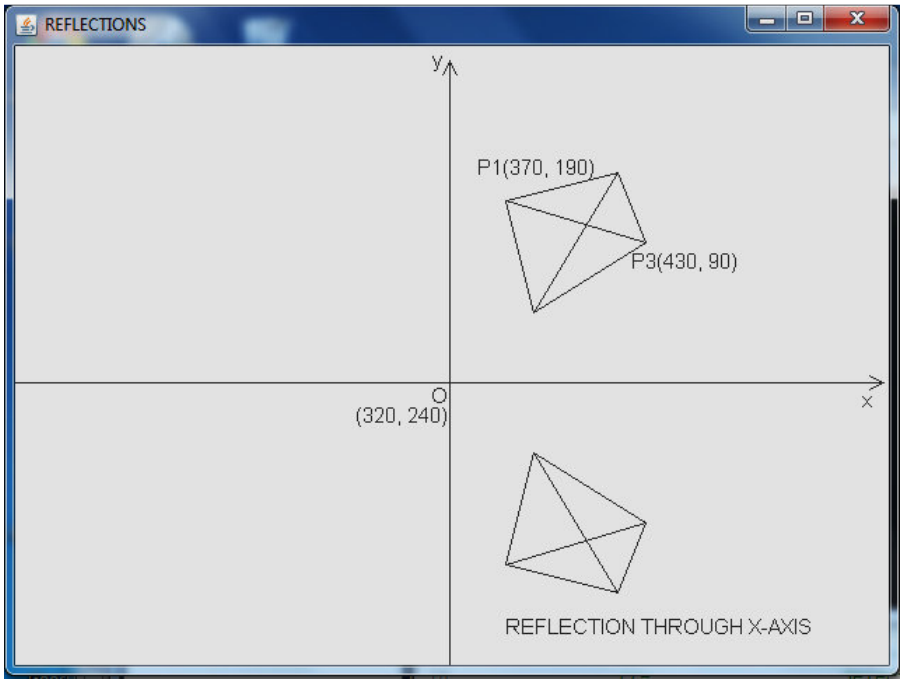

Figure 4: Reflection through x-axis

IV.1.2. Reflection through y-axis

Supposed we want to reflect the object in figure 3 with respect to the y-axis of the conventional coordinate system. The y-coordinate of the image point Q would be the same, while the x-coordinate is the calculated by subtracting the distance between point P, and the y-axis ($P_x - O_x$) from the x-coordinate of the origin, we have:

$$Q_x = O_x - (P_x - O_x) = 2O_x - P_x \quad (8)$$
$$Q_y = P_y$$

Four points in (5) would have images with coordinates:

$Q_1(270, 190)$
$Q_2(190, 140)$
$Q_3(210, 330)$
$Q_4(290, 110)$ (9)

After performing the reflection through x-axis, draw new picture that contains both original and new shapes (Figure 5).

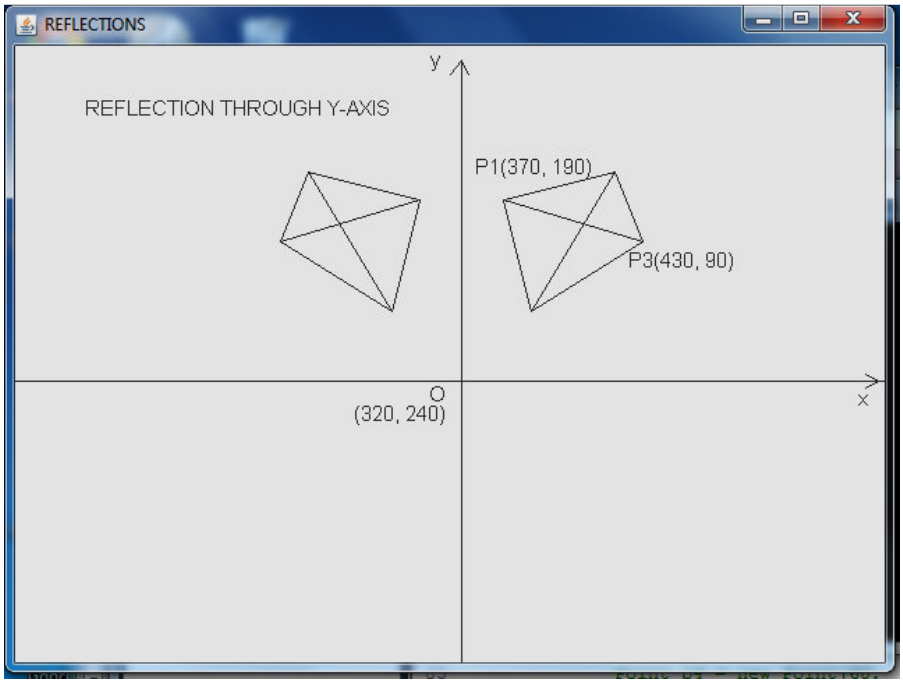

Figure 5: Reflection through y-axis

IV.1.3. Reflection through the origin O

Supposed we want to reflect the object in figure 3 with respect to the origin of the conventional coordinate system. This reflection is a combination of the previous two steps, we have:

From equation (8), $Q_x = O_x - (P_x - O_x) = 2O_x - P_x$ (10)

From equation (6), $Q_y = O_y + O_y - P_y = 2Q_y - P_y$ (11)

Four points in (5) would have images with coordinates:

$Q_1(270, 290)$

$Q_2(190, 340)$

$Q_3(210, 390)$

$Q_4(290, 370)$

After performing the reflection through x-axis, draw new picture that contains both original and new shapes (Figure 6).

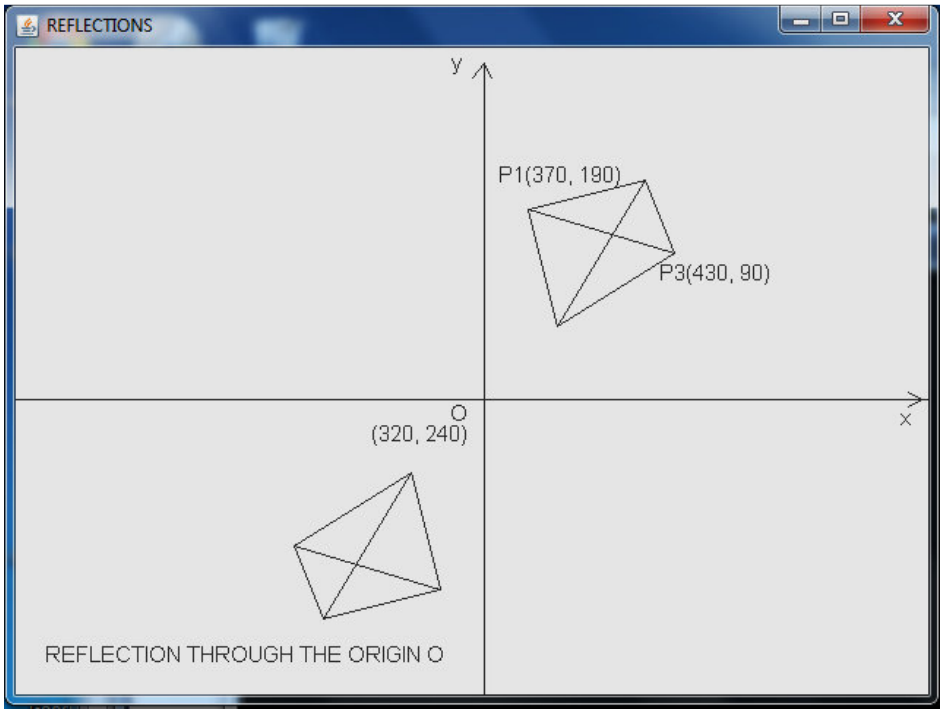

Figure 6: Reflection through the origin O

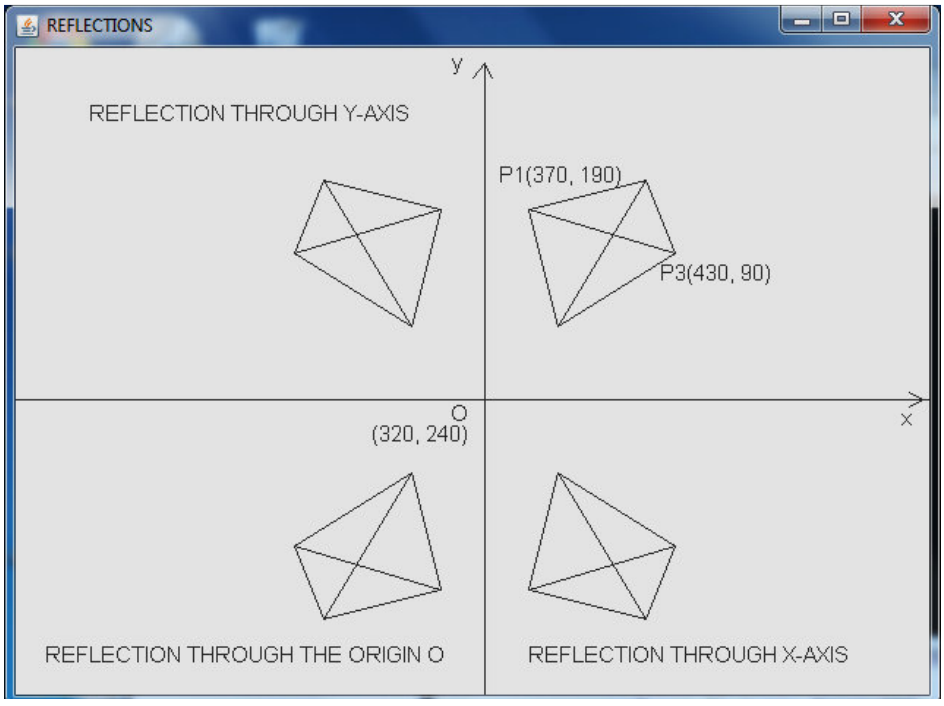

Figure 7: All combination

V. Conclusion

This paper can be used as a guide for graphics programming in Java. As can be seen through the example above, the translation process for an object in a conventional coordinate system can be used in Java. However, to make this happened, we first need to convert the Java coordinate system to conventional system and also the coordinates of points of the original object to their relative coordinates in the new conventional system before applying the translation process. Other transformations such as scaling, rotation, refection, and shearing can be done in the same way.

References:

Liang. Introduction to Java Programming.Seventh edition. Pearson Education, Upper Saddle River, New Jersey, 2009.ISBN-13: 978-0-13-814626-9

Renka, R. J.,Department of Computer Science & Engineering, University of North Texas, 9/23/2013