COS 344: L5 Chapter 7: 2D Shapes and Transformations

Cobus Redelinghuys

University of Pretoria

04/03/2024

Introduction

- Today we will look at modeling and animations for 2D objects.
- After today's lecture, you should be able to start planning your model for Practical 2.
- Today's lecture contains a set of examples, which are posted on ClickUp.
- Not all of this week's content is in the textbook!

2D Shapes

- Before animating an object we need to model said object.
- In computer graphics, triangles are used to create more shapes.
  - Why triangles?

2D Shapes

- Before animating an object we need to model said object.
- In computer graphics, triangles are used to create more shapes.
  - Why triangles?
  - Simplicity:
    - Most basic polygon.
    - Triangles are the least amount of points needed to create an area.
  - Planarity:
    - Three **non-collinear** points always define a unique plane in 3D space.
    - Which ensures that triangles are always flat.
  - `https://www.educative.io/answers/why-do-3d-engines-use-triangles-to-draw-object-surfaces`

- Triangles, thus, consists out of three vertices.
  - Is the order in which we “connect” the vertices to form the triangle important?

- Triangles, thus, consists out of three vertices.
  - Is the order in which we “connect” the vertices to form the triangle important?
    - Yes, the triangle needs to be created such that the normal of the plane created by the triangle faces outward.
- Open `COS344 L5 Normals` in `https://www.geogebra.org/3d?lang=en`.
- Why must the normal face outward?

- Now that we can create triangles correctly, how can we use them to create other shapes?
- For each of the following shapes describe how to model them using triangles.
  - Rectangle
  - Pentagon
  - Hexagon
  - Circle

Introduction

- In order to achieve transformation, a transformation matrix is needed.
  - Why a matrix and not just the formulae?
- For the moment, ignore how the matrix is created.
  - This will be discussed later.

Transformation application algorithm

The pseudocode below describes how to apply the transformation matrix.

Algorithm 1 Pseudo-code for applying a transformation matrix

Require: Matrix **T** {Transformation matrix}
Require: Object o {Object modeled by a set of vertices}
  List vertices = o.getVertices()
  for each vertex **v** in vertices **do**
    v' = **Tv**; {Matrix multiplied with a vector}
    updatePoint(o, **v**, **v'**)
  end for

Section 7.1.1: Scaling

- Scaling is the most basic transformation.
- Used to change the shape along coordinate axes.

$$scale(s_x, s_y) = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix}$$

- Example: Calculate the resultant matrix with the vertex $\begin{bmatrix} x \\ y \end{bmatrix}$.

$$\begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} s_x x + 0y \\ 0x + s_y y \end{bmatrix} = \begin{bmatrix} s_x x \\ s_y y \end{bmatrix}$$

- If $s_x == s_y$, the object's shape is maintained.
- If $s_x \neq s_y$, the object is deformed.

Look at `Examples/2D/Transformations/Scale` example

Illustrations

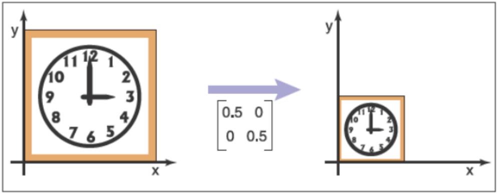
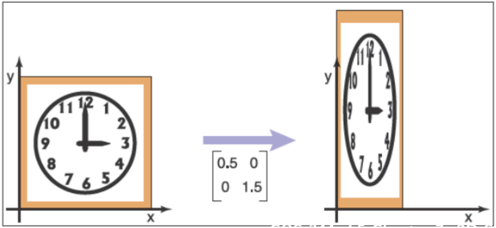


Figure 7.1. Scaling uniformly by half for each axis: The axis-aligned scale matrix has the proportion of change in each of the diagonal elements and zeroes in the off-diagonal elements.



Section 7.1.2: Shearing

- Shearing causes the "illusion" of pushing the object sideways like a deck of cards.
  - The top card is further to the side while the bottom card is at the same position.

$$shear_x(s) = \begin{bmatrix} 1 & s \\ 0 & 1 \end{bmatrix},\ shear_y(s) = \begin{bmatrix} 1 & 0 \\ s & 1 \end{bmatrix}$$

- Example: Calculate we can the resulting matrix with the vertex $\begin{bmatrix} x \\ y \end{bmatrix}$ using $shear_x$.

$$\begin{bmatrix} 1 & s \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x + sy \\ 0x + y \end{bmatrix} = \begin{bmatrix} x + sy \\ y \end{bmatrix}$$

Look at `Examples/2D/Transformations/Shear` example

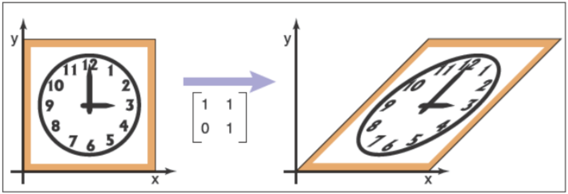
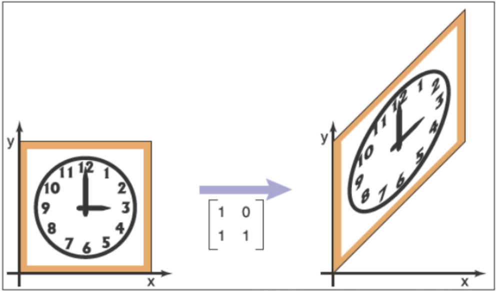


Figure 7.3. An x-shear matrix moves points to the right in proportion to their y-coordinate. Now the square outline of the clock becomes a parallelogram and, as with scaling, the circular face of the clock becomes an ellipse.



Alternative form of shearing

- An alternative way to express shearing:
  - Is to just rotate the object by a single axis, instead of both like in standard rotations.

$$shear_{\phi}(s) = \begin{bmatrix} 1 & tan(\phi) \\ 0 & 1 \end{bmatrix}, shear_{y}(\phi) = \begin{bmatrix} 1 & 0 \\ tan(\phi) & 1 \end{bmatrix}$$

- $shear_x(\phi)$ tilts the object by an angle ϕ clockwise from the vertical axis.
- $shear_y(\phi)$ tilts the object by an angle ϕ counterclockwise from the horizontal axis.

Section 7.1.4: Reflection

- Reflection about a coordinate axis by using the scaling matrix with a negative one scale factor (s_x or s_y).
  $reflect_y = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}$, $reflect_x = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$
- Example: Calculate we can the resulting matrix with the vertex $\begin{bmatrix} x \\ y \end{bmatrix}$ using $reflect_y$.

$$\begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} -x + 0y \\ 0x + y \end{bmatrix} = \begin{bmatrix} -x \\ y \end{bmatrix}$$

Look at `Examples/2D/Transformations/Reflection` example

Section 7.1.3: Rotations

- Suppose we want to rotate **a** by angle ϕ to form **b**.
- Suppose further that the angle between **a** and the x-axis is α, and **a** has a length $r = \sqrt{x_a^2 + y_a^2}$.

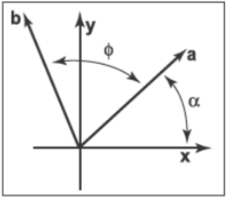


Figure 7.5. The geometry for Equation (7.1).

Thus, we know

- $x_a = rcos(\alpha)$
- $y_a = rsin(\alpha)$
- As **b** is only a rotation of **a**, r remains the same.
- The angle between **b** and the x-axis: $\alpha + \phi$.

- Using the facts on the previous slide:
  - $x_b = rcos(\alpha + \phi) = rcos(\alpha)cos(\phi) - rsin(\alpha)sin(\phi)$
  - $y_b = rsin(\alpha + \phi) = rsin(\alpha)cos(\phi) + rcos(\alpha)sin(\phi)$
- We can substitute $x_a = rcos(\alpha)$ and $y_a = rsin(\alpha)$ which results:
  - $x_b = x_a cos(\phi) - y_a sin(\phi)$
  - $y_b = y_a cos(\phi) + x_a sin(\phi)$
- What does this look like?

- Using the facts on the previous slide:
  - $x_b = rcos(\alpha + \phi) = rcos(\alpha)cos(\phi) - rsin(\alpha)sin(\phi)$
  - $y_b = rsin(\alpha + \phi) = rsin(\alpha)cos(\phi) + rcos(\alpha)sin(\phi)$
- We can substitute $x_a = rcos(\alpha)$ and $y_a = rsin(\alpha)$ which results:
  - $x_b = x_a cos(\phi) - y_a sin(\phi)$
  - $y_b = y_a cos(\phi) + x_a sin(\phi)$
- What does this look like?

$$\begin{bmatrix} x_a cos(\phi) - y_a sin(\phi) \\ y_a cos(\phi) + x_a sin(\phi) \end{bmatrix} = \begin{bmatrix} cos(\phi) & -sin(\phi) \\ sin(\phi) & cos(\phi) \end{bmatrix} \begin{bmatrix} x_a \\ y_a \end{bmatrix}$$

$$\begin{bmatrix} x_b \\ y_b \end{bmatrix} = \begin{bmatrix} cos(\phi) & -sin(\phi) \\ sin(\phi) & cos(\phi) \end{bmatrix} \begin{bmatrix} x_a \\ y_a \end{bmatrix}$$

Look at `Examples/2D/Transformations/Rotation` example

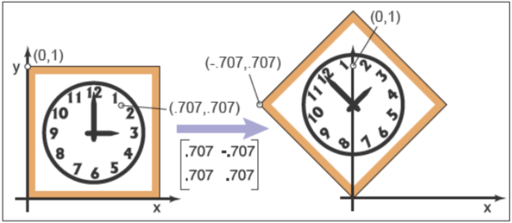


Figure 7.6. A rotation by 45°. Note that the rotation is counterclockwise and that $\cos(45°) = \sin(45°) \approx .707$.

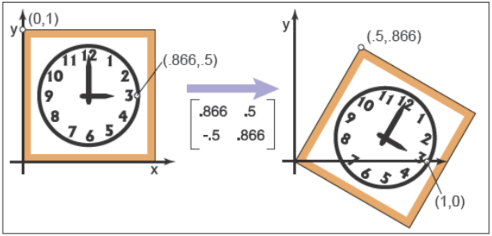


Figure 7.7. A rotation by $-30°$. Note that the rotation is clockwise and that $\cos(-30°) \approx .866$ and $\sin(-30°) = -.5$.

Section 7.3: Translation and Affine Transformations

- Say we would like to rotate an object around itself, which is not at the origin.
- Can the defined rotational matrices work?
- Look at `Examples/2D/Transformations/OffCenterRotation` example.
- How can the defined rotation matrix be used to rotate the object about itself, when the object is not at the center?
  1. Move the object such that the point of rotation is at the origin.
  2. Apply the rotation.
  3. Move the object back.

Naive approach

- To apply translation using direction **d**, we can use:

$$x' = x + d_x$$

$$y' = y + d_y$$

- But this clashes with our current approach of using matrices to perform transformations.
- How can this be implemented using a matrix?

- To use matrices for transformations, we add an additional dimension to the transformation matrix.
- Thus, for 2D coordinate spaces, we use a 3×3 matrix.

$$\begin{bmatrix} m_{11} & m_{12} & x_t \\ m_{21} & m_{22} & y_t \\ 0 & 0 & 1 \end{bmatrix}$$

- These are known as homogeneous coordinates (adding an extra dimension to the matrix).
- We can express the translation as:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} m_{11} & m_{12} & x_t \\ m_{21} & m_{22} & y_t \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} m_{11}x + m_{12}y + x_t \\ m_{21}x + m_{22}y + y_t \\ 1 \end{bmatrix}$$

- This allows a single matrix to apply a linear transformation followed by a translation!

Translation matrix

- Using the homogeneous coordinates, we can create the following translation matrix in direction $\begin{bmatrix} x_t \\ y_t \end{bmatrix}$:

$$translation(x_t, y_t) = \begin{bmatrix} 1 & 0 & x_t \\ 0 & 1 & y_t \\ 0 & 0 & 1 \end{bmatrix}$$

- This can then achieve the following movement:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & x_t \\ 0 & 1 & y_t \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

- Is the number represented by the 1 significant?

Rule of thumb

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

is a location or point.

$$\begin{bmatrix} x \\ y \\ 0 \end{bmatrix}$$

is a direction.

- Exercise: Verify that when translating a direction, the direction remains the same.
- `Examples/2D/Transformations/Translation` example.
- Remainder of Chapter 7 will be discussed during our next lecture.

Summary

- Homogeneous scaling matrix: $\begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$
- Homogeneous shearing matrix: $\begin{bmatrix} 1 & h_{xy} & 0 \\ h_{yx} & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
- Homogeneous y-reflection matrix: $\begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
- Homogeneous rotational matrix: $\begin{bmatrix} cos(\phi) & -sin(\phi) & 0 \\ sin(\phi) & cos(\phi) & 0 \\ 0 & 0 & 1 \end{bmatrix}$
- Homogeneous tranlsation matrix: $\begin{bmatrix} 1 & 0 & x_t \\ 0 & 1 & y_t \\ 0 & 0 & 1 \end{bmatrix}$

Joke of the day - By ChatGPT

Why did the square refuse to spin?

Joke of the day - By ChatGPT

Why did the square refuse to spin?

It couldn't handle the twist!