ECE 600: Introduction to Shape Analysis

Lab #1 – Vectors, Matrices and Transforms

(Assigned Thursday 5/14/09 – Due Thursday 5/21/09)

In this lab, we will discuss/experiment the basic building blocks of objects in computer graphics and computational geometry. We will get a flavor of how to represent shapes in terms of points, and how to manipulate them in the sense of transformations (translation, scaling and rotation for example). The building blocks for objects/shapes and transforms are vectors and matrices, in latter labs we will also discuss curves and surfaces, however in this lab we will focus on points as the basic mean of describing objects and how to transform these points in some fashion.

1. Experimentation

1.1 Supporting Materials

In the supporting material of this lab, you will find classes which define a two dimensional point and line in MATLAB (Point2D.m and Line2D.m), another class is defined to retrieve different transformation matrices given the transformation parameters (TransformMatrix2D.m), also you will find a sample file (test.m) which displays the 2-dimensional Cartesian coordinate centered at the origin (0,0), a line in 2D is defined using the aforementioned classes, different transformations were applied to the defined line and results were displayed one at a time.

1.2 Lab Tasks

You are required to do the following tasks;

Task 1: Extend the given classes to the 3-dimensional space.

Task 2: Define a new class to form a polygon (in 2D and 3D), which can be considered as a list of ordered points (vertices) having lines connecting between them, use the provided class(es) build your new class. If you have any problems regarding programming classes in MATLAB, either refer to matlab_oop.pdf provided with the supporting material or simply read through the provided classes as a sample code to direct you.
Task 3: Build your own graphical user interface (GUI) to allow the user to pick points from the space (2D and 3D) to form his own shapes (e.g. lines, triangles, rectangle, polygons with arbitrary number of points). Allow the user to select the type of transformation required to be performed on the drawn shape associated with its parameters and draw the shape after the transformation is applied. Note that you are required to draw the Cartesian coordinate system (2D and 3D) to understand the transformation effect.

Task 4: Write a report to summarize the theoretical background needed for this lab and your experimental results (experiment your program with different shapes and transformations). Your report should begin with a cover page introducing the project title and group members. It is important to note that all figure axes should be labeled properly. You are required to submit your MATLAB codes (fully commented) with a readme file describing your files and how to use them in terms of input and output.

2. Theoretical Background

2.1 Vector Spaces

We will consider point sets which are subsets of the real Euclidean \( n \)-space which is defined as follows. (refer to the reading material)

**Definition 1.18:** Real Euclidean \( n \)-space is given by \( \mathbb{R}^n = \{ \mathbf{x} = (x_1, x_2, ..., x_n) | x_i \in \mathbb{R} \} \) where \( \mathbf{x} \) denotes a point with \( n \)-coordinates.

The set \( \mathbb{R}^n \) associated with the operations of addition and scalar multiplication is said to form a **vector space** if the following properties are satisfied:

1. If \( \mathbf{x} \in \mathbb{R}^n \) and \( \mathbf{y} \in \mathbb{R}^n \) then \( \mathbf{x} + \mathbf{y} \in \mathbb{R}^n \) (Closure under addition).
2. If \( \mathbf{x} \in \mathbb{R}^n \) and \( \mathbf{y} \in \mathbb{R}^n \) then \( \mathbf{x} + \mathbf{y} = \mathbf{y} + \mathbf{x} \) (Commutative)
3. If \( \mathbf{x}, \mathbf{y}, \mathbf{z} \in \mathbb{R}^n \) then \( \mathbf{x} + (\mathbf{y} + \mathbf{z}) = (\mathbf{x} + \mathbf{y}) + \mathbf{z} \) (Associative)
4. There is a zero vector \( \mathbf{0} \in \mathbb{R}^n \) such that \( \mathbf{0} + \mathbf{x} = \mathbf{x} + \mathbf{0} = \mathbf{x} \) for all \( \mathbf{x} \in \mathbb{R}^n \) (Additive identity)
5. For each \( \mathbf{x} \in \mathbb{R}^n \), there is \( \mathbf{a} - \mathbf{x} \in \mathbb{R}^n \), called the negative of \( \mathbf{x} \) such that \( (\mathbf{a} - \mathbf{x}) + \mathbf{x} = \mathbf{x} + (\mathbf{a} - \mathbf{x}) = \mathbf{0} \). (Additive inverse)
6. If \( \alpha \) is any scalar (i.e. \( \alpha \in \mathbb{R} \)) and \( \mathbf{x} \in \mathbb{R}^n \), then \( \alpha \mathbf{x} \in \mathbb{R}^n \) (Closure under scalar multiplication)
7. If \( \alpha \) is any scalar and \( \mathbf{x}, \mathbf{y} \in \mathbb{R}^n \), then \( \alpha(\mathbf{x} + \mathbf{y}) = \alpha \mathbf{x} + \alpha \mathbf{y} \)
8. If \( \alpha, \beta \) are scalars and \( \mathbf{x} \in \mathbb{R}^n \), then \( (\alpha + \beta)\mathbf{x} = \alpha \mathbf{x} + \beta \mathbf{x} \)
9. If \( \alpha, \beta \) are scalars and \( \mathbf{x} \in \mathbb{R}^n \), then \( \alpha(\beta \mathbf{x}) = (\alpha \beta)\mathbf{x} \)
10. If \( \mathbf{x} \in \mathbb{R}^n \), then \( 1\mathbf{x} = \mathbf{x} \).

Here we consider vectors as column vectors. Thus \( \mathbf{x}^T \) denotes a row vector where \( ^T \) denotes the transpose.
We can define a scalar product in \( \mathbb{R}^n \) as follows:

\[
\langle \mathbf{x}, \mathbf{y} \rangle = \mathbf{x} \cdot \mathbf{y} = \mathbf{x}^T \mathbf{y} = \sum_{i=1}^{n} x_i y_i
\]

Where \( \mathbf{x} = (x_1, x_2, ..., x_n)^T \) and \( \mathbf{y} = (y_1, y_2, ..., y_n)^T \)

Two vectors are called perpendicular/normal to each other if \( \langle \mathbf{x}, \mathbf{y} \rangle = 0 \), for example \( \mathbf{x} = \begin{pmatrix} 1 \\ 1 \\ -1 \end{pmatrix} \) and \( \mathbf{y} = \begin{pmatrix} -1 \\ 0 \\ -1 \end{pmatrix} \) are perpendicular.

The Euclidean norm can be defined as follows:

\[
\|\mathbf{x}\| = \sqrt{\langle \mathbf{x}, \mathbf{x} \rangle} = \sqrt{\sum_{i=1}^{n} x_i^2}
\]

A vector \( \mathbf{x} \) is called normalized if \( \|\mathbf{x}\| = 1 \).

The Euclidean distance between the vectors \( \mathbf{x} = (x_1, x_2, ..., x_n)^T \) and \( \mathbf{y} = (y_1, y_2, ..., y_n)^T \) is then given by:

\[
\|\mathbf{x} - \mathbf{y}\| = \sqrt{\langle \mathbf{x} - \mathbf{y}, \mathbf{x} - \mathbf{y} \rangle} = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + \cdots + (x_n - y_n)^2}
\]

The addition of the vectors \( \mathbf{x} = (x_1, x_2, ..., x_n)^T \) and \( \mathbf{y} = (y_1, y_2, ..., y_n)^T \) is given by;

\[
\mathbf{x} + \mathbf{y} = (x_1, x_2, ..., x_n)^T + (y_1, y_2, ..., y_n)^T = (x_1 + y_1, x_2 + y_2, ..., x_n + y_n)^T
\]

The scalar multiplication the vector \( \mathbf{x} = (x_1, x_2, ..., x_n)^T \) with a scalar \( \alpha \in \mathbb{R} \) is given by;

\[
\alpha \mathbf{x} = \alpha (x_1, x_2, ..., x_n)^T = (\alpha x_1, \alpha x_2, ..., \alpha x_n)^T
\]

While vector multiplication between the vectors \( \mathbf{x} = (x_1, x_2, ..., x_n)^T \) and \( \mathbf{y} = (y_1, y_2, ..., y_n)^T \) is defined by;

\[
\mathbf{xy}^T = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} \begin{pmatrix} y_1 & y_2 & \cdots & y_n \end{pmatrix} = \begin{pmatrix} x_1 y_1 & x_1 y_2 & \cdots & x_1 y_n \\ x_2 y_1 & x_2 y_2 & \cdots & x_2 y_n \\ \vdots & \vdots & \ddots & \vdots \\ x_n y_1 & x_n y_2 & \cdots & x_n y_n \end{pmatrix}
\]
The cross product is a binary operation on two vectors in a three-dimensional Euclidean space \( \mathbb{R}^3 \) that results in another vector which is perpendicular to the plane containing the two input vectors (i.e. is perpendicular to the plane spanned by the two input vectors). It is defined as;

\[
\mathbf{x} \times \mathbf{y} = \begin{pmatrix}
    x_2y_3 - x_3y_2 \\
    x_3y_1 - x_1y_3 \\
    x_1y_2 - x_2y_1
\end{pmatrix}
\]

We have the following properties:

1. \( \mathbf{x} \times (\mathbf{y} + \mathbf{z}) = \mathbf{x} \times \mathbf{y} + \mathbf{x} \times \mathbf{z} \)
2. \( \mathbf{x} \times \mathbf{y} = -\mathbf{y} \times \mathbf{x} \)
3. \( \mathbf{x} \times (\mathbf{y} \times \mathbf{z}) + \mathbf{z} \times (\mathbf{x} \times \mathbf{y}) + \mathbf{y} \times (\mathbf{z} \times \mathbf{x}) = \mathbf{0} \) (Jacobi identity)
4. \( \mathbf{x} \times (\mathbf{y} \times \mathbf{z}) = (\mathbf{z} \cdot \mathbf{y})\mathbf{x} - (\mathbf{x} \cdot \mathbf{y})\mathbf{z} \) (Vector identity)

### 2.2 Points and Vectors

It is important to know the difference between points and vectors. Vectors are quantities with magnitude and direction while points are locations in the Cartesian space. Points can be represented by vectors, indicating the direction and distance from the origin of the Cartesian plane. However, points and vectors differ in that points can be moved whereas vectors cannot. For example, a normal to a surface (vector perpendicular to the surface) remains the same, no matter where the surface is, as long as the orientation of the surface remains constant. However, points describing the surface are affected by the position of the surface.

To represented a point in \( \mathbb{R}^3 \) we use the format \( \mathbf{x} = (x_1, x_2, x_3)^T \) which is the vector representing the location of the point relative to the origin. The same notation is used for points in 2-dimensional space where \( \mathbf{x} = (x_1, x_2)^T \).

Now line segments can be defined by two points. Let \( \mathbf{p}_1 \) and \( \mathbf{p}_2 \) be the two points defining the line segment, then, \( \mathbf{p} = (1 - \alpha)\mathbf{p}_1 + \alpha\mathbf{p}_2 \) defines points on the line segment where \( \alpha \in [0,1] \).

#### 2.2.1 Homogeneous Coordinates

In many situations it is necessary to differentiate between points and vectors. To do so we embed the vector space \( \mathbb{R}^3 \) in the vector space \( \mathbb{R}^4 \) to create the homogeneous coordinate system.

**Definition:** if \( u_1, u_2, u_3, w \in \mathbb{R} \) and \( w \neq 0 \), then \( (u_1, u_2, u_3, w)^T \) is a homogeneous coordinate representation of the point \( \left( \frac{u_1}{w}, \frac{u_2}{w}, \frac{u_3}{w} \right)^T \). The points \( (u_1, u_2, u_3, 0)^T \) are so called points at infinity and are often used to represent vectors.
We now have separate representations for points and vectors. Points have the form \((u_1, u_2, u_3, w)^T\) with \(w \neq 0\), while vectors have the form \((u_1, u_2, u_3, 0)^T\). Vectors and points in \(\mathbb{R}^4\) form a Grassmann space. The homogeneous form of vectors and points will prove to be useful when defining various transforms.

### 2.3 Representing Objects by Points

Most objects can be represented by a set of points \(P\). These points may describe points on the surface of the object or may be used as control points to describe a surface associated with additional parameters. The simplest form of object is a polygonal mesh which is described by a set of polygons created from the point set \(P\). These polygons can be described by a set of faces \(F\) where each element of \(F\) is a set of edges that describe that face. The edges are simply described by two points \((p_1, p_2)\) where \(p_1, p_2 \in P\). Hence the positional information is stored in \(P\) while the connectivity information is stored in \(F\). Thus any transform applied to the points in \(P\) will transform the whole object. We typically use affine transforms to transform the points.

### 2.4 Affine Transformations

Basically, we would like to be able to transform a curve or a surface without manually calculating the each individual point in the new curve or surface. Affine transforms will allow us to do this. But first what is a transform?

**Definition:** A transform/warp on \(\mathbb{R}^n\) is any mapping \(W: \mathbb{R}^n \to \mathbb{R}^n\). That is, each point \(x \in \mathbb{R}^n\) is mapped to exactly one point \(W(x)\) also in \(\mathbb{R}^n\).

**Definition:** Let \(W: \mathbb{R}^n \to \mathbb{R}^n\) be a transform. \(W\) is said to be a linear transform/warp if and only if:

\[
\begin{align*}
(a) & \quad \text{For all } \alpha \in \mathbb{R} \text{ and all } x \in \mathbb{R}^n \text{ we have } W(\alpha x) = \alpha W(x). \\
(b) & \quad \text{For all } x, y \in \mathbb{R}^n \text{ we have } W(x + y) = W(x) + W(y).
\end{align*}
\]

This implies that \(W(0) = 0\) since \(W(0, x) = 0, W(x) = 0\). An example of a linear transform/warp is the identity transform given by \(W(x) = x\).

**Definition:** Let \(W: \mathbb{R}^n \to \mathbb{R}^n\) be a transform. \(W\) is said to be a translation if there exists \(t \in \mathbb{R}^n\) so that for all \(x \in \mathbb{R}^n\) we have \(W(x) = x + t\). A translation moves all vectors or points by a fixed distance in a fixed direction.

**Definition:** An affine transform is a transform that can be written as \(W(x) = T(L(x))\) where \(L(\cdot)\) is a linear transform and \(T(\cdot)\) is a translation. This can also be written as \(W(x) = L(x) + t\) or \(W = T_t L\).

Any linear transform in \(\mathbb{R}^3\) can be represented by a 3x3 matrix of the following form;
2.4.1 Scaling

In order to change the size of an object in $\mathbb{R}^3$, if we assume that the object is centered at the origin, then scaling is given by:

$$ S = \begin{pmatrix} s & 0 & 0 \\ 0 & s & 0 \\ 0 & 0 & s \end{pmatrix} $$

To scale a point we apply the matrix $S$ to the point $\mathbf{x} \in \mathbb{R}^n$ where $\mathbf{x} = \{x_1, x_2, x_3\}$ to get

$$ \mathbf{x'} = S \mathbf{x} = \begin{pmatrix} s & 0 & 0 \\ 0 & s & 0 \\ 0 & 0 & s \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} s x_1 \\ s x_2 \\ s x_3 \end{pmatrix} $$

This is called *uniform scaling* since we change the size of the object in all directions with the same amount, however, it is not necessary to scale evenly in all directions, in this case we can define the *non-uniform scaling* matrix as follows;

$$ S = \begin{pmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & s_z \end{pmatrix} $$

Where $x, y, z$ connotes the three dimensions in a 3D Euclidean space.

In the same manner, scaling matrices can be defined in the 2D space where $S = \begin{pmatrix} s & 0 \\ 0 & s \end{pmatrix}$ in case of uniform scaling and $S = \begin{pmatrix} s_x & 0 \\ 0 & s_y \end{pmatrix}$ in case of non-uniform scaling.

2.4.2 Translation

Translations in $\mathbb{R}^3$ cannot be written as 3x3 matrices, however to allow a unified representation for affine transforms, we can use the homogeneous coordinate system which allows us to represent an affine transform as a matrix. A translation by $\mathbf{t} = (t_x, t_y, t_z)^T$ is given by a 4x4 matrix defined as:
\[ T_t = \begin{pmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \]

Linear transforms can also be represented as transforms on homogeneous coordinate systems as follows;

\[ L = \begin{pmatrix} \ell_{11} & \ell_{12} & \ell_{13} & 0 \\ \ell_{21} & \ell_{22} & \ell_{23} & 0 \\ \ell_{31} & \ell_{32} & \ell_{33} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \]

Affine transforms can be represented as combination of a linear transform and a translation, thus can be represented by the matrix product \( W = T_tL \) in the homogeneous coordinate system.

In the same manner, translation matrices can be defined in the 2D space as \( S = \begin{pmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{pmatrix} \)

**2.4.3 Rotation**

In the Cartesian plane in \( \mathbb{R}^2 \), rotation can be derived as follows. In the following figure, we have a point \( x = (x_1, x_2)^T \) is rotated anticlockwise by an angle \( \alpha \) to obtain \( x' = (x'_1, x'_2)^T \).

It can be seen that \( x_1 = ||x|| \cos \theta \) and \( x_2 = ||x|| \sin \theta \), where the norm \( ||x|| \) is the distance between the origin \((0,0)\) and the point \( x \), since this distance will not be changed by rotation, we will have \( x'_1 = ||x|| \cos(\theta + \alpha) = ||x|| \cos \theta \cos \alpha - ||x|| \sin \theta \sin \alpha = x_1 \cos \alpha - x_2 \sin \alpha \).

Likewise we can compute \( x'_2 \) as follows;

\[ x'_2 = ||x|| \sin(\theta + \alpha) = ||x|| \cos \theta \sin \alpha + ||x|| \sin \theta \cos \alpha = x_1 \sin \alpha + x_2 \cos \alpha \]

The transform can be written as the orthogonal matrix;

\[ R_{z,\alpha} = \begin{pmatrix} \cos \alpha & -\sin \alpha \\ \sin \alpha & \cos \alpha \end{pmatrix} \]
$R_{z,\alpha}$ rotates in the Cartesian plane, which is essentially rotation around the $z$-axis. In $\mathbb{R}^3$, the rotation matrix about the $z$-axis can be written as:

$$R_{z,\alpha} = \begin{pmatrix} \cos \alpha & -\sin \alpha & 0 \\ \sin \alpha & \cos \alpha & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

In the same manner, the rotation matrices about $x$ and $y$ axes are defined as follows:

$$R_{x,\alpha} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos \alpha & -\sin \alpha \\ 0 & \sin \alpha & \cos \alpha \end{pmatrix}$$

And

$$R_{y,\alpha} = \begin{pmatrix} \cos \alpha & 0 & -\sin \alpha \\ 0 & 1 & 0 \\ \sin \alpha & 0 & \cos \alpha \end{pmatrix}$$

The *Euler transform* is defined as $E(\alpha, \beta, \gamma) = R_{z,\gamma}R_{x,\alpha}R_{y,\beta}$, where the angles $\alpha, \beta$ and $\gamma$ define yaw, pitch and roll angles. The Euler transform is often used to represent the orientation of an object. However, the Euler transform suffers from *gimbal lock* that is a degree of freedom can be lost in some cases. Instead of using the Euler transform, we use a general rotation matrix and use concatenated transforms to specify rotations.

*Good Luck*