Objectives

- Introduce the elements of geometry
  - Scalars
  - Vectors
  - Points
- Develop mathematical operations among them in a coordinate-free manner
- Define basic primitives
  - Line segments
  - Polygons

Basic Elements

- Geometry is the study of the relationships among objects in an n-dimensional space
  - In computer graphics, we are interested in objects that exist in three dimensions
- Want a minimum set of primitives from which we can build more sophisticated objects
- We will need three basic elements
  - Scalars
  - Vectors
  - Points

Coordinate-Free Geometry

- When we learned simple geometry, most of us started with a Cartesian approach
  - Points were at locations in space \( p = (x, y, z) \)
  - We derived results by algebraic manipulations involving these coordinates
- This approach was nonphysical
  - Physically, points exist regardless of the location of an arbitrary coordinate system
  - Most geometric results are independent of the coordinate system
  - Example Euclidean geometry: two triangles are identical if two corresponding sides and the angle between them are identical

Transformations to Change Coordinate Systems

- 4 coordinate systems
- 1 point \( P \)
  
  \[
  M_{1 \to 2} = T(4, 2) \\
  M_{2 \to 3} = T(2, 3) \cdot S(0.5, 0.5) \\
  M_{3 \to 4} = T(6.7, 1.8) \cdot R(45\degree) \\
  M_{i \to k} = M_{i \to j} \cdot M_{j \to k}
  \]

Scalars

- Need three basic elements in geometry
  - Scalars, Vectors, Points
- Scalars can be defined as members of sets which can be combined by two operations (addition and multiplication) obeying some fundamental axioms (associativity, commutativity, inverses)
- Examples include the real and complex number systems under the ordinary rules with which we are familiar
- Scalars alone have no geometric properties
Vectors

• Physical definition: a vector is a quantity with two attributes
  - Direction
  - Magnitude

• Examples include
  - Force
  - Velocity
  - Directed line segments
    - Most important example for graphics
    - Can map to other types

Vector Operations

• Every vector has an inverse
  - Same magnitude but points in opposite direction

• Every vector can be multiplied by a scalar

• There is a zero vector
  - Zero magnitude, undefined orientation

• The sum of any two vectors is a vector
  - Use head-to-tail axiom

Linear Vector Spaces

• Mathematical system for manipulating vectors

• Operations
  - Scalar-vector multiplication: \( u = \alpha v \)
  - Vector-vector addition: \( w = u + v \)

• Expressions such as
  \[ \mathbf{v} = \mathbf{u} + 2 \mathbf{w} - 3 \mathbf{r} \]
  Make sense in a vector space

Vectors Lack Position

• These vectors are identical
  - Same direction and magnitude (length)

• Vectors spaces insufficient for geometry
  - Need points

Points

• Location in space

• Operations allowed between points and vectors
  - Point-point subtraction yields a vector
  - Equivalent to point-vector addition

Affine Spaces

• Point + a vector space

• Operations
  - Vector-vector addition
  - Scalar-vector multiplication
  - Point-vector addition
  - Scalar-scalar operations

• For any point define
  - \( 1 \cdot \mathbf{P} = \mathbf{P} \)
  - \( 0 \cdot \mathbf{P} = \mathbf{0} \) (zero vector)
Lines

- Consider all points of the form $P(\alpha) = P_0 + \alpha d$
- Set of all points that pass through $P_0$ in the direction of the vector $d$

Parametric Form

- This form is known as the parametric form of the line
- More robust and general than other forms
- Extends to curves and surfaces

Two-dimensional forms
- Explicit: $y = mx + h$
- Implicit: $ax + by + c = 0$
- Parametric:
  $x(\alpha) = \alpha x_0 + (1-\alpha)x_1$
  $y(\alpha) = \alpha y_0 + (1-\alpha)y_1$

Rays and Line Segments

- If $\alpha \geq 0$, then $P(\alpha)$ is the ray leaving $P_0$ in the direction $d$
- If we use two points to define $v$, then
  $P(\alpha) = Q + \alpha (R-Q) = Q + \alpha v$
  $= \alpha R + (1-\alpha)Q$
- For $0 \leq \alpha \leq 1$ we get all the points on the line segment joining $R$ and $Q$

Convexity

- An object is convex iff for any two points in the object all points on the line segment between these points are also in the object

Affine Sums

- Consider the “sum” $P = \alpha_1 P_1 + \alpha_2 P_2 + \ldots + \alpha_n P_n$
- If $\alpha_1 + \alpha_2 + \ldots + \alpha_n = 1$
in which case we have the affine sum of the points $P_1, P_2, \ldots, P_n$
- If, in addition, $\alpha_i \geq 0$, we have the convex hull of $P_1, P_2, \ldots, P_n$

Convex Hull

- Smallest convex object containing $P_1, P_2, \ldots, P_n$
- Formed by “shrink wrapping” points
Curves and Surfaces

• Curves are one parameter entities of the form $P(\alpha)$ where the function is nonlinear
• Surfaces are formed from two-parameter functions $P(\alpha, \beta)$
  - Linear functions give planes and polygons

Planes

• A plane can be defined by a point and two vectors or by three points

Triangles

- A triangle is convex so any point inside can be represented as an affine sum
  $P(\alpha_1, \alpha_2, \alpha_3) = \alpha_1 P + \alpha_2 Q + \alpha_3 R$
  where
  $\alpha_1 + \alpha_2 + \alpha_3 = 1$
  $\alpha_i \geq 0$
  The representation is called the barycentric coordinate representation of $P$

Normals

- Every plane has a vector $n$ normal (perpendicular, orthogonal) to it
- From point-two vector form $P(\alpha, \beta) = R + \alpha u + \beta v$, we know we can use the cross product to find $n = u \times v$ and the equivalent form
  $(P(\alpha) - P) \cdot n = 0$
Objectives

- Introduce concepts such as dimension and basis
- Introduce coordinate systems for representing vectors spaces and frames for representing affine spaces
- Discuss change of frames and bases
- Introduce homogeneous coordinates

Linear Independence

- A set of vectors \( v_1, v_2, \ldots, v_n \) is linearly independent if
  \[ \alpha_1 v_1 + \alpha_2 v_2 + \ldots + \alpha_n v_n = 0 \iff \alpha_1 = \alpha_2 = \ldots = 0 \]
- If a set of vectors is linearly independent, we cannot represent one in terms of the others
- If a set of vectors is linearly dependent, at least one can be written in terms of the others

Dimension

- In a vector space, the maximum number of linearly independent vectors is fixed and is called the dimension of the space
- In an \( n \)-dimensional space, any set of \( n \) linearly independent vectors form a basis for the space
- Given a basis \( v_1, v_2, \ldots, v_n \), any vector \( v \) can be written as
  \[ v = \alpha_1 v_1 + \alpha_2 v_2 + \ldots + \alpha_n v_n \]
  where the \( \{a_i\} \) are unique

Representation

- Until now we have been able to work with geometric entities without using any frame of reference, such as a coordinate system
- Need a frame of reference to relate points and objects to our physical world.
  - For example, where is a point? Can’t answer without a reference system
  - World coordinates
  - Camera coordinates

Coordinate Systems

- Consider a basis \( v_1, v_2, \ldots, v_n \)
- A vector is written \( v = \alpha_1 v_1 + \alpha_2 v_2 + \ldots + \alpha_n v_n \)
- The list of scalars \( \{\alpha_1, \alpha_2, \ldots, \alpha_n\} \) is the representation of \( v \) with respect to the given basis
- We can write the representation as a row or column array of scalars
  \[ a = [\alpha_1, \alpha_2, \ldots, \alpha_n]^T = \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_n \end{bmatrix} \]

Example

- \( v = 2v_1 + 3v_2 - 4v_3 \)
- \( a = [2, 3, -4]^T \)
- Note that this representation is with respect to a particular basis
- For example, in WebGL we start by representing vectors using the object basis but later the system needs a representation in terms of the camera or eye basis
Coordinate Systems

- Which is correct?
- Both are because vectors have no fixed location

Frames

- A coordinate system is insufficient to represent points
- If we work in an affine space we can add a single point, the origin, to the basis vectors to form a frame

Representation in a Frame

- Frame determined by \((P_0, v_1, v_2, v_3)\)
- Within this frame, every vector can be written as
  \[ v = a_1 v_1 + a_2 v_2 + \ldots + a_6 v_6 \]
- Every point can be written as
  \[ P = P_0 + b_1 v_1 + b_2 v_2 + \ldots + b_6 v_6 \]

Confusing Points and Vectors

Consider the point and the vector
\[ P = P_0 + \beta_1 v_1 + \beta_2 v_2 + \ldots + \beta_6 v_6 \]
\[ v = a_1 v_1 + a_2 v_2 + \ldots + a_6 v_6 \]

They appear to have the similar representations
\[ p = [\beta_1 \beta_2 \beta_3] \quad v = [a_1 a_2 a_3] \]
which confuses the point with the vector.

A vector has no position

Vector can be placed anywhere

Homogeneous Coordinates

If we define \(0 \cdot P = \mathbf{0}\) and \(1 \cdot P = P\) then we can write
\[ v = a_1 v_1 + a_2 v_2 + a_3 v_3 = [a_1 a_2 a_3 0] v_1 v_2 v_3 P_0^T \]
\[ P = P_0 + \beta_1 v_1 + \beta_2 v_2 + \beta_3 v_3 = [\beta_1 \beta_2 \beta_3 1] v_1 v_2 v_3 P_0^T \]

Thus we obtain the four-dimensional homogeneous coordinate representation
\[ v = [a_1 a_2 a_3 0]^T \]
\[ p = [\beta_1 \beta_2 \beta_3 1]^T \]
Homogeneous Coordinates

The homogeneous coordinates form for a three dimensional point \([x \ y \ z]\) is given as
\[ p = [x' \ y' \ z' \ w]^T = [wx \ wy \ wz \ w]^T \]

We return to a three dimensional point (for \(w \neq 0\)) by
\[ x = x'/w \]
\[ y = y'/w \]
\[ z = z'/w \]
If \(w=0\), the representation is that of a vector
Note that homogeneous coordinates replaces points in three dimensions by lines through the origin in four dimensions
For \(w=1\), the representation of a point is \([x \ y \ z \ 1]\)

Change of Coordinate Systems

Consider two representations of the same vector with respect to two different bases. The representations are
\[ a = [a_1 \ a_2 \ a_3]^T \]
\[ b = [b_1 \ b_2 \ b_3]^T \]
where
\[ v = a_1v_1 + a_2v_2 + a_3v_3 = [a_1 \ a_2 \ a_3]^T [v_1 \ v_2 \ v_3]^T \]
\[ = b_1u_1 + b_2u_2 + b_3u_3 = [b_1 \ b_2 \ b_3]^T [u_1 \ u_2 \ u_3]^T \]

Matrix Form

The coefficients define a 3 x 3 matrix
\[ M = \begin{bmatrix} \gamma_{11} & \gamma_{12} & \gamma_{13} \\ \gamma_{21} & \gamma_{22} & \gamma_{23} \\ \gamma_{31} & \gamma_{32} & \gamma_{33} \end{bmatrix} \]
and the bases can be related by
\[ a = M^T b \]
see text for numerical examples

Homogeneous Coordinates and Computer Graphics

- Homogeneous coordinates are key to all computer graphics systems
- All standard transformations (rotation, translation, scaling) can be implemented with matrix multiplications using 4 x 4 matrices
- Hardware pipeline works with 4 dimensional representations
- For orthographic viewing, we can maintain \(w=0\) for vectors and \(w=1\) for points
- For perspective we need a perspective division

Representing second basis in terms of first

Each of the basis vectors, \(u_1, u_2, u_3\), are vectors that can be represented in terms of the first basis

\[ u_1 = \gamma_{11}v_1 + \gamma_{12}v_2 + \gamma_{13}v_3 \]
\[ u_2 = \gamma_{21}v_1 + \gamma_{22}v_2 + \gamma_{23}v_3 \]
\[ u_3 = \gamma_{31}v_1 + \gamma_{32}v_2 + \gamma_{33}v_3 \]

Change of Frames

We can apply a similar process in homogeneous coordinates to the representations of both points and vectors

Consider two frames:
\((P_0, v_1, v_2, v_3)\)
\((Q_0, u_1, u_2, u_3)\)

- Any point or vector can be represented in either frame
- We can represent \(Q_0, u_1, u_2, u_3\) in terms of \(P_0, v_1, v_2, v_3\)
Representing One Frame in Terms of the Other

Extending what we did with change of bases

\[ u^1 = g^{11}v^1 + g^{12}v^2 + g^{13}v^3 \]
\[ u^2 = g^{21}v^1 + g^{22}v^2 + g^{23}v^3 \]
\[ u^3 = g^{31}v^1 + g^{32}v^2 + g^{33}v^3 \]
\[ Q^0 = g^{41}v^1 + g^{42}v^2 + g^{43}v^3 + g^{44}P^0 \]

Defining a 4 x 4 matrix

\[
M = \begin{bmatrix}
\gamma_{11} & \gamma_{12} & \gamma_{13} & 0 \\
\gamma_{21} & \gamma_{22} & \gamma_{23} & 0 \\
\gamma_{31} & \gamma_{32} & \gamma_{33} & 0 \\
\gamma_{41} & \gamma_{42} & \gamma_{43} & 1
\end{bmatrix}
\]

Working with Representations

Within the two frames any point or vector has a representation of the same form

\[ a = [\alpha_1 \alpha_2 \alpha_3 \alpha_4] \] in the first frame
\[ b = [\beta_1 \beta_2 \beta_3 \beta_4] \] in the second frame

where \( \alpha_4 = \beta_4 = 1 \) for points and \( \alpha_4 = \beta_4 = 0 \) for vectors and

\[ a = M^\top b \]

The matrix \( M \) is 4 x 4 and specifies an affine transformation in homogeneous coordinates

The World and Camera Frames

- When we work with representations, we work with n-tuples or arrays of scalars
- Changes in frame are then defined by 4 x 4 matrices
- In WebGL, the base frame that we start with is the world frame
- Eventually we represent entities in the camera frame by changing the world representation using the model-view matrix
- Initially these frames are the same (\( M = I \))

Moving the Camera

If objects are on both sides of \( z = 0 \), we must move camera frame

\[
M = \begin{bmatrix}
0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & -d \\
0 & 0 & 0 & 1
\end{bmatrix}
\]