

Objectives

- Describing 3D objects
- Coordinate systems
 - Homogeneous Coordinates
- Frames
- Fractal terrain



Spaces

- Vector Spaces
 - Scalars and vectors
 - Vector addition, scalar-vector multiplication
- Affine spaces
 - Scalars, vectors and points
 - Point-point subtraction
- Euclidean spaces
 - Scalars, vectors and points
 - Inner product => distance



3D Primitives

- Objects in 3 dimensions:
 - Can be described by their surfaces
 - Specified as a set of vertices in 3D
 - Can be approximated by flat, convex polygons.
- Graphics systems can render millions of polygons per second
- Surfaces are 2D objects embedded in 3D



Coordinate systems

- Linear combination of vectors u_1, u_2, \dots, u_n
$$u = \alpha_1 u_1 + \alpha_2 u_2 + \dots + \alpha_n u_n$$
- If the only set of scalars that satisfy
$$\alpha_1 u_1 + \alpha_2 u_2 + \dots + \alpha_n u_n = 0$$
is $\alpha_1 = \alpha_2 = \dots = \alpha_n = 0$ then the vectors are **linearly independent**.
- Maximum number of independent vectors is the **dimension n** of the space.
- A set of n independent vectors forms a **basis**.
- In an Affine space we can also specify a reference point as the **origin**, basis plus origin define a **frame**.



Representation

- Until now we have been able to work with geometric entities without using any frame of reference, such 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

- In a 3D space we can represent vector w uniquely in terms of independent vectors v_1, v_2, v_3 , as $w = \alpha_1 v_1 + \alpha_2 v_2 + \alpha_3 v_3$
- Scalars α_1, α_2 , and α_3 are the **components** of w with respect to the basis v_1, v_2, v_3 .
- The **representation** of w is the column matrix:

$$a = \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{bmatrix}$$



Example

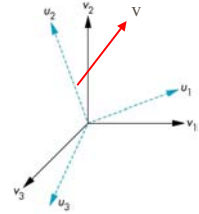
- $V=2v_1+3v_2-4v_3$
- $A=[2\ 3\ -4]$
- Note that this representation is with respect to a particular basis
- For example, in OpenGL we start by representing vectors using the world basis but later the system needs a representation in terms of the camera or eye basis



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

$$\begin{aligned}u_1 &= \gamma_{11}v_1 + \gamma_{12}v_2 + \gamma_{13}v_3 \\u_2 &= \gamma_{21}v_1 + \gamma_{22}v_2 + \gamma_{23}v_3 \\u_3 &= \gamma_{31}v_1 + \gamma_{32}v_2 + \gamma_{33}v_3\end{aligned}$$



Matrix Form

The coefficients define a 3 x 3 matrix

$$M = \begin{bmatrix} \gamma_{11} & \gamma_{12} & \gamma_{13} \\ \gamma_{21} & \gamma_{22} & \gamma_{23} \\ \gamma_{31} & \gamma_{32} & \gamma_{33} \end{bmatrix}$$

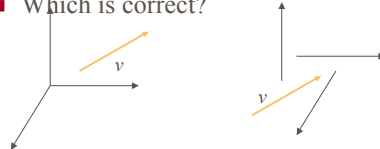
and the basis can be related by

$$a = M^T b$$



Coordinate Systems

- Which is correct?



- Both are because vectors have no fixed location



Frames

- Basis v_1, v_2, v_3 , and Origin P_0 :
 - Vectors : $v = \alpha_1 v_1 + \alpha_2 v_2 + \alpha_3 v_3$
 - Points : $P = P_0 + \beta_1 v_1 + \beta_2 v_2 + \beta_3 v_3$
- Typical unit basis for Euclidean R^3
 - $i = (1, 0, 0)^T$
 - $j = (0, 1, 0)^T$
 - $k = (0, 0, 1)^T$



Changing Coordinate Systems

- Points and vectors exist independent of reference system, but have to be represented wrt one.
- Frequently we have a model in a world frame and a different camera frame.
- We want to know how objects appear to the camera.
- Use **Model View Matrix** to convert from world to camera.



Confusing Points and Vectors

Consider the point and the vector

$$P = P_0 + \beta_1 v_1 + \beta_2 v_2 + \dots + \beta_n v_n$$

$$v = \alpha_1 v_1 + \alpha_2 v_2 + \dots + \alpha_n v_n$$

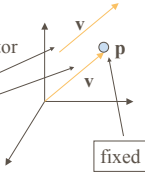
They appear to have the similar representations

$$p = [\beta_1 \beta_2 \beta_3] \quad v = [\alpha_1 \alpha_2 \alpha_3]$$

which confuse the point with the vector

A vector has no position

can place anywhere



Homogeneous Coordinates

- If we use the same 3-tuple representation

- for points: $P = P_0 + x v_1 + y v_2 + z v_3$

- And lines: $w = \delta_1 v_1 + \delta_2 v_2 + \delta_3 v_3$
 $p = [x, y, z]^T \quad w = [\delta_1, \delta_2, \delta_3]^T$

- How do we distinguish between them?

- **Homogenous coordinate representation**

use $[v_1, v_2, v_3, P_0]^T$ to represent the frame:

$$P = [x \ y \ z \ 1]^T \quad w = [\delta_1 \ \delta_2 \ \delta_3 \ 0]^T$$



Homogeneous Coordinates

The general form of four dimensional homogeneous coordinates is

$$p = [x \ y \ x \ w]^T$$

We return to a three dimensional point (for $w \neq 0$) by

$$x \leftarrow x/w$$

$$y \leftarrow y/w$$

$$z \leftarrow z/w$$

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



Homogeneous Coordinates and Computer Graphics

- Homogeneous coordinates are key to all computer graphics systems
 - All standard transformations (rotation, translation, scaling) can be implemented by matrix multiplications with 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*



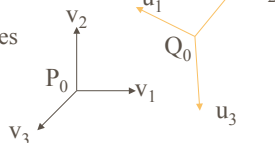
Change of Frames

- 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 each



Representing One Frame in Terms of the Other

Extending what we did with change of bases

$$u_1 = \gamma_{11} v_1 + \gamma_{12} v_2 + \gamma_{13} v_3$$

$$u_2 = \gamma_{21} v_1 + \gamma_{22} v_2 + \gamma_{23} v_3$$

$$u_3 = \gamma_{31} v_1 + \gamma_{32} v_2 + \gamma_{33} v_3$$

$$Q_0 = \gamma_{41} v_1 + \gamma_{42} v_2 + \gamma_{43} v_3 + \gamma_{44} P_0$$

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

$\mathbf{a} = [\alpha_1 \ \alpha_2 \ \alpha_3 \ \alpha_4]$ in the first frame

$\mathbf{b} = [\beta_1 \ \beta_2 \ \beta_3 \ \beta_4]$ in the second frame

where $\alpha_4 = \beta_4 = 1$ for points and $\alpha_4 = \beta_4 = 0$ for vectors
and

$$\mathbf{a} = \mathbf{M}^T \mathbf{b}$$

The matrix \mathbf{M} is 4 x 4 and specifies an affine transformation in homogeneous coordinates



Frames in OpenGL

- Two frames used in OpenGL:
 - Camera frame
 - World frame
- **Model view matrix** transforms homogenous-coordinate representations of vectors and points from world to camera frame.



Moving the Camera

If objects are on both sides of $z=0$, we must move

$$\mathbf{M} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -d \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

