15-462: Computer Graphics	**Carnegie Mellon University**

Topic (9/7/6)

Lecturer: James Kuffner *Scribes: Rosen Diankov*

Basis: An array of linearly independent vectors that span a certain coordinate system. The span of vectors **a** and **b** is

$$span(\mathbf{a}, \mathbf{b}) = \{\mathbf{v} : \mathbf{v} = v_1\mathbf{a} + v_2\mathbf{b}, v_1, v_2 \in \mathbb{R}\}$$

The dimension of a space is the minimum number of basis vectors needed to fully span it. So a basis for an n-dimensional space $\mathbb{R}^n$ requires n linearly independent vectors. Interested in orthogonal bases $\mathbf{e_1}, \mathbf{e_2}, ...$ such that

$$\mathbf{e_i} \cdot \mathbf{e_j} = \delta_{i,j} \quad \text{where } \delta_{i,j} \text{ is the Kronecker delta}$$

Let's take the 2D plane and the most common basis: $\mathbf{e_1} = (1, 0)$ and $\mathbf{e_2} = (0, 1)$, and let's rotate the basis vectors counter-clockwise by θ. Then $\mathbf{e'_1} = (\cos(\theta), \sin(\theta))$ and $\mathbf{e'_2} = (-\sin(\theta), \cos(\theta))$. So any vector $\mathbf{x} = x\mathbf{e_1} + y\mathbf{e_2}$ can be rotated by θ just by changing its basis to $\mathbf{e'_1}$ and $\mathbf{e'_2}$ so that the rotated vector is $\mathbf{x'} = x\mathbf{e'_1} + y\mathbf{e'_2}$. This can be shorted to matrix multiplication by a rotation matrix:

$$R(\theta) = \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix}$$

$$\mathbf{x'} = R(\theta)\mathbf{x}$$

3D rotations can easily be generalized from the 2D case. A rotation usually happens counter-clockwise around a certain axis (right hand axis). For example a rotation about the **x**-axis will rotate in the **yz** plane. The **yz** rotation just becomes a 2D rotation. The following matrices are the rotations for each of the 3 axes:

$$R_x(\theta) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) \\ 0 & \sin(\theta) & \cos(\theta) \end{pmatrix}, \qquad R_y(\theta) = \begin{pmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{pmatrix}, \qquad R_z(\theta) = \begin{pmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{pmatrix},$$

A common method to represent arbitrary rotations of an object is Euler angles in which rotations around each axis are combined like

$$R = R_x(\theta_x)R_y(\theta_y)R_z(\theta_z)$$

Since matrix multiplication is not commutative, a different permutation of the axis rotations will yield a different rotation. The biggest problem with Euler angles are that singularities exist. Given certain parameters of θ_x and θ_y, then varying θ_z might produce very little to no rotation[1]! Another problem is the

[1] This is called Gimbal lock

interpolation properties. If we want to animate an object from one rotation to the next, a solution might be to linearly interpolate the Euler angles between the first and second rotations. However, this will produce a very uneven and weird animation. What we would like instead is find the shortest rotation that we can apply to move an object to the second rotation. Later on, we'll see how quaternions can be used to achieve this. The basic idea is that we'll be representing every rotation by an axis of rotation and the angle to rotate around that axis.

Because a rotation matrix's column vectors are orthogonal and unit, the inverse of a rotation matrix R is R^T. Any rotation matrix R has $det(R) = 1$.

There are two types of transformations we're interested in: rigid body and affine transformations. Rigid body transformations only deal with rotations and translations, so they preserve the angles and local structure of the object. Affine transformations further add scaling and shearing. To translate a point by (t_x, t_y), each point will need to have a 3rd homogeneous coordinate that is always 1. In 2D the operation will look like:

$$\underbrace{\begin{pmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{pmatrix}}_{T} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} x + t_x \\ y + t_y \\ 1 \end{pmatrix}$$

A rigid transformation in 2D that rotates the point first and then translates will then look like:

$$TR(\theta) = \begin{pmatrix} \cos(\theta) & -\sin(\theta) & t_x \\ \sin(\theta) & \cos(\theta) & t_y \\ 0 & 0 & 1 \end{pmatrix}$$

A matrix that scales s_x and s_y along each axis is represented by:

$$\begin{pmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

The 3D equivalent of the transformations trivially follow in derivation. If a matrix M represents an affine transformation, then $det(M) >= 0$.[2] For example the matrix $\begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix}$ is a reflection about the **x**-axis and its determinant is -1.

[2] The converse doesn't hold because non-affine transformations can have a positive determinant.