Geometry for Graphics

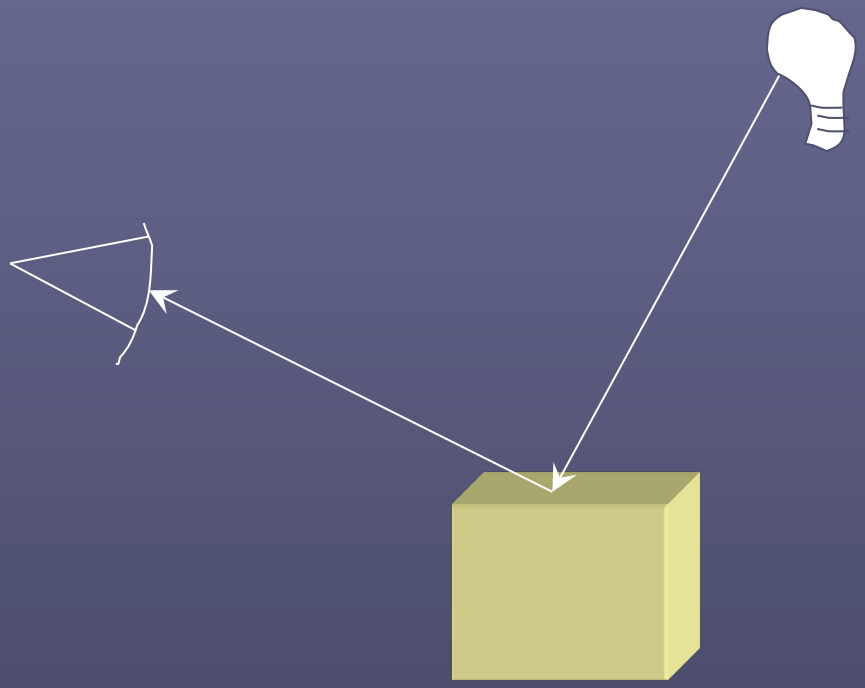
January 10, 2007

Cleanup from last time

- Remaining questions?

Review from last time

- Ray tracing: Following paths of light to an imaginary camera



Geometry for graphics

- If it doesn’t have any math, it probably isn’t worth doing
- It it has too much math, it definitely isn’t worth doing

Geometry for graphics

What types of geometric queries would be useful to perform?

- Given a line and a point, which side of the line does the point lie on?
- Given two lines, do they intersect and where?
- Others?

Geometric entities for ray tracing

- Point
- Vector
- Triangle
- Line
- Line segment
- Ray
- Plane
- Hyperplane

Vector space

If V is a vector space, then the following are true ($\vec{X}, \vec{Y}, \vec{Z}$ are vectors, r, s are scalars):

- Commutativity:
$$\vec{X} + \vec{Y} = \vec{Y} + \vec{X}$$
- Associativity of vector addition:
$$(\vec{X} + \vec{Y}) + \vec{Z} = \vec{X} + (\vec{Y} + \vec{Z})$$
- Additive Identity:
$$\vec{X} + \vec{0} = \vec{0} + \vec{X} = \vec{X}$$

Vector space, continued

- Existence of additive inverse:
  $\vec{X} + (-\vec{X}) = 0$
- Associativity of scalar multiplication:
  $(rs)\vec{X} = r(s\vec{X})$
- Distributivity of scalar sums:
  $(r + s)\vec{X} = r\vec{X} + s\vec{X}$
- Distributivity of vector sums:
  $r + (\vec{X} + \vec{Y}) = r\vec{X} + r\vec{Y}$
- Scalar multiplication identity:
  $1\vec{X} = \vec{X}$

Euclidean space

- A Euclidean space is a vector space where the vector is three real numbers: $\Re^3$

Geometries

- Plane Geometry
- Parabolic Geometry
- Hyperbolic Geometry
- Euclidean Geometry
- Elliptic Geometry

Euclidean geometry

Euclidean geometry defines 5 postulates:

1. A straight line segment can be drawn joining any two points.
2. Any straight line segment can be extended indefinitely in a straight line.
3. Given any straight line segment, a circle can be drawn having the segment as radius and one endpoint as center.
4. All right angles are congruent.
5. If two lines are drawn which intersect a third in such a way that the sum of the inner angles on one side is less than two right angles, then the two lines inevitably must intersect each other on that side if extended far enough.

Another geometry

- What about projective geometry?

Image from www.confluence.org

Affine space

An affine space defines points with these properties (P,Q are points, A,B are vectors):

$$\vec{P} + \vec{0} = \vec{P}$$

$$\vec{P} + (\vec{A} + B) = (\vec{P} + \vec{A}) + \vec{B}$$

For any $\vec{Q}$, there exists $\vec{A}$ such that:

$$\vec{Q} = \vec{P} + \vec{A}$$

$$(\vec{A} = \vec{P} - \vec{Q})$$

Points and Vectors

- Why did we define points differently than vectors?

- Definition of linear transform T:

$$T(A+B) = T(A) + T(B)$$

$$sT(A) = T(sA)$$

Affine transformations

- An affine transformation preserves colinearity and ratios of distances
- Projective transform (ala OpenGL) is not Affine
- Rotation, scale, shear, translation, reflection are all affine

Affine transformations

$$M\vec{p} + \vec{T} = \vec{p}'$$

(where M is a 3x3 transform and T is a translation vector)

Can also be written as a 4x3 matrix:

$$\begin{bmatrix} M_{00} & M_{01} & M_{02} & T_X \\ M_{10} & M_{11} & M_{12} & T_Y \\ M_{20} & M_{21} & M_{22} & T_Z \end{bmatrix}$$

- Affine transformations of points:

$$\begin{bmatrix} M_{00} & M_{01} & M_{02} & T_x \\ M_{10} & M_{11} & M_{12} & T_y \\ M_{20} & M_{21} & M_{22} & T_z \end{bmatrix} \begin{bmatrix} P_x \\ P_y \\ P_z \\ 1 \end{bmatrix} = \begin{bmatrix} P'_x \\ P'_y \\ P'_z \\ 1 \end{bmatrix}$$

- Affine transformations of vectors:

$$\begin{bmatrix} M_{00} & M_{01} & M_{02} & T_x \\ M_{10} & M_{11} & M_{12} & T_y \\ M_{20} & M_{21} & M_{22} & T_z \end{bmatrix} \begin{bmatrix} V_x \\ V_y \\ V_z \\ 0 \end{bmatrix} = \begin{bmatrix} V'_x \\ V'_y \\ V'_z \\ 0 \end{bmatrix}$$

Programming note

There are at least three ways of distinguishing points and vectors:

1. Make separate Point and Vector classes
2. Make a single Vector class and store four values with 0 or 1 in the fourth component
3. Have two methods for transformation, one for points and one for vectors

Geometric consistency

- We defined axioms for adding vectors but what is the geometric meaning of $P_1 + P_2$?
- What about $s*P_1$?

Interpolation

- What about finding the midpoint of a line between two points?

$$P_{mid} = 0.5\ P_1 + 0.5\ P_2$$

- This is called an affine combination, and is valid for an arbitrary number of points only if the weights sum to 1

Affine transformations

- Affine transformations are also called Affine maps
- They have a few properties:

1. An Affine transform is linear when applied to vectors:
   a) T(A+B) = T(A)+T(B)
   b) T(sA) = sT(A)
2. An Affine transform preserves the plus operator for points/vectors
   a) T(P+V) = T(P)+T(V)
3. Affine transformations can be composed

Implementation

- Recommendation: Don’t define multiplication on points

Interpolate(double, Point, Point)

AffineCombination(double, Point, double, Point)

AffineCombination(double, Point, double, Point, double, Point)

Operators

- Operators that do make sense:

P = P+V

P = P-V

V = P-P

V = V + V

V = V - V

V = s * V

V = - V

Cheating

- Sometimes all this religion forces you to be inefficient
- This will allow you to work around it if necessary:

explicit Point(const Vector&)

explicit Vector(const Point&)

And/or:

Point Vector::asPoint()

Vector Point::asVector()

Also define V*V and a way to access each component

Vector operations

- There are a few other operations that we will need: first, an inner product
- Inner product: a way to multiply vectors resulting in a scalar with these properties:

$$\langle \vec{A} + \vec{B}, \vec{C} \rangle = \langle \vec{A}, \vec{B} \rangle + \langle \vec{A}, \vec{C} \rangle$$

$$\langle s\vec{A}, \vec{B} \rangle = s\langle \vec{A}, \vec{B} \rangle$$

$$\langle \vec{A}, \vec{B} \rangle = \langle \vec{B}, \vec{A} \rangle$$

$$\langle \vec{A}, \vec{A} \rangle \geq 0 \text{ and } \langle \vec{A}, \vec{A} \rangle = 0 \textit{ iff } \vec{A} = \vec{0}$$

More spaces

- Eulerian space is an inner product space
- More specifically, a Hilbert space

Dot product

- The inner product for Eulerian space is called the Dot product:

$A \cdot B = A_x B_x + A_y B_y + A_z B_z$

- Useful properties:
  - $A \cdot B = |A|\,|B| \cos \theta$
  - Invariant under rotation
  - is commutative, distributive, and associative

Distances

A·A is “magnitude” or length squared

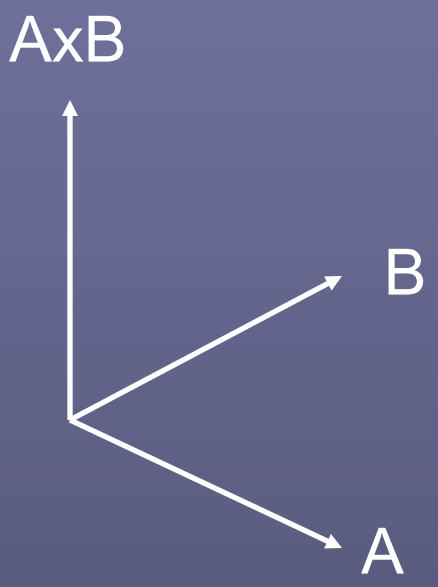
The distance between P1 and P2 is the length of the vector P2-P1

If A·A == 1, the vector is called a “unit vector” or is normalized

Multiply a vector by the inverse of the length to find a unit vector having the same direction

Cross product

$$\vec{C} = \vec{A} \times \vec{B}$$



Can be written as a determinant:

$$\begin{vmatrix} \vec{i} & \vec{j} & \vec{k} \\ A_x & A_y & A_z \\ B_x & B_y & C_z \end{vmatrix}$$

Expanded:

$$\left[A_y B_z + A_z B_y, A_z B_x + A_x B_z, A_x B_y + A_y B_x \right]$$

Cross product properties

$$\left|\vec{A} \times \vec{B}\right| = \left|\vec{A}\right| \sin\theta$$

$$\vec{A} \times \vec{B} = -\vec{B} \times \vec{A}$$

$$\vec{A} \times (\vec{B} + \vec{C}) = (\vec{A} \times \vec{B}) + (\vec{A} \times \vec{C})$$

$$(s\vec{A}) \times \vec{B} = s(\vec{A} \times \vec{B})$$

$$\left|\vec{A} \times \vec{B}\right| = \text{Area of swept rectangle}$$

If $\vec{C} = \vec{A} \times \vec{B}$, then $\vec{C}$ is perpendicular to A and B

$$(\vec{C} \cdot \vec{A} = \vec{C} \cdot \vec{B} = 0)$$

If $\vec{A} = \vec{B}$ or $\vec{A} = -\vec{B}$ then $\vec{A} \times \vec{B} = \vec{0}$

Scalar triple product

A·(BxC) = B·(CxA) = C·(AxB)

Can also be written as the determinant:

$$\begin{vmatrix} A_x & A_y & A_z \\ B_x & B_y & B_z \\ C_x & C_y & C_z \end{vmatrix}$$

This one is pretty cool, but we won't use it often.

Implementation notes

- Dot product and cross product are awkward. I have done it two ways in C++:
  - As a standalone function: Dot(A, B)
  - As a member method: A.dot(B)

- I don't recommend overloading ^ or some other obscure operator. Operator precedence will bite you and nobody will be able to read your code