Computational Geometry

Basics of Vectors and Vector Maths

Scalar – normal number

Vector – consists of components $\mathbf{A} = \langle A_x, A_y, A_z \rangle$

Magnitude of vector **A** (length of line from origin to **A**) $|\mathbf{A}| = \sqrt{A_x^2 + A_y^2 + A_z^2}$

Vector addition and subtraction work on individual components:

$\mathbf{A}+\mathbf{B} = \langle A_x+B_x, A_y+B_y, A_z+B_z \rangle$ $\mathbf{A}-\mathbf{B} = \langle A_x-B_x, A_y-B_y, A_z-B_z \rangle$

Vector multiplication has two forms:

Dot product: (gives a scalar result)

$\mathbf{A}\cdot\mathbf{B} = A_x \times B_x + A_y \times B_y + A_z \times B_z$

If θ is the angle between **A** and **B**: $\mathbf{A}\cdot\mathbf{B} = |\mathbf{A}| \times |\mathbf{B}| \times \cos\theta$

Cross Product:

$\mathbf{A}\times\mathbf{B} = \langle A_y \times B_z - B_y \times A_z, A_z \times B_x - B_z \times A_x, A_x \times B_y - B_x \times A_y \rangle$

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

Gives a vector result orthogonal to vectors **A** and **B** (perpendicular to the plane defined by vectors **A** and **B**). Direction of the result can be determined by the right-hand rule.

Vector Properties and Simple Geometric Problems

Useful basic properties:

$\mathbf{A}\cdot\mathbf{B} = 0 \quad \leftrightarrow \quad \mathbf{A} \perp \mathbf{B}$

$\mathbf{A}\times\mathbf{B} = 0 \quad \leftrightarrow \quad \mathbf{A} \parallel \mathbf{B}$

$\text{area } \Delta\mathbf{ABC} \quad = \quad \tfrac{1}{2} |(\mathbf{A}-\mathbf{B})\times(\mathbf{A}-\mathbf{C})|$

When working in 2D, we let $A_z = B_z = 0$. Under such conditions, the sign of the Z component of **A**×**B** gives us information on the orientation of **A** and **B**.

In 2D, with a clockwise angle θ between **A** and **B** when positioned head-to-tail:

$\mathbf{A}\times\mathbf{B} > 0$	$\leftrightarrow$	$\theta < 180°$	(left-hand turn)
$\mathbf{A}\times\mathbf{B} = 0$	$\leftrightarrow$	$\theta = 180°$	(corresponds with basic properties)
$\mathbf{A}\times\mathbf{B} < 0$	$\leftrightarrow$	$\theta > 180°$	(right-hand turn)

This can also be used to determine which side of a line a point is on.

Other formulae:

Distance from point **D** to line AB

$|(\mathbf{A}-\mathbf{D})\times(\mathbf{A}-\mathbf{B})| \div |\mathbf{A}-\mathbf{B}|$

Distance from point **D** to plane ABC

$|((\mathbf{A}-\mathbf{B})\times(\mathbf{A}-\mathbf{C})) \cdot (\mathbf{P}-\mathbf{A})| \div |(\mathbf{A}-\mathbf{B})\times(\mathbf{A}-\mathbf{C})|$

Standard problems:

- Given points **A**, **B**, **C** and **D**, determine if line segments AB and CD intersect – Use cross products to check that **A** and **B** are on either side of CD and that **C** and **D** are on either side of AB. If it is in 3D, you must also check that all 4 points lie on the same plane.

- Check whether a point **A** lies inside a *convex* polygon – Average the vertices of the polygon to find a point **B** inside it. Then for every edge of the polygon, check that **A** and **B** are on the same side of that edge (cross products).
- Check whether a point **A** lies inside a *concave* polygon – Use a technique called ray casting: draw a line in a random direction from **A** to the extent of your coordinate system and counter the number of times it intersects with edges of the polygon. An odd number of intersections means **A** lies inside, an even number means it lies outside.

<u>Applications to More Complex Problems</u>

- Create a simple closed path for a given set of points – Find the anchor point **A** with the lowest y-coordinate (break ties by lowest x-coordinate). Sort all other points according to the angle from point **A** to that point. The simple closed path is the path starting at **A** and visiting the other vertices in the sorted order.
- Find the convex hull of a given set of points – Start by finding the order of points in the simple closed path for the set. Add the anchor point and the first of the sorted points to the hull. Thereafter, iterate through the sorted points in order. If the current point forms a left-hand turn with the last two points of the convex hull, it can be added to the hull. If it forms a right-hand turn, points must be removed from the end of the convex hull until the current point does form a left-hand turn with the last two and the current point is added to the hull. The cross product can be used to determine which way it is turning.

<u>Implementation</u>

Considerations:

1. Make provision for border cases, such as multiple collinear points, points falling on lines or lines passing through vertices.
2. When working with real numbers, never check for exact equality. Rather check a value to within a certain range (called tolerance).

Using the complex number class:

Variables declared and used as such:

```
complex<type> point1,point2; //type can be any number type eg. int or double
real(point1) = 0; //assignment to the real part
imag(point1) = 0; //assignment to the imaginary part
real(point2) = imag(point1)+2;
```

For complex numbers $(a + bi)$ and $(c + di)$:

$$(a + bi)(c + di) = (ac - bd) + (ad + bc)i$$

or $$(a - bi)(c + di) = (ac + bd) + (ad - bc)i$$

From which we can see that the real and imaginary components correspond to the dot and cross products of two vectors $\langle a, -b, 0\rangle$ and $\langle c, d, 0\rangle$.

The conjugate of $(a + bi)$ is $(a - bi)$.

Therefore we can easily write the dot and cross products in terms of the components of the product of two complex numbers.

```
dot_product = real( conj(point1) * point2 );
cross_product = imag( conj(point1) * point2 );
```