Introduction to Graphics

Lecture 6:

Manipulation of 3D Objects

Lecture Overview

- Halfspaces
- Convex Objects
- Convexity and Containment Tests
- Vector formulations
- Clipping Algorithms
- Introduction to OpenGL

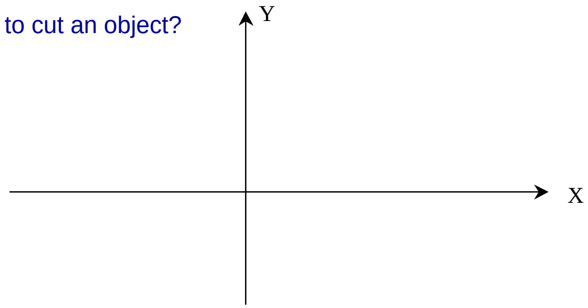
Viewpoint + Orientation ⇒ Clipping in 3D!

- Need to find out which parts of the 3D object are inside 3D window
- The 3D window is a convex object, usually bound by planes.
- This object defines the space that is visible from the viewpoint in the direction the viewer is looking.

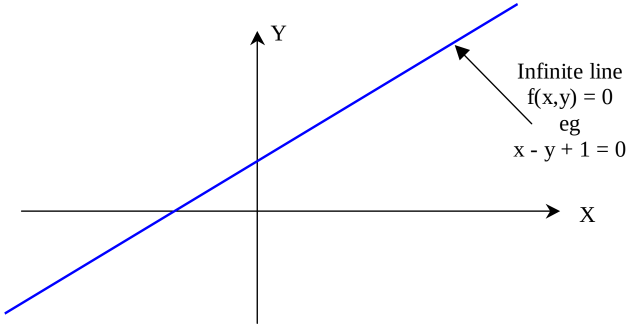
3D clipping = cutting 3D objects with the planes of the window

The concept of a halfspace

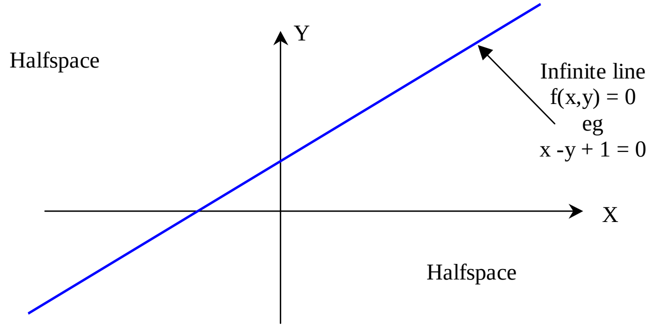
How to cut an object?



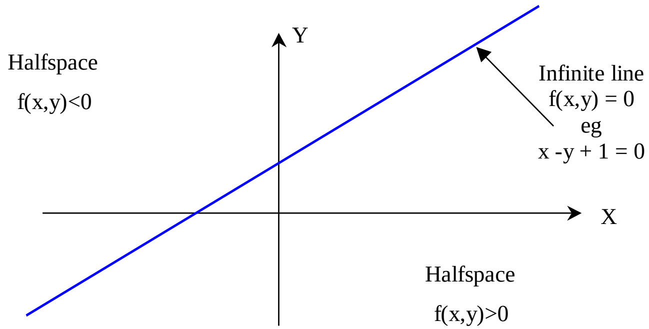
The concept of a halfspace



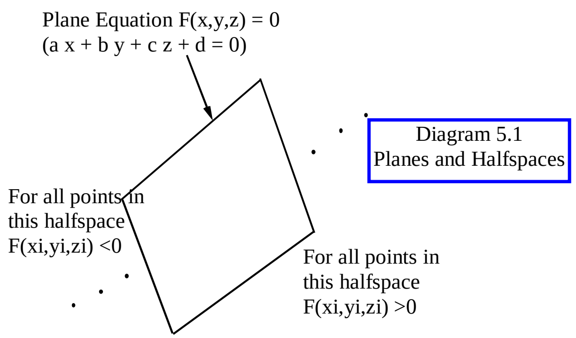
The concept of a halfspace



The concept of a halfspace



The same idea extends to three dimensions



Using the halfspace concept

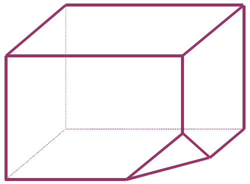
- Use the halfspace property for a number of algorithms for manipulating graphics scenes.

  e.g. 3D clipping, convex or not, removing hidden lines, etc

- Consider convex objects and an algorithm to determine whether an object is convex or not.

Two Definitions of Convex

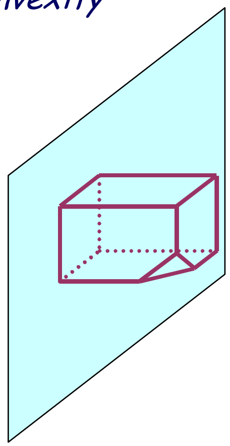
1. A line joining any two points on the boundary lies inside the object.
2. The object is the intersection of planar halfspaces.



Algorithm for determining if an object is convex

convex = *true*
for each face of the object
{ find the plane equation of the face f (x,y,z) = 0
choose one object point (xi,yi,zi) not on the face
and find *sign*(f (xi,yi,zi))
for all other points of the object
{ *if* (*sign*(f(xj,yj,zj)) not = *sign*(f (xi,yi,zi)))
then convex = *false*
}
}

Testing for Convexity



Testing for Containment

- A frequently encountered problem is to determine whether a point is inside an object or not.
- For clipping algorithms:

  Determine if a point is inside the 3D window (a convex object by itself!)

Algorithm for Containment

let the test point be [xt,yt,zt]

contained = *true*

for each face of the object

{ find the plane equation of the face f (x,y,z) = 0

choose one object point (xi,yi,zi) not on the face

and find *sign*(f (xi,yi,zi))

if (*sign*(f(xt,yt,zt)) not = *sign*(f (xi,yi,zi)))

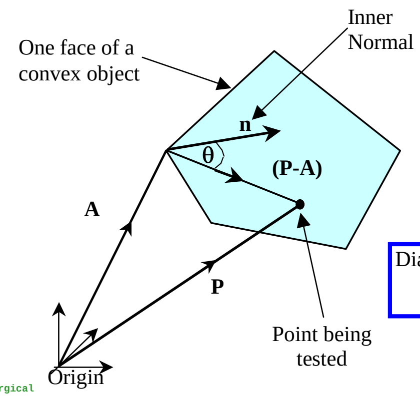
then contained = *false*

}

Vector formulation

- The same test can be expressed in vector form.
- To avoid the need to calculate the cartesian equation of the plane, store the normal vector **n** to each face of the object.

Vector test for containment



Contained of θ is acute
ie Cos(θ) is positive
or **n**.(**P-A**) is positive
n.(**P-A**) = |**n**||**P-A**|Cos(θ)

Diagram 5.2 Containment within a convex object

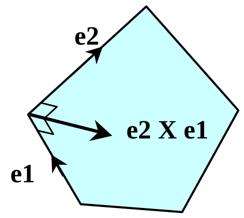
Normal vector to a face

- The vector formulation does not require finding the plane equation of a face, but it does require finding a normal vector to the plane.

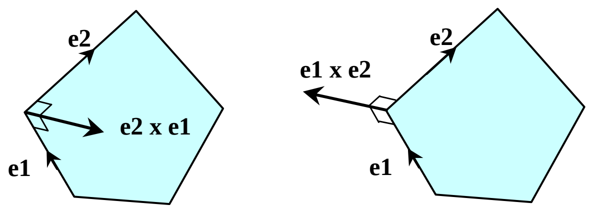
(same thing really since for plane ax + by +cz + d=0 a normal vector is [a,b,c]!)

Finding a normal vector

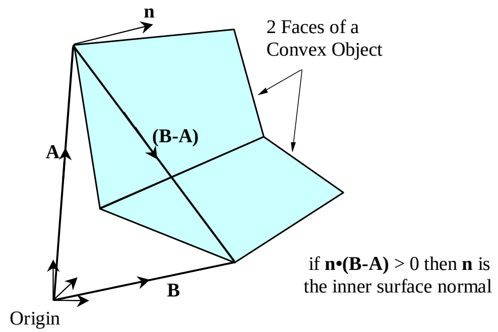
The normal vector can be found from the cross product of two vectors on the plane, say two edge vectors.



But which normal vector points inwards?



Checking the normal direction



An Example

A face of a convex object lies in the plane

3x+5y+7z +1 = 0 and a vertex is {-1,-1,1}

The normal vector is therefore **n** = {3,5,7}

1. If another vertex of the object is {1,1,1} determine whether **n** is an inner or outer surface normal.

2. Determine whether the point {1,0,-1} is on the inside or the outside of the face.

Solution 1

P-A = {1,1,1} - {-1,-1,1}

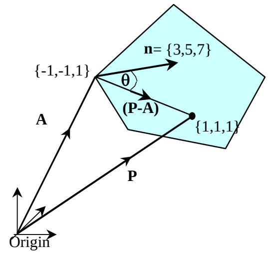
= {2,2,0}

n.(P-A) = 6 + 10 = 16

n.(P-A) is positive,

θ is acute

n is an inner normal



Solution 2

Method 1:

The plane has equation 3x+5y+7z+1=0

i.e. f(x,y,z) = 3x+5y+7z+1

For the internal point {1,1,1} f(1,1,1) = 16

For the test point {1,0,-1} f(1,0,-1) = -3

The signs are different, so the test point is on the outside

Solution 3

Method 2:

The inner surface normal is **n** = {3,5,7}

for the test point **P** = {1,0,-1} and vertex **A**={-1,-1,1}

P-A = {2,1,-2}

n.(P-A) = -3

Thus the angle to the normal is > 90 an the point is on the outside

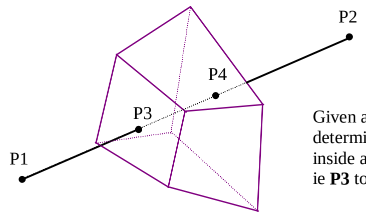
Clipping

Containment is an important property used in clipping algorithms.

Clipping is used to remove unwanted parts of a graphics scene before drawing.

It can be applied in computer aided design, and graphics scene design.

Clipping a line to a convex polyhedron



Given a line segment **P1** to **P2** determine the part of the line inside a convex object, ie **P3** to **P4**

Clipping algorithm

✓The algorithm checks the line against every face of the convex polyhedron.

✓It determines whether the end points of the line are on the inside or the outside of the face

✓This can be done by the halfspace or the dot product with the inner normal as before [**n.**(**P1 - A**) & **n.**(**P2 - A**)].

Case 1: Both P_1 and P_2 are on the outside

Both signs -ve or zero

The line is completely clipped (no part of it is inside the polyhedron)

The algorithm terminates

Case 2: Both P_1 and P_2 are on the inside

Both signs -ve or zero

There is no new information.

If there are more faces to test the algorithm continues to the next face.

Otherwise the line is completely inside the volume.

Case 3/4: P_1/ P_2 is outside and P_2/ P_1 is inside

Compute the intersection between the line and plane.

- for any vector **p** lying on the plane **n.p** = 0
- let the intersection point be $\mu_i \mathbf{P}_{2/1} + (1-\mu_i)\mathbf{P}_{1/2}$

if A is a vertex of the object a vector on the plane is

- $\mu_i \mathbf{P}_{2/1} + (1-\mu_i)\mathbf{P}_{1/2} - \mathbf{A}$
- thus $\mathbf{n}.(\mu_i \mathbf{P}_{2/1} + (1-\mu_i)\mathbf{P}_{1/2} - \mathbf{A}) = 0$

we can solve this for μ_i and find the point of intersection

Replace $\mathbf{P}_1$/ $\mathbf{P}_2$ with the intersection