Interactive Computer Graphics: Lecture 3

Clipping

Some slides adopted from
F. Durand and B. Cutler, MIT

The Graphics Pipeline

Modelling Transformations

Illumination (Shading)

Viewing Transformation (Perspective / Orthographic)

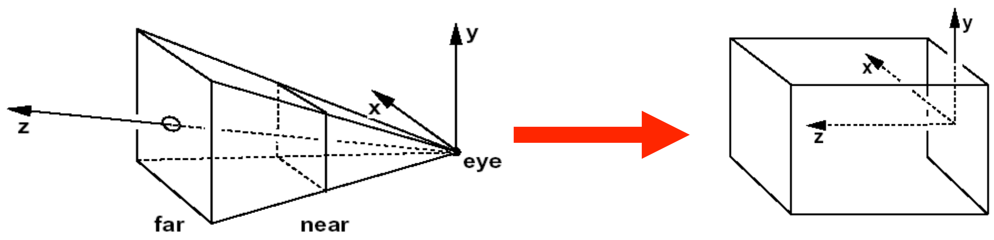
Clipping

Projection (to Screen Space)

Scan Conversion (Rasterization)

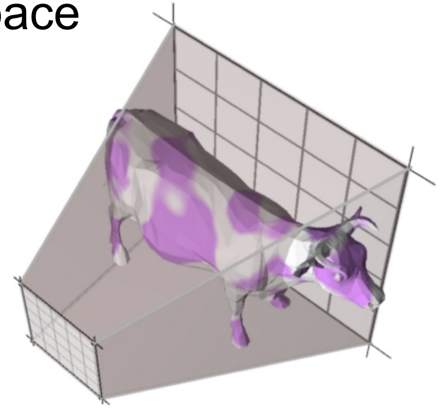
Visibility / Display

- Portions of the scene outside the viewing volume (view frustum) are removed (clipped)
- Transform to Normalized Device Coordinates



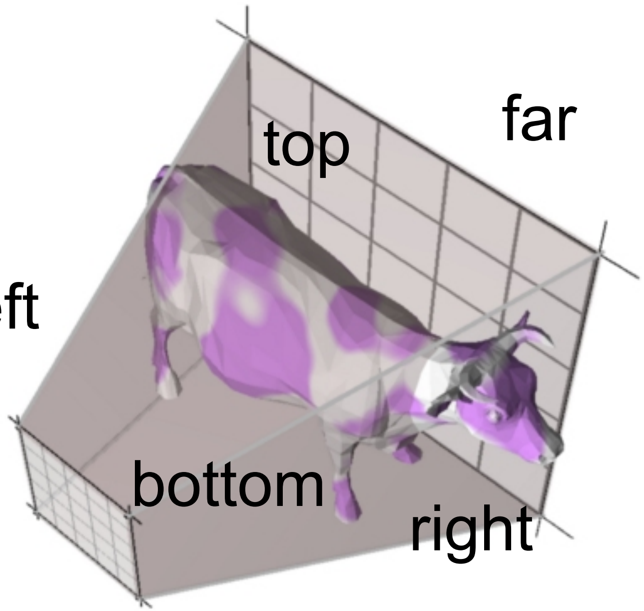
Eye space

NDC



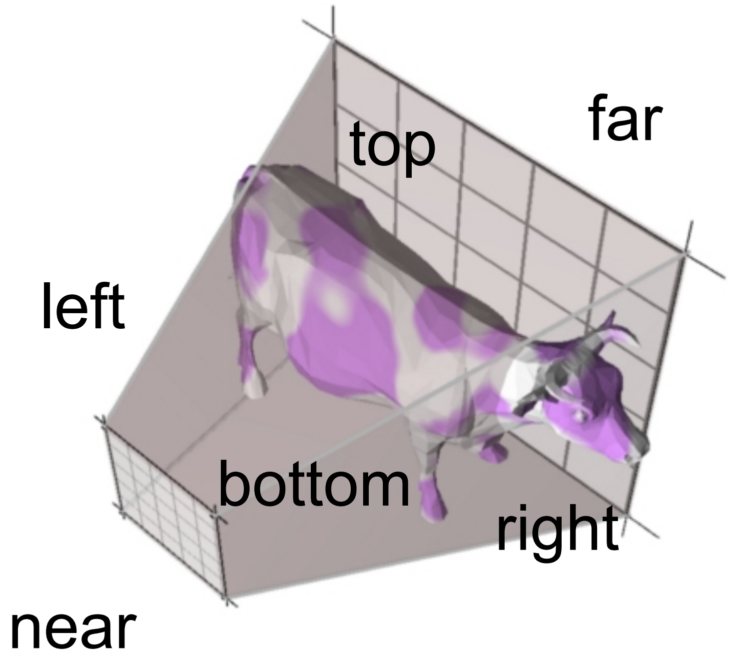
Clipping

- Eliminate portions of objects outside the viewing frustum
- View frustum
  - boundaries of the image plane projected in 3D
  - a near & far clipping plane
- User may define additional clipping planes



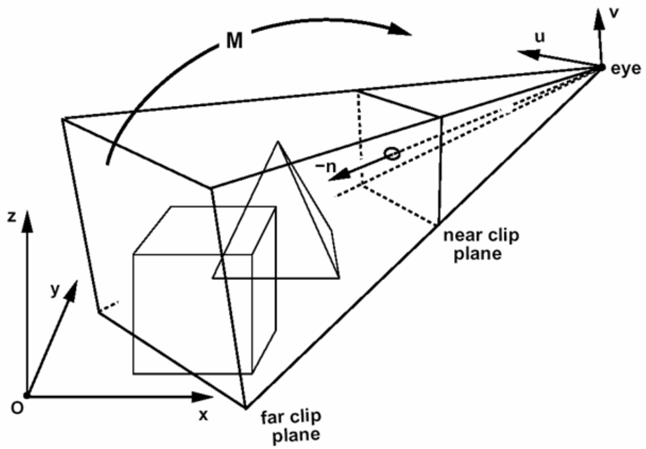
Why clipping ?

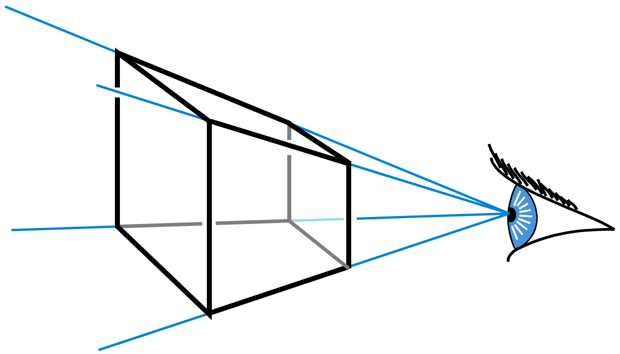
- Avoid degeneracy
  - e.g. don't draw objects behind the camera
- Improve efficiency
  - e.g. do not process objects which are not visible

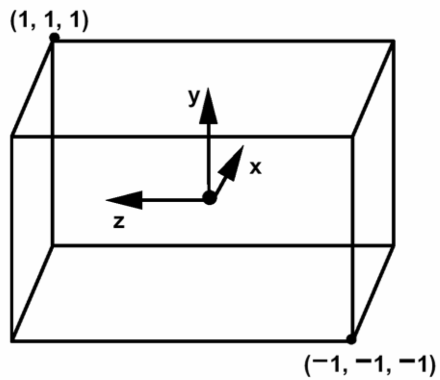


When to clip?

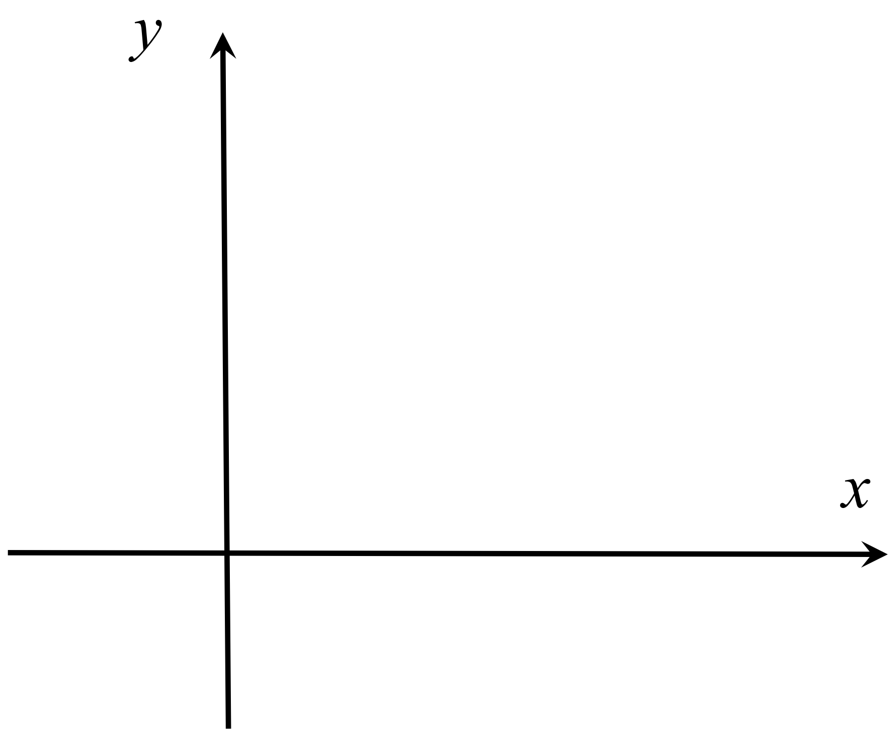
- Before perspective transform in 3D space
  - use the equation of 6 planes
  - natural, not too degenerate
- In homogeneous coordinates after perspective transform (clip space)
  - before perspective divide (4D space, weird w values)
  - canonical, independent of camera
  - simplest to implement
- In the transformed 3D screen space after perspective division
  - problem: objects in the plane of the camera



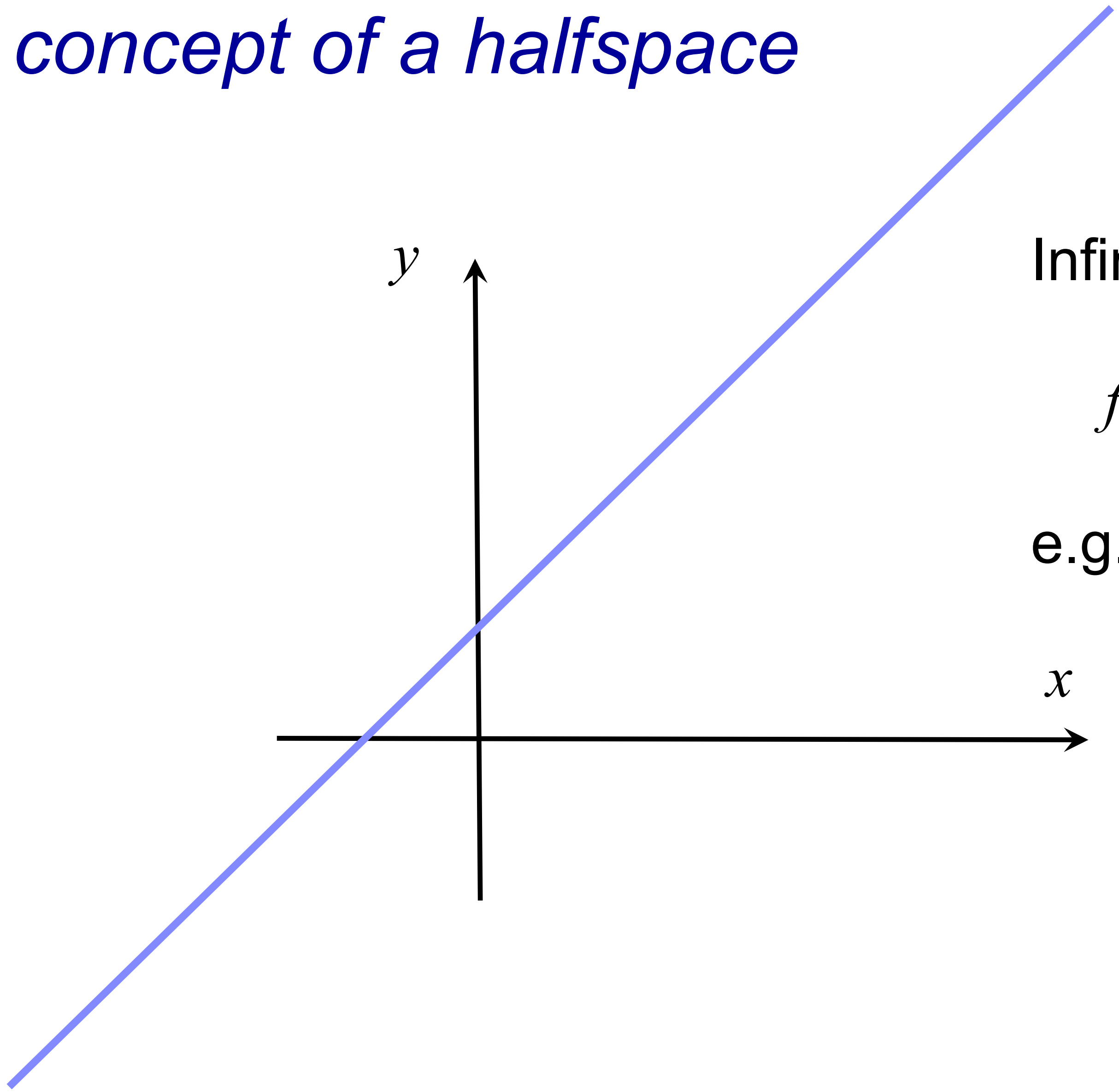




The concept of a halfspace



The concept of a halfspace

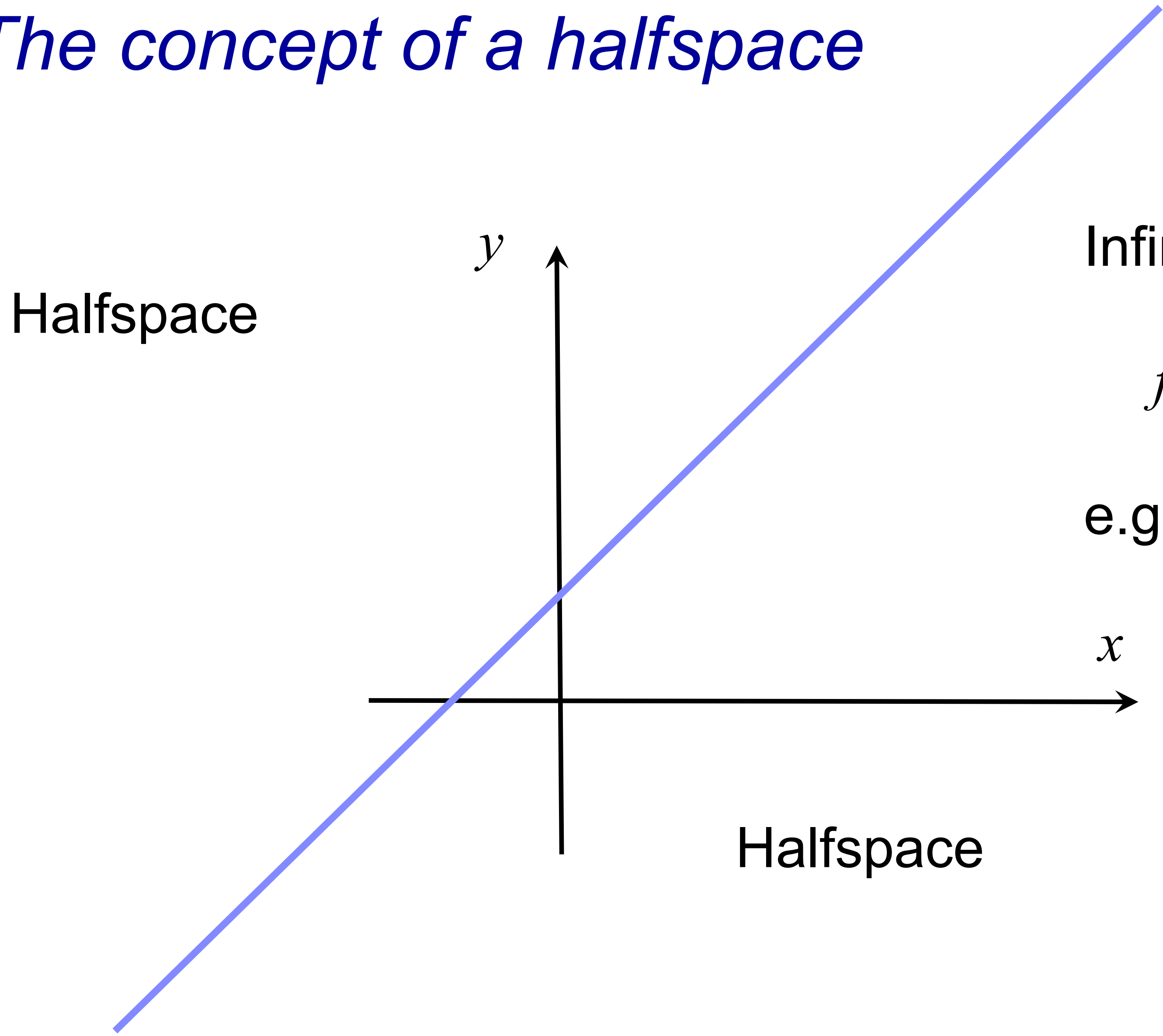


Infinite line:

$$f(x, y) = 0$$

e.g. $x - y + 1 = 0$

The concept of a halfspace

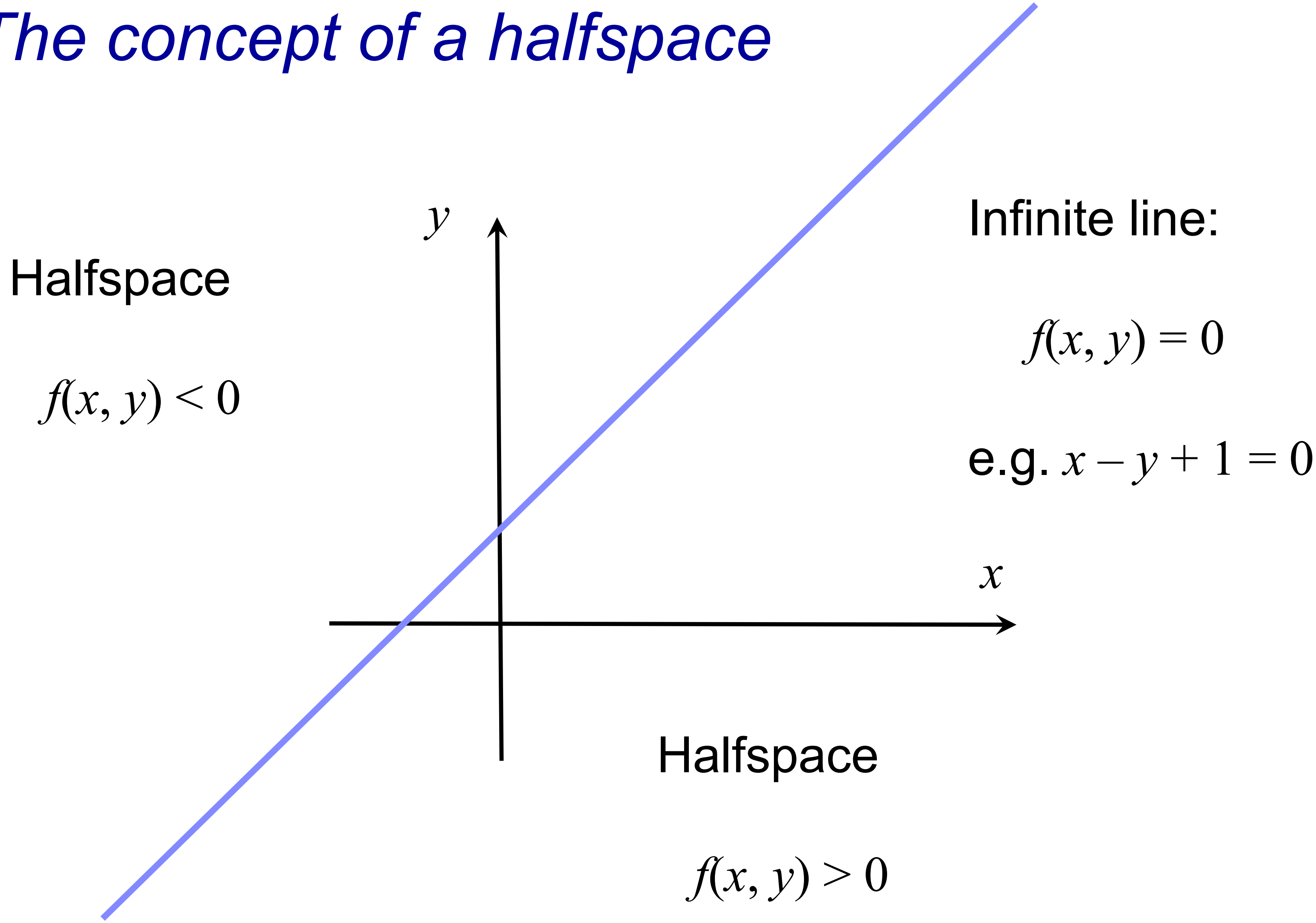


Infinite line:

$$f(x, y) = 0$$

e.g. $x - y + 1 = 0$

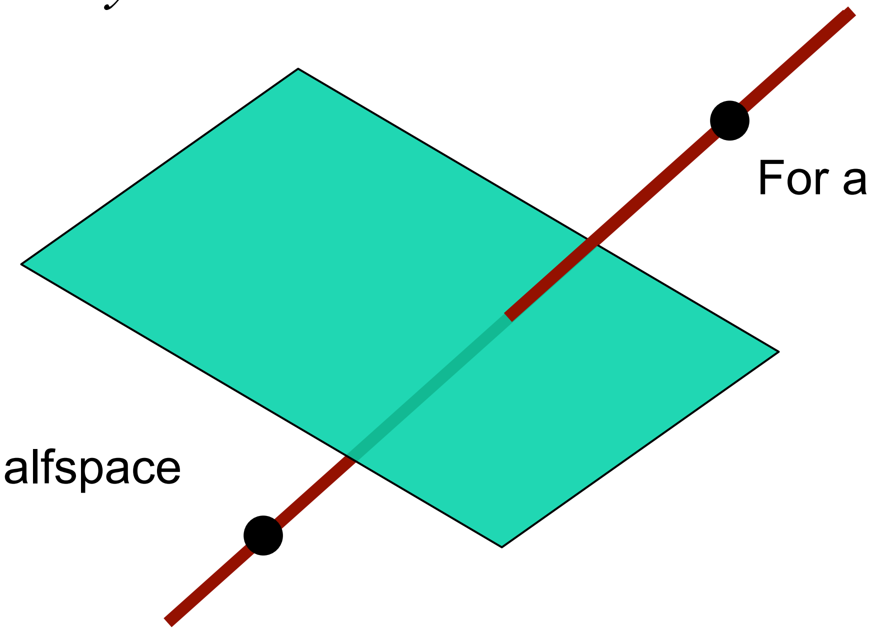
The concept of a halfspace



The concept of a halfspace in 3D

Plane equation $f(x, y, z) = 0$

or $Ax + By + Cz + D = 0$



For all points in this halfspace

$f(x, y, z) > 0$

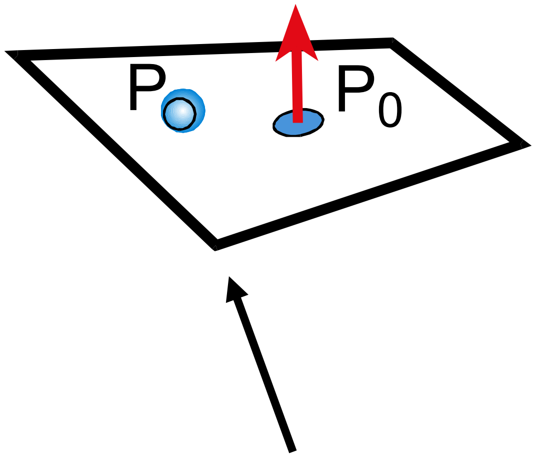
For all points in this halfspace

$f(x, y, z) < 0$

Reminder: Homogeneous Coordinates

- Link plane equation $Ax + By + Cz + D = 0$ with vector $\mathbf{H} = (A, B, C, D)^T$ in homogeneous coordinates
- Each point (x, y, z, w) has an infinite number of equivalent homogenous coordinates:

$$(sx, sy, sz, sw)\ ,\ s \neq 0$$



$\mathbf{H} = (A, B, C, D)^T$

- Relates to infinite number of equivalent plane equations:

$$sAx + sBy + sCz + sD = 0 \rightarrow \mathbf{H} = \begin{pmatrix} sA \\ sB \\ sC \\ sD \end{pmatrix}$$

Point-to-Plane Distance

- Scale **H** so that (A, B, C) becomes normalized, i.e. that

  $$A^2 + B^2 + C^2 = 1$$

- Then distance is easily calculated

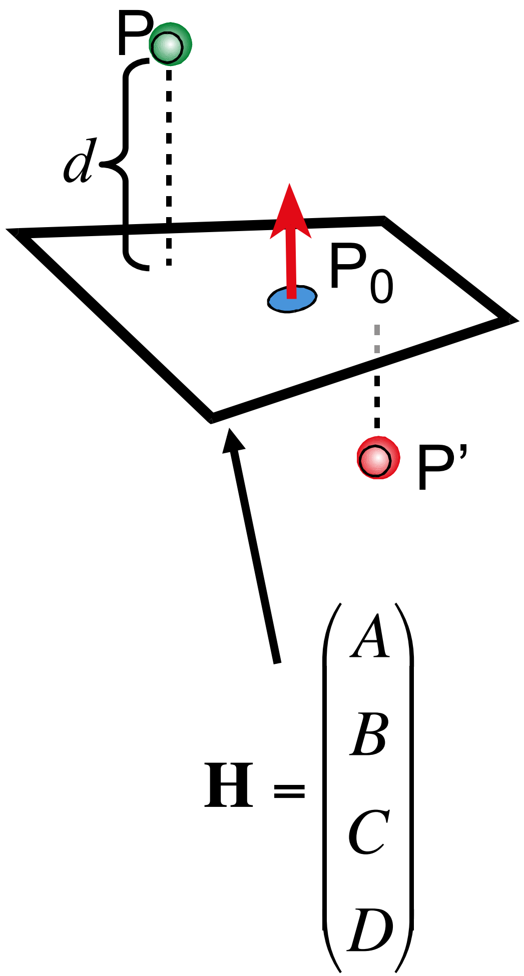
  $$d = \mathbf{H} \bullet \mathbf{p} = \mathbf{H}^T \mathbf{p}$$

  n.b. dot product is in *homogeneous* coordinates

- d is a *signed distance:*

  positive = "inside"

  negative = "outside"



Which side of the plane is a point on?

(Recall the planes in the frustum)

- If $d = \mathbf{H} \bullet \mathbf{p} \geq 0$

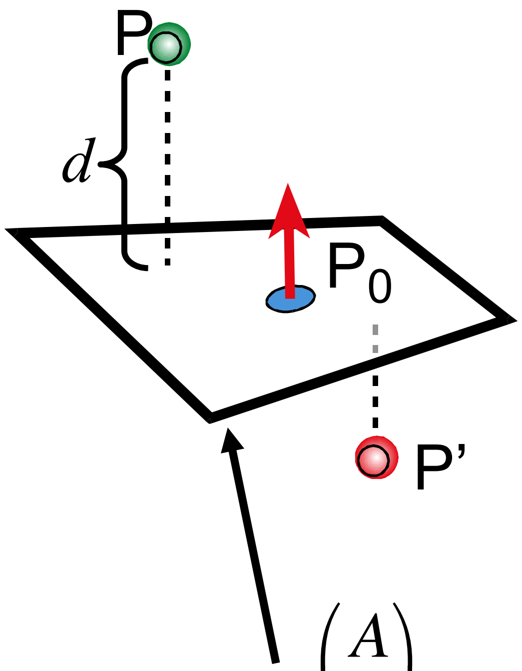
  Pass through

- If $d = \mathbf{H} \bullet \mathbf{p} < 0$

  Clip (or cull or reject)

Don't really need to normalize A,B,C

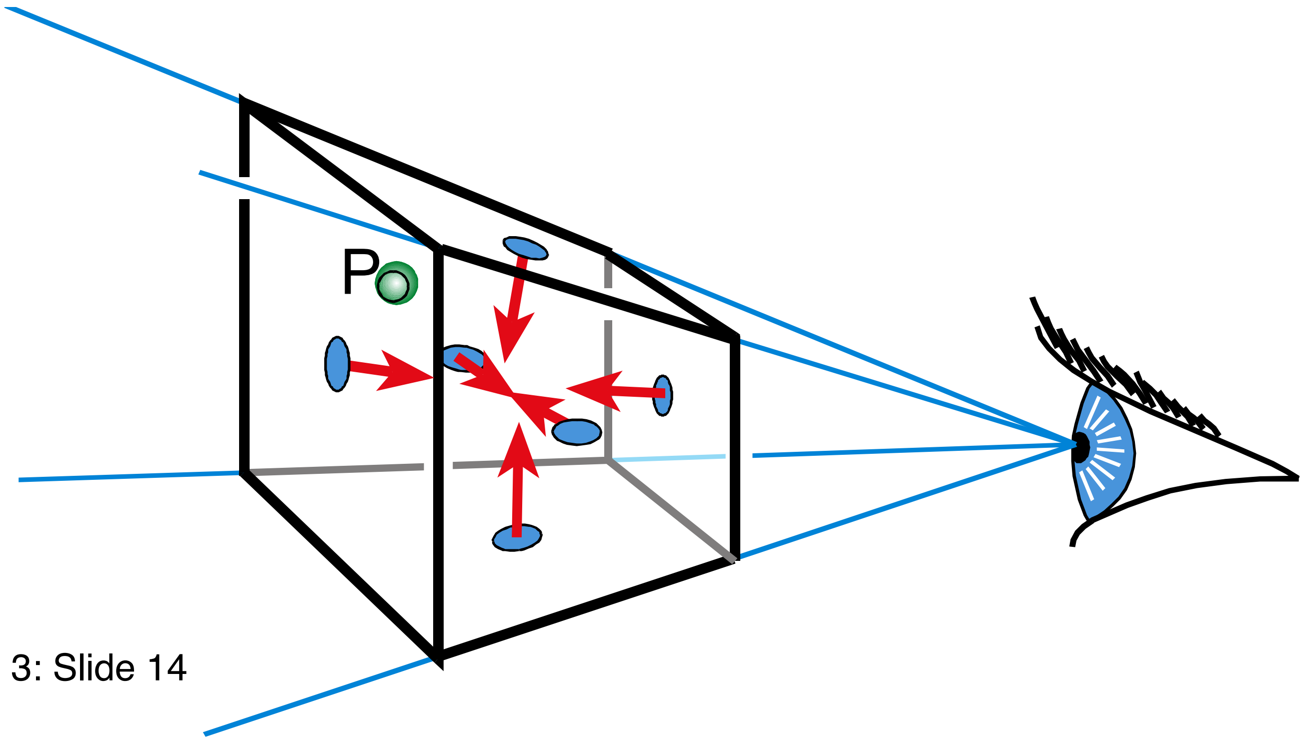
We only test the *sign* of $\mathbf{H}\bullet\mathbf{p}$



$$\mathbf{H} = \begin{pmatrix} A \\ B \\ C \\ D \end{pmatrix}$$

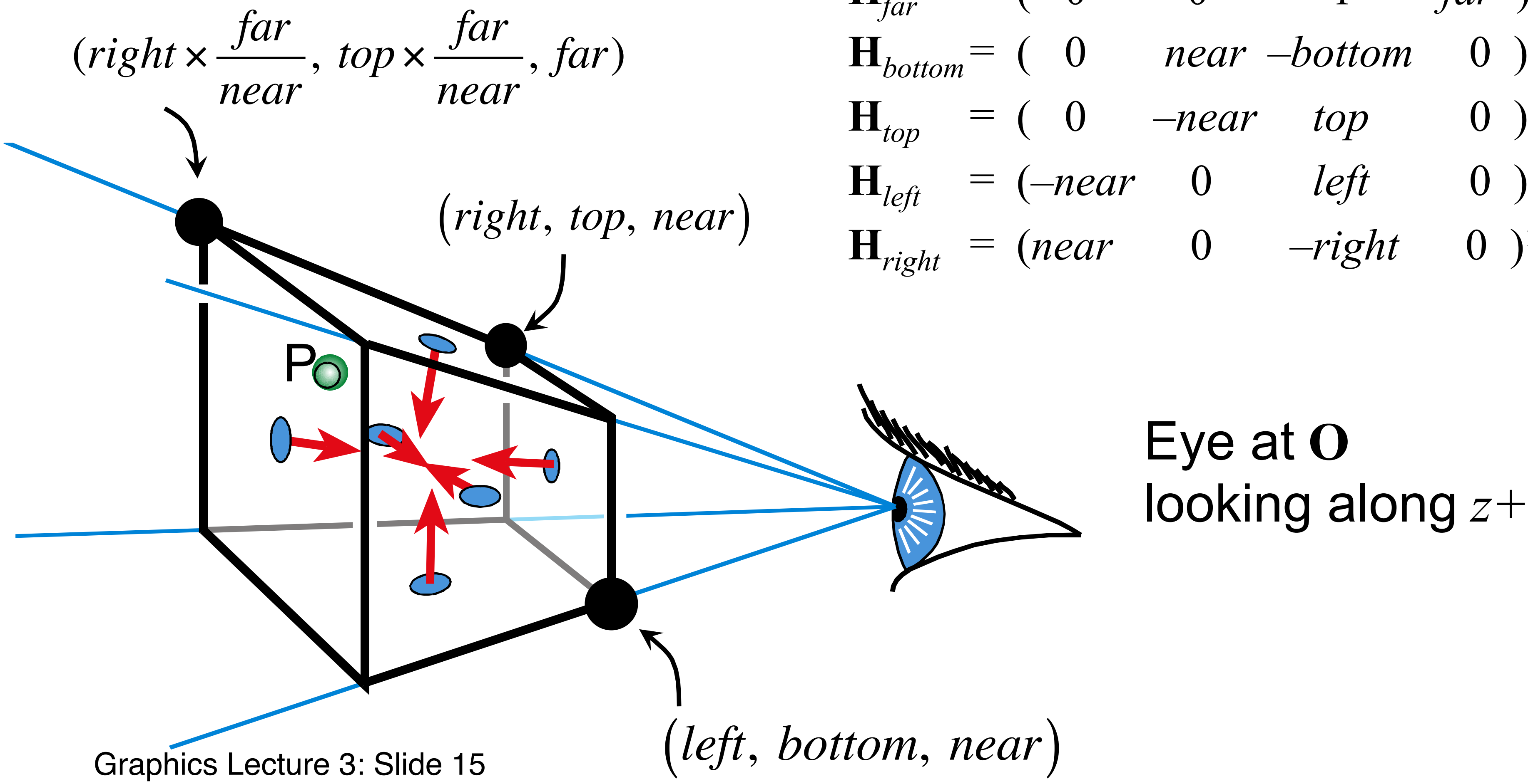
Clipping with respect to View Frustum

- Test point $\mathbf{p}$ against each of the 6 planes
  - Normals oriented towards the interior
  - Each has its own $\mathbf{H}$
- If $\mathbf{H} \bullet \mathbf{p} < 0$ for any $\mathbf{H}$ then clip $\mathbf{p}$ ('cull' / 'reject')



What are the View Frustum Planes?

$$
\begin{array}{lcrrrrl}
\mathbf{H}_{near} & = & (& 0 & 0 & 1 & -near\ )^T \\
\mathbf{H}_{far} & = & (& 0 & 0 & -1 & far\ )^T \\
\mathbf{H}_{bottom} & = & (& 0 & near & -bottom & 0\ )^T \\
\mathbf{H}_{top} & = & (& 0 & -near & top & 0\ )^T \\
\mathbf{H}_{left} & = & (& -near & 0 & left & 0\ )^T \\
\mathbf{H}_{right} & = & (& near & 0 & -right & 0\ )^T
\end{array}
$$



Example derivation

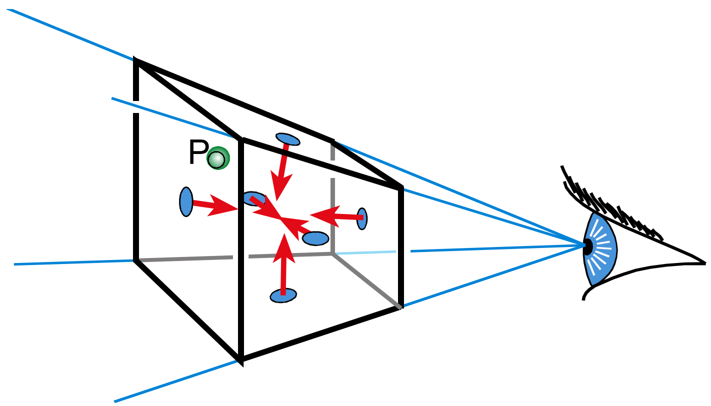
$$\begin{pmatrix} l \\ b \\ n \end{pmatrix} \times \begin{pmatrix} r \\ b \\ n \end{pmatrix} = \begin{vmatrix} \hat{i} & \hat{j} & \hat{k} \\ l & b & n \\ r & b & n \end{vmatrix}$$

$$\hat{i}(bn - bn) + \hat{j}(rn - ln) + \hat{k}(lb - rb) \equiv (0, n, -b)^T$$

$$(\mathbf{P} - \mathbf{P}_1) \cdot \mathbf{n} = \mathbf{0}$$

$$\Rightarrow \quad ny - bz = 0$$

$$\Rightarrow \quad \mathbf{H}_{bottom} = (0, n, -b, 0)^T$$



Line-Plane Intersection

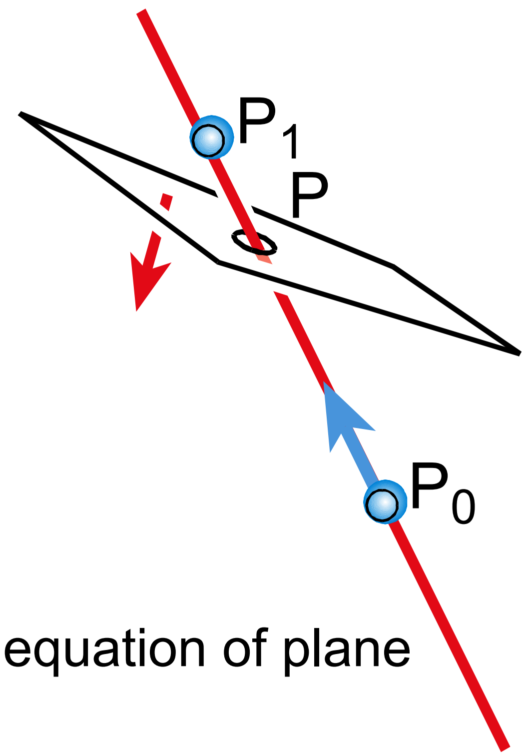
- Sometimes we need to clip lines and line segments!

- Explicit (Parametric) Line Equation

$$\mathbf{L}(\mu) = \mathbf{p}_0 + \mu\,(\mathbf{p}_1 - \mathbf{p}_0)$$

or

$$\mathbf{L}(\mu) = \mu\,\mathbf{p}_1 + (1-\mu)\,\mathbf{p}_0$$



- How do we intersect?
  - Insert explicit equation of line into implicit equation of plane
  - use the normal vector

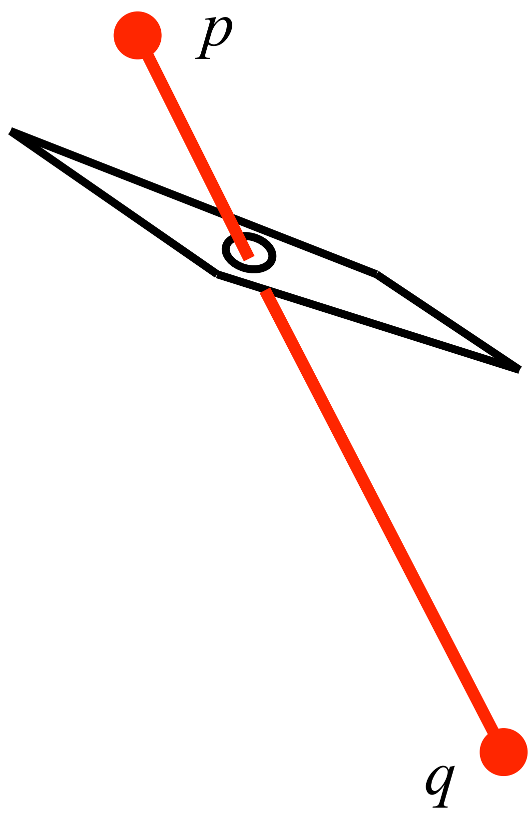
Line-Plane Intersection: Example method

To compute intersection line joining $\boldsymbol{p}_0$, $\boldsymbol{p}_1$ and plane*:*

1. For any vector $\mathbf{w}$ lying in the plane $\mathbf{n} \bullet \mathbf{w} = 0$
2. Let the intersection point be $\mu\,\mathbf{p}_1 + (1-\mu)\,\mathbf{p}_0$
3. Choose $\mathbf{v}$ to be *any* point on the plane.
4. A vector in the plane is given by $\mu\mathbf{p}_1 + (1-\mu)\mathbf{p}_0 - \mathbf{v}$
5. So $\mathbf{n} \bullet (\mu\mathbf{p}_1 + (1-\mu)\mathbf{p}_0 - \mathbf{v}) = 0$
6. We can solve this for μ and hence find the point of intersection

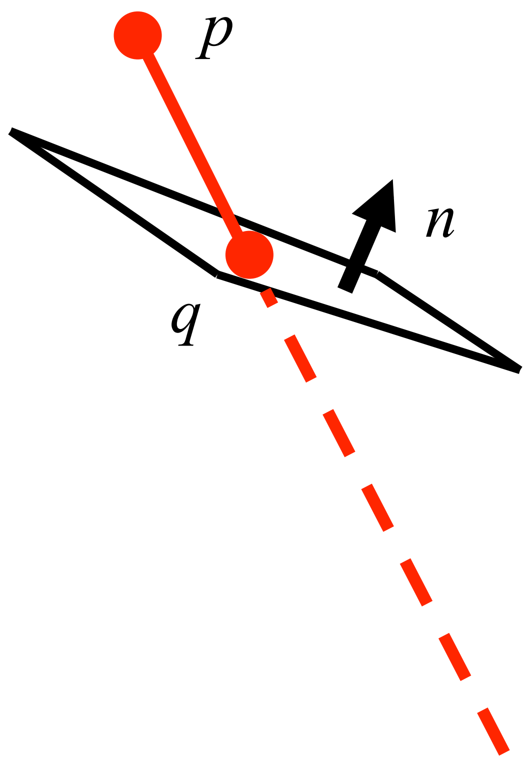
Segment Clipping

- If $\mathbf{H} \bullet \mathbf{p} > 0$ and $\mathbf{H} \bullet \mathbf{q} < 0$
- If $\mathbf{H} \bullet \mathbf{p} < 0$ and $\mathbf{H} \bullet \mathbf{q} > 0$
- If $\mathbf{H} \bullet \mathbf{p} > 0$ and $\mathbf{H} \bullet \mathbf{q} > 0$
- If $\mathbf{H} \bullet \mathbf{p} < 0$ and $\mathbf{H} \bullet \mathbf{q} < 0$



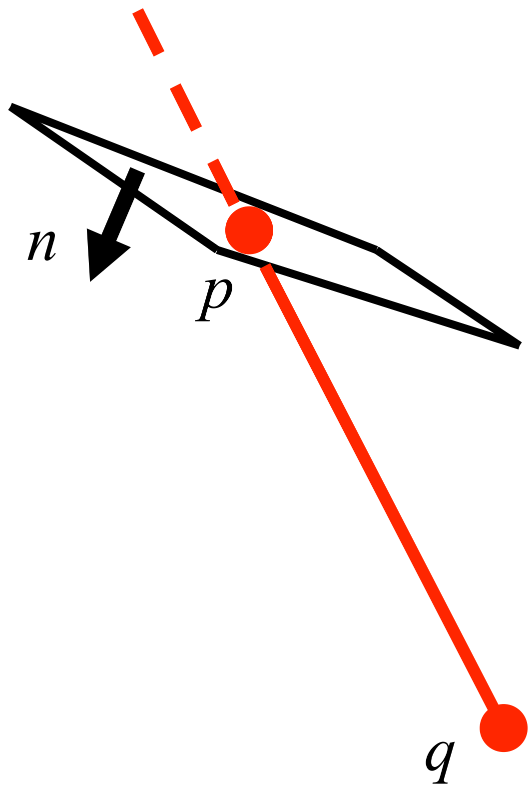
Segment Clipping

- If $\mathbf{H} \bullet \mathbf{p} > 0$ and $\mathbf{H} \bullet \mathbf{q} < 0$
  - clip q to plane
- If $\mathbf{H} \bullet \mathbf{p} < 0$ and $\mathbf{H} \bullet \mathbf{q} > 0$
- If $\mathbf{H} \bullet \mathbf{p} > 0$ and $\mathbf{H} \bullet \mathbf{q} > 0$
- If $\mathbf{H} \bullet \mathbf{p} < 0$ and $\mathbf{H} \bullet \mathbf{q} < 0$



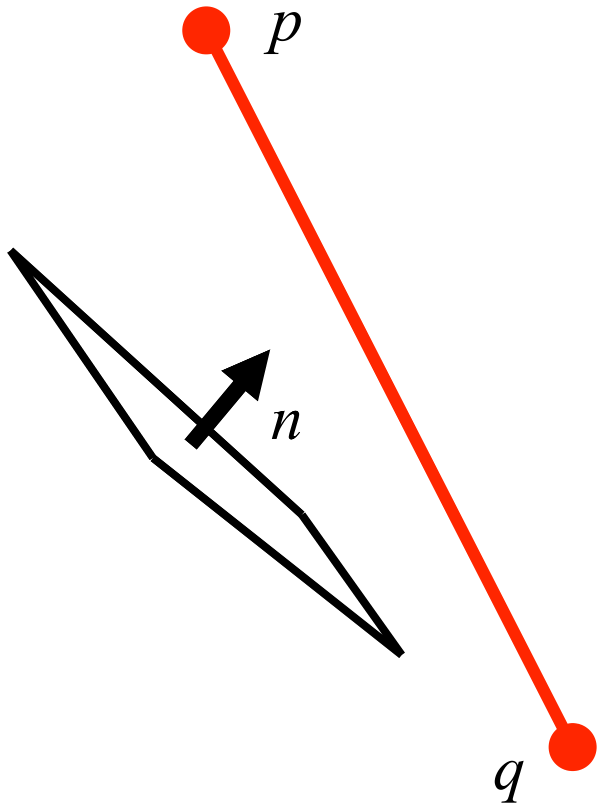
Segment Clipping

- If $\mathbf{H} \bullet \mathbf{p} > 0$ and $\mathbf{H} \bullet \mathbf{q} < 0$
  - clip q to plane
- If $\mathbf{H} \bullet \mathbf{p} < 0$ and $\mathbf{H} \bullet \mathbf{q} > 0$
  - clip p to plane
- If $\mathbf{H} \bullet \mathbf{p} > 0$ and $\mathbf{H} \bullet \mathbf{q} > 0$
- If $\mathbf{H} \bullet \mathbf{p} < 0$ and $\mathbf{H} \bullet \mathbf{q} < 0$



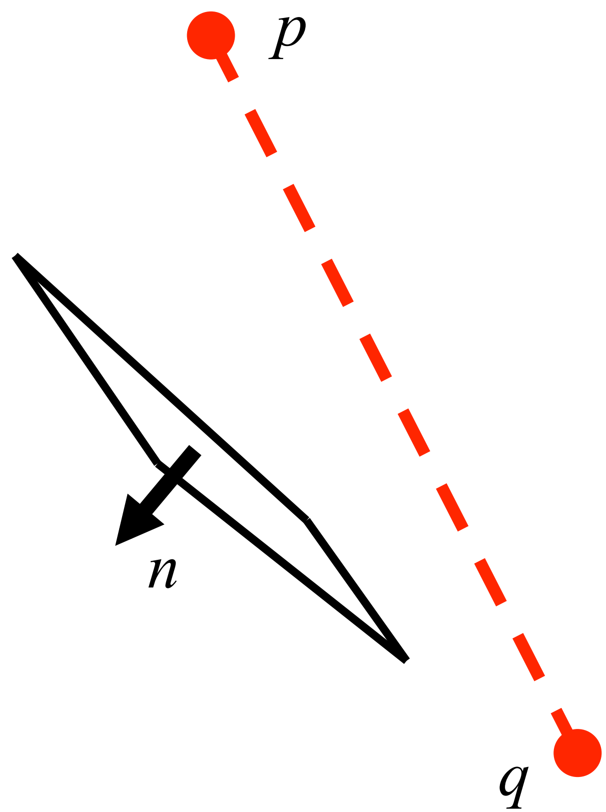
Segment Clipping

- If $\mathbf{H} \bullet \mathbf{p} > 0$ and $\mathbf{H} \bullet \mathbf{q} < 0$
  - clip q to plane
- If $\mathbf{H} \bullet \mathbf{p} < 0$ and $\mathbf{H} \bullet \mathbf{q} > 0$
  - clip p to plane
- If $\mathbf{H} \bullet \mathbf{p} > 0$ and $\mathbf{H} \bullet \mathbf{q} > 0$
  - pass through
- If $\mathbf{H} \bullet \mathbf{p} < 0$ and $\mathbf{H} \bullet \mathbf{q} < 0$

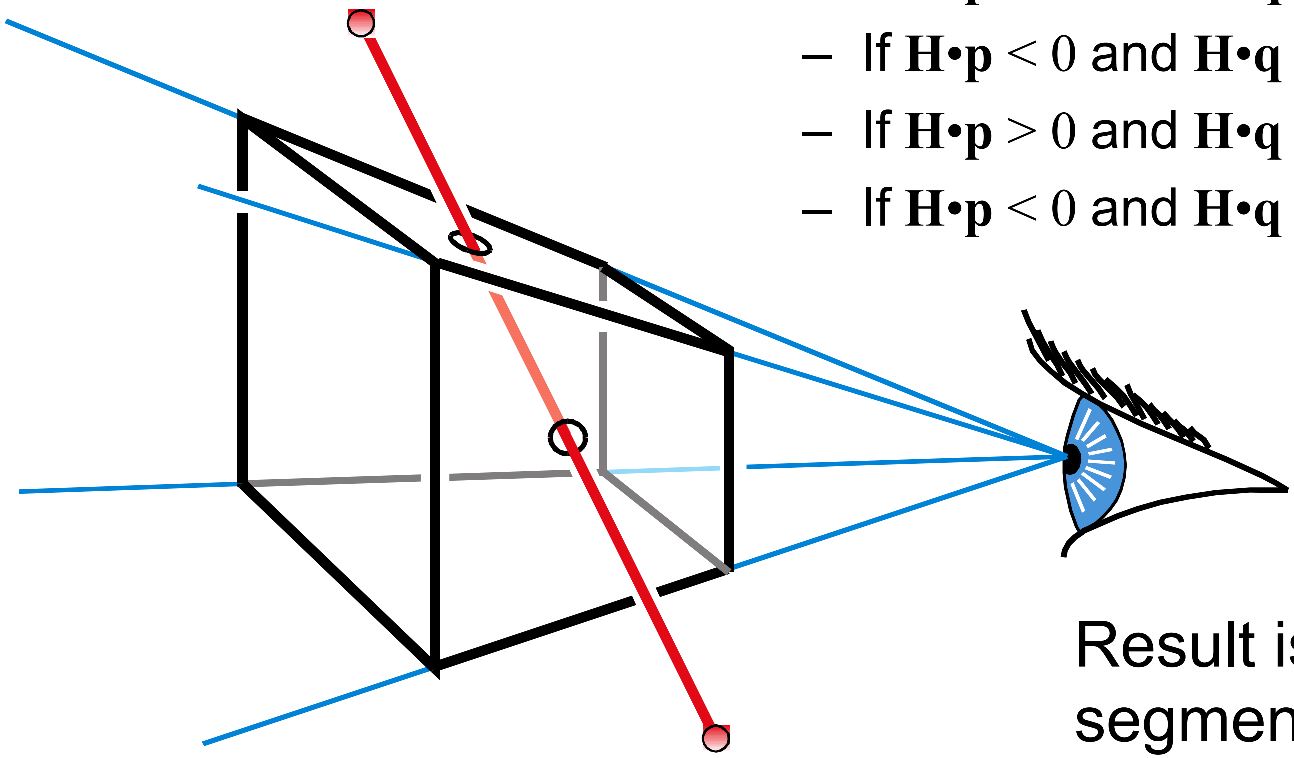


Segment Clipping

- If $\mathbf{H} \bullet \mathbf{p} > 0$ and $\mathbf{H} \bullet \mathbf{q} < 0$
  - clip q to plane
- If $\mathbf{H} \bullet \mathbf{p} < 0$ and $\mathbf{H} \bullet \mathbf{q} > 0$
  - clip p to plane
- If $\mathbf{H} \bullet \mathbf{p} > 0$ and $\mathbf{H} \bullet \mathbf{q} > 0$
  - pass through
- If $\mathbf{H} \bullet \mathbf{p} < 0$ and $\mathbf{H} \bullet \mathbf{q} < 0$
  - clipped out



Clipping against the frustum



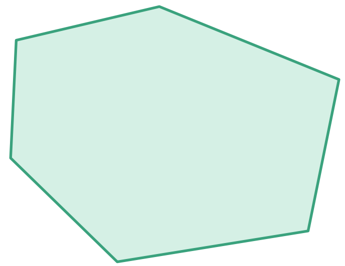
For each frustum plane $\mathbf{H}$

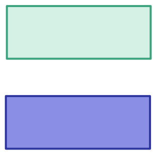
- If $\mathbf{H} \bullet \mathbf{p} > 0$ and $\mathbf{H} \bullet \mathbf{q} < 0$, clip $\mathbf{q}$
- If $\mathbf{H} \bullet \mathbf{p} < 0$ and $\mathbf{H} \bullet \mathbf{q} > 0$, clip $\mathbf{p}$
- If $\mathbf{H} \bullet \mathbf{p} > 0$ and $\mathbf{H} \bullet \mathbf{q} > 0$, pass through
- If $\mathbf{H} \bullet \mathbf{p} < 0$ and $\mathbf{H} \bullet \mathbf{q} < 0$, clipped out

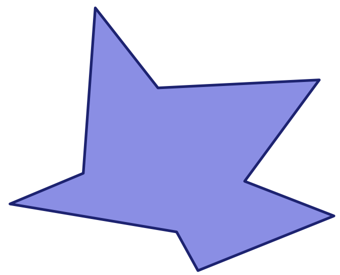
Result is a single segment.

Clipping and containment

- Clipping can be carried out against any object
  - Not just a viewing frustum
- Clipping against an arbitrary object
- Need a test for containment
  - i.e. is a point inside or outside the object
- Can develop containment test for
  - Convex objects: Common problem, e.g. convex polyhedra
  - Concave objects: Harder than convex case

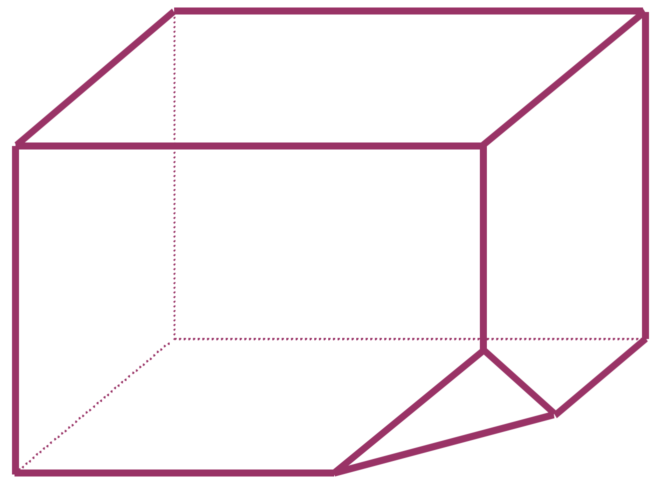






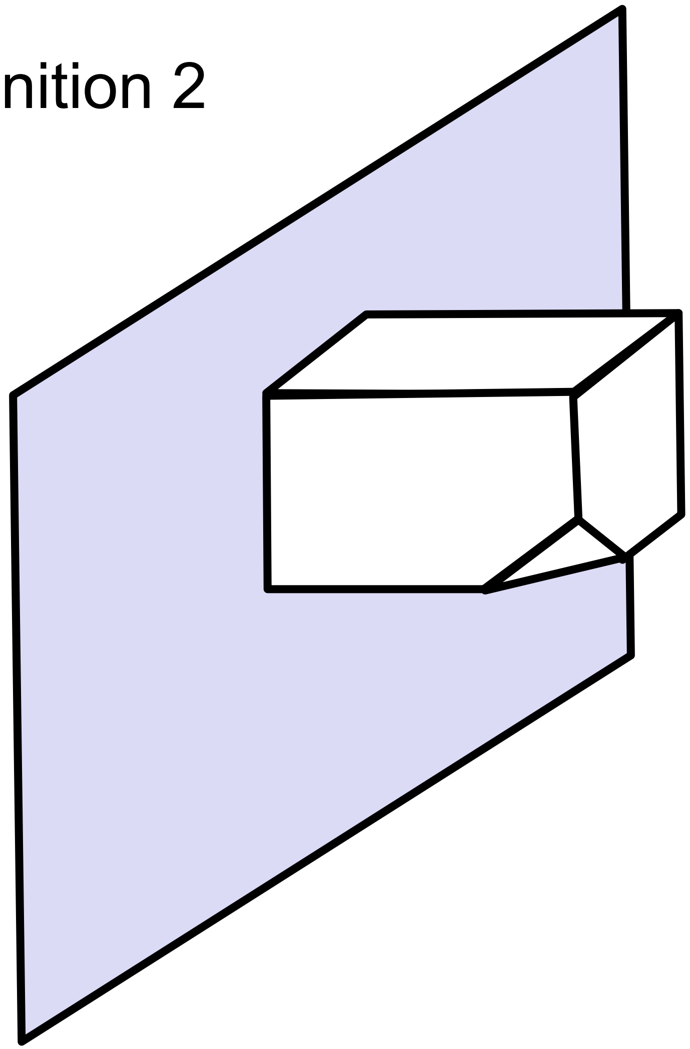
Convex objects: Two Definitions

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



Testing if an object is convex

Illustration of definition 2



Testing if an object is convex: Algorithm

```
convex = true
for each face of the object {
  find plane equation of face: F(x,y,z) = 0
  choose object point (xi,yi,zi) not on the face

  for all other points of the object {
        if (sign(F(xj,yj,zj)) != sign(F(xi,yi,zi)))
              then convex = false
  }
}
```

Works due to definition 2, all points of the object must lie entirely to one side of each face

Test containment within a convex object: Algorithm

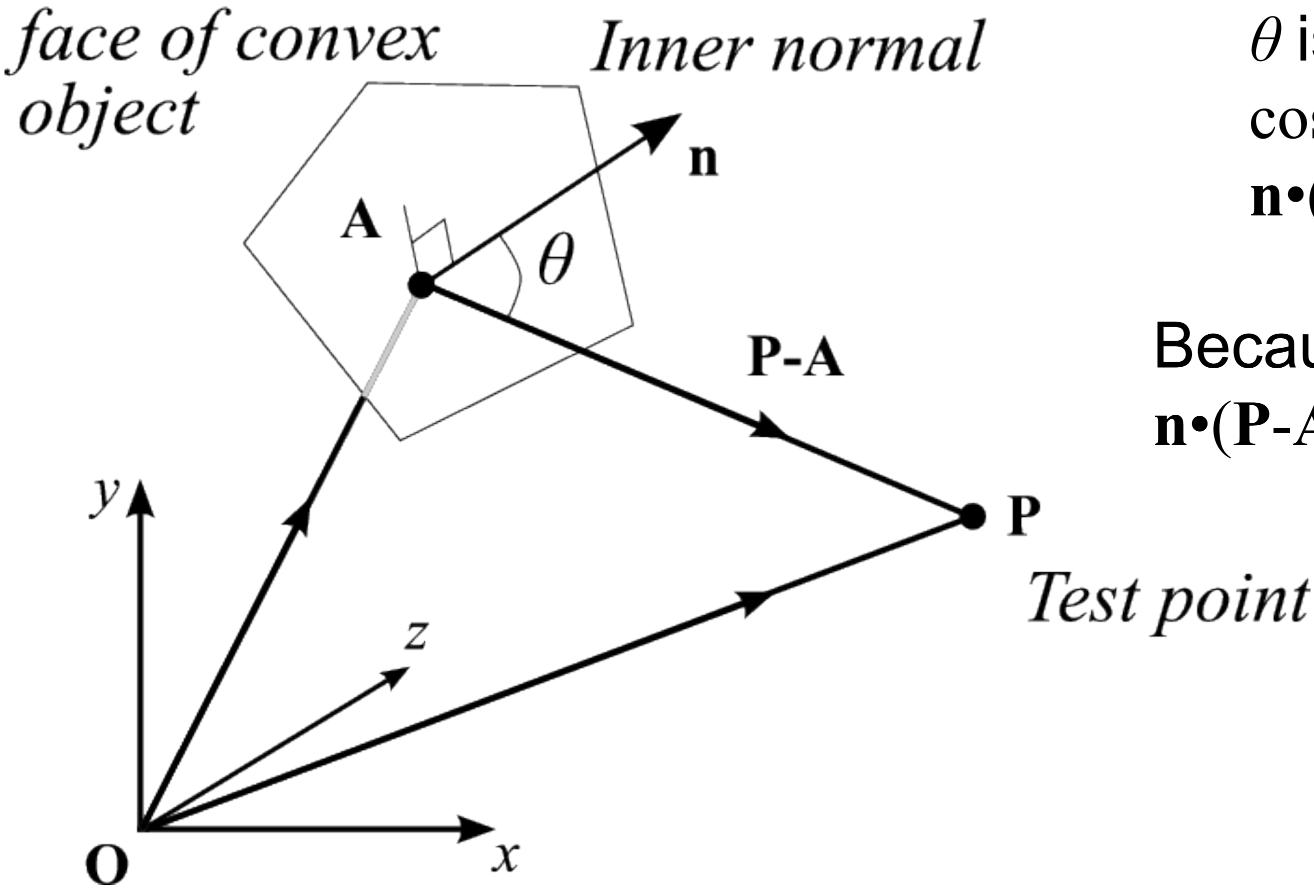
```
let the test point be (x_t,y_t,z_t)
contained = true
for each face of the convex object {
  find plane equation of face: F(x,y,z) = 0
  choose an object point (x_i,y_i,z_i) not on the face

  if (sign( F(x_t,y_t,z_t) ) != sign( F(x_i,y_i,z_i) ))
      then contained = false
}
```

Vector formulation

- The same test can be expressed in vector form.
- This avoids the need to calculate the Cartesian equation of the plane, if, in our model we store the normal vector $\mathbf{n}$ for each face of our object.

Vector test for containment



P is on the 'inside' of the face if:

θ is acute

$\cos\theta > 0$

$\mathbf{n} \bullet (\mathbf{P}-\mathbf{A}) > 0$

Because

$\mathbf{n} \bullet (\mathbf{P}-\mathbf{A}) = |\mathbf{n}|\,|\mathbf{P}-\mathbf{A}| \cos\theta$

Normal vector to a face

- The vector formulation does not require us to find the plane equation of a face, but it does require us to find a normal vector to the plane;
- Same thing really since for plane

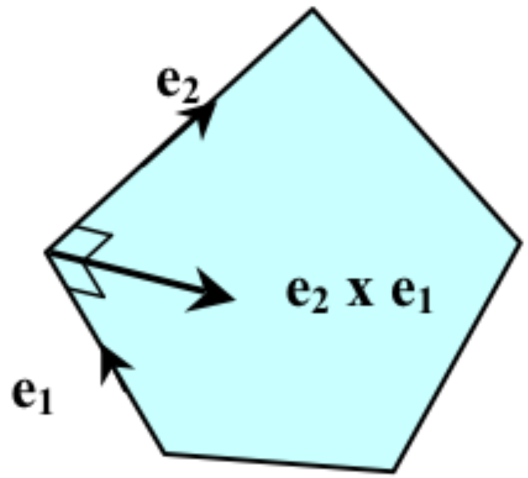
$$Ax + By + Cz + D = 0$$

- A normal vector is

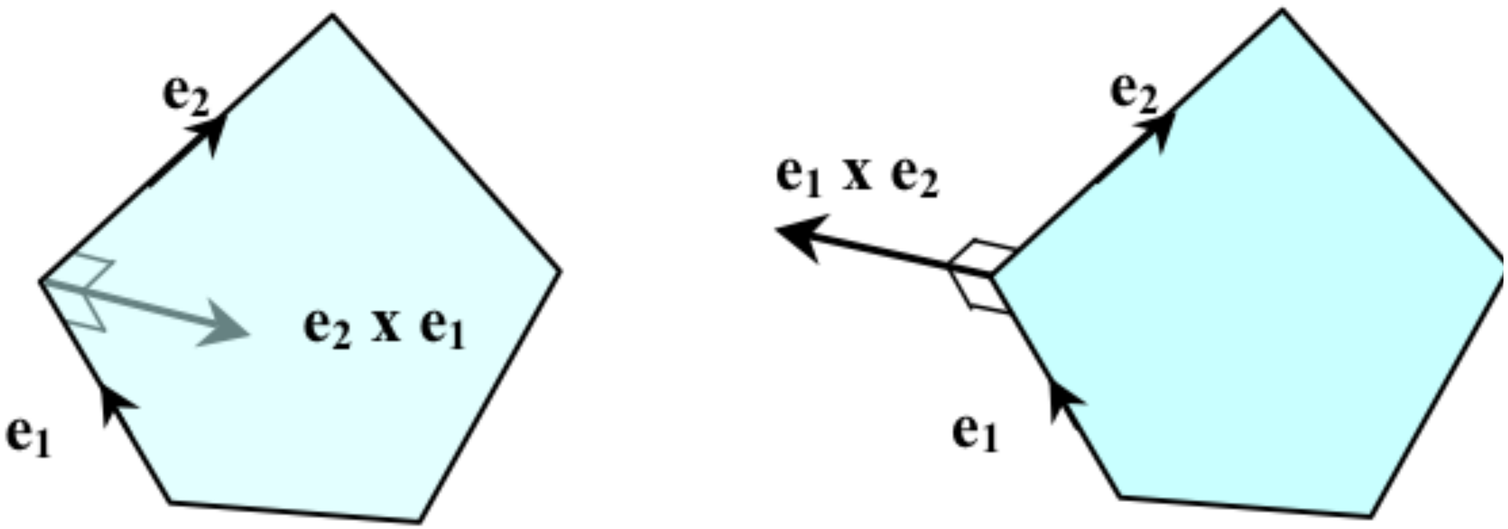
$$\mathbf{n} = \begin{pmatrix} A \\ B \\ C \end{pmatrix}$$

Finding a normal vector

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

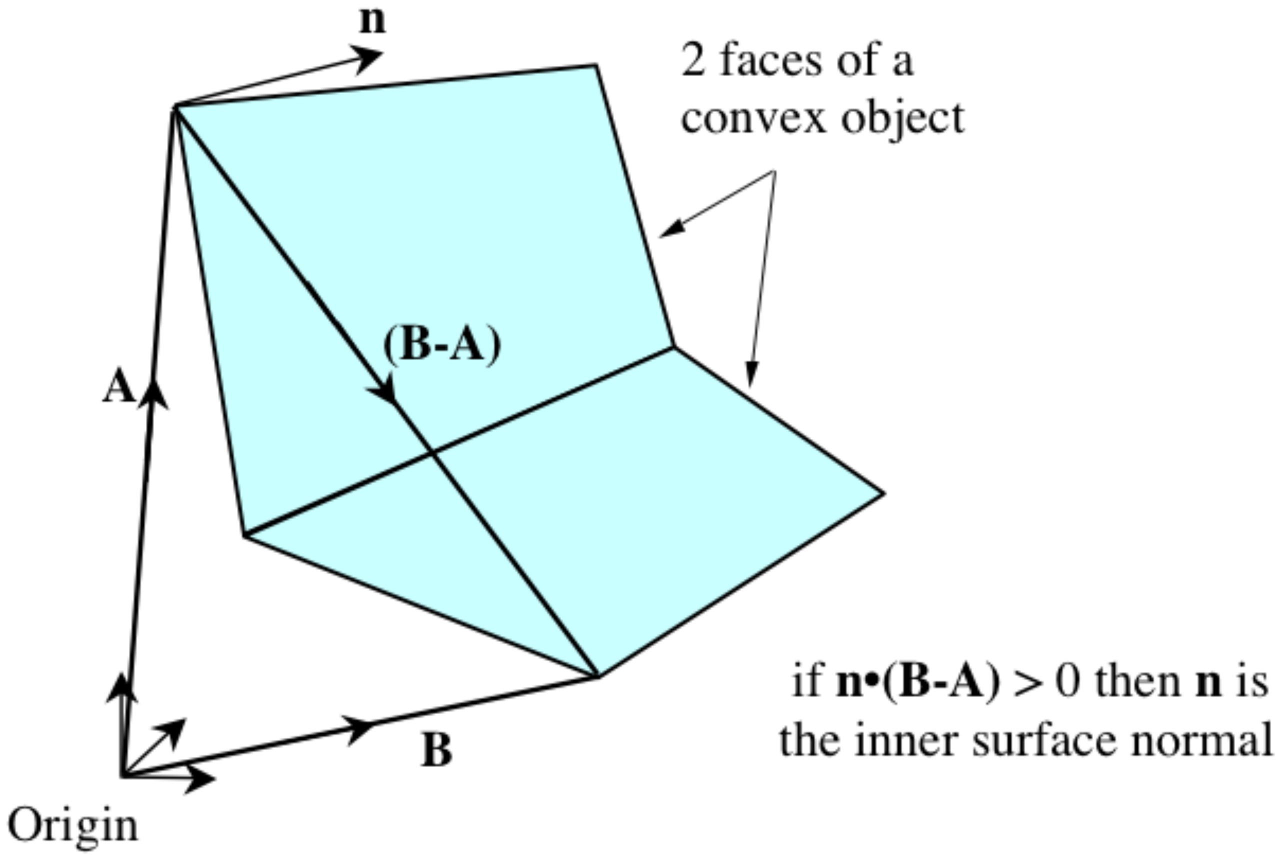


But which normal vector points inwards?



Checking normal direction (convex object)

Is **n** an inner normal?



Problem Break

- A face of a convex object lies in the plane

$$3x + 5y + 7z + 1 = 0$$

and a vertex $\mathbf{v}$ is $(-1, -1, 1)$. A normal vector is therefore

$$\mathbf{n} = (3,5,7)^T$$

- Problems:

1. If another vertex of the object is $\mathbf{w} = (1, 1, 1)$ determine whether $\mathbf{n}$ is an inner or outer surface normal.
2. Determine whether the point $\mathbf{p} = (1, 0, -1)$ is on the inside or the outside of the face.

Solution to Q1

'P−A'

$= \mathbf{w} - \mathbf{v}$

$= (1, 1, 1) - (-1,-1,1)$

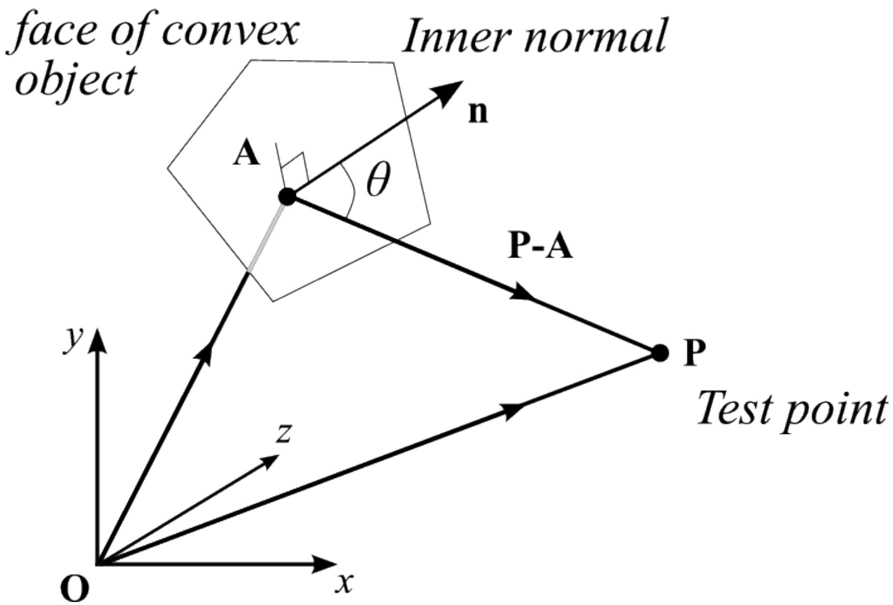
$= (2, 2, 0)$

$\mathbf{n}\bullet(\mathbf{w}-\mathbf{v}) = 6 + 10 = 16$

$\mathbf{n}\bullet(\mathbf{w}-\mathbf{v})$ is positive,

θ is acute

So $\mathbf{n}$ is an inner normal



Solution to Q2

Method 1:

- The plane has equation

$$F(x, y, z) = 3x + 5y + 7z + 1 = 0$$

  - For the internal point $\mathbf{w} = (1, 1, 1)$:

    $F(1, 1, 1) = 16$
  - For the test point $\mathbf{p} = (1, 0, -1)$:

    $F(1, 0, -1) = -3$
- The signs are different, so the test point is on the outside

Solution to Q2

Method 2:

The inner surface normal is $\mathbf{n} = (3, 5, 7)$

for the test point $\mathbf{p} = (1, 0, -1)$

and face vertex $\mathbf{v} = (-1, -1, 1)$

$$\mathbf{p} - \mathbf{v} = (2, 1, -2)$$

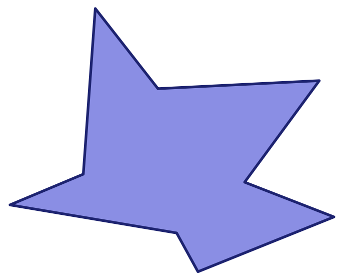
$$\mathbf{n} \bullet (\mathbf{p} - \mathbf{v}) = -3$$

Thus the angle to the normal is > 90

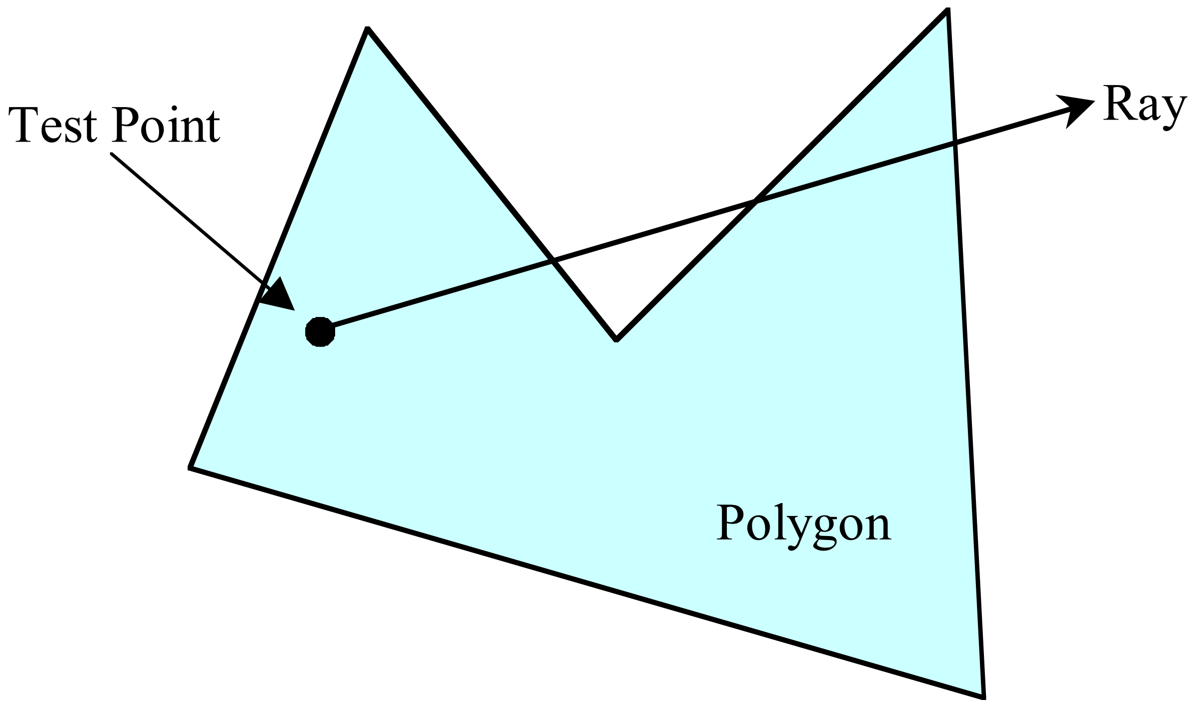
So the point $\mathbf{p}$ is on the outside

Concave Objects

- Containment and clipping can also be carried out with concave objects.
- Most algorithms are based on the ray containment test.



The Ray test in two dimensions



Find all intersections between the ray and the polygon edges.
If the number of intersections is odd the point is contained

Calculating intersections with rays

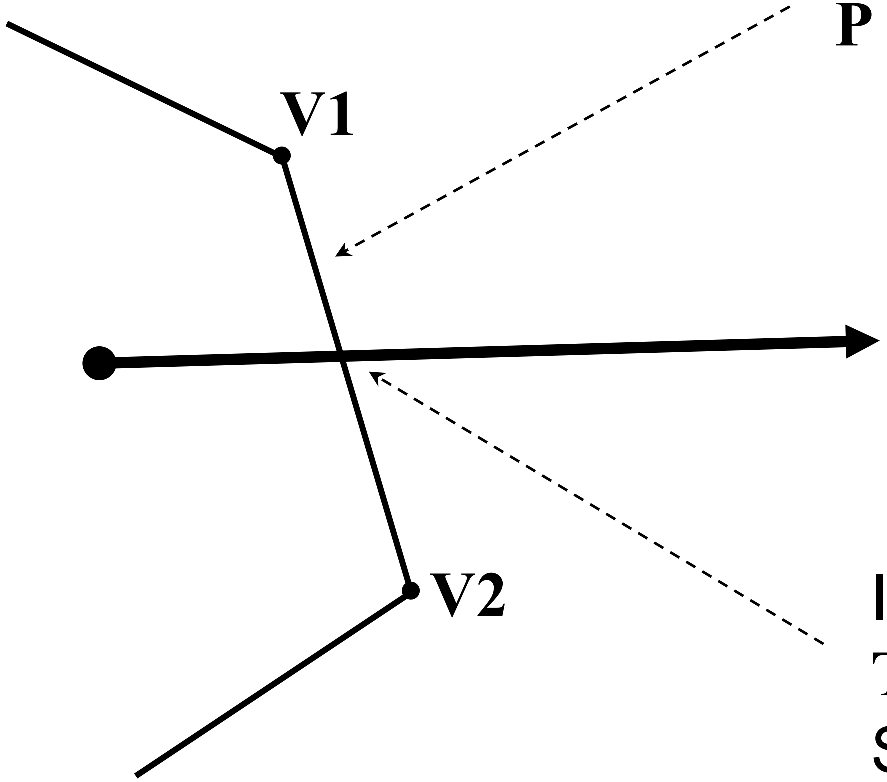
- Rays have equivalent equations to lines, but go in only one direction. For test point $\mathbf{T}$ a ray is defined as

$$\mathbf{R} = \mathbf{T} + \mu\, \mathbf{d} \quad , \quad \mu > 0$$

- We choose a simple to compute direction e.g.

$$\mathbf{d} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \quad \text{or} \quad \mathbf{d} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$$

Valid Intersections



Line segment

$\mathbf{P} = \mathbf{V2} + \nu\,(\mathbf{V1} - \mathbf{V2})$

Ray

$\mathbf{P} = \mathbf{T} + \mu\,\mathbf{d} \quad , \mathbf{d} = (1,0)^T$

Intersection

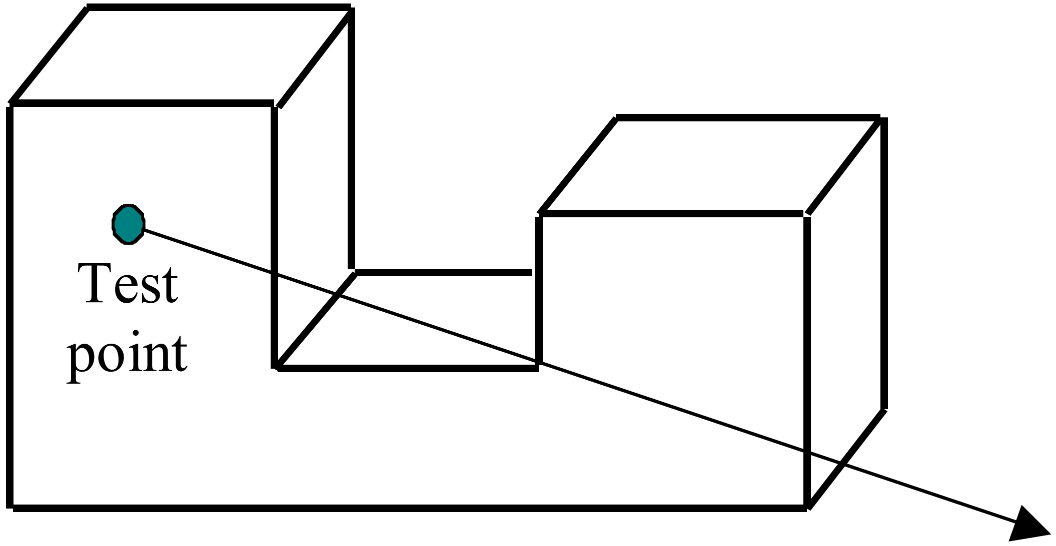
$\mathbf{T} + \mu\,\mathbf{d} = \mathbf{V2} + \nu\,(\mathbf{V1} - \mathbf{V2})$

Solve for ν and μ

Valid intersection if

$\mu > 0 \ , \ 0 \leq \nu < 1$

Extending the ray test to 3D

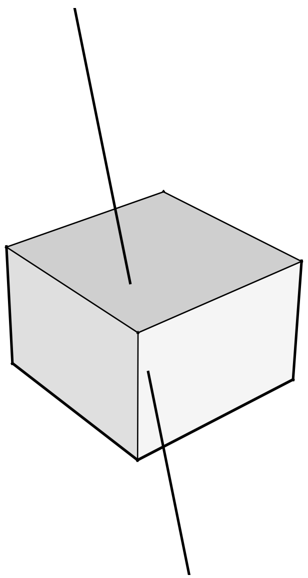


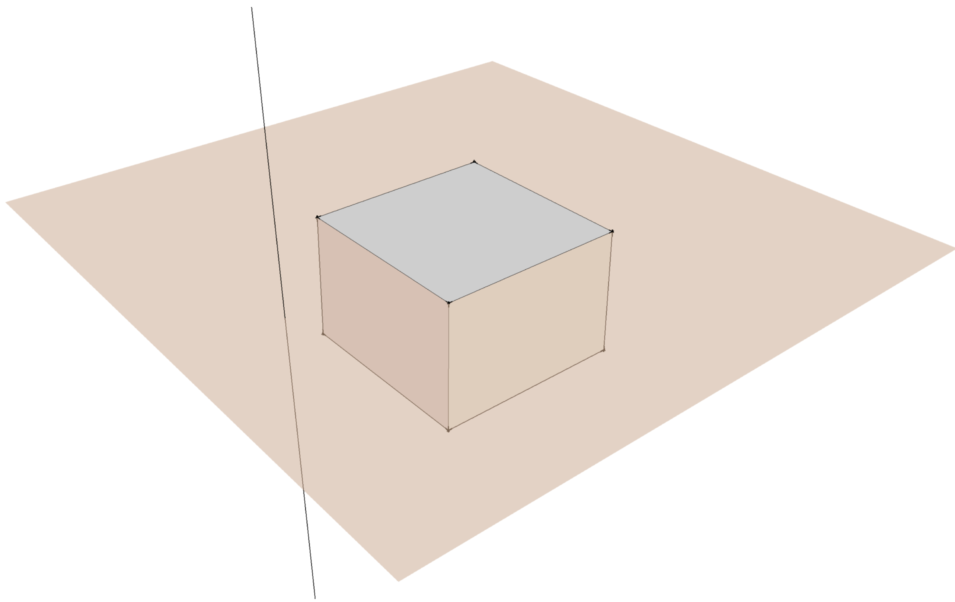
A ray is projected in any direction.

If the number of intersections with the object is odd, then the test point is inside

3D Ray test

- There are two stages:
  1. Compute the intersection of the ray with the plane of each face.
  2. If the intersection is in the positive part of the ray ($\mu > 0$) check whether the intersection point is contained in the face (i.e. not just in the planar *extension* of the face).





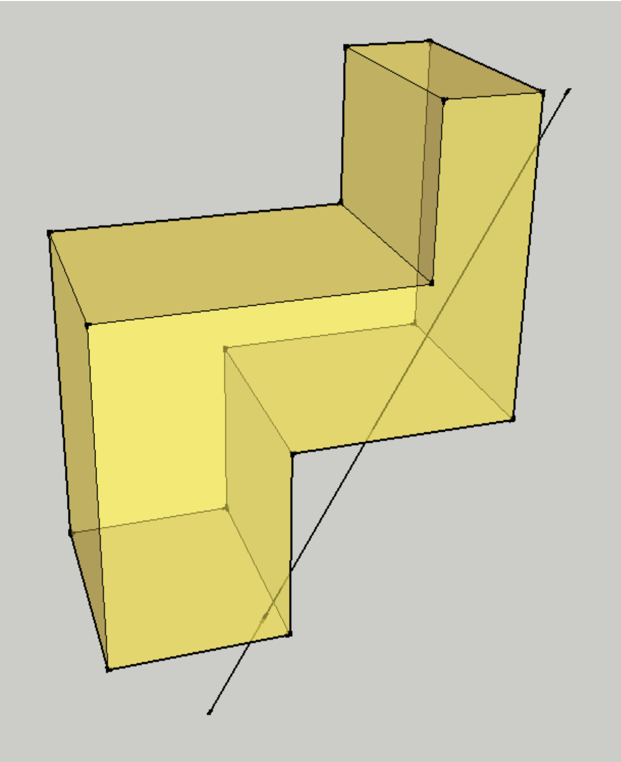
The plane of a face

- Unfortunately the plane of a face does not in general line up with the Cartesian axes, so the second part is not a two dimensional problem.

- However, containment is invariant under orthographic projection, so it can be simply reduced to two dimensions.

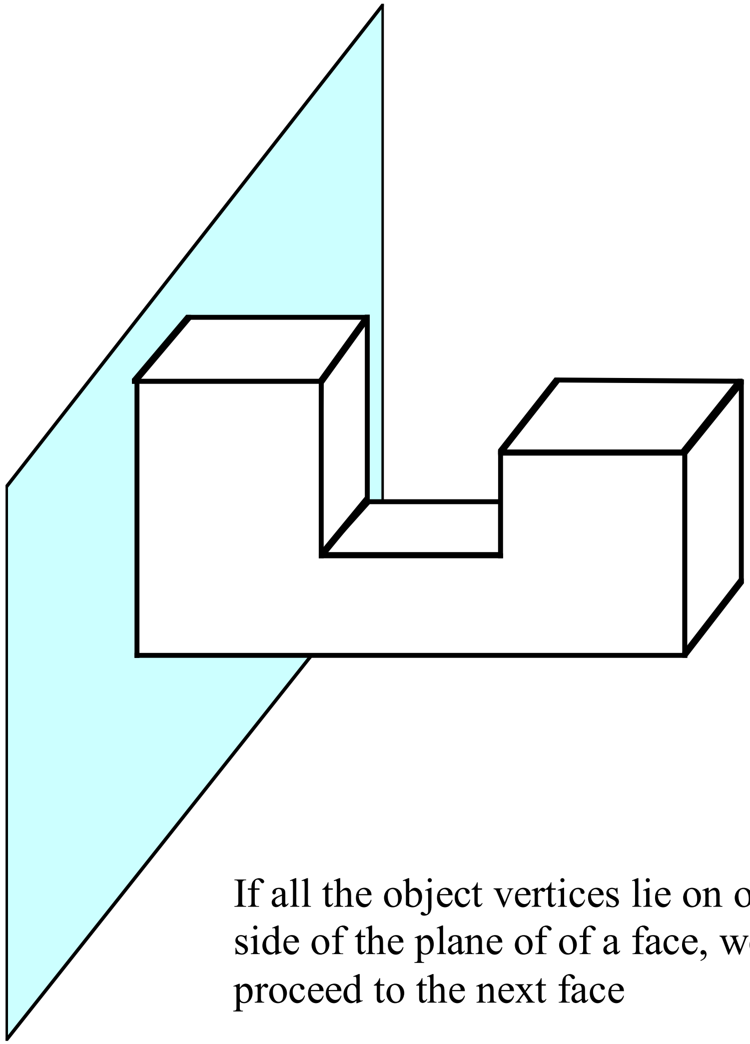
Clipping to concave volumes

- Find every intersection of the line to be clipped with the volume
- This divides the line into one or more segments.
- Test a point on the first segment for containment
- Adjacent segments will be alternately inside and out.



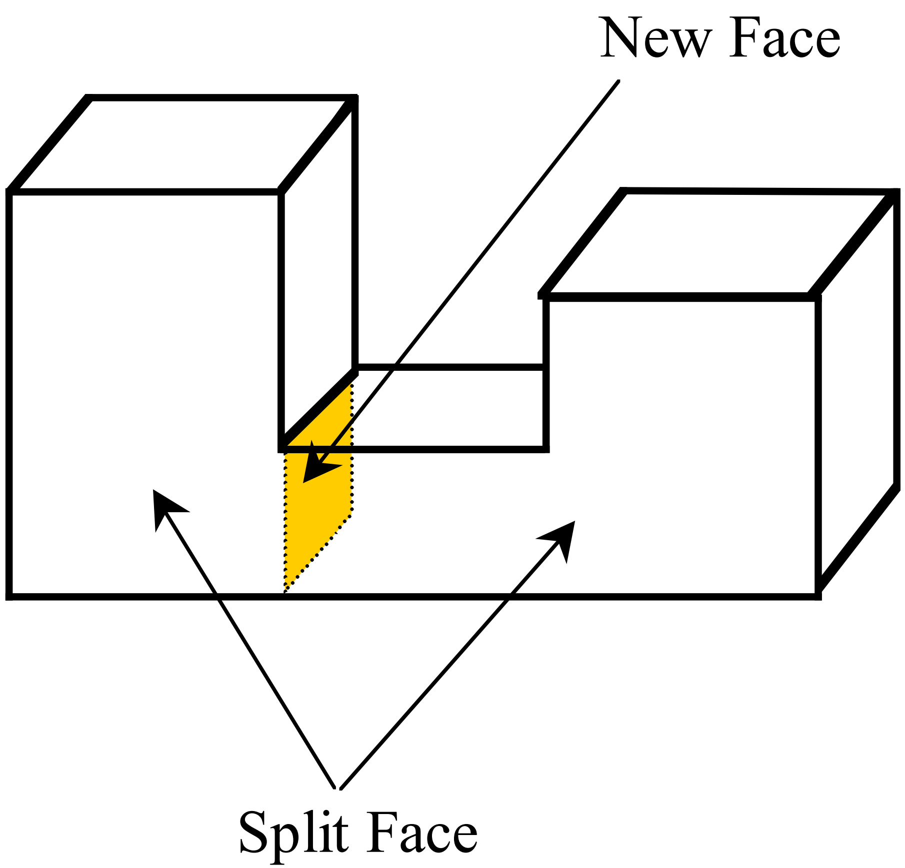
Splitting a volume into convex parts

- Split concave volume into convex parts
- Can apply tests for convex objects to each of the parts
- Consider each face

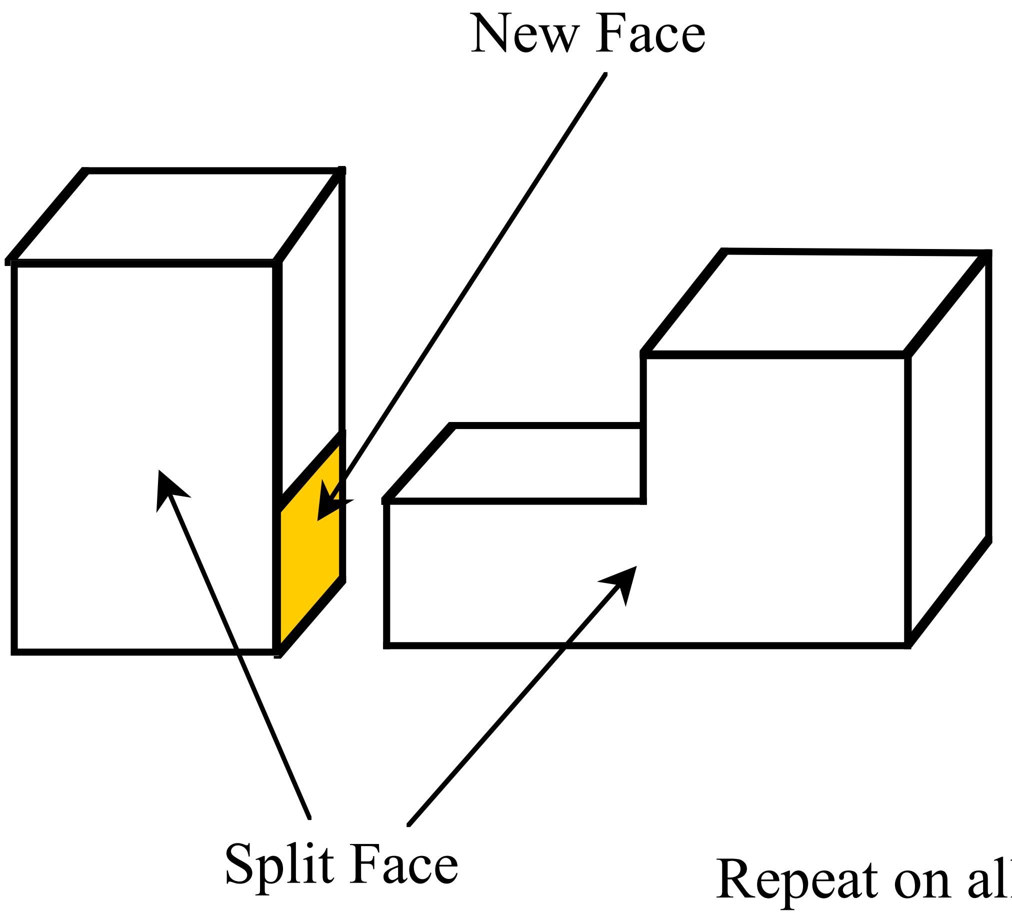


If all the object vertices lie on one side of the plane of of a face, we proceed to the next face

If the plane of a face cuts the object:



Split the Object



Repeat on all concave sub parts