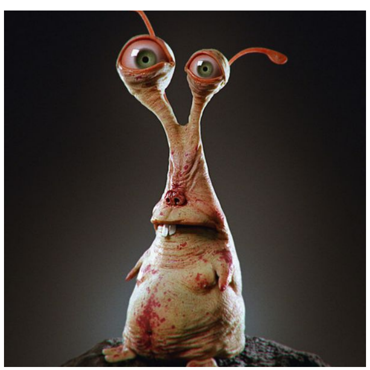
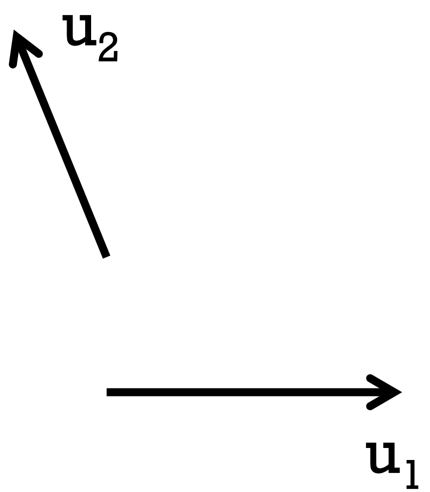
Ray tracing polygonal meshes

3D Computer Graphics (Lab 5)



Remember …

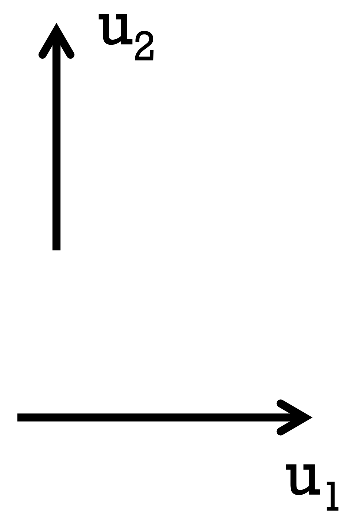
$$\boxed{\cos(\theta) = \frac{u_1.u_2}{|u_1||u_2|}}$$



Angle θ between u_1 and u_2 > 90°

$$\cos\theta < 0$$

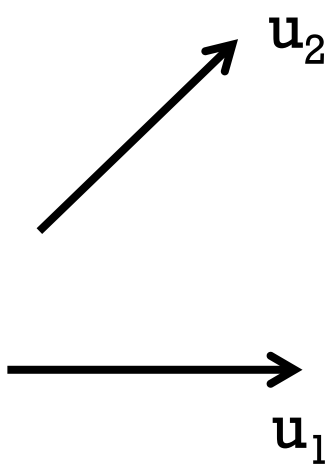
$$\boxed{u_1.u_2 < 0}$$



Angle θ between u_1 and u_2 = 90°

$$\cos\theta = 0$$

$$\boxed{u_1.u_2 = 0}$$

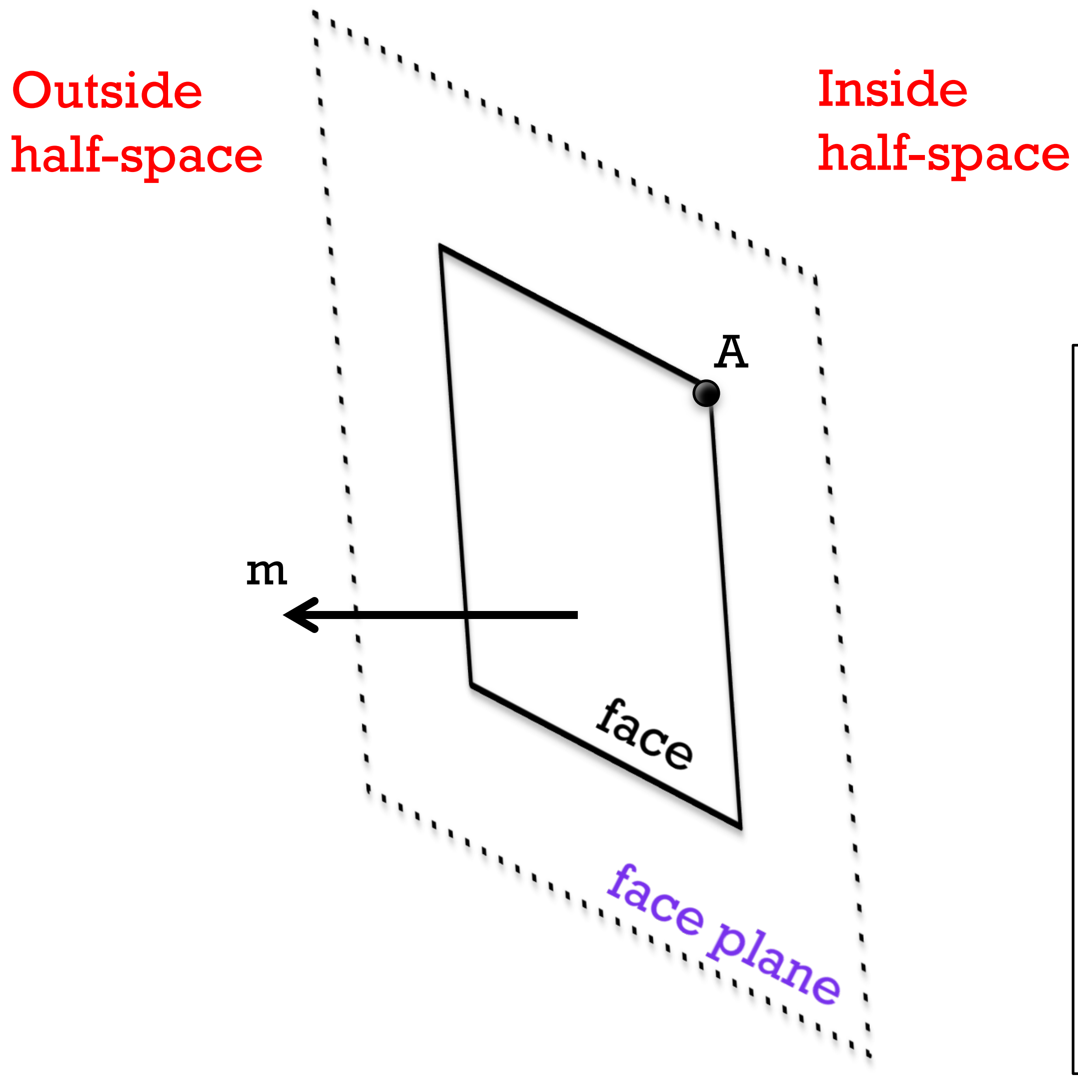


Angle θ between u_1 and u_2 < 90°

$$\cos\theta > 0$$

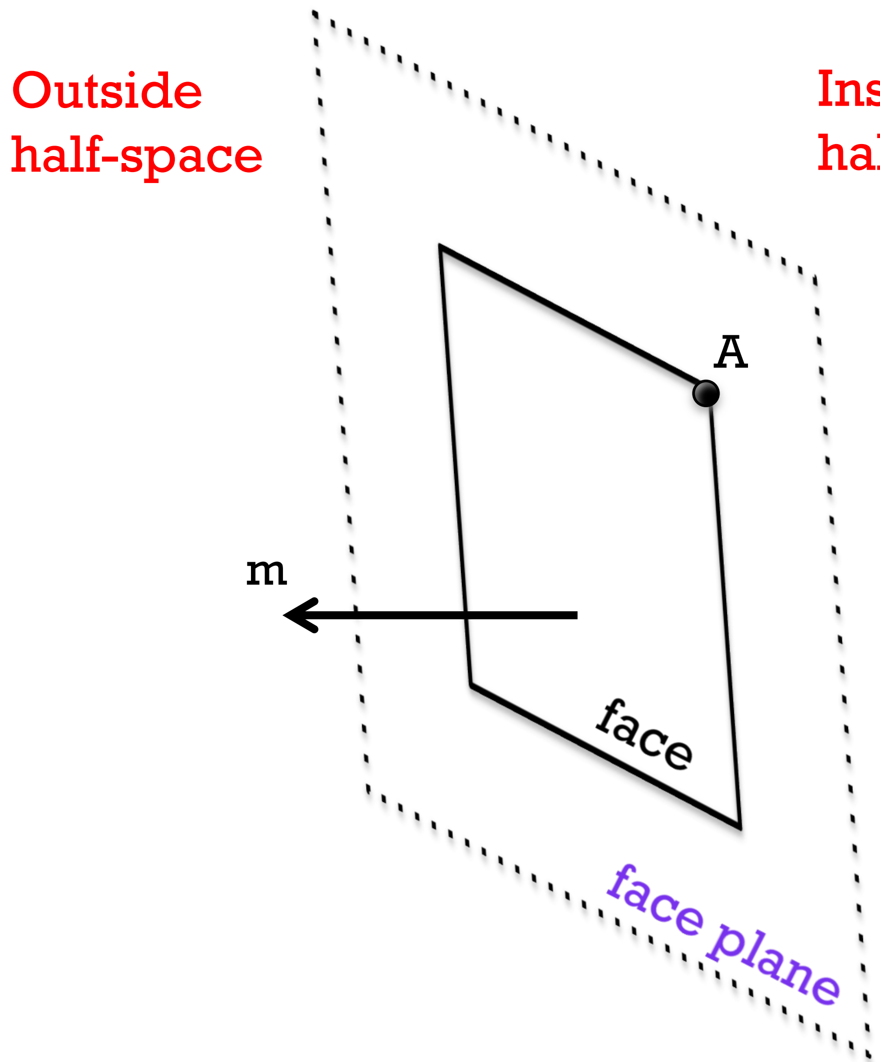
$$\boxed{u_1.u_2 > 0}$$

Face plane and half-spaces



- A = a vertex of the face
  m = normal vector of the face
- A face plane is the plane in which the face lies.
- The face plane divides the 3D space in two: the outside and inside half-space.
- m points in the direction of the outside half-space.

Face plane and half-spaces



- Note that we associated a normal vector with each vertex of a face in Lab 2.
- The normal vectors associated with these vertices may be different so how do we set m?
- The purpose of m is to be able to differentiate between the inside and outside half-space, so any of the normal vectors of the face will do.

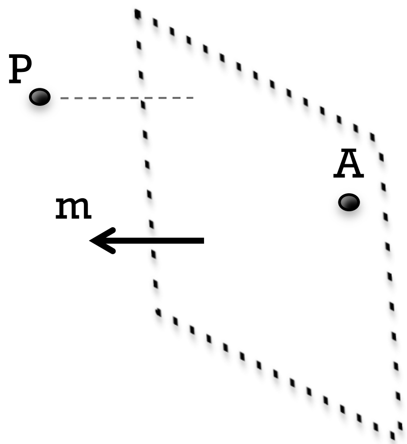
Position of a point with respect to a plane

Position of a point w.r.t. a plane

How can we determine the difference?

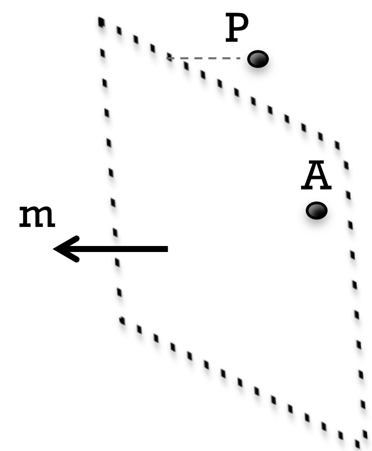
$\boxed{m.(A-P)<0}$

P lies in outside half-space



$\boxed{m.(A-P)>0}$

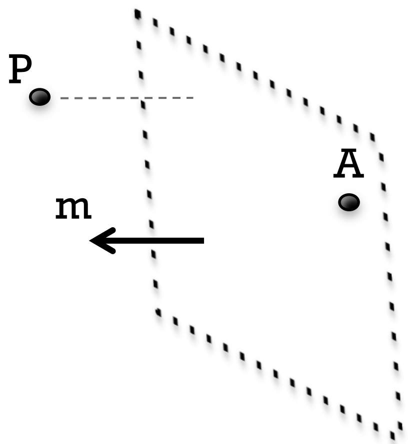
P lies in inside half-space



Position of a point w.r.t. a plane

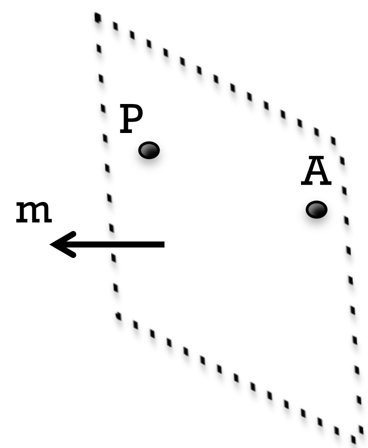
$\boxed{m.(A-P)<0}$

P lies in outside half-space



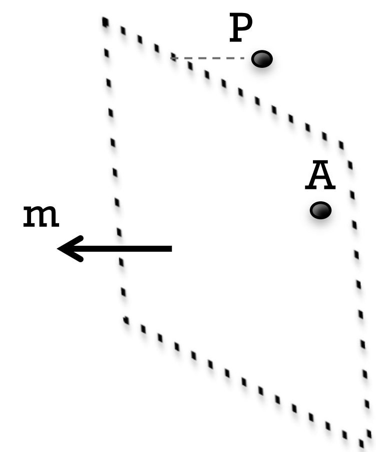
$\boxed{m.(A-P)=0}$

P lies in face plane



$\boxed{m.(A-P)>0}$

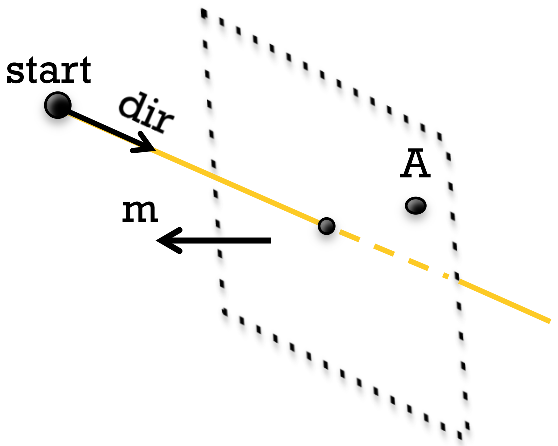
P lies in inside half-space



Position of a ray with respect to a plane

$\boxed{m.dir < 0}$

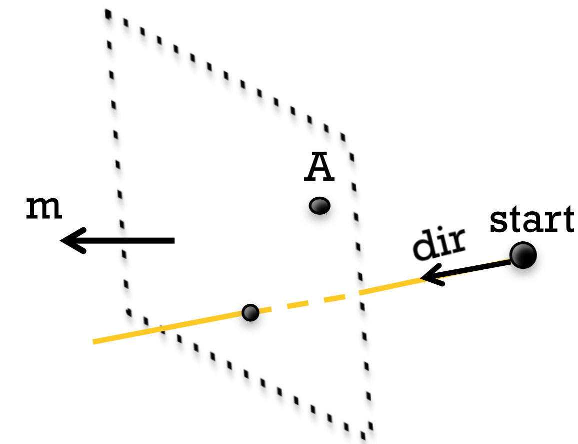
ray enters inside half-space



How can we determine the difference?

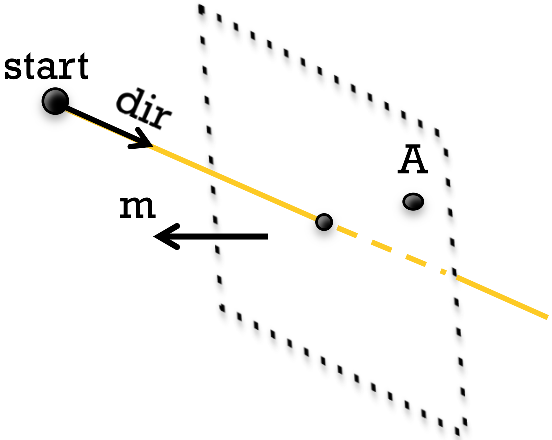
$\boxed{m.dir > 0}$

ray exits inside half-space



$\boxed{m.dir < 0}$

ray enters inside half-space



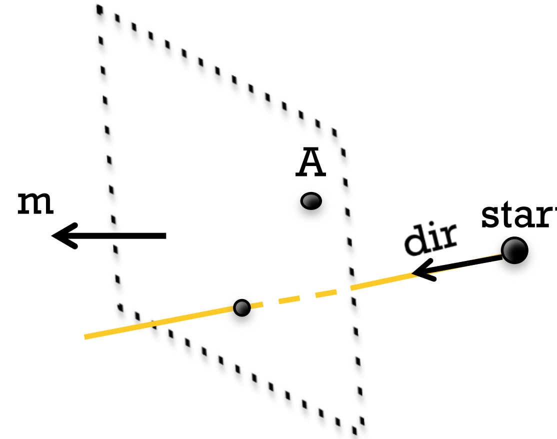
$\boxed{m.dir = 0}$

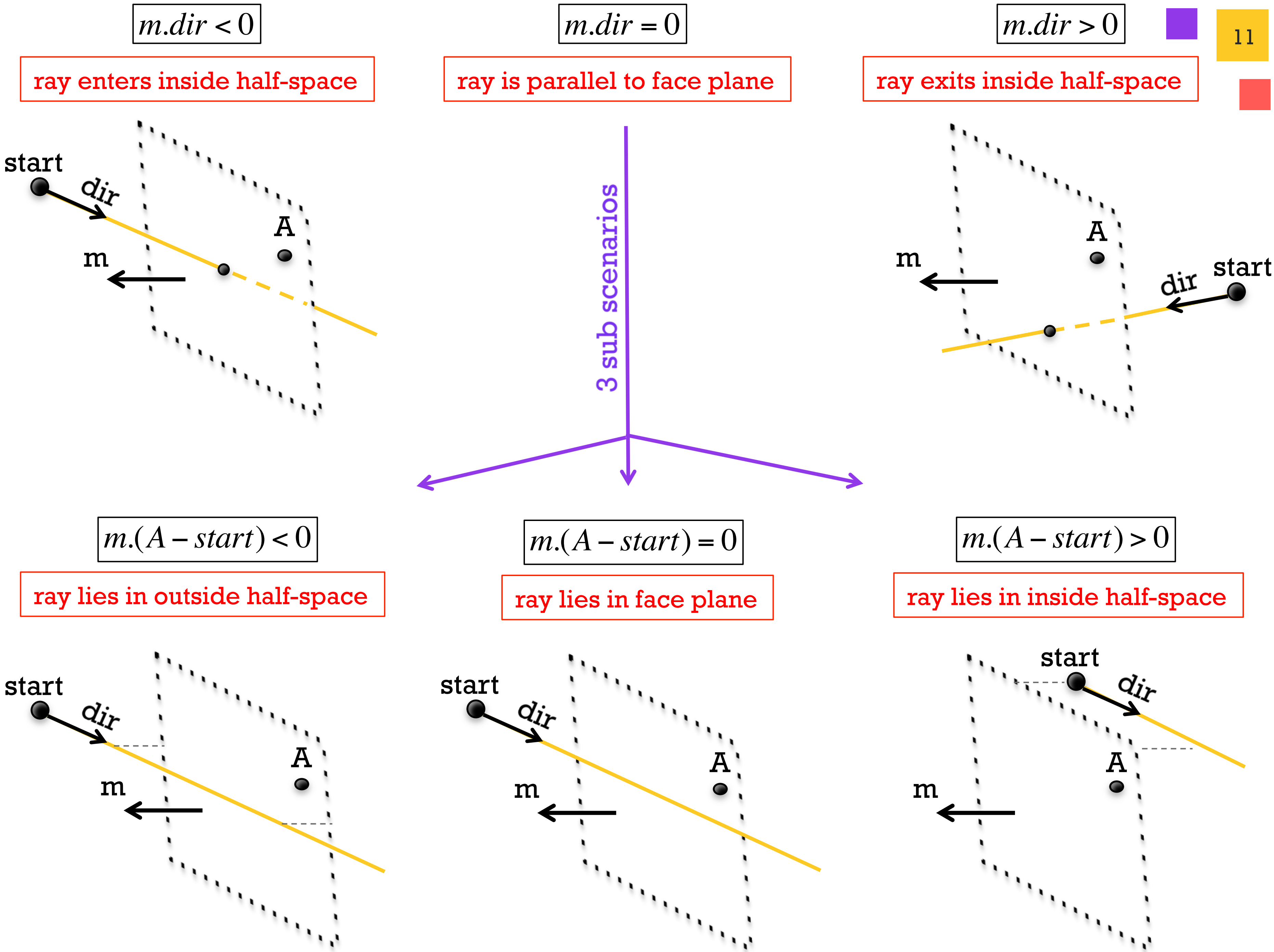
ray is parallel to face plane

Depending on the position of the start point, there are three sub scenarios.

$\boxed{m.dir > 0}$

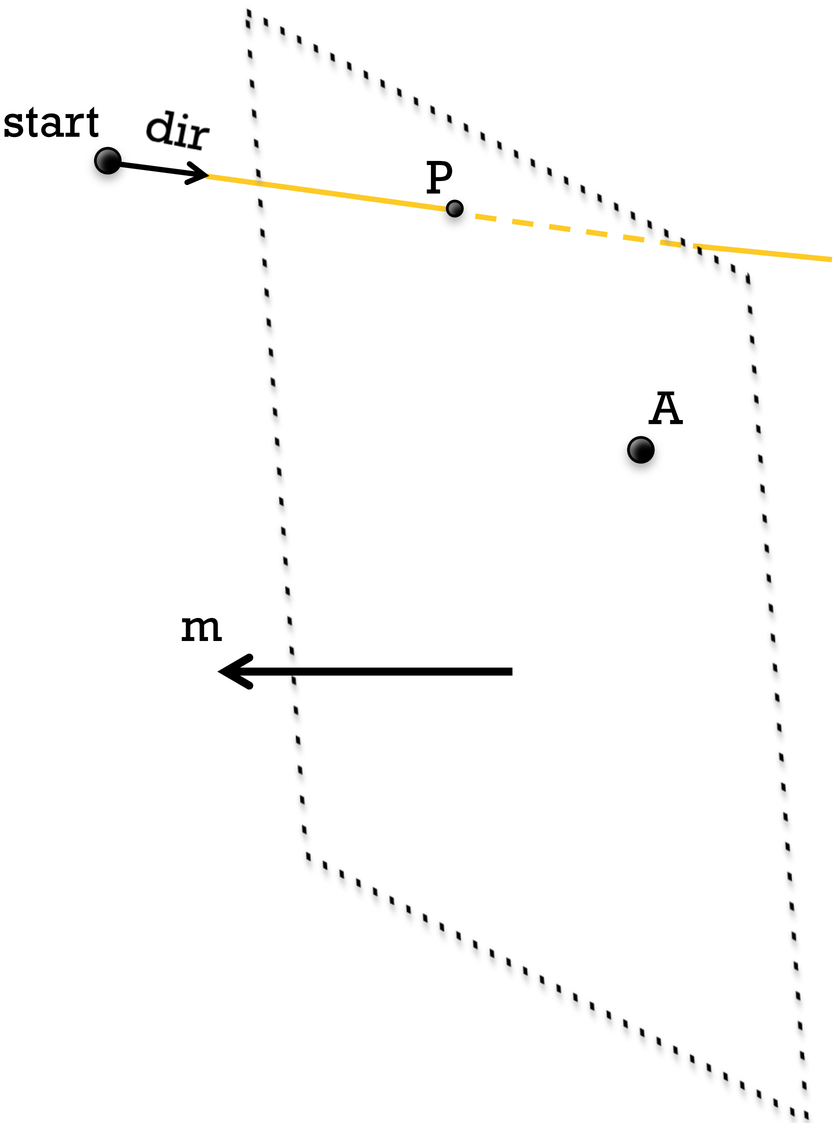
ray exits inside half-space




$m.dir < 0$
ray enters inside half-space
$m.dir = 0$
ray is parallel to face plane
$m.dir > 0$
ray exits inside half-space
start
dir
m
A
3 sub scenarios
$m.(A - start) < 0$
ray lies in outside half-space
$m.(A - start) = 0$
ray lies in face plane
$m.(A - start) > 0$
ray lies in inside half-space

Intersection ray - plane

Intersection ray - plane



We are searching for the point P which satisfies the folowing system

$$\begin{cases} P = start + t.dir & \boxed{\text{P is on ray}} \\ m.(A - P) = 0 & \boxed{\text{P is in plane}} \end{cases}$$

$$m.(A - (start + t.dir)) = 0$$

$$m.(A - start - t.dir) = 0$$

$$m.(A - start) - t(m.dir) = 0$$

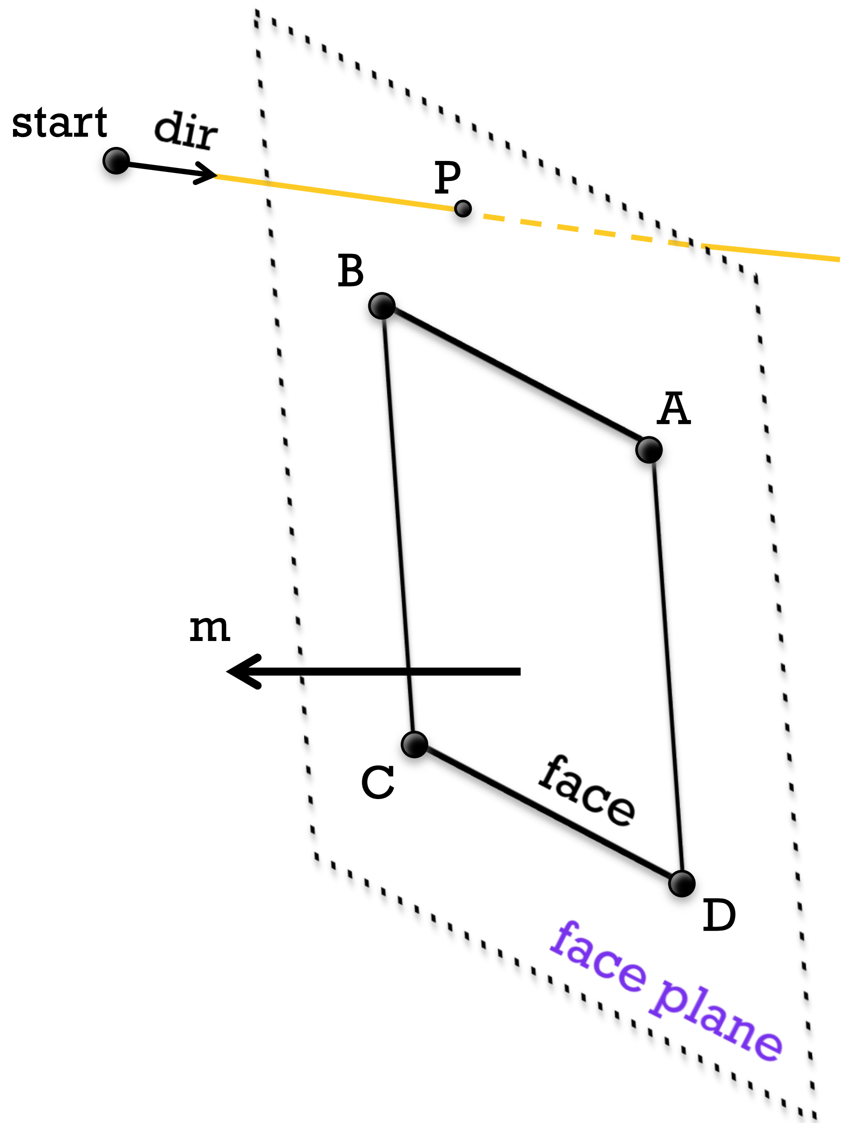
$$m.(A - start) = t(m.dir)$$

$$\boxed{t = t_{hit} = \frac{m.(A - start)}{m.dir}}$$

What if $m.dir = 0$?

Intersection
ray – polygonal mesh

Intersection ray – polygonal mesh



Pseudo code

```
for each face
    if m.dir is zero
        ignore face
    compute t_hit
    if t_hit > 0
        add hitInfo to list
```

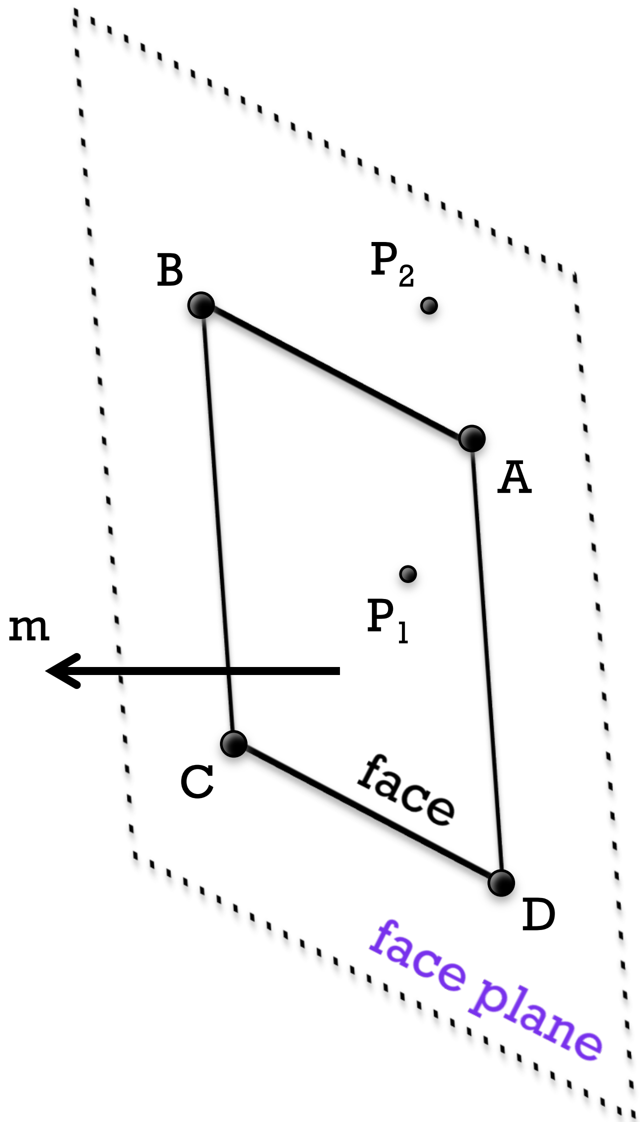
Correct?

No, the hitPoint P does not necessarily lie in the face!

$$t = t_{hit} = \frac{m.(A - start)}{m.dir}$$

Point-in-polygon test

Point-in-polygon test

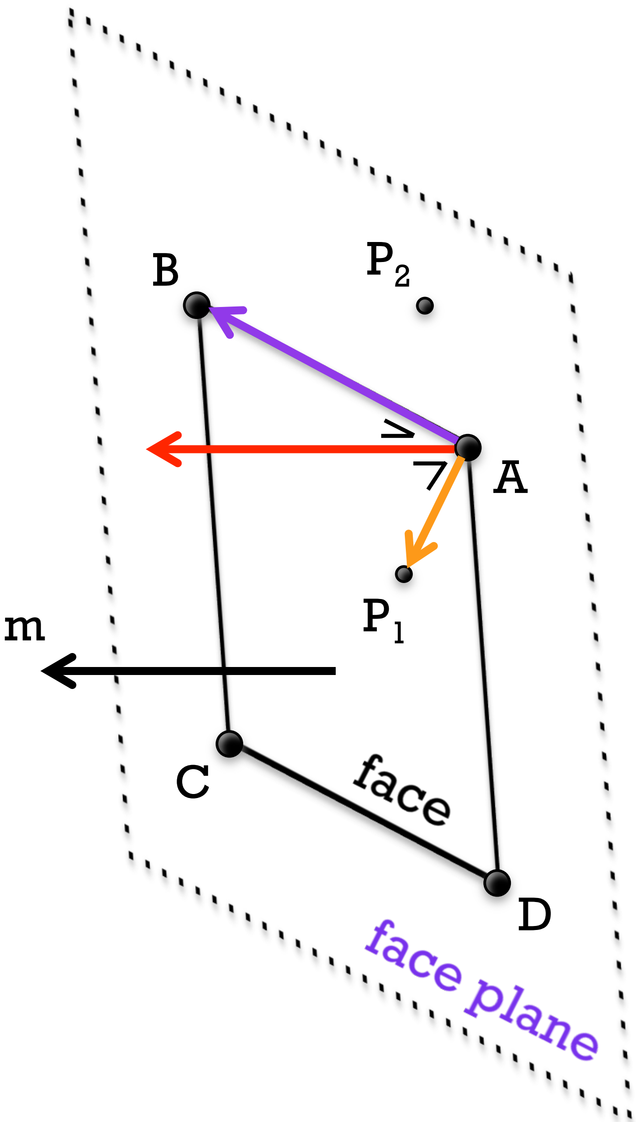


Idea …

- Walk along the edges on the outside of the face so that you visit the vertices in the order in which they were specified in the face.
- If the point lies to the left of each edge, the point is inside the polygon.
- Else the point is outside the polygon.

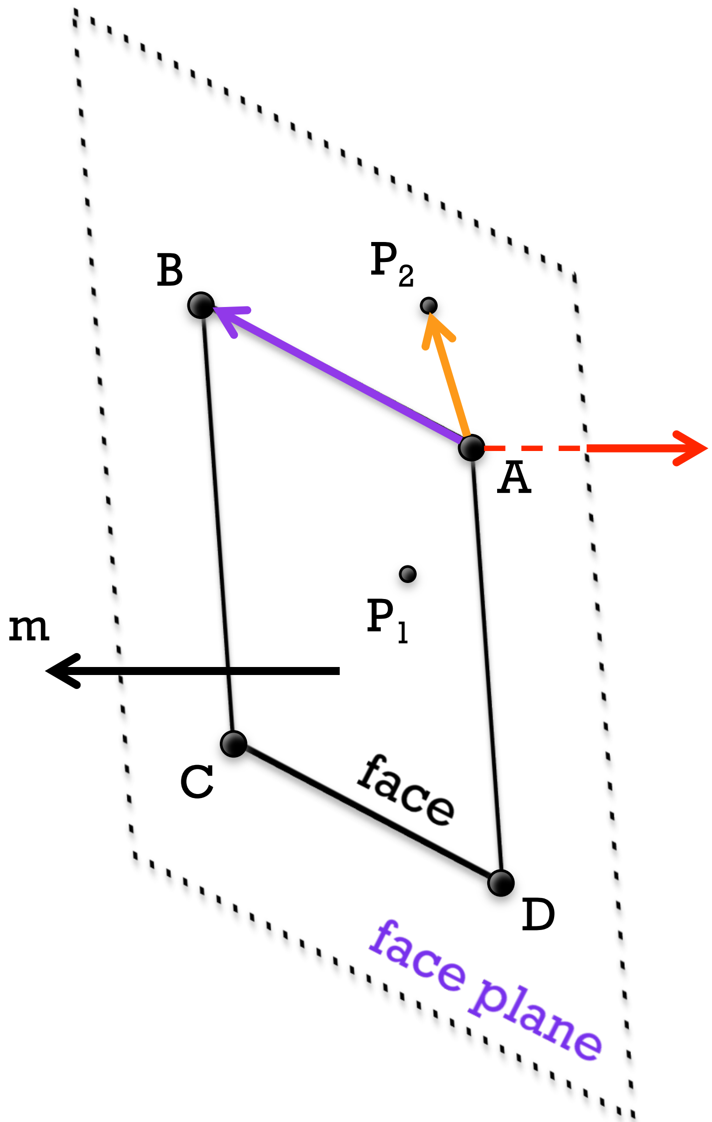
Note that this idea only works because of the convention we introduced in Lab 2 to order the vertices in a face counterclockwise as seen from outside the object.

Point-in-polygon test



- $\boxed{v_1}$ = vector from A to B
- $\boxed{v_2}$ = vector from A to P_1
- $\boxed{v_3}$ = v_1 x v_2
- v_3 points to outside half-space

Point-in-polygon test



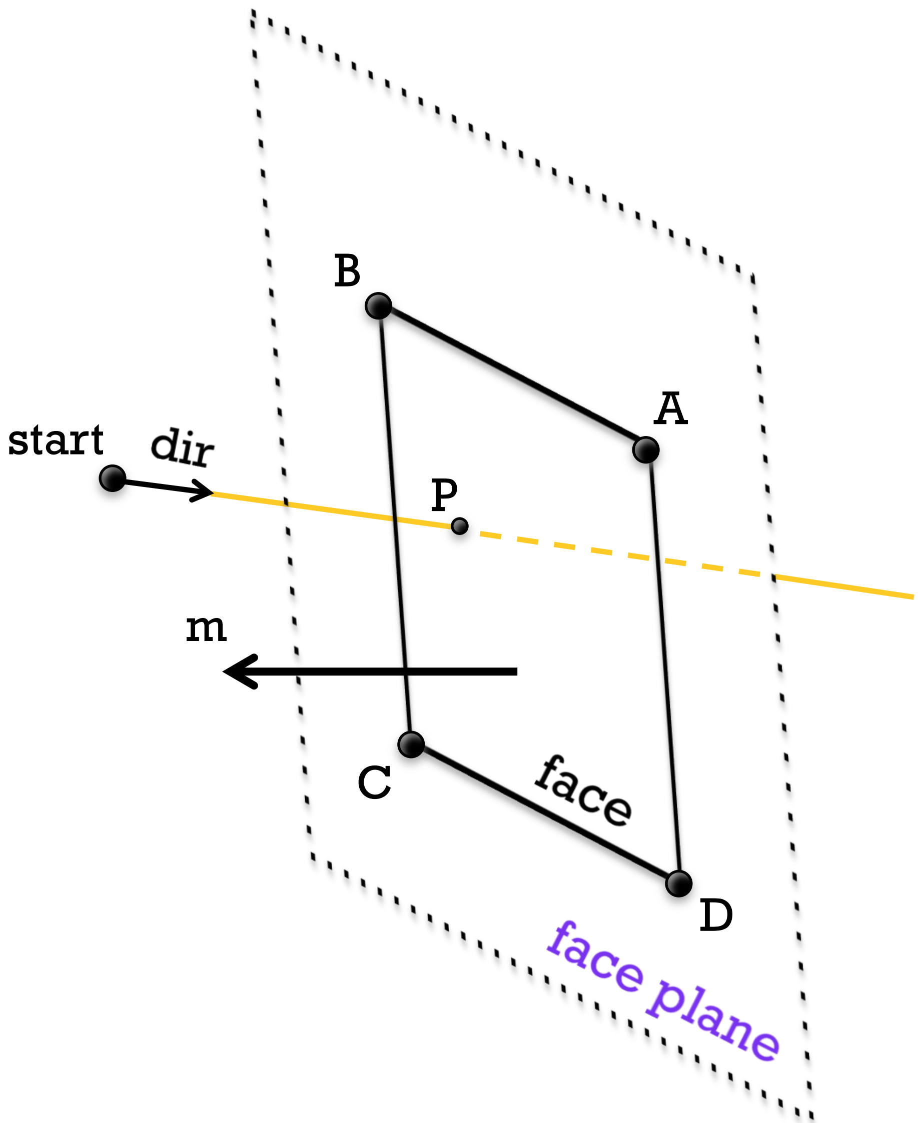
- v_1 = vector from A to B
- v_2 = vector from A to P_1
- $v_3 = v_1 \times v_2$
- v_3 points to outside half-space

- $\boxed{v_1}$ = vector from A to B
- $\boxed{v_2}$ = vector from A to P_2
- $\boxed{v_3} = v_1 \times v_2$
- v_3 points to inside half-space

A point P lies to the left of an edge if v_3 points to the outside half-space. This is the case if

$$\boxed{v_3.m > 0}$$

Intersection ray – polygonal mesh



Pseudo code

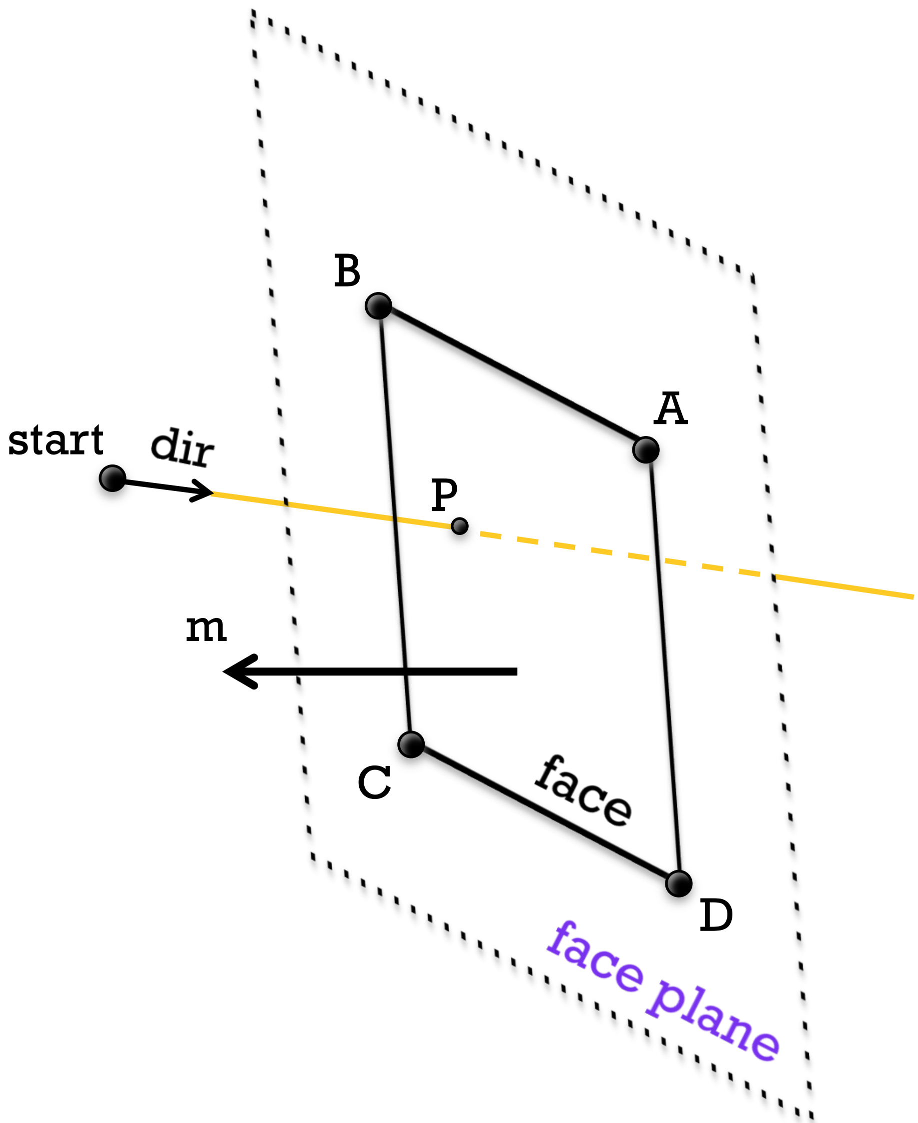
```
for each face
    if m.dir is zero, ignore face
    compute t_hit
    if t_hit < 0, ignore face
    for each edge
        if hitPoint not to the left of edge
            ignore face
    if hitPoint in face
        add hitInfo to list
return list
```

Correct?

No, the hit with the bestHit-Time should be at index 0!

How?

Intersection ray – polygonal mesh



Pseudo code

In practise:
|m.dir| < 0.0000001

for each face

    if m.dir is zero, ignore face

    compute t_{hit}

    if t_{hit} < 0, ignore face

    for each edge

        if hitPoint not to the left of edge

            ignore face

    if hitPoint in face

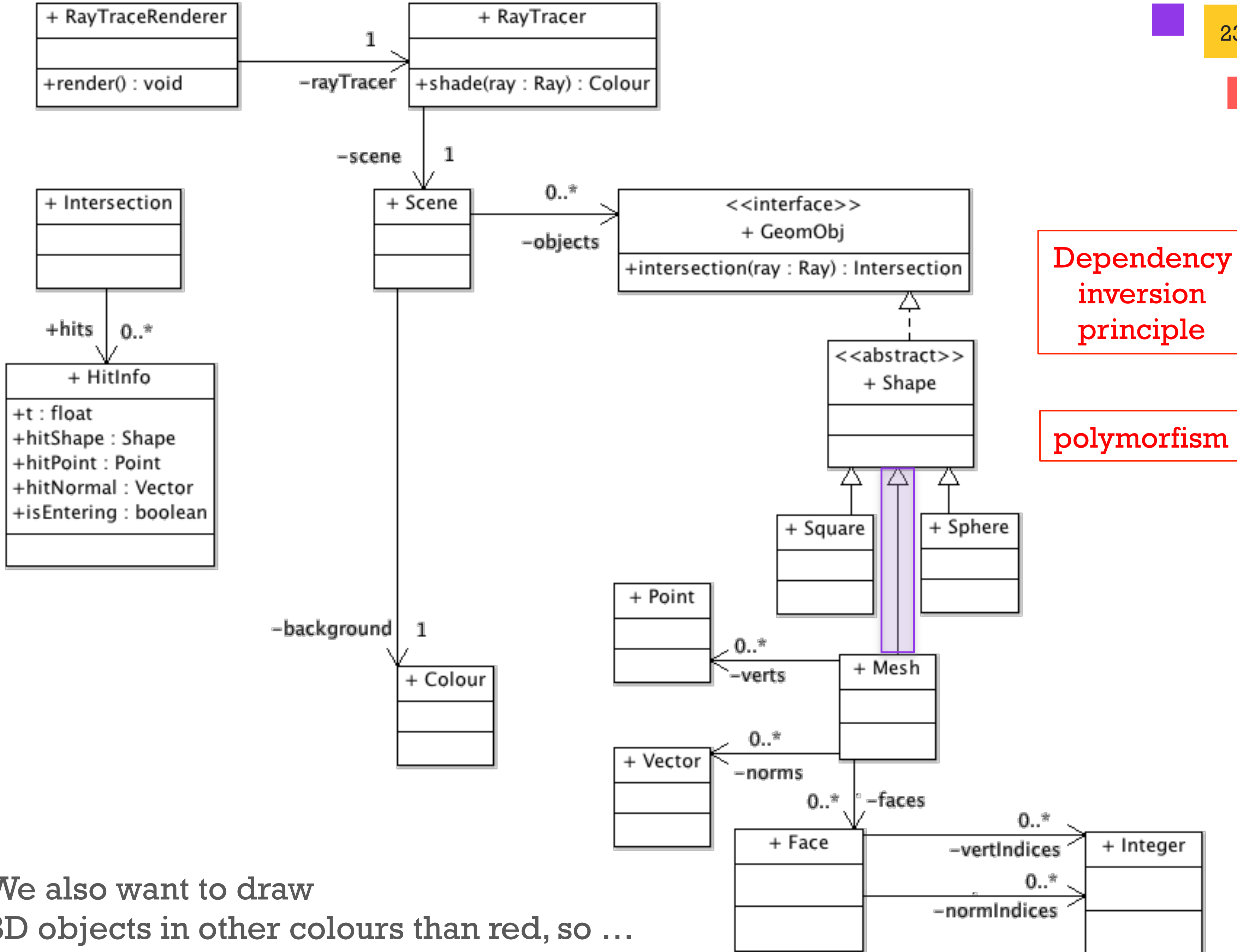
        add hitInfo to list

    set hit with bestHitTime at index 0

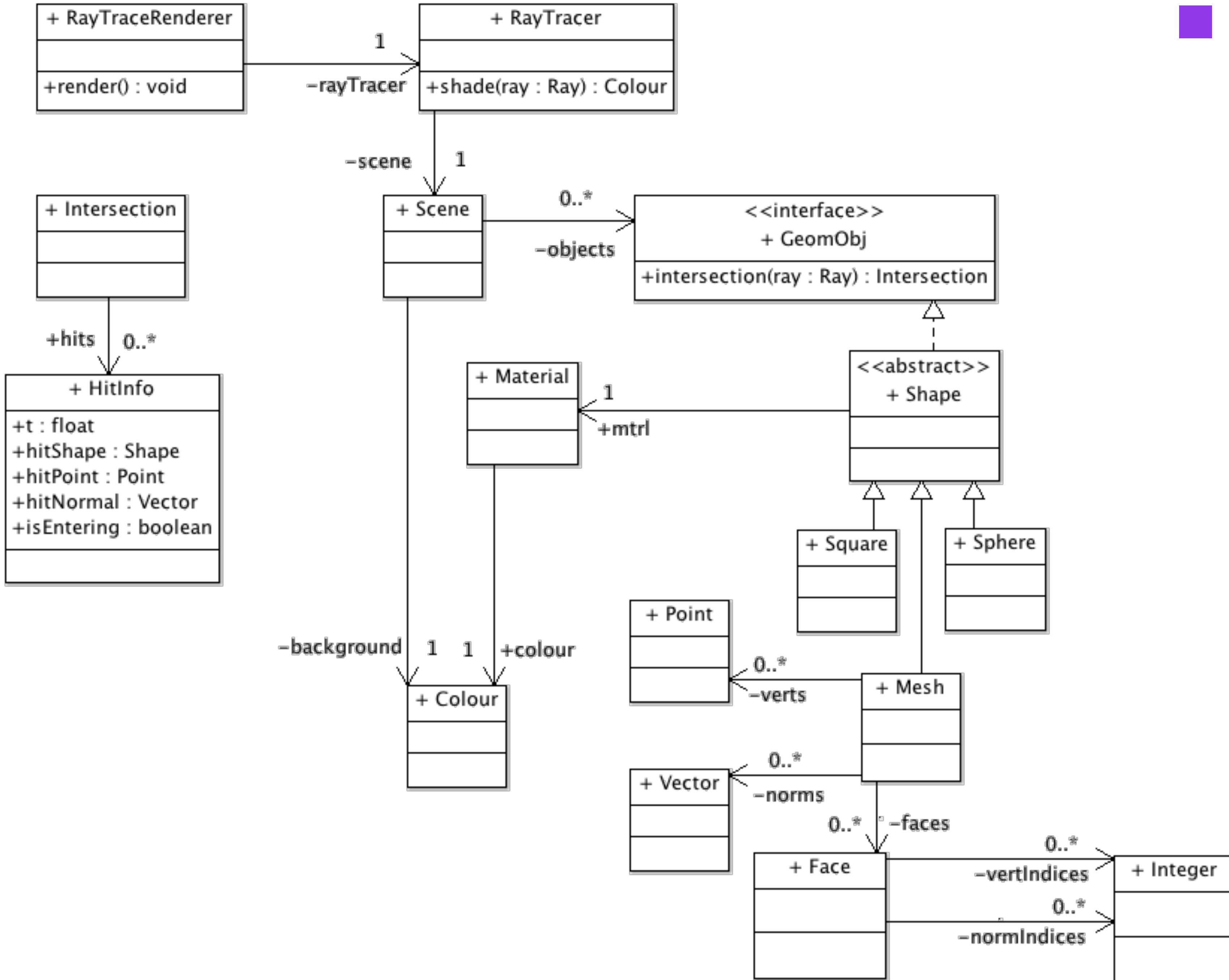
return list

Correct? **Yes!**

(Simplified)
rendering framework



We also want to draw
3D objects in other colours than red, so …


+ RayTraceRenderer
+render() : void
1
-rayTracer
+ RayTracer
+shade(ray : Ray) : Colour
-scene
1
+ Intersection
+hits
0..*
+ HitInfo
+t : float
+hitShape : Shape
+hitPoint : Point
+hitNormal : Vector
+isEntering : boolean
+ Scene
0..*
-objects
<<interface>>
+ GeomObj
+intersection(ray : Ray) : Intersection
+ Material
1
+mtrl
<<abstract>>
+ Shape
+ Square
+ Sphere
+ Point
0..*
-verts
+ Mesh
-background
1
1
+colour
+ Colour
0..*
+ Vector
-norms
0..*
-faces
+ Face
0..*
-vertIndices
0..*
-normIndices
+ Integer

Questions?