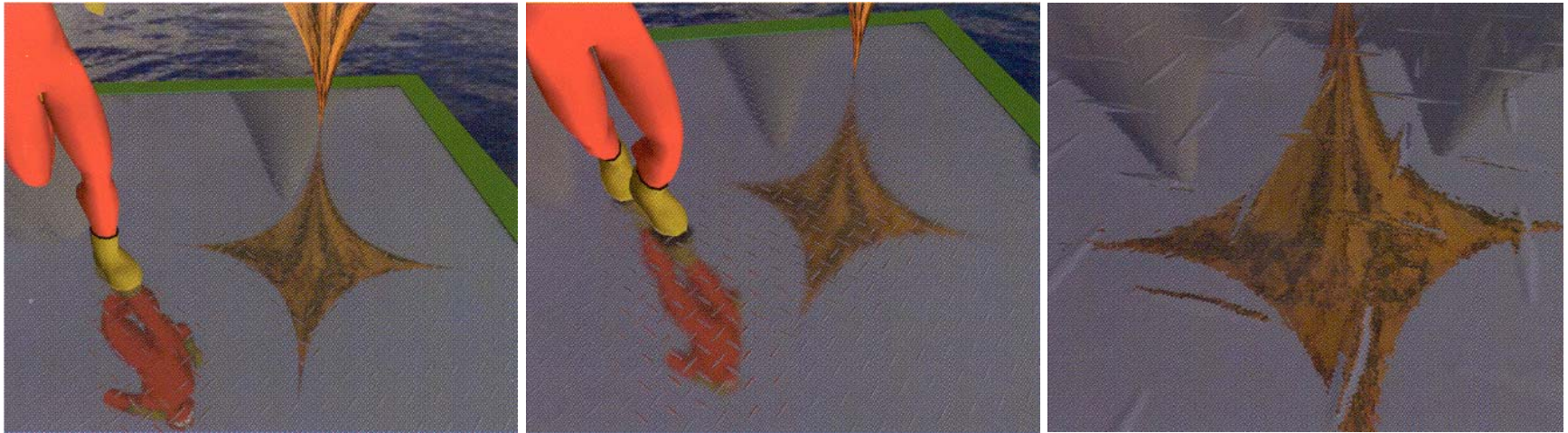


Reflections from Bumpy Surfaces

GPU Graphics

What are we trying to achieve?

- Most surfaces are not flat like glass
- Some of these surfaces still give off reflections, we would like to model this effect

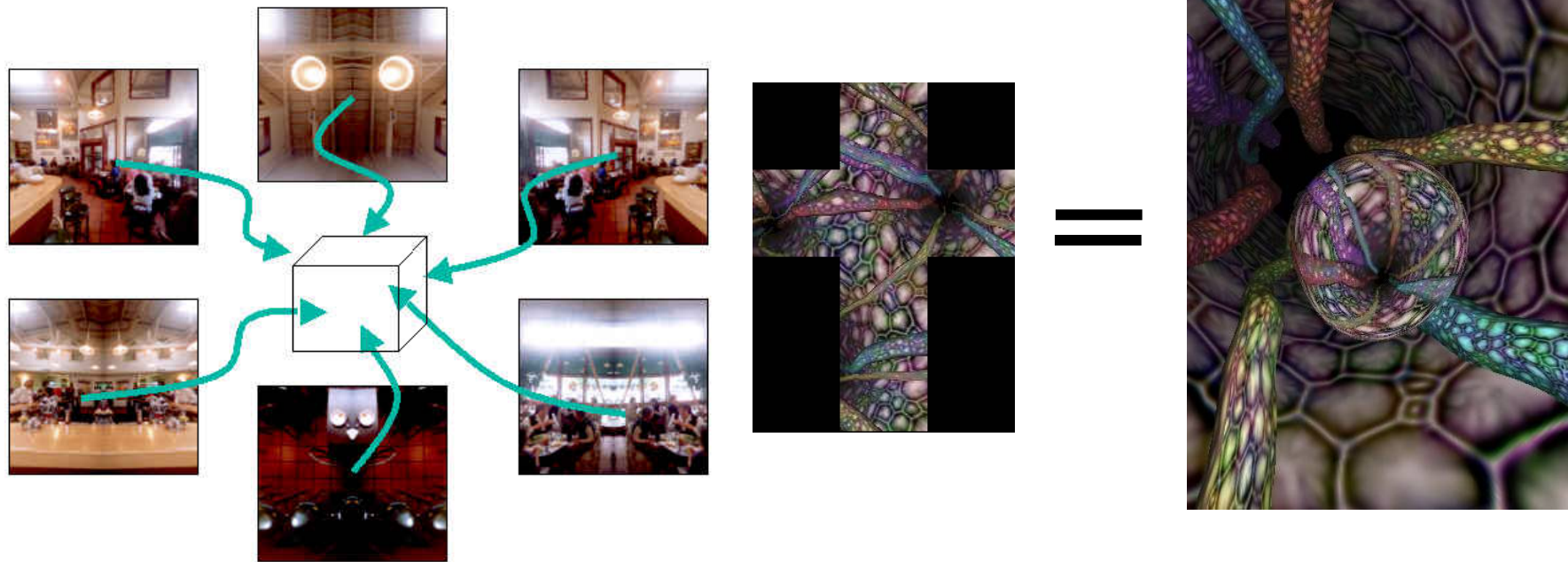




Methods for Reflection

1. Ray Tracing (expensive)
2. Environment Mapping (cube maps)
 1. Is an option in the shader 4 model
 2. In shader model 3 we can not update the cube map in real time easily
3. Spherical Environment Mapping (with cube map)

Cube Map Example



- ❑ Can not be updated in real-time easily
- ❑ Only currently works well for static scenes
- ❑ Shader 4 will allow arrays of textures, will make real-time cube maps easy to implement

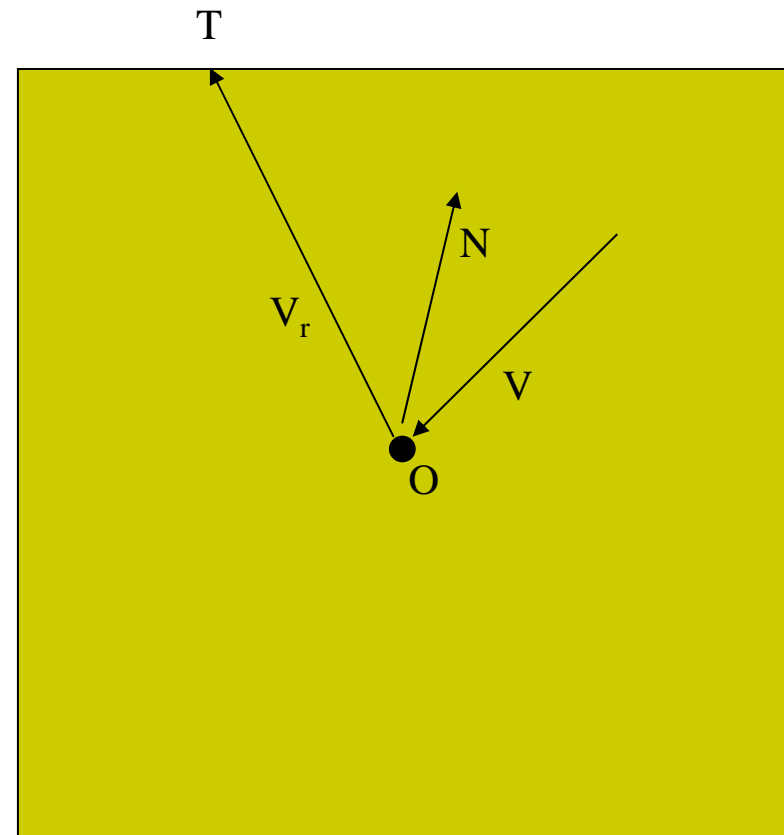
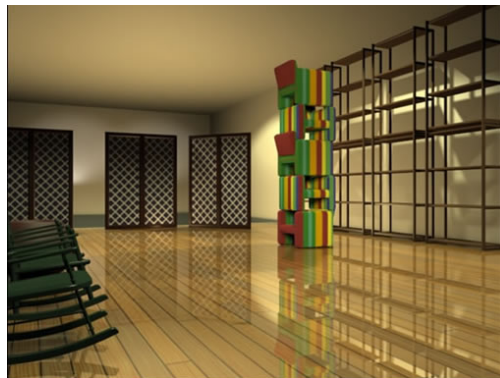


Simple Cube Map Lookup

1. Reflect incoming View vector around Normal
2. Intersect V_r with environment map and perform texture lookup

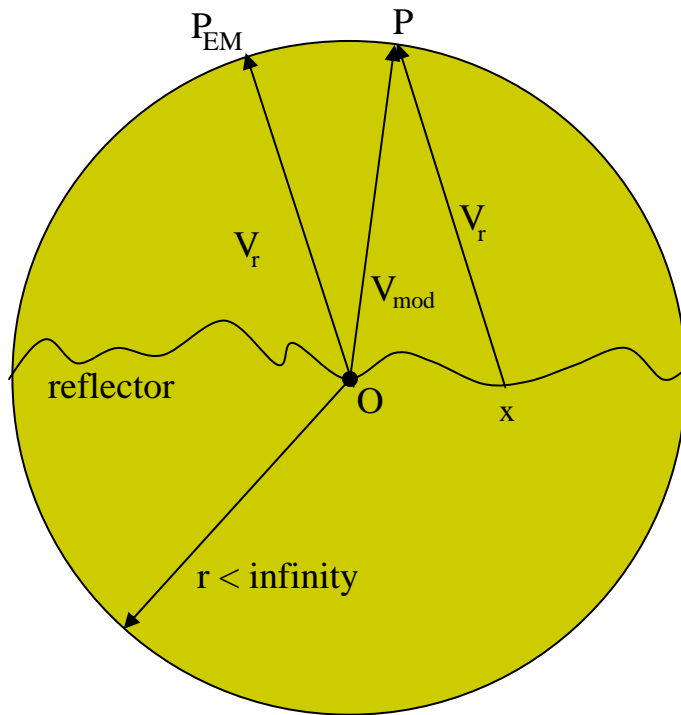
What is the issue here?

Two points with identical mirror directions will look up the same direction in the environment map



2D view of an environment map

Spherical Environment Mapping



V_r = view vector,
reflected about the point normal at x

P_{EM} = point on sphere interior
using regular cube map

P = intersection between line $x + V_r t$
and sphere using $V_{mod} = P - O$ to index
cube map based at O

V_{mod} = modified view vector

- ❑ Assume the environment map exists on the interior of a sphere with a finite radius and center
- ❑ Define a ray using the position and mirror direction of the point, calculate the ray intersection with environment sphere
- ❑ Use intersection as the environment lookup direction

An even simpler case: The planar reflection

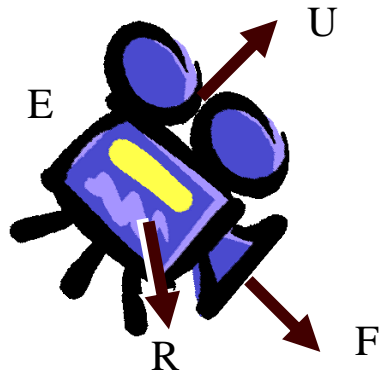
Two Pass algorithm

1. Reflected image is rendered to a texture
2. Reflected surface is shaded with reflected texture from first pass

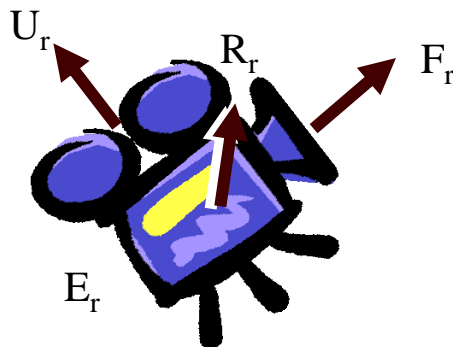
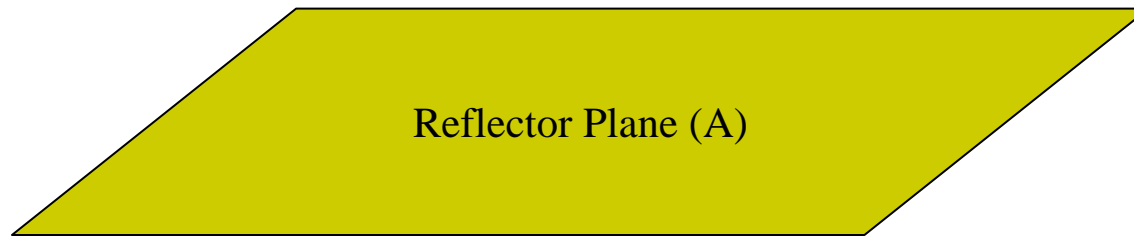
Only useful for something like floors



Rendering Reflection to Texture



Camera Pos: $E_r = \text{reflect}_a(E)$
Front: $F_r = \text{reflect}_a(F)$
Right: $R_r = \text{reflect}_a(R)$
Up: $U_r = -\text{reflect}_a(U)$



1. Reflect the current viewpoint about the reflector plane.
2. Render the relevant parts of the scene from the new viewpoint.



Reflection Texture Lookup

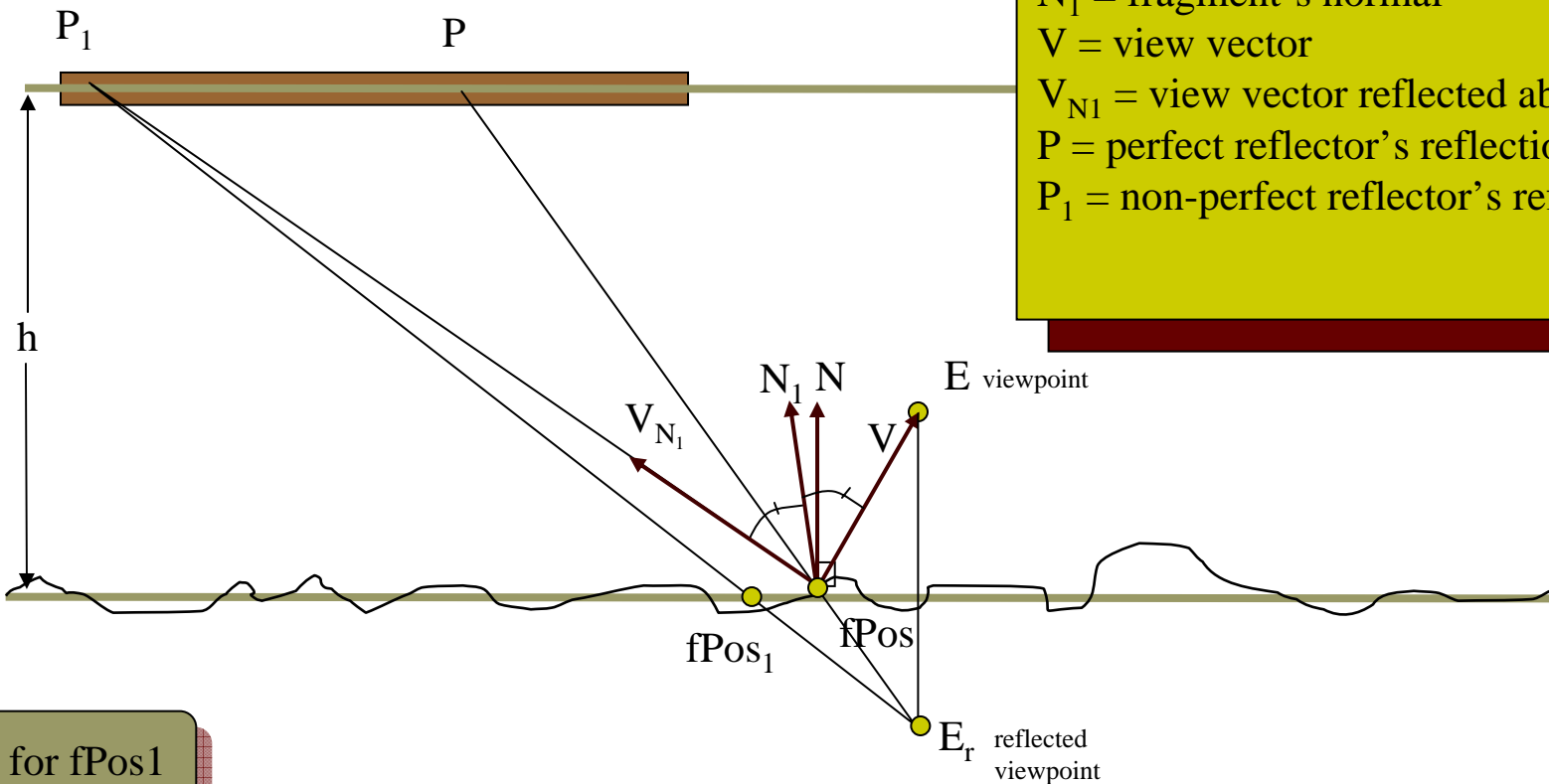
□ For the Perfect Reflector:

```
for_each fragment {  
    Fpos = fragment world position  
    Vr = reflected view matrix  
    Epos = eye space position = Fpos * Vr  
    Ppos = post projective space = Epos * P  
    S = scale translate matrix  
  
    texture_lookup = (S*P*Vr*Fpos).xy  
  
    color = tex2d(texture_lookup)  
}
```

Reflection Texture Lookup

□ For the Bumpy Reflector

N = normal
 E = viewpoint
 E_r = reflected viewpoint
 N_1 = fragment's normal
 V = view vector
 V_{N_1} = view vector reflected about N_1
 P = perfect reflector's reflection
 P_1 = non-perfect reflector's reflection



Solve for $fPos_1$