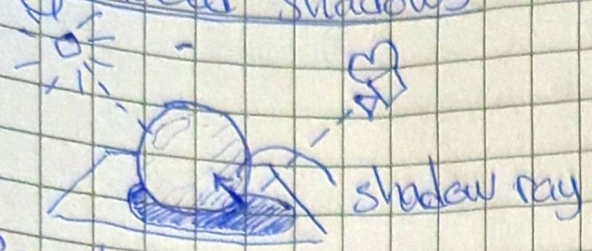
20 - Shadows & reflections

Ray traced shadows



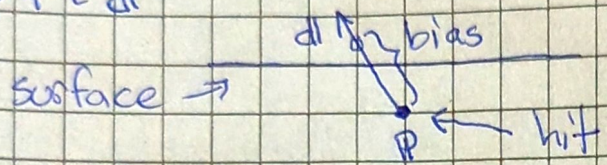
Rounding errors

Algorithm

Found ray hit at t
  If $t \geq \delta$, ignore hit
  (δ is a small positive value (bias))

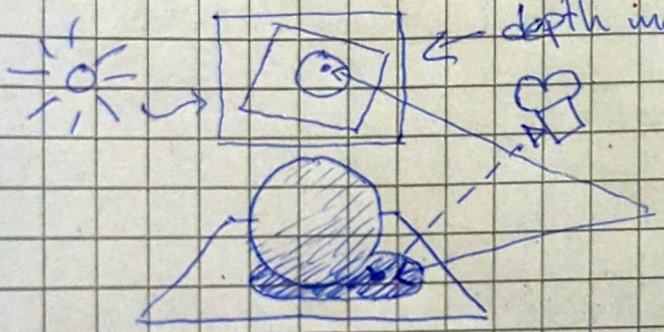
$X = P + t\,dl$

Because of rounding errors, sometimes a hit will be directly behind a surface, so it will create a shadow. We can introduce a bias to fix this.



Shadows without ray tracing → shadow mapping

Faster than ray traced shadows, but more complex. Works by creating a depth map (shadow map) raster image from a light source and then use that for lookup if the point is in shadow or not. Can not be used for complex lights like mesh lights.

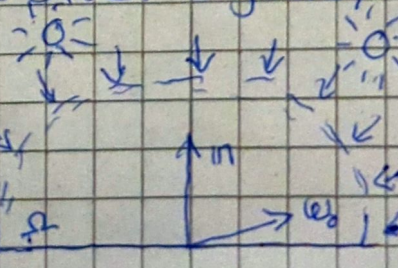


The point from the depth map isn't matching the point from the camera, so it must be in shadow.

Reflections

Reflections refers to reflection from other objects, not the light sources.

The rendering equation accounts for global illumination, i.e. light bouncing off objects.



"Perfect" Specular reflections of objects

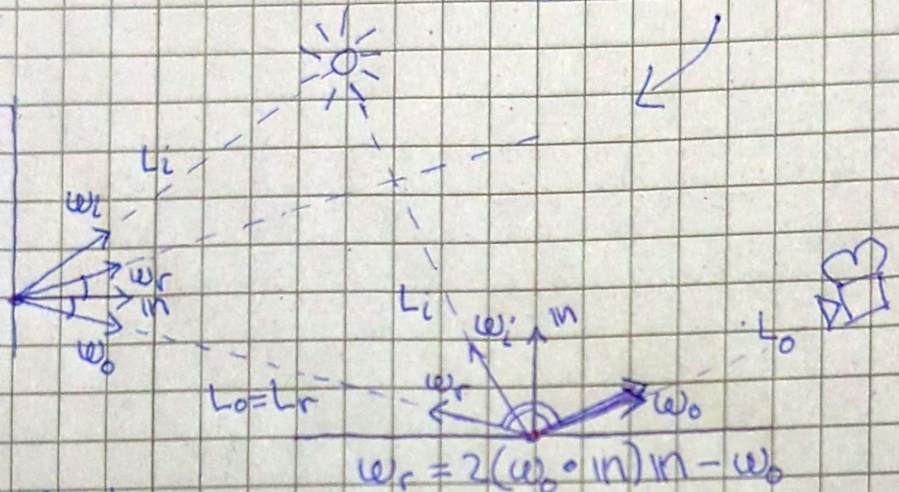
Perfect refers to mirror like reflection, imperfect would be blurry

$$L_o(\omega_o) = \sum_i L_i(\omega_i)\cos\theta_i f_r(\omega_i, \omega_o) + \underbrace{L_r(\omega_r)K_r}_{\text{reflection component}}$$

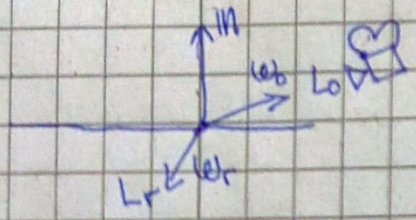
(K_r: Material reflection constant, often $= k_s$)

$$L_r(\omega_r) = \sum_i L_i(\omega_i)\cos\theta_i f_r(\omega_i, \omega_o) + L_r(\omega_r)K_r$$

This is a recursive process, we need to set a limit



Refractions



Almost like reflections, but for semi-transparent surfaces. We need to have a refraction index for the two mediums, like glass and air.

Ray types

- Primary rays ← Rays from camera/eyes
- Secondary rays ← Rays after first hit
  - Reflection/refraction rays
    - Trace to infinity
    - Find the closest hit
    - Record the hit information
  - shadow rays
    - Trace until reaching the light source
    - Find any hit
    - No need for the hit information

NB! shadow rays can be its own procedure if we want to optimize, or included with reflection/refraction