Game Engines

CMPM 164, F2019

Prof. Angus Forbes (instructor)
angus@ucsc.edu

Montana Fowler (TA)
mocfowle@ucsc.edu

Website: creativecoding.soe.ucsc.edu/courses/cmpm164
Slack: https://ucsccmpm164.slack.com

Homework 2B - extended

2B: Implement a classic Whitted recursive ray tracer (due 10/17 at 11:59pm)

For a C grade: Ray trace 3 or more solid spherical objects with 2 or more lights using the Phong lighting model

For a B grade: Ray trace 3 or more solid and transparent and mirrored spherical objects (i.e., use reflection and refraction), including shadows

For an A grade: Same as above, but include at least one additional kind of shape (i.e., besides a sphere)

Comment code clearly to demonstrate that you understand what you are implementing.

Homework 2B – extra credit

Extra credit for each of the following:

- interactive camera
- implementing Fresnel effect
- complex shape or mesh
- outstanding aesthetics

Ideas for scene:

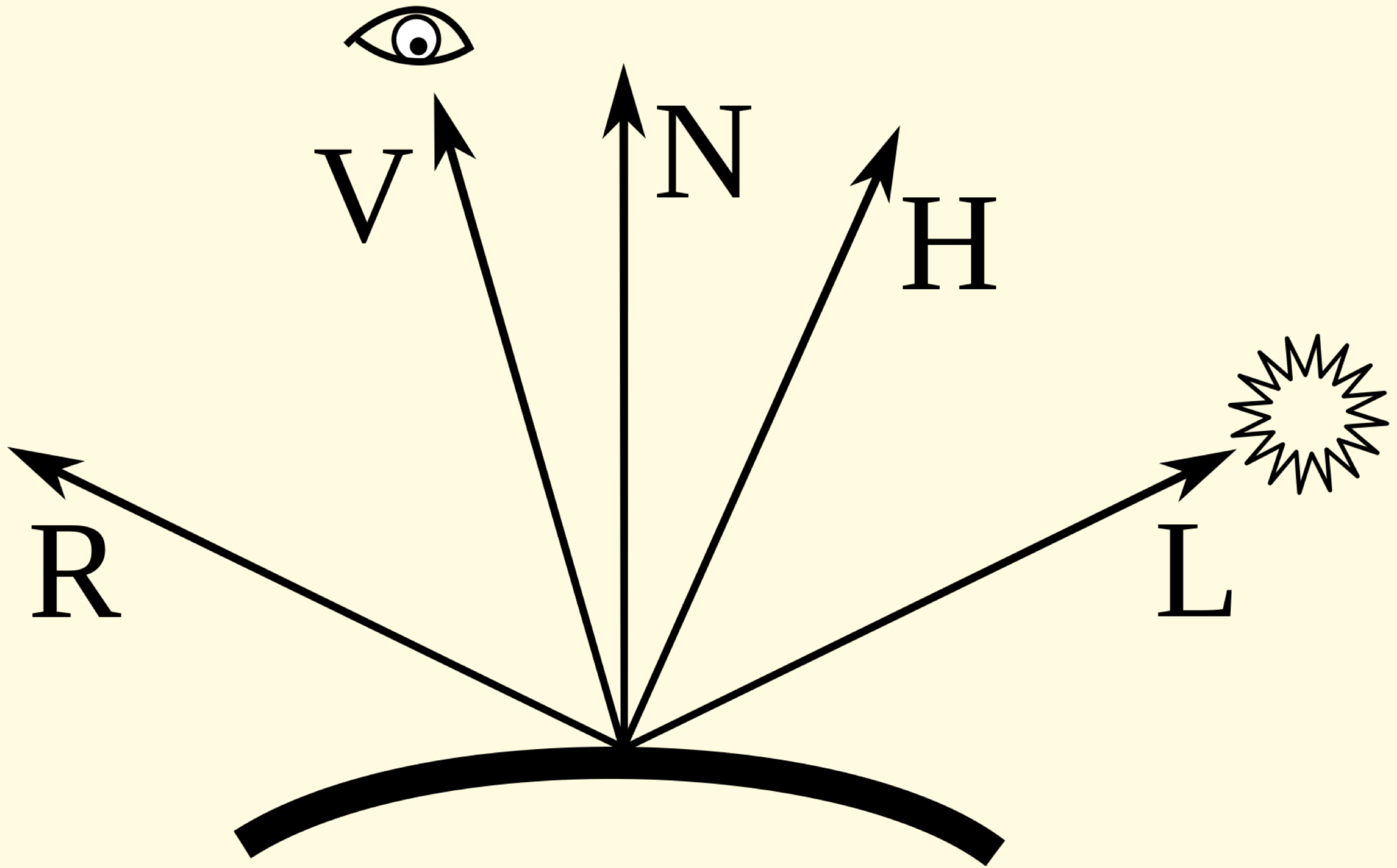
Disco ball?

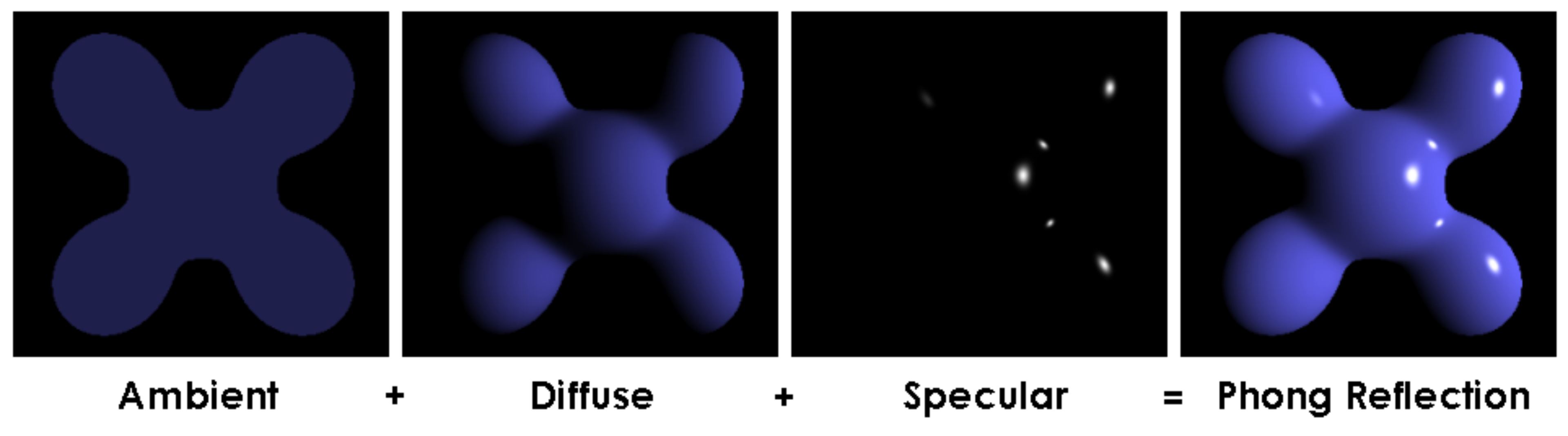
Mirrored planes?

Import character mesh?

Use different refractive indices to simulate different materials?

Implement chromatic dispersion where different color channels have slightly different refractive behavior?


V
N
H
R
L


Ambient
+
Diffuse
+
Specular
=
Phong Reflection

Calculating color at intersections

Diffuse calculation:

Requires *Normal* vector and *Light* vector

DiffuseContribution =
    diffuseColor * max((Normal . Light), 0.0)

The diffuse term is largest when the normal and the light are parallel. The diffuse term is 0 when the normal and the light are >= perpendicular.

Calculating color at intersections

Specular calculation:

Requires *Reflect* vector and *View* vector. That is, it can change based on the position of the camera

SpecularContribution =
    specularColor * (Reflect . View)^shininessFactor

The specular term is large only when the viewer direction is aligned with the reflection direction. The highlights are sharper the greater the shininessFactor is.

(*Reflect* vector is calculated using *Light* vector and *Normal* vector)

Calculating color at intersections

Alternative Specular calculation:

Requires *Normal* vector and a *Half* vector that is in between the *View* and the *Light* vectors.

SpecularContribution =
    specularColor * (Normal . Half)^shininessFactor

The specular term is large only when the viewer direction is aligned with the reflection direction. The highlights are sharper the greater the shininessFactor is.

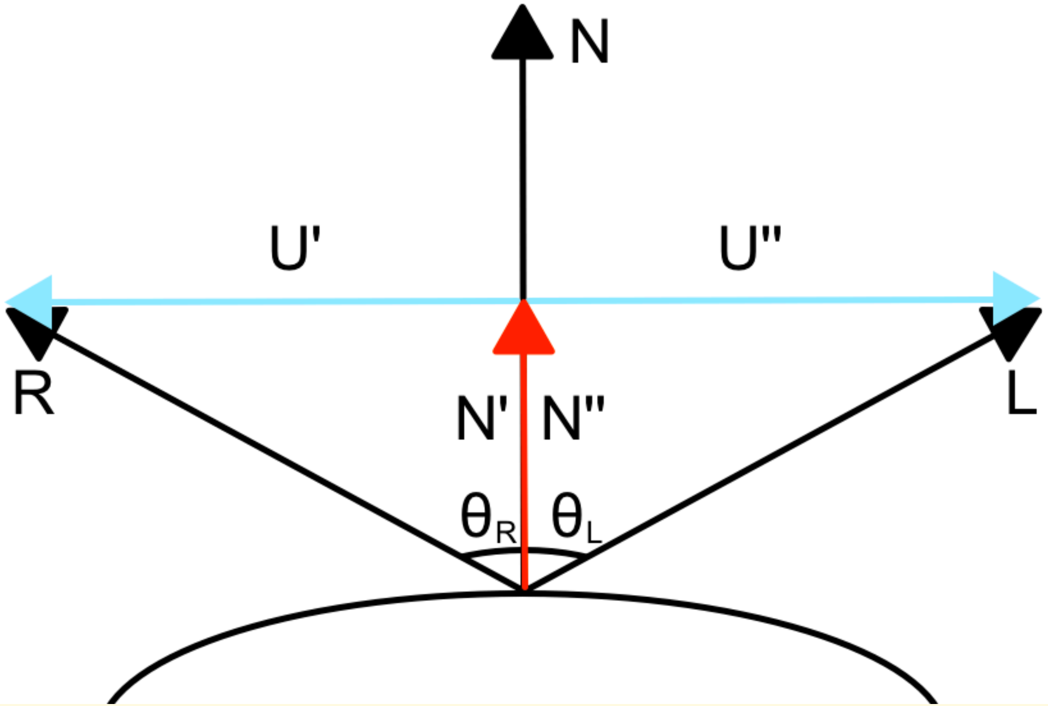
Calculating color at intersections

Calculate diffuse and specular contribution for each light and add them together

Check if light is not occluded by another object, otherwise move to next to light. If all lights are occluded, then the point is completely in shadow.

Reflection

Reflection



$$\hat{U}' = \hat{R} - \hat{N}' = \hat{R} - (\hat{R} \cdot \hat{N})\hat{N}$$

$$\hat{U}'' = \hat{L} - \hat{N}'' = \hat{L} - (\hat{L} \cdot \hat{N})\hat{N}$$

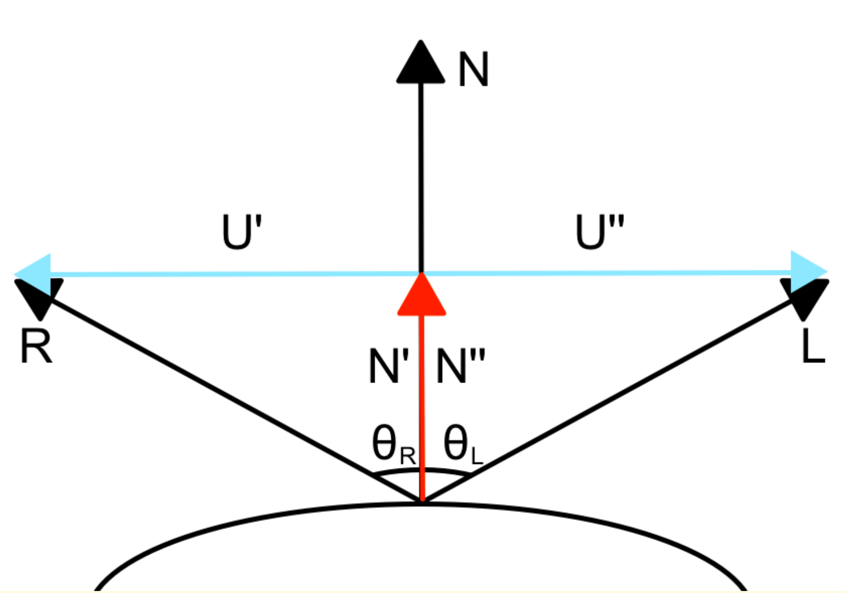
$$\hat{R} - (\hat{R} \cdot \hat{N})\hat{N} = -(\hat{L} - (\hat{L} \cdot \hat{N})\hat{N})$$

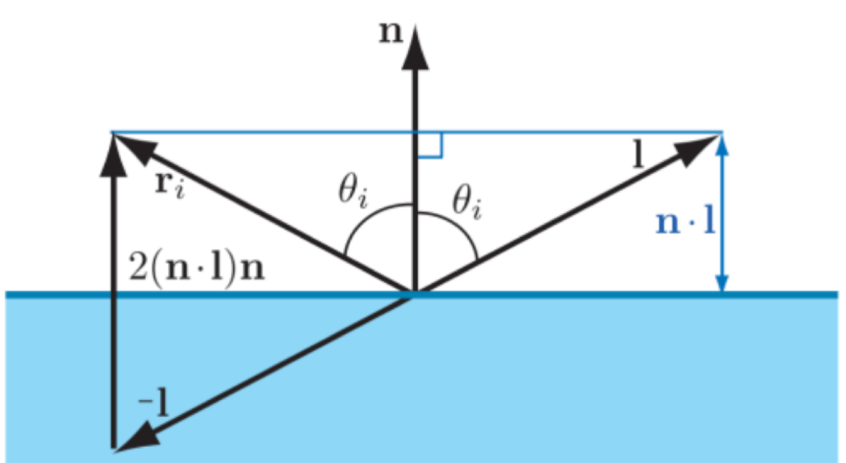
$$\hat{R} - (\hat{L} \cdot \hat{N})\hat{N} = -(\hat{L} - (\hat{L} \cdot \hat{N})\hat{N})$$

$$\hat{R} = (\hat{L} \cdot \hat{N})\hat{N} - (\hat{L} - (\hat{L} \cdot \hat{N})\hat{N})$$

$$\hat{R} = (\hat{L} \cdot \hat{N})\hat{N} - \hat{L} + (\hat{L} \cdot \hat{N})\hat{N}$$

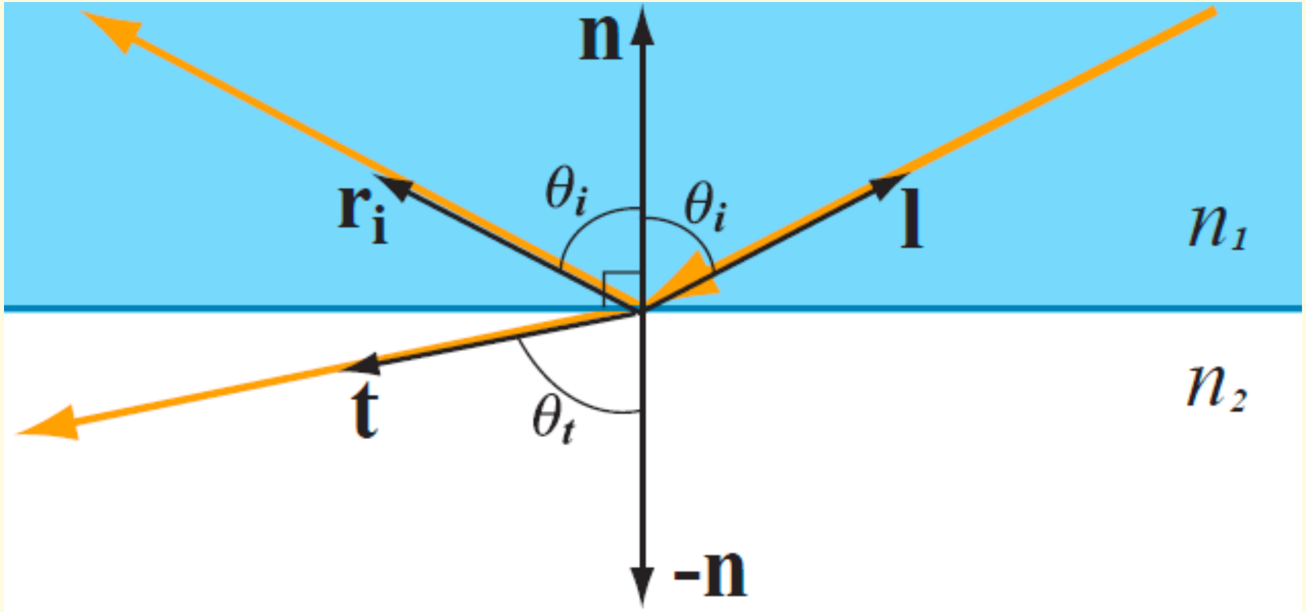
$$R = 2(\hat{N} \cdot \hat{L})\hat{N} - \hat{L}$$





Refraction

Refraction



Snell's Law

https://en.wikipedia.org/wiki/List_of_refractive_indices

Vacuum = 1.0
Air = 1.000293
Water = 1.33
Glass = 1.52
Diamond = 2.417

$$\frac{\sin\theta_2}{\sin\theta_1} = \frac{v_2}{v_1} = \frac{n_1}{n_2}$$

"The indices of refraction of the media are used to represent the factor by which a light ray's speed decreases when traveling through a refractive medium, such as glass or water, as opposed to its velocity in a vacuum."

"Refraction between two surfaces is *reversible* because if all conditions were identical, the angles would be the same for light propagating in the opposite direction."

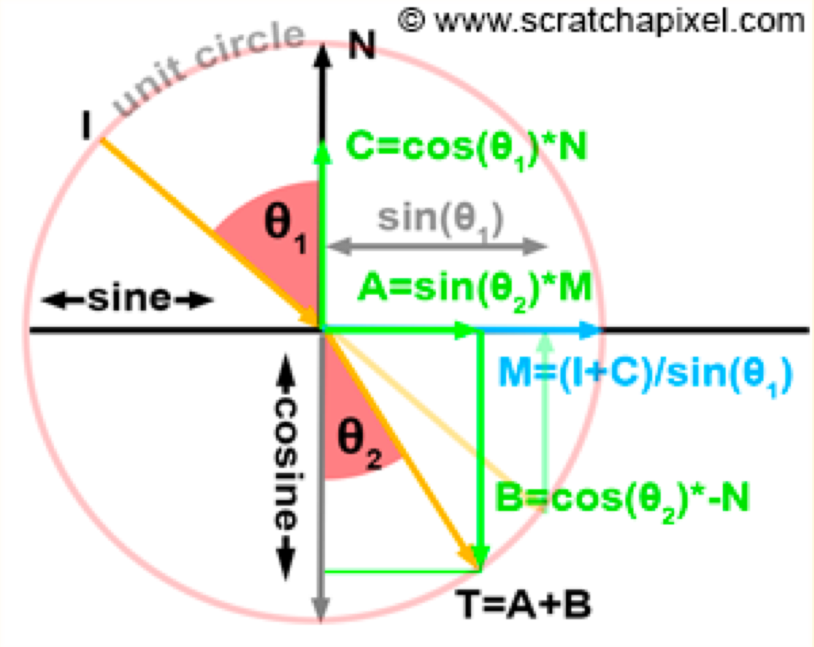
Refraction

$$A = M\sin(\theta_2),$$

$$B = -N\cos(\theta_2).$$

$$M = \frac{(I + C)}{\sin(\theta_1)}.$$

$$C = \cos(\theta_1)N.$$

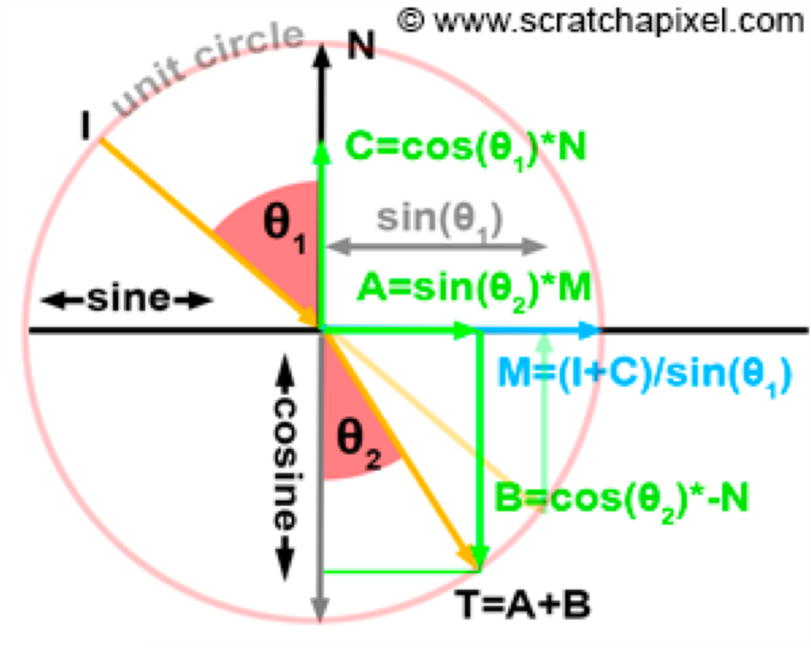


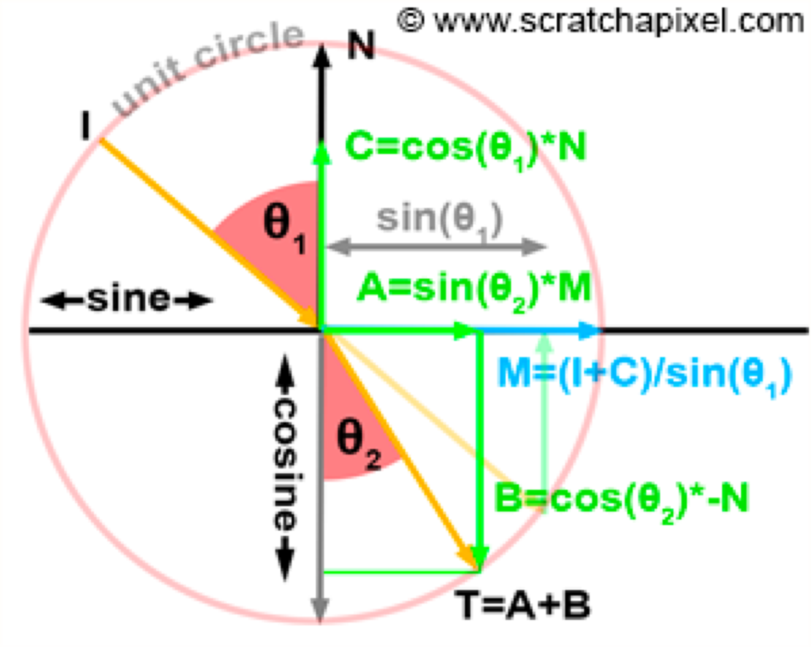
$$T = A + B,$$

$$T = M \sin(\theta_2) - N \cos(\theta_2),$$

$$T = \frac{(I + C)\sin(\theta_2)}{\sin(\theta_1)} - N \cos(\theta_2),$$

$$T = \frac{(I + \cos(\theta_1)N)\sin(\theta_2)}{\sin(\theta_1)} - N \cos(\theta_2).$$

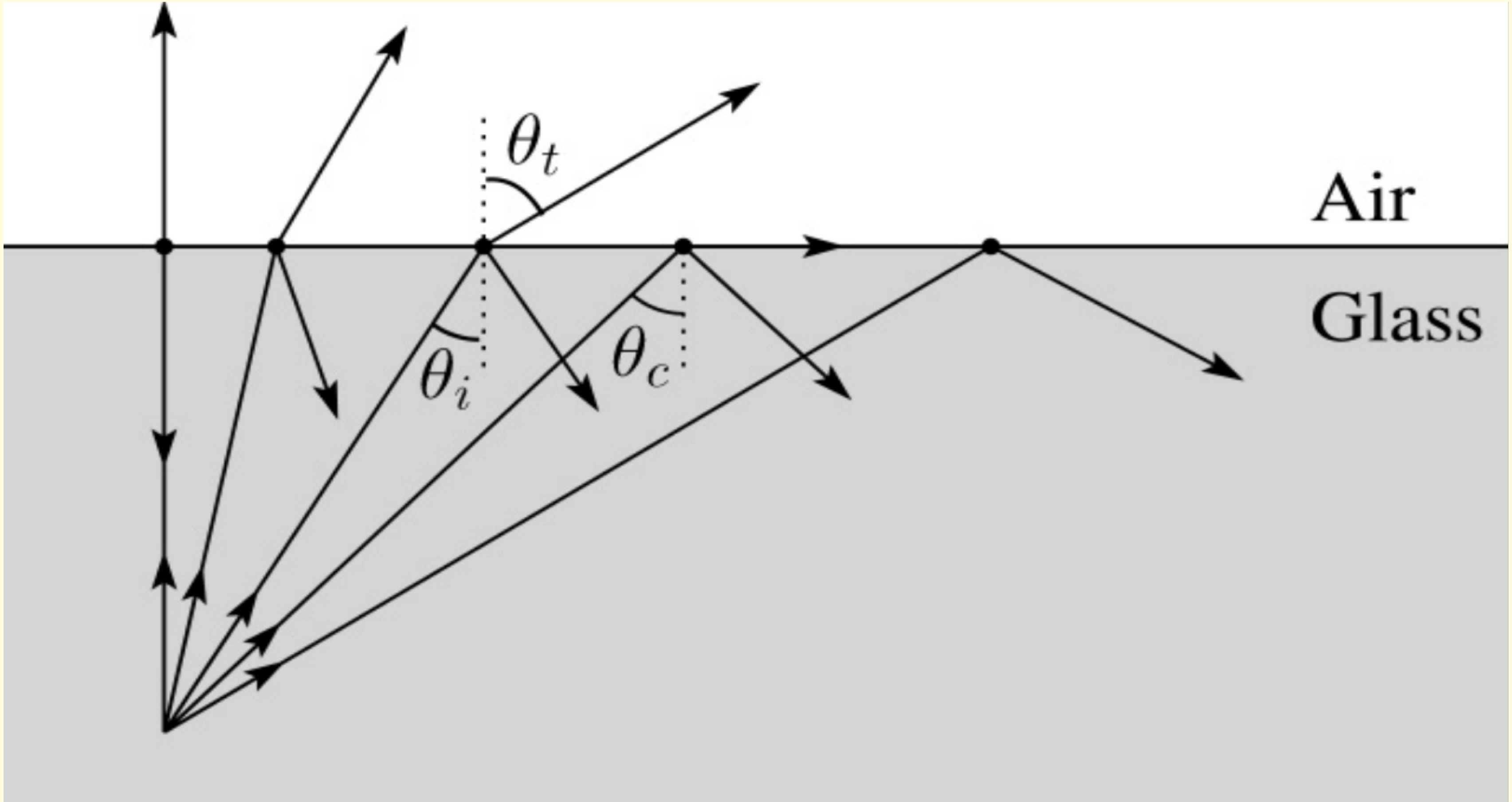




$$T = \frac{\eta_1}{\eta_2}(I + \cos(\theta_1)N) - N\cos(\theta_2).$$

$$T = \frac{\eta_1}{\eta_2}(I + \cos(\theta_1)N) - N\sqrt{1 - \left(\frac{\eta_1}{\eta_2}\right)^2 \sin^2(\theta_1)}.$$

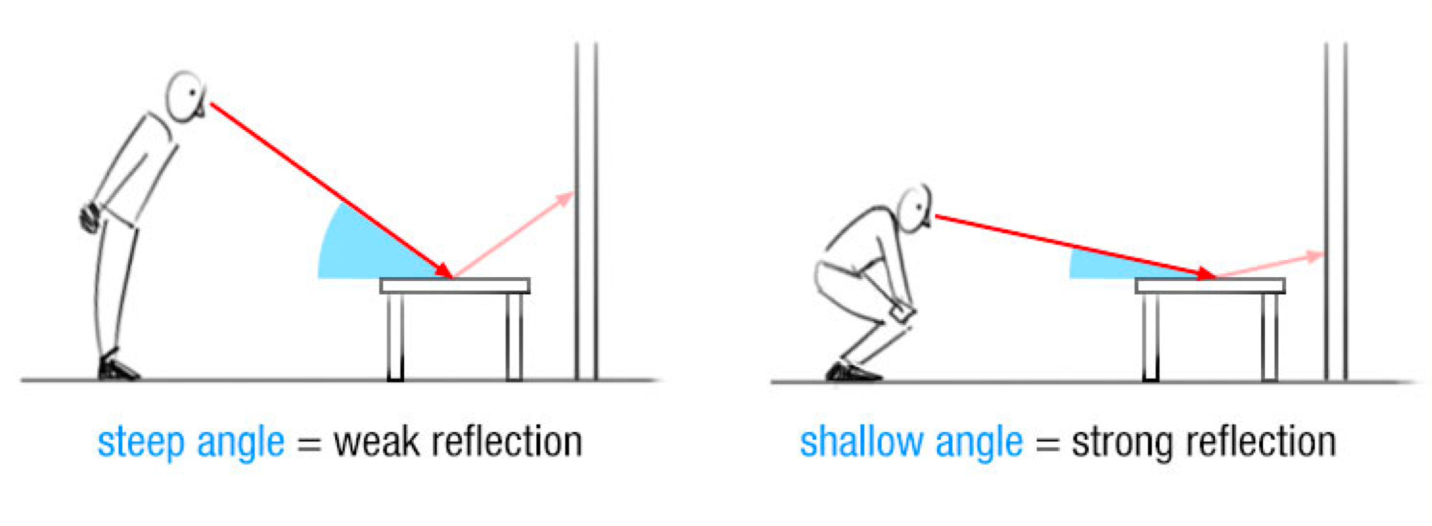
Refraction



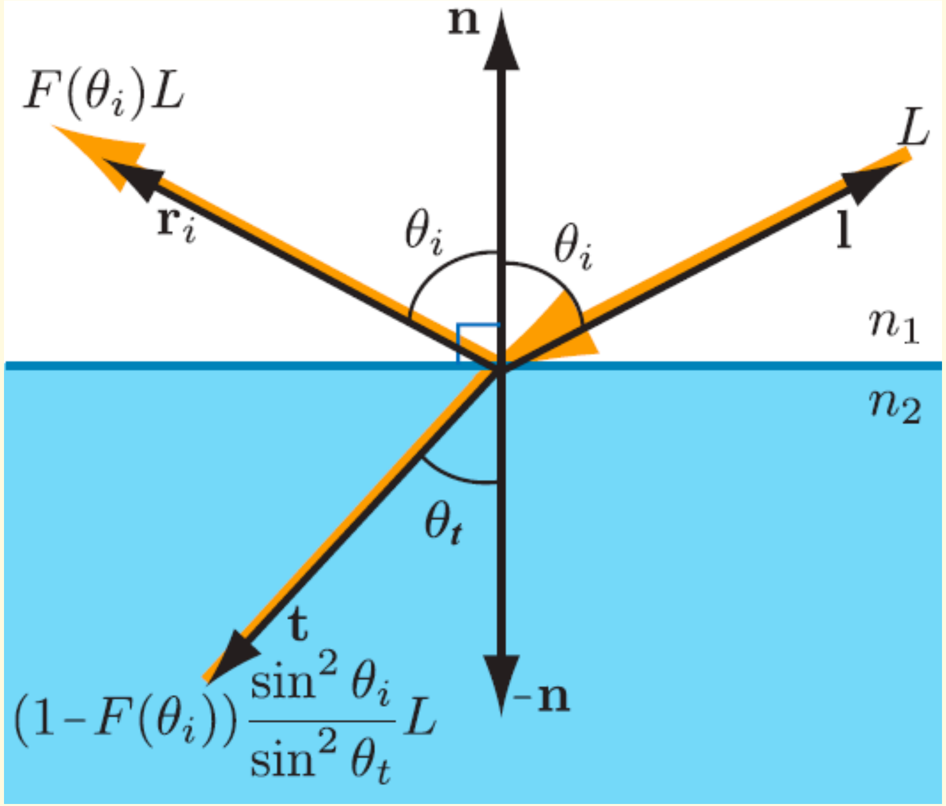
Fresnel Effect

steep angle = weak reflection

shallow angle = strong reflection



Refraction



n
$F(\theta_i)L$
L
$\mathbf{r}_i$
θ_i
θ_i
l
n_1
n_2
θ_t
t
$(1-F(\theta_i))\frac{\sin^2\theta_i}{\sin^2\theta_t}L$
-n

```
#include <stdlib.h>   // card > aek.ppm
#include <stdio.h>
#include <math.h>
typedef int i;typedef float f;struct v{f x,y,z;v operator+(v r){return
v(x+r.x,y+r.y,z+r.z);}v operator*(f r){return v(x*r,y*r,z*r);}f operator%(v r){return
x*r.x+y*r.y+z*r.z;}v(){}v operator^(v r){return v(y*r.z-z*r.y,z*r.x-x*r.z,x*r.y-
y*r.x);}v(f a,f b,f c){x=a;y=b;z=c;}v operator!(){return*this*(1/sqrt(*this%*this));}};i
G[]={247570,280596,280600,249748,18578,18577,231184,16,16};f R()
{return(f)rand()/RAND_MAX;}i T(v o,v d,f&t,v&n){t=1e9;i m=0;f p=-
o.z/d.z;if(.01<p)t=p,n=v(0,0,1),m=1;for(i k=19;k--;)for(i j=9;j--;)if(G[j]&1<<k){v p=o+v(-
k,0,-j-4);f b=p%d,c=p%p-1,q=b*b-c;if(q>0){f s=-b-sqrt(q);if(s<t&&s>.01)t=s,n=!
(p+d*t),m=2;}}return m;}v S(v o,v d){f t;v n;i m=T(o,d,t,n);if(!m)return v(.7,.6,1)*pow(1-
d.z,4);v h=o+d*t,l=!(v(9+R(),9+R(),16)+h*-1),r=d+n*(n%d*-2);f
b=l%n;if(b<0||T(h,l,t,n))b=0;f p=pow(l%r*(b>0),99);if(m&1){h=h*.2;return((i)
(ceil(h.x)+ceil(h.y))&1?v(3,1,1):v(3,3,3))*(b*.2+.1);}return v(p,p,p)+S(h,r)*.5;}i main()
{printf("P6 512 512 255 ");v g=!v(-6,-16,0),a=!(v(0,0,1)^g)*.002,b=!(g^a)*.002,c=
(a+b)*-256+g;for(i y=512;y--;)for(i x=512;x--;){v p(13,13,13);for(i r=64;r--;){v t=a*
(R()-.5)*99+b*(R()-.5)*99;p=S(v(17,16,8)+t,!(t*-1+(a*(R()+x)+b*
(y+R())+c)*16))*3.5+p;}printf("%c%c%c",(i)p.x,(i)p.y,(i)p.z);}}
```