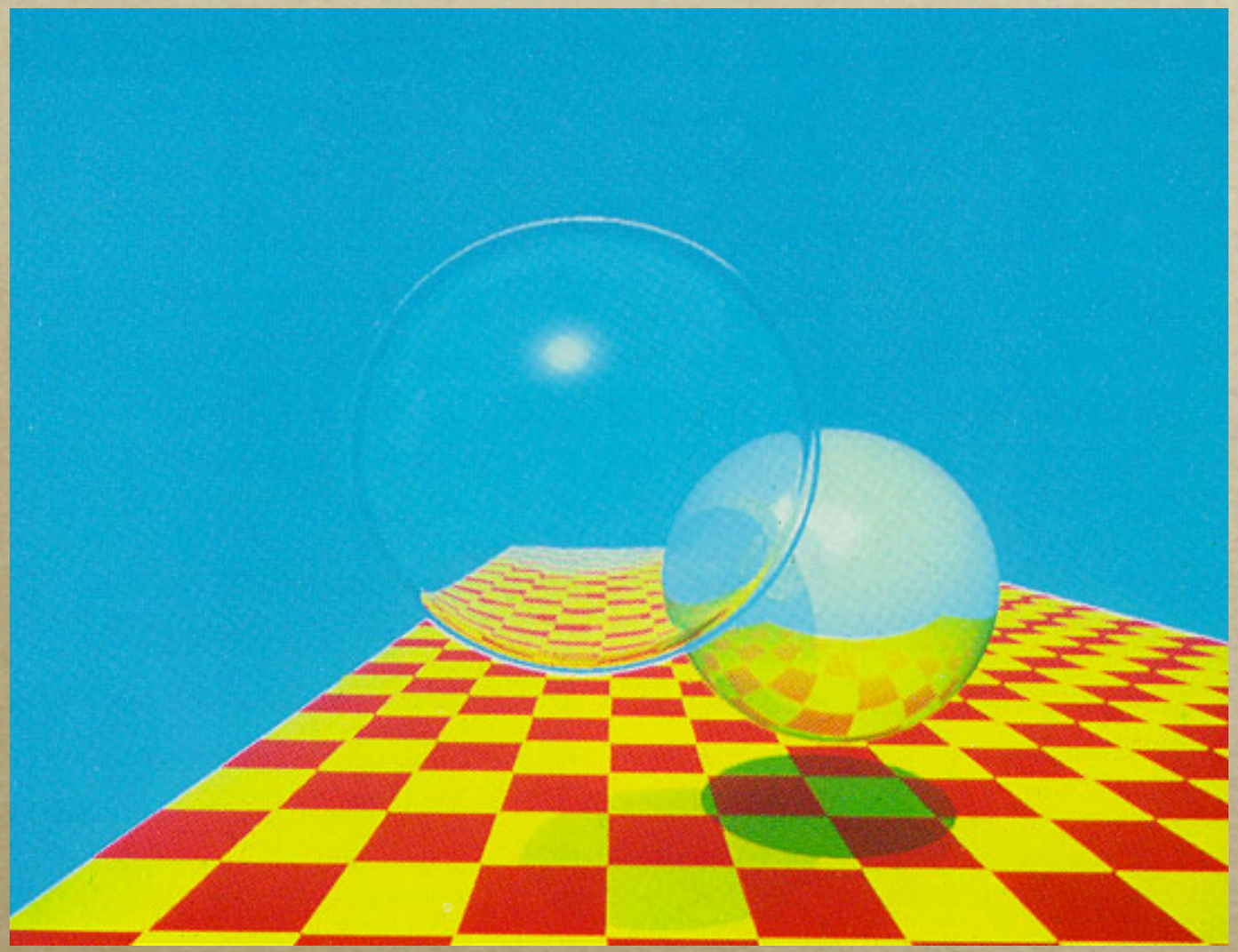
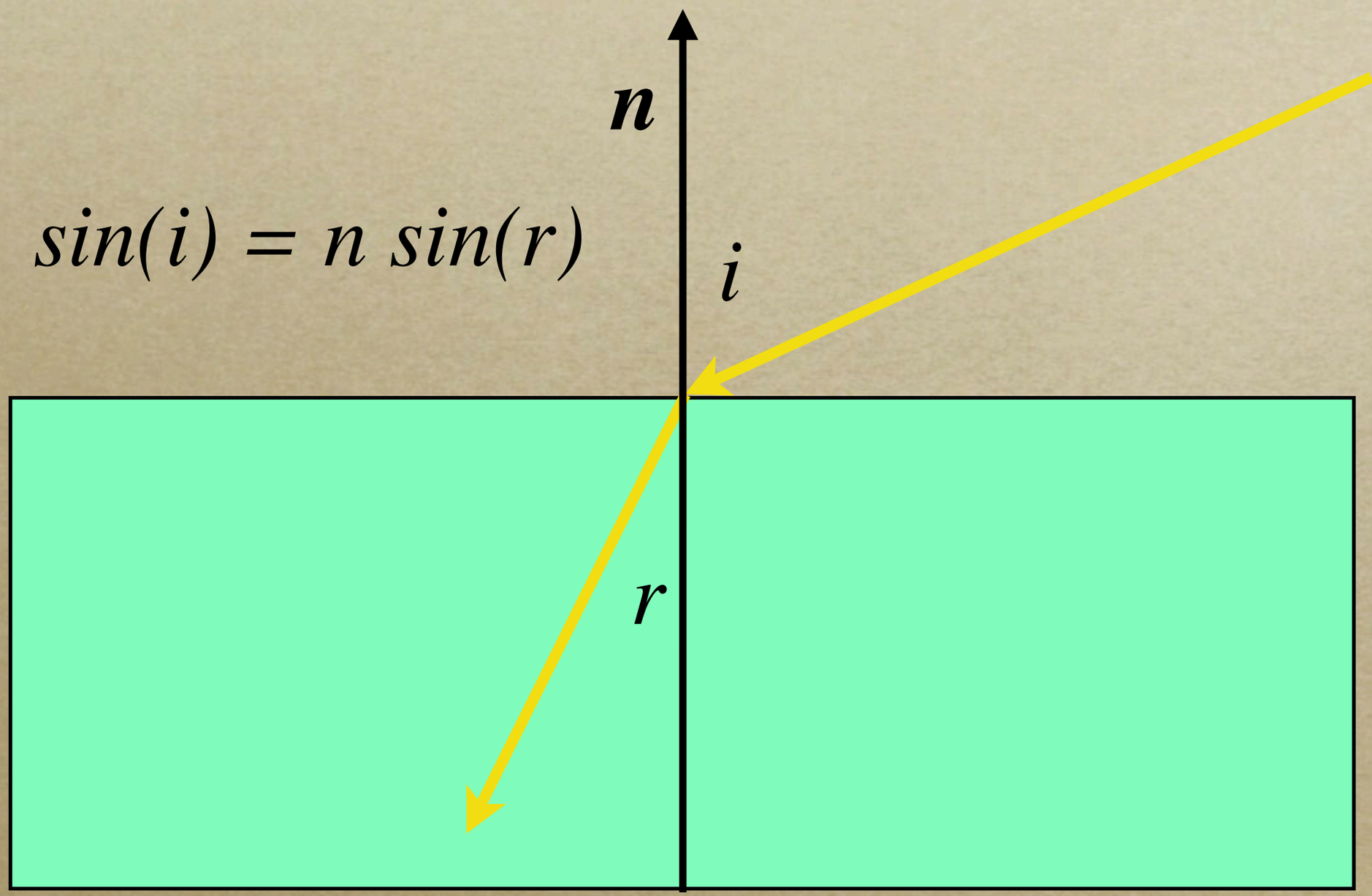
Refraction in Ray Tracing

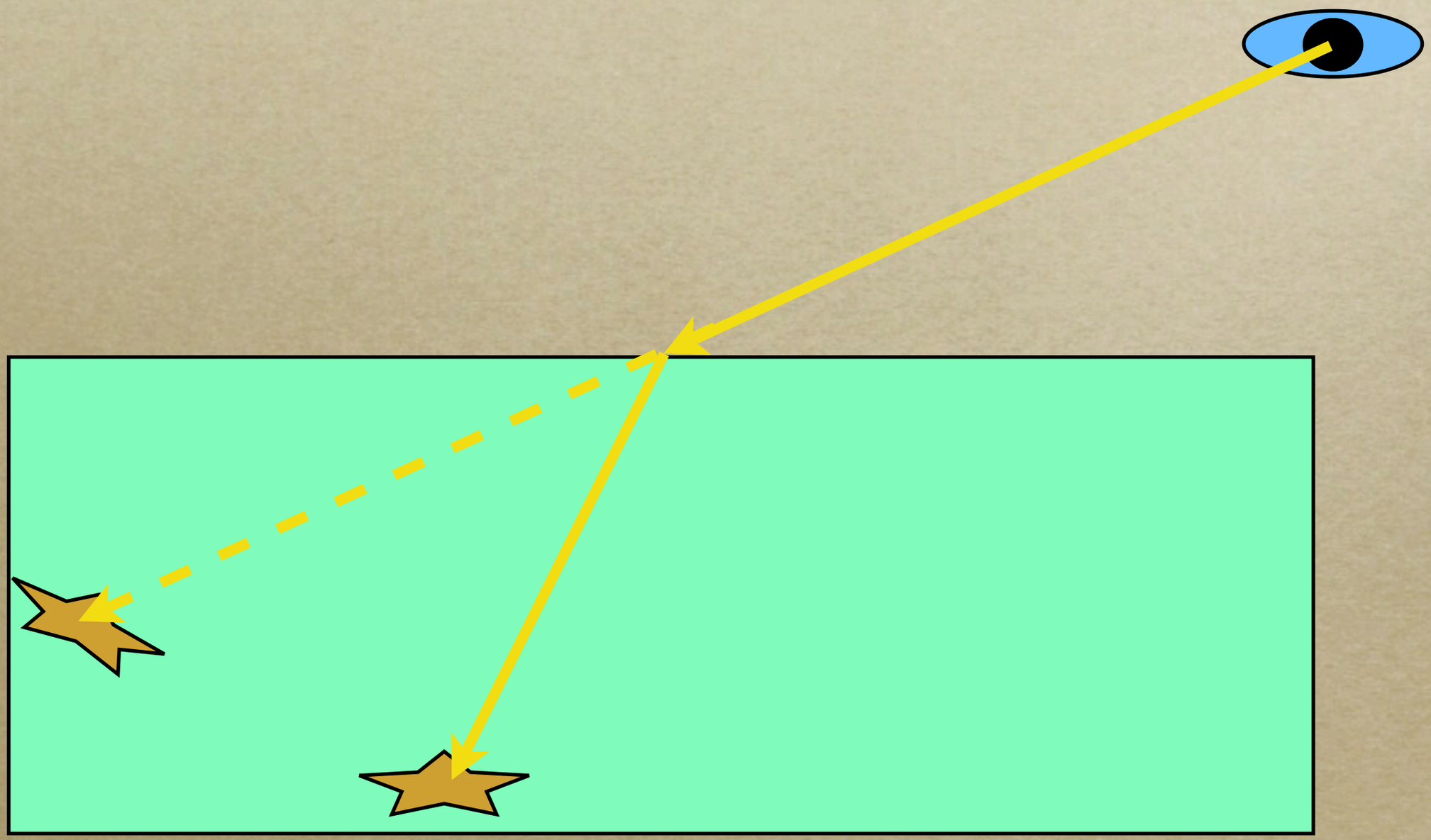


What is refraction?

$$sin(i) = n\ sin(r)$$



Effect of refraction



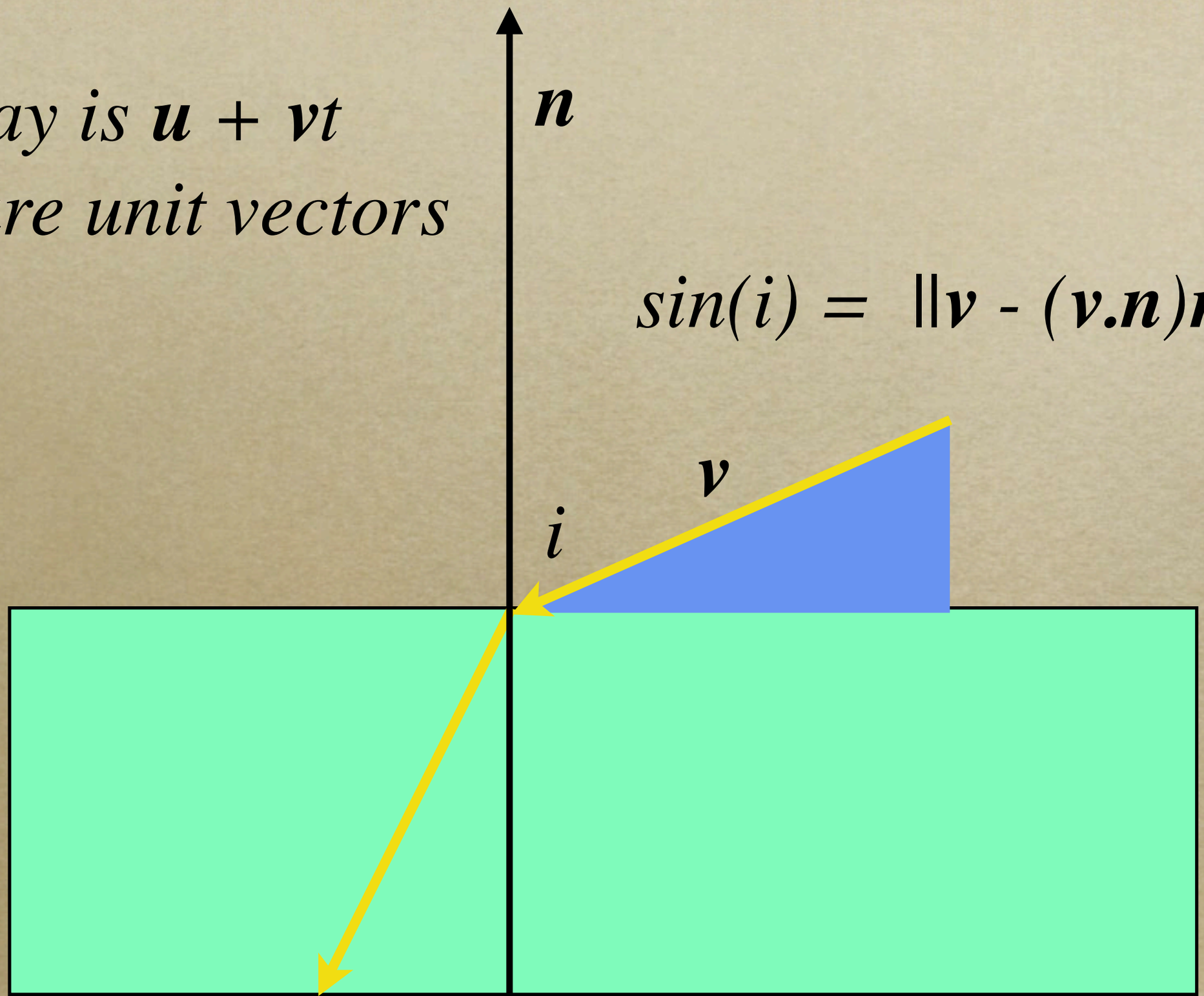
Refractive index, n

- *Actually a ratio*
- *Velocity of light in a medium = c/n*
- *n depends on the medium*
- $n_1 \sin(i) = n_2 \sin(r)$
- *n for air is almost 1.0*

Using vectors

Ray is $\boldsymbol{u} + \boldsymbol{v}t$

$\boldsymbol{n}$, $\boldsymbol{v}$ *are unit vectors*



$$sin(i) = \|\boldsymbol{v} - (\boldsymbol{v}.\boldsymbol{n})\boldsymbol{n}\|$$

Refracted ray

The refracted ray will have a 'parallel' part in the direction of $\boldsymbol{v} - (\boldsymbol{v}.\boldsymbol{n})\boldsymbol{n}$, *and magnitude par* $= \|\boldsymbol{v} - (\boldsymbol{v}.\boldsymbol{n})\boldsymbol{n}\| \, (n_1/n_2)$

and a 'normal' part, the right size to make up a unit vector $= k\,(\boldsymbol{v}.\boldsymbol{n})\boldsymbol{n}$

$$par^2 + k^2((\boldsymbol{v}.\boldsymbol{n})\boldsymbol{n})^2 = 1$$

$$k = \sqrt{(1 - par^2)/((\boldsymbol{v}.\boldsymbol{n})\boldsymbol{n})^2}$$

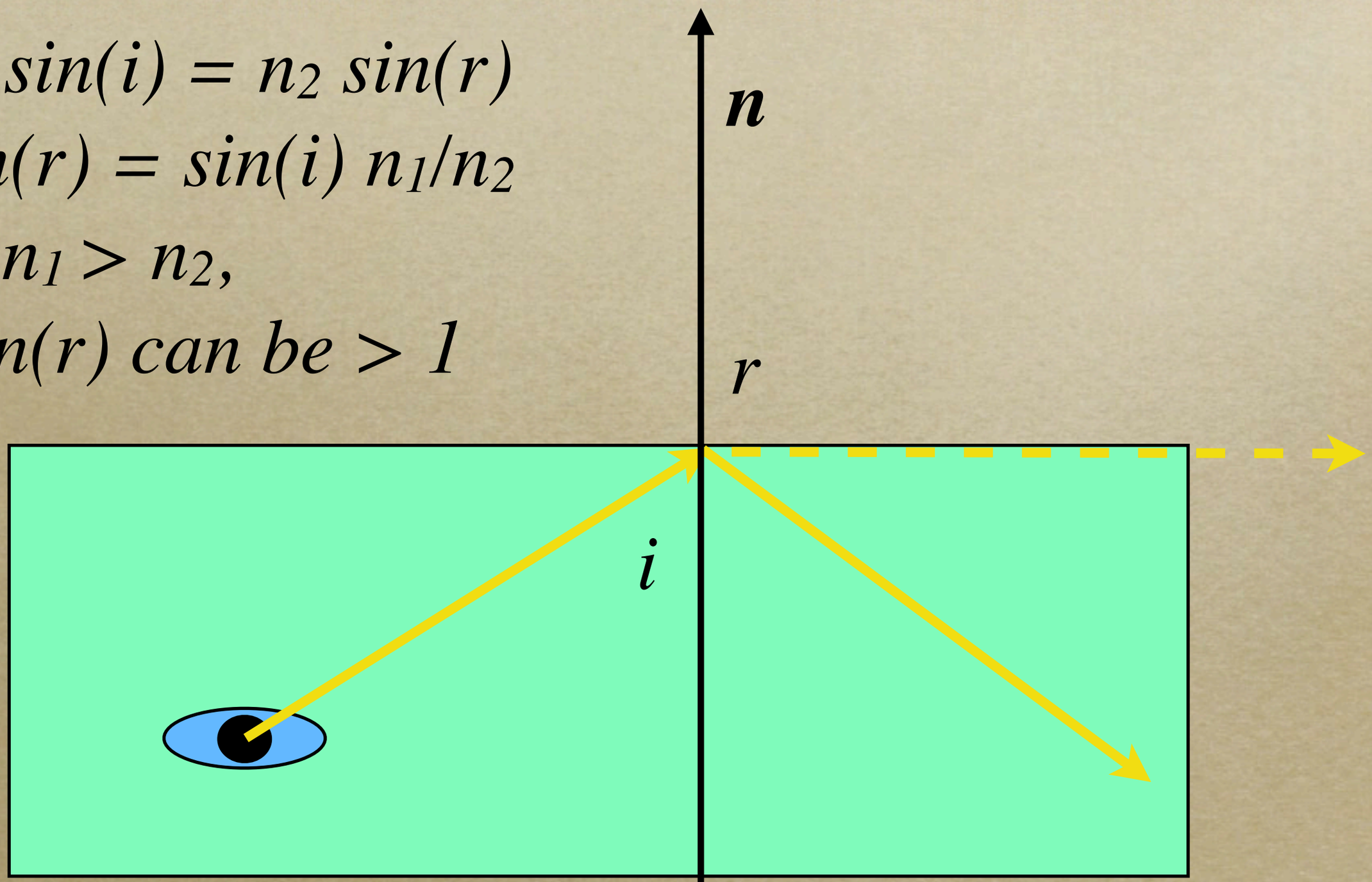
Total internal reflection

$n_1 \sin(i) = n_2 \sin(r)$

$\sin(r) = \sin(i)\, n_1/n_2$

if $n_1 > n_2$,

sin(r) can be > 1



The truth is more complicated

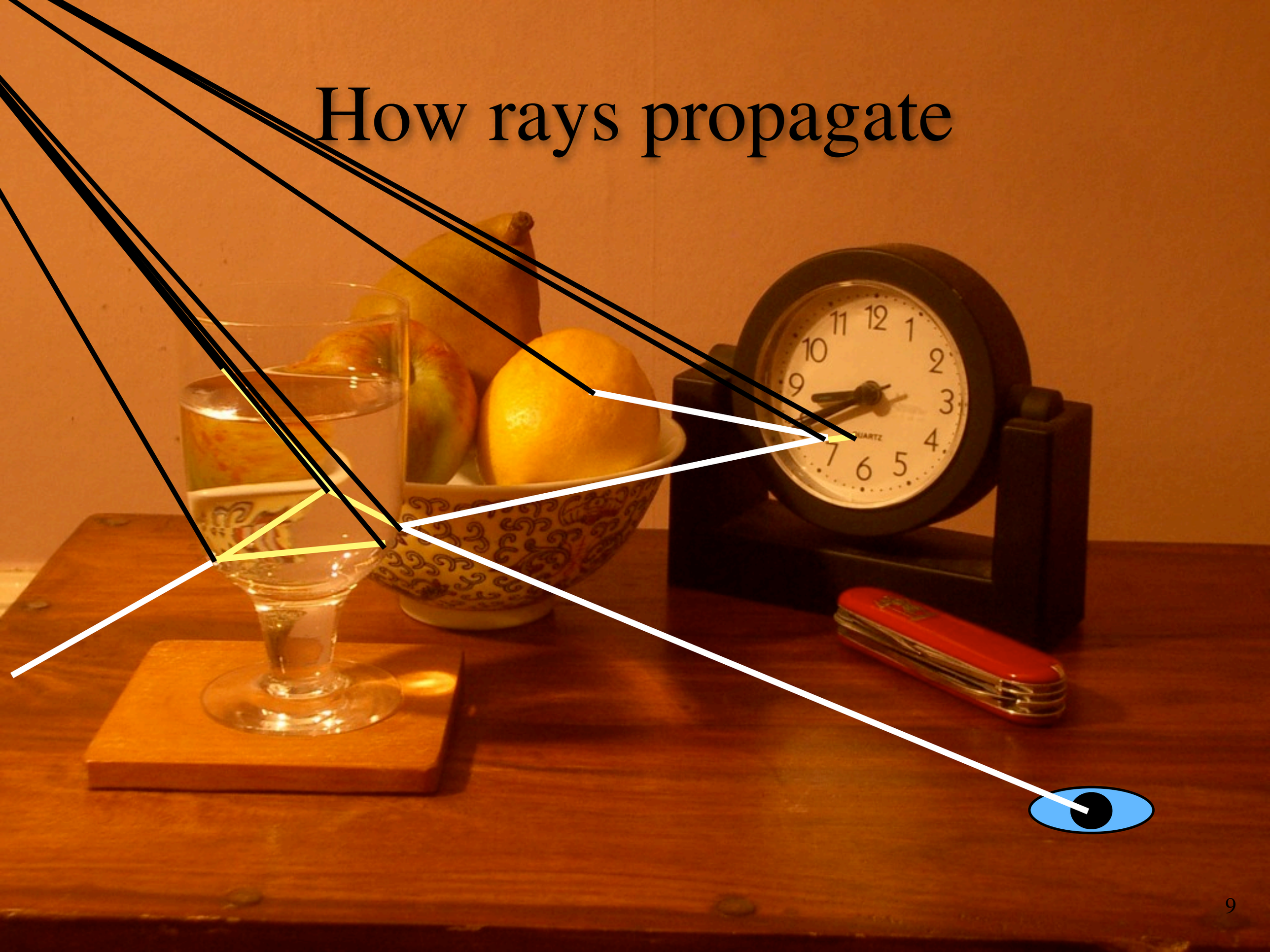
As the incident ray approaches the critical angle the proportion of light reflected goes up to 100%

The amount reflected and transmitted is given by Fresnel's equations…

and those are not part of COSC342

(phew!)

How rays propagate



Recursion is inescapable

Where we just have reflections, we can avoid recursion by doing the rest of the illumination before sending the reflective ray. But when we have refraction, we can spawn two, new, recursive rays at each intersection point.

How deep do we go?

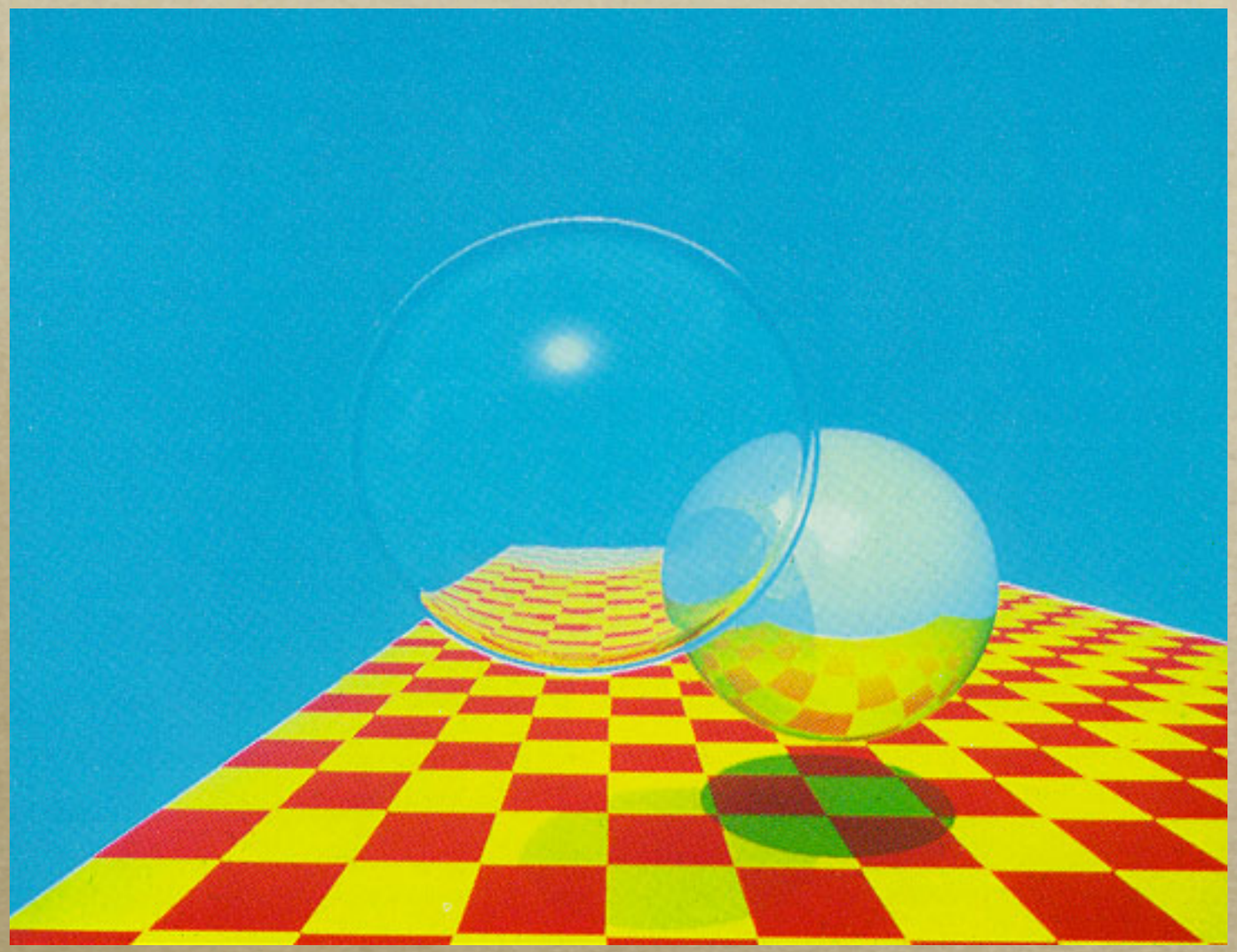
Real reflection and refraction is never perfect. We always lose a little light. So our coefficients k_d, k_s, k_r, k_t, for diffuse, specular, reflection and refraction are all less than 1.0

For $k_r = 0.8$, after 30 reflections the contribution will be $0.8^{30} < 1/1000$

Do it in the right order...

```
void ray(fraction, depth, u, v ...)
{
   if (depth > LIMIT) return;
   if (fraction < TINY) return;
   get hit... do local lighting...
        shadows...
   ray(fraction*kr, depth+1,
        hit, r(v) ...)
   ray(fraction*kt, depth+1,
        hit, t(v) ...)
}
```

Whitted 1980

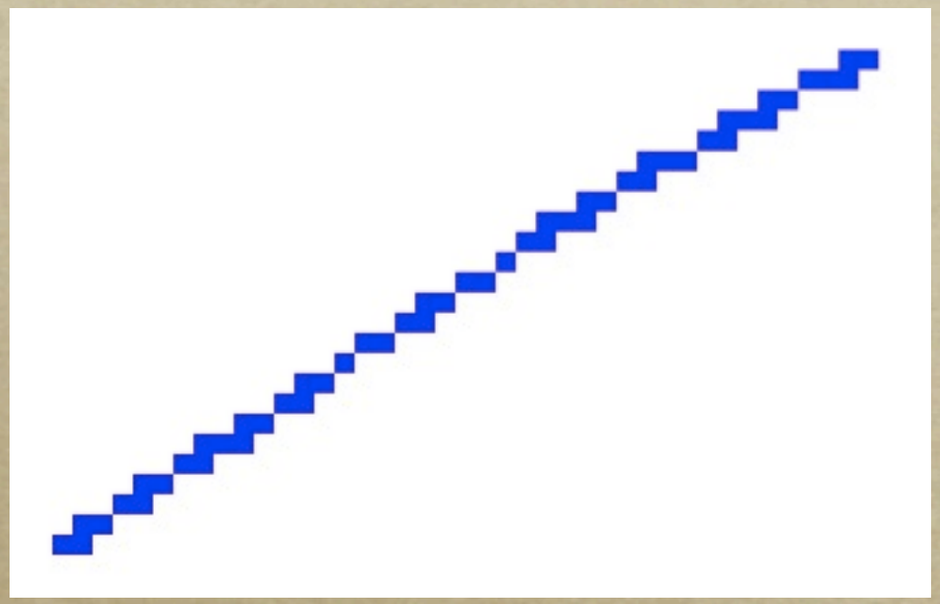


Ambient, Lambert, Phong, reflection, refraction, point light sources.

Just the beginning…

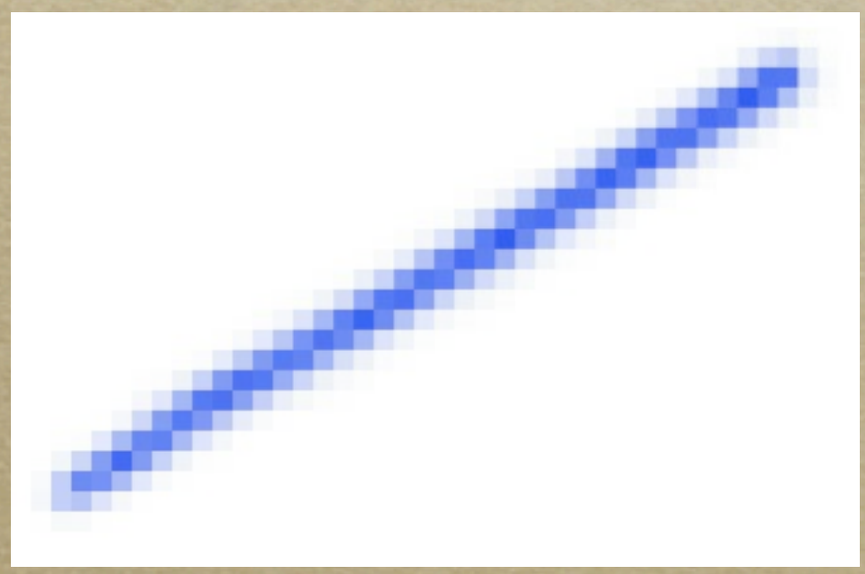
- *Aliasing artefacts*
- *No surface/surface illumination*
- *No caustics*
- *Shadows are hard (i.e. have hard edges)*
- *Colour problems*
- *Very slow*

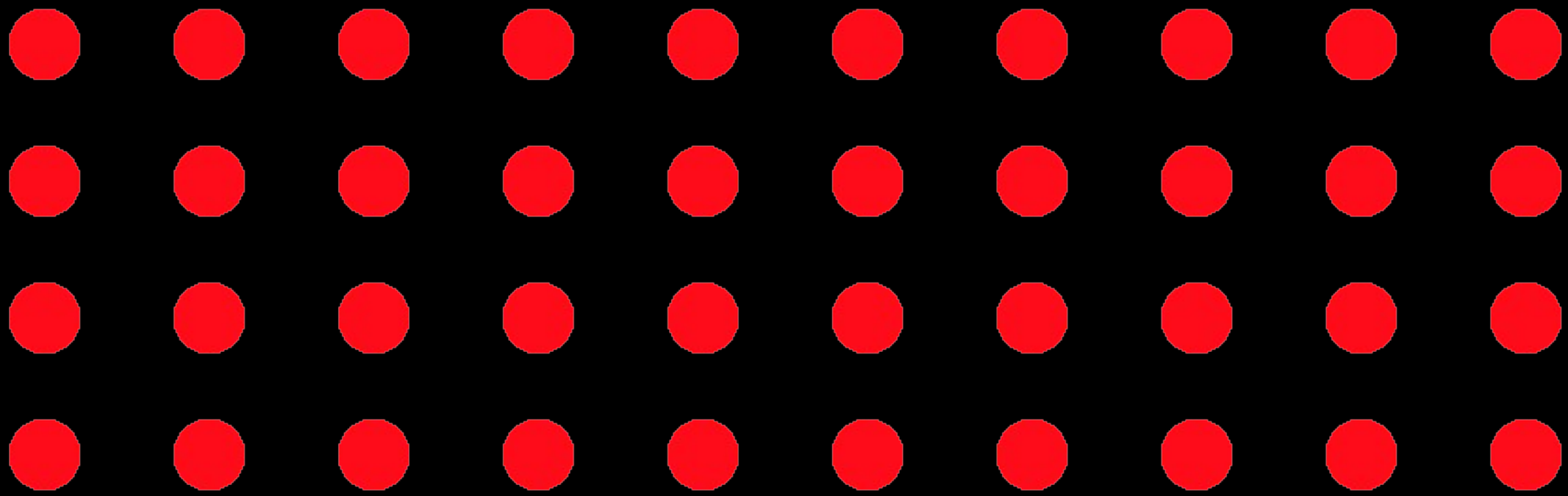
Aliasing?



Jaggies

Antialiased

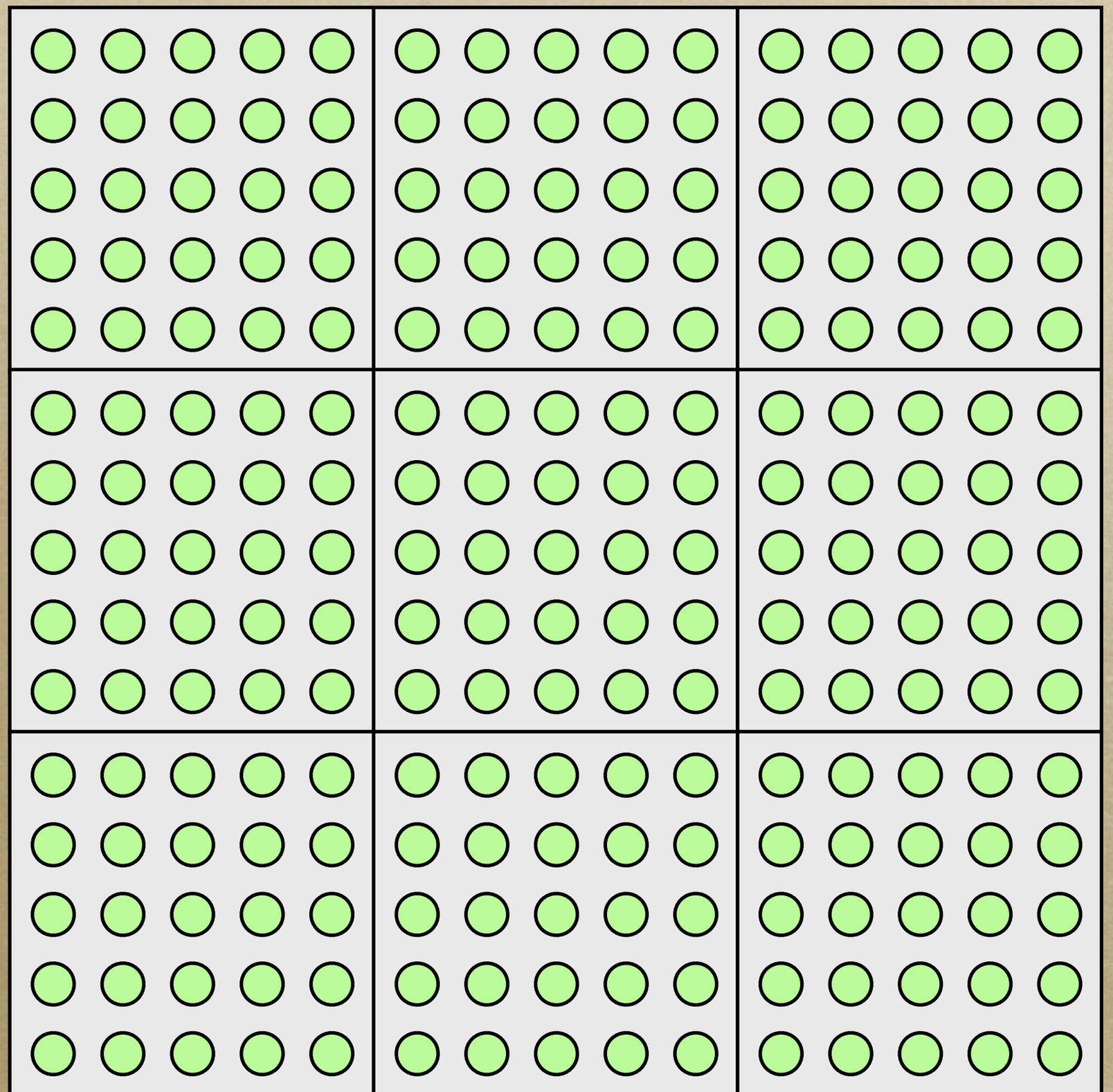




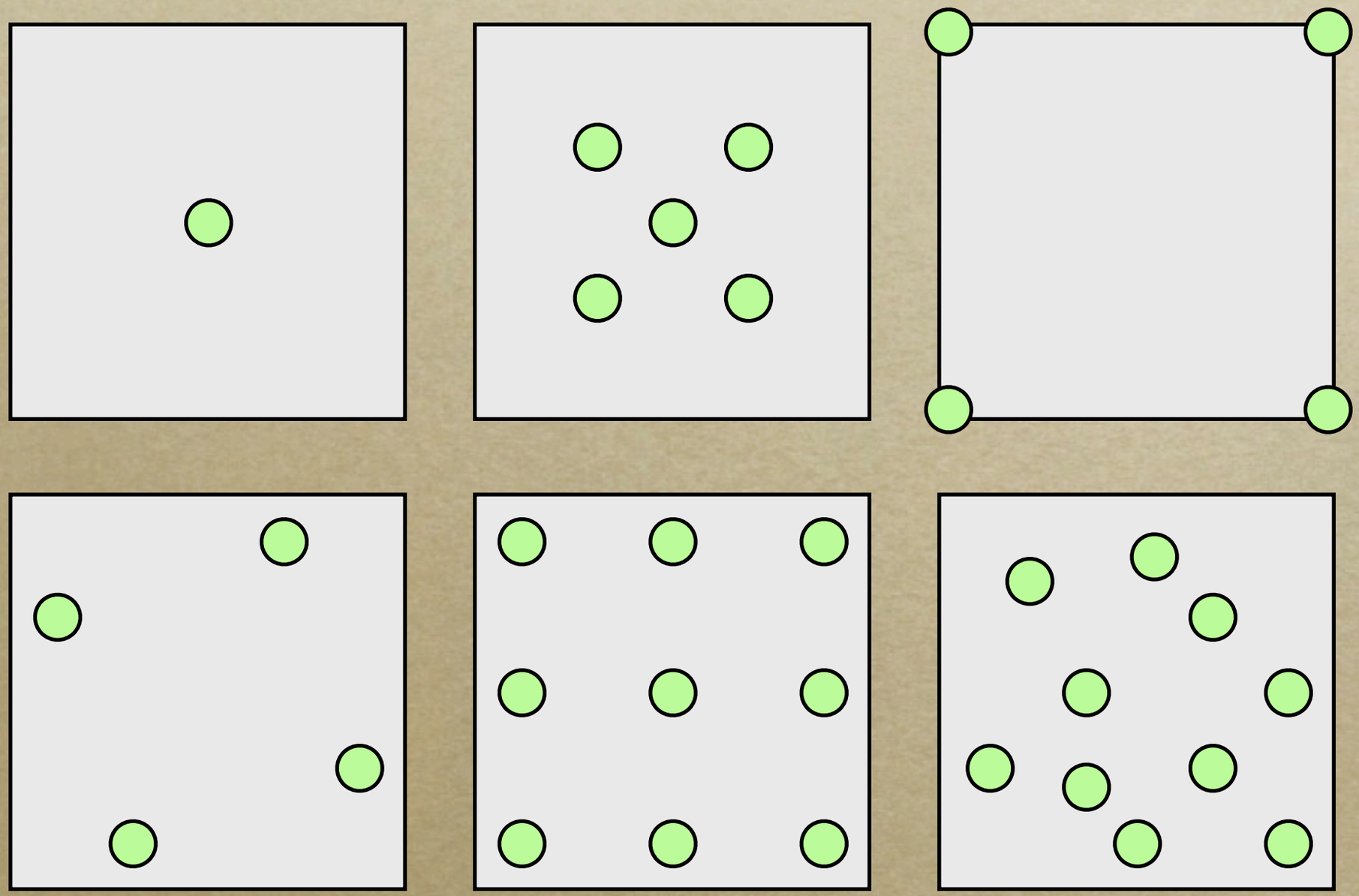
Aliases

- *A given set of samples can represent more than one picture.*
- *The alternatives are called aliases.*
- *Antialiasing is mostly about getting the samples to look like the picture we want.*

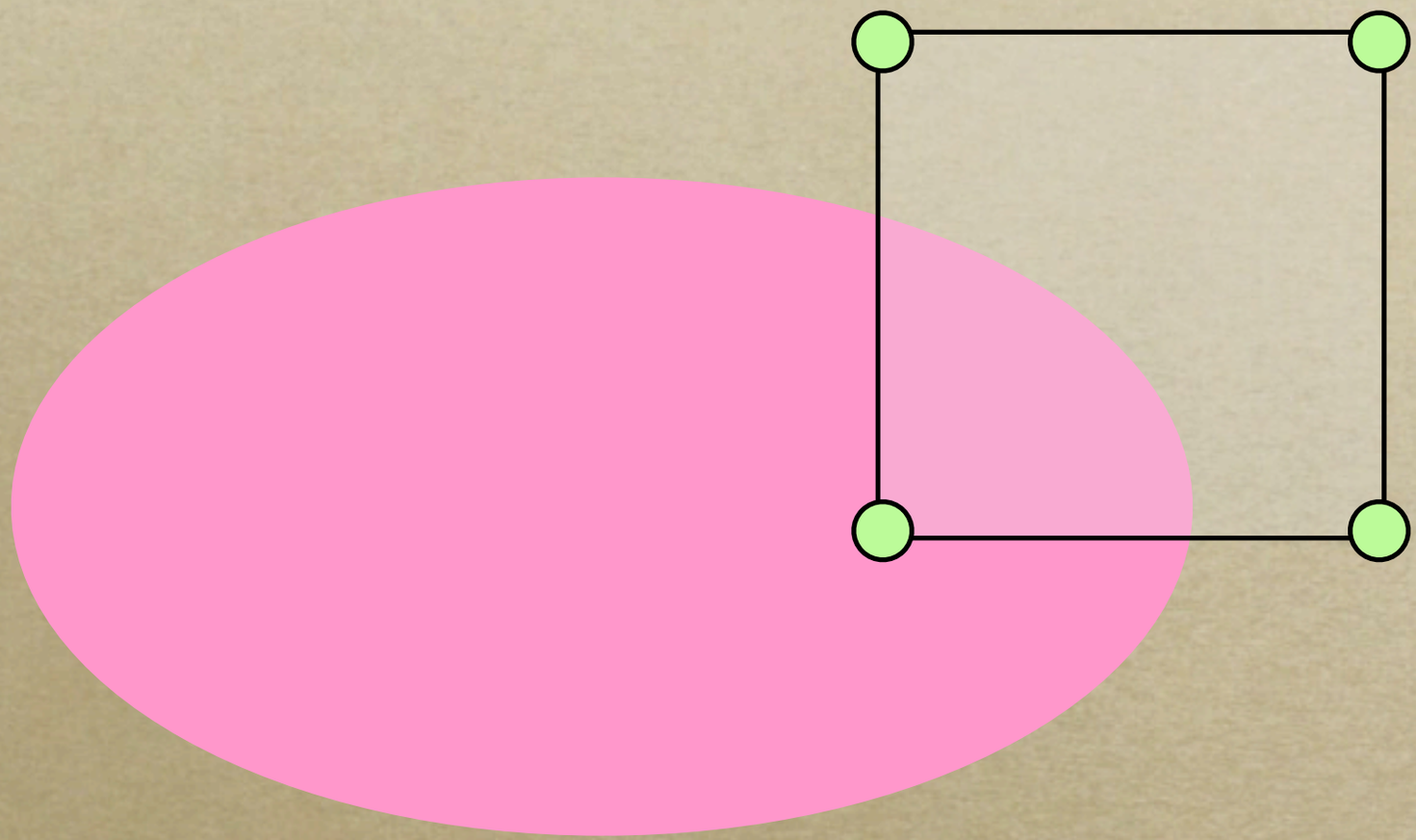
Supersampling



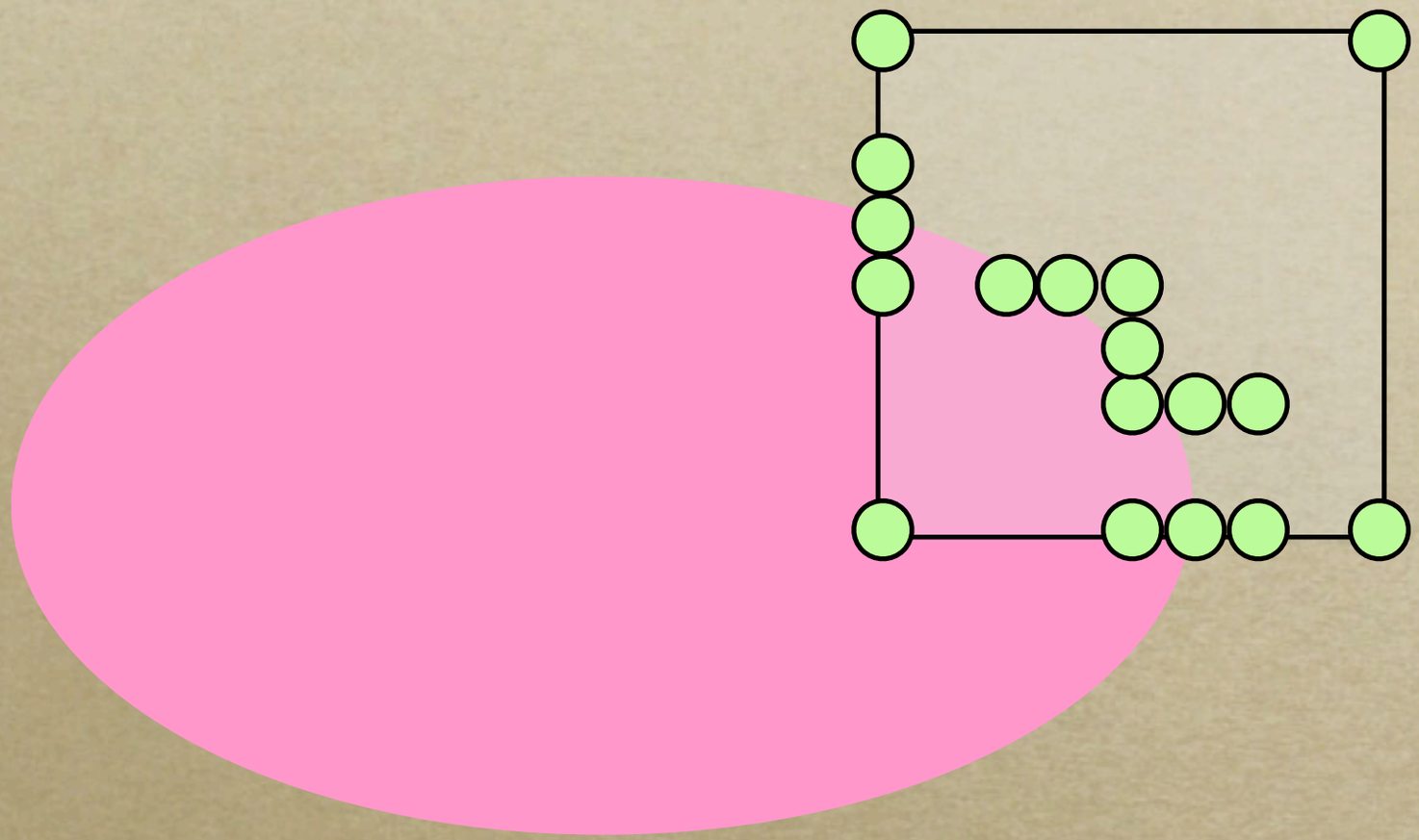
Where to cast the rays



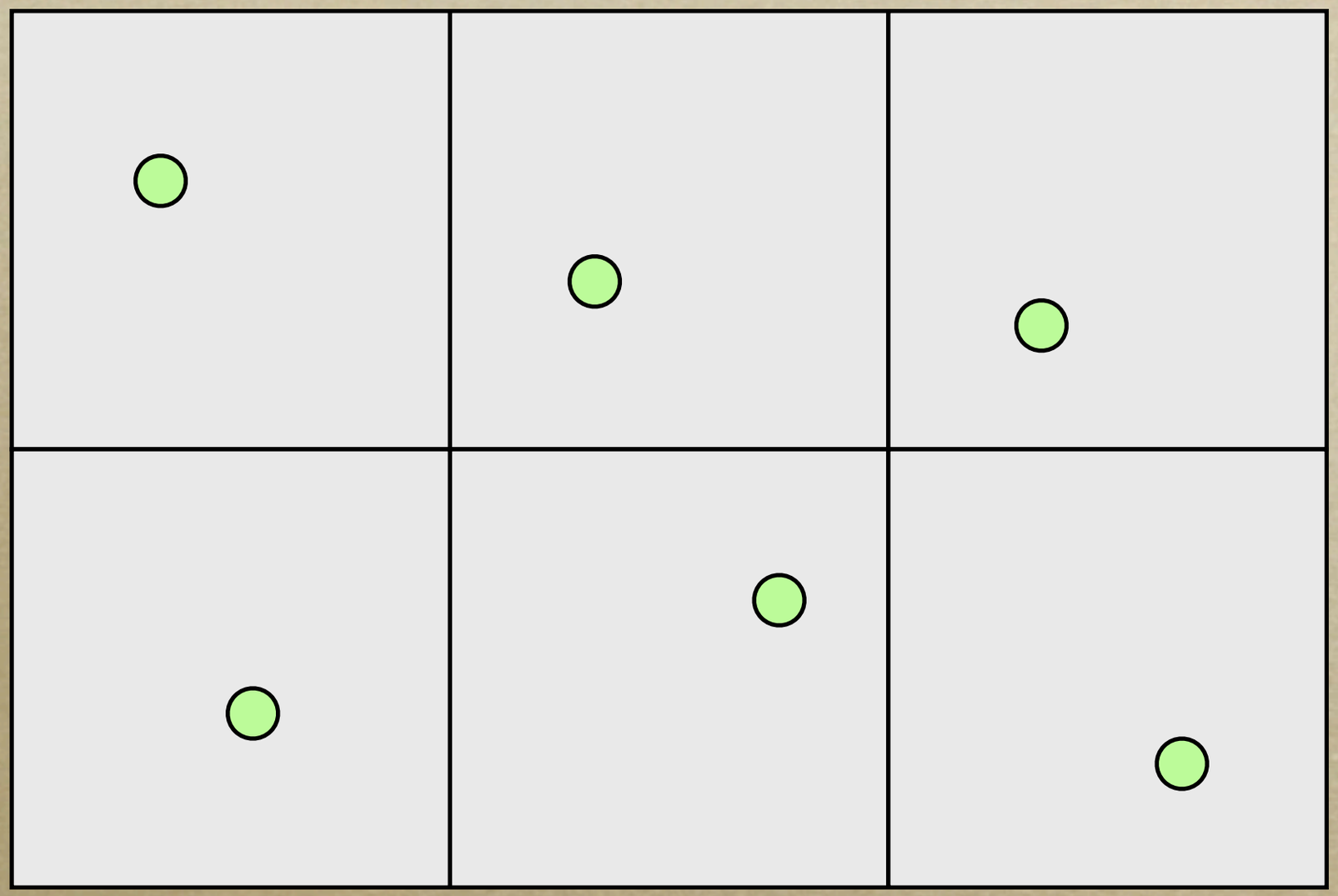
Adaptive Sampling



Adaptive Sampling



Jittering



Even with one sample per pixel this is an improvement on uniform sampling.

Links

http://escience.anu.edu.au/lecture/cg/GlobalIllumination/printNotes.en.html