
Lighting model

© 1999, Denis Zorin

Local lighting model

Describes interaction of the light with the surface.

Almost never truly based on physics: perception plays a greater role.

Visible light: electromagnetic waves, with wavelengths 400nm (violet) - 700nm (red); intensity can vary over many orders of magnitude.

Computer model: only three “frequencies”: RGB, intensity varies over a small range, typically only 255 discrete values/ color.

© 1999, Denis Zorin

Physics vs. graphics

Computer graphics terms are somewhat confusing and disagree with physics:

Graphics: color of an object; physics: reflection spectrum (i.e. fraction of light of each frequency that gets reflected).

Graphics: intensity or color of a light ray; physics: radiance distribution (measured in watts/steradian/meter²/meter)

Graphics: intensity or color of a point light source; physics: intensity spectrum of a point light source (almost in agreement!) measured in watts/steradian/meter.

© 1999, Denis Zorin

Illumination model

Two main components:

- light source characteristics

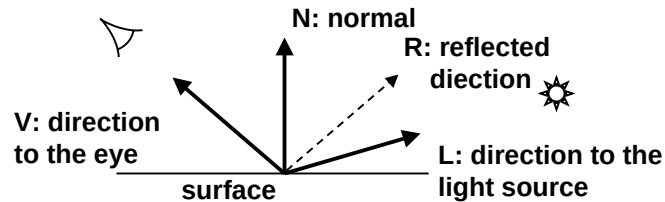
- position
- intensity for each freq. (color)
often, different intensity can be specified for different colors
- directional distribution

- surface properties

- reflectance for each freq. (color)
- different reflectance can be specified for diffuse and specular light

© 1999, Denis Zorin

Reflection geometry



$$R = 2(N, V)N - V$$

© 1999, Denis Zorin

Illumination components

In the model commonly used in graphics applications, there are several components

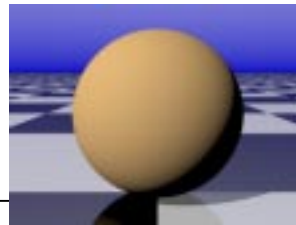
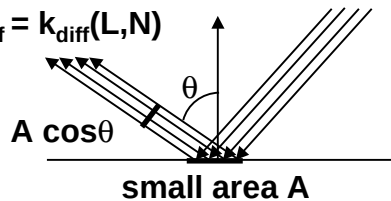
- **diffuse relection:** intensity does not depend on the direction to the viewer
- **specular:** simulates relective surfaces and specular highlights depending on the direction to the viewer
- **ambient:** a crude approximation to the illumination created by the light diffusely reflected from surfaces

© 1999, Denis Zorin

Diffuse component

Diffuse surfaces are surfaces following the Lambert's law: the energy of the light reflected from a surface in a direction D is proportional to the cosine of the angle between the normal and D. As intensity (radiance) is proportional to the energy times cross section of the ray, it does not depend on the view direction, but is proportional to the cosine of the angle between the normal and the direction to the light.

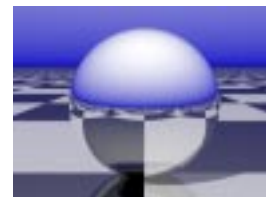
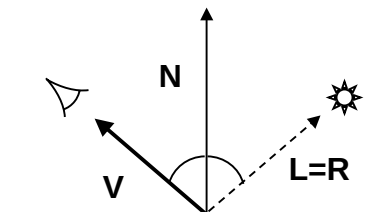
$$L_{\text{diff}} = k_{\text{diff}}(L, N)$$



© 1999, Denis Zorin

Specular component

Specular component approximates behavior of shiny surfaces. If a surface is an ideal mirror, the light from a source reaches the eye bouncing off a fixed point of the surface, only if the direction to the light coincides with the reflected direction to the eye:

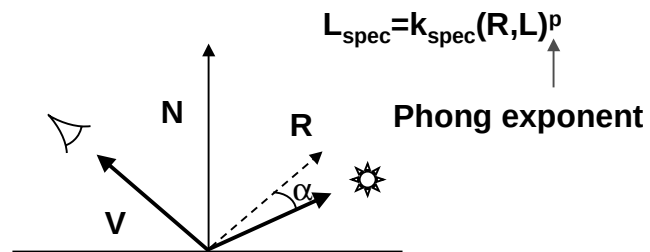


ideal mirror
sphere

© 1999, Denis Zorin

Specular reflection

For non-ideal reflectors, the reflection of the light is still the brightest when $L=R$ but decays, rather than disappears, as the angle between L and R increases. One way to achieve this effect is to use cosine of the angle to scale the reflected intensity:



© 1999, Denis Zorin

Specular reflection



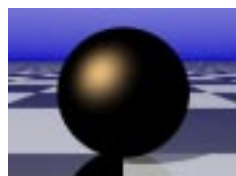
$p = 0$



$p = 1$



$p = 2$



$p = 10$



$p = 25$



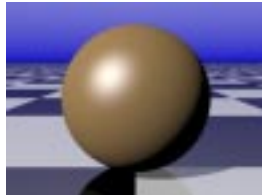
$p = 100$

© 1999, Denis Zorin

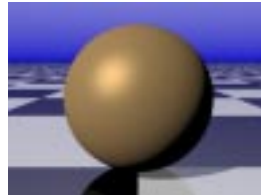
Metal vs. plastic

Natural look of metallic surfaces is difficult to simulate, but the first approximation is obtained using proper highlight color.

For plastic objects, highlights are close in color to the color of the light. For metals, to the color of the surface. Assuming white lights, for plastic set $k_{\text{spec}}=[c,c,c]$, where c is a constant, for more metallic look set $k_{\text{spec}}=k_{\text{diff}}$



plastic look



metallic look

© 1999, Denis Zorin

Ambient component

Not all light illuminating a surface comes from light sources, or reflections of light sources in ideal mirrors; however, the light diffusely reflected from other surfaces is difficult to take into account, especially for real-time rendering. It is approximated by the ambient component: a constant is added to all objects. To have more control over ambient contribution, surfaces can be assigned ambient reflectivity.

$$L_{\text{amb}} = k_{\text{amb}} I_{\text{amb}}$$



© 1999, Denis Zorin

Complete equation

$$I_{\text{total}} = k_{\text{amb}} I_{\text{amb}} + \sum_{\text{all lights}} I_i (k_{\text{diff}}(L_i, N) + k_{\text{spec}}(L_i, R)^p)$$

↑
↑

intensity of
i-th light
 direction to
i-th light

If we are ray tracing for rendering, in summation only visible lights are present, and there are two additional terms: contribution from the reflected ray and transmitted ray. If we are using Z-buffering, then all active light sources are regarded as visible.

© 1999, Denis Zorin

Attenuation

In real life, radiance reaching us from a light source decreases with distance as $1/r^2$ (the stars are much less bright than the sun). However, due to the nature of approximations used in graphics, the inverse-square law typically results in pictures that are too dark; the fix is to allow the programmer to control how fast the decay is. I_i in the formulas is replaced not by I_i/r^2 , as physics suggest, but by

$$\frac{l_i}{a + br + br^2}$$

and the most common choice of constants is

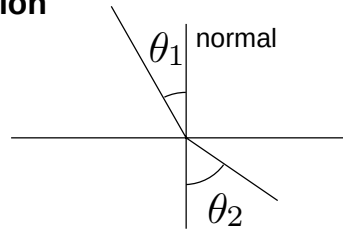
$a = 1, b = c = 0$, that is, no attenuation!

© 1999, Denis Zorin

Snell's law

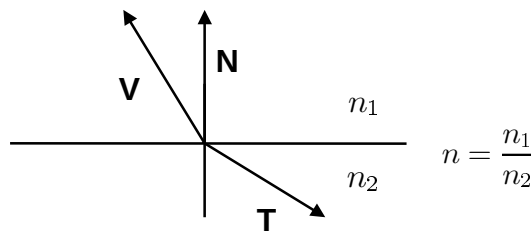
If a surface separates two media with different refraction indices (e.g. air and water) the light rays change direction when they go through.

Snell's law: the refracted ray is stays in the plane spanned by the normal and the direction of the original ray. The angles between the normal and the rays are related by $n_1 \sin \theta_1 = n_2 \sin \theta_2$ where n_1 and n_2 are refraction indices.



© 1999, Denis Zorin

Refracted ray direction



$$\mathbf{T} = -n\mathbf{V} + \left(n(\mathbf{V}, \mathbf{N}) - \sqrt{1 - n^2(1 - (\mathbf{V}, \mathbf{N})^2)} \right) \mathbf{N}$$

© 1999, Denis Zorin

Ray tracing

For each pixel shoot a ray R from camera;
 $\text{pixel} = \text{TraceRay}(R)$

The recursive ray tracing procedure:

```
RGBvalue TraceRay(Ray R)
  shoot rays to all light sources;
  for all visible sources, compute RGB values  $r_i$ 
  shoot reflected ray  $R_{\text{refl}}$ ;  $r_{\text{refl}} = \text{TraceRay}(R_{\text{refl}})$ 
  shoot refracted ray  $R_{\text{trans}}$ ;  $r_{\text{trans}} = \text{TraceRay}(R_{\text{trans}})$ 
  compute resulting RGB value from
   $r_i, r_{\text{refl}}, r_{\text{trans}}$  using the lighting model
```

© 1999, Denis Zorin

Image processing

© 1999, Denis Zorin

How computer images work?

Continuous real image

↓
*Digitization
(e.g. scanning)*

square array of numbers
(abstract pixels)

↓
*each physical pixel
covers an area*

display (physical pixels)

↓
*The eye blurs pixels into
continuous image*

perceived image

© 1999, Denis Zorin

What can go wrong?

Pipeline: sample - process - reconstruct

All kinds of artifacts can appear

- jaggies
- alias patterns
- moire patterns
- temporal aliasing (wheels going wrong way)

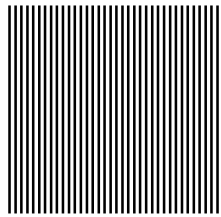
Question: how do we avoid all this?



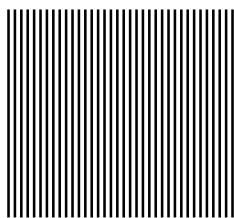
© McMillan '98

© 1999, Denis Zorin

Aliasing



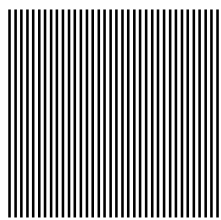
**Slightly different
frequency**



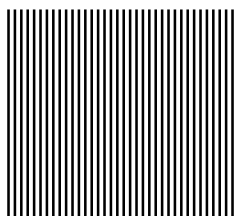
What can go wrong?

© 1999, Denis Zorin

Aliasing



**Slightly different
frequency**

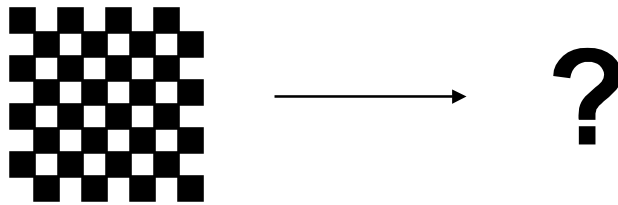


**Lower frequency
appears**

© 1999, Denis Zorin

Shrinking

Naïve 1.5x shrinking : drop 2 out of 3



What do we get?

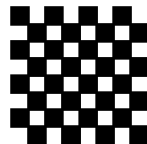
© 1999, Denis Zorin

Shrinking

Naïve 1.5x shrinking: drop 2 out of 3



But we want



(impossible)

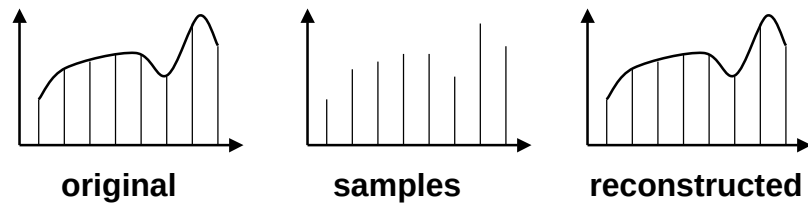
© 1999, Denis Zorin

Frequency analysis

The key to fighting aliasing is to
avoid frequencies we cannot represent

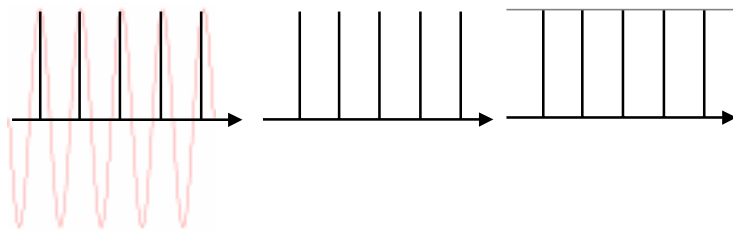
Big question:

When can we reconstruct a continuous
signal from samples?



© 1999, Denis Zorin

Frequency analysis



original

samples

reconstructed

Need enough samples, to be more precise,
sampling frequency should be twice the frequency
of the wave

© 1999, Denis Zorin

Frequency analysis

- decompose everything into waves
- mathematically convenient to use complex waves

$$A \cos(\omega t + \phi)$$

amplitude frequency phase

$$Ae^{i(\omega t + \phi)} = A (\cos(\omega t + \phi) + i \sin(\omega t + \phi))$$

© 1999, Denis Zorin

Fourier transform

Decomposes into freq. components

$$\mathcal{F}[x](\omega) = \int_{-\infty}^{\infty} x(t) e^{-i\omega t} dt$$

Inverse transform reconstructs the function

$$\mathcal{F}^{-1}[X] = \frac{1}{2\pi} \int_{-\infty}^{\infty} X(\omega) e^{i\omega t} d\omega$$

© 1999, Denis Zorin

Examples of FT pairs

FT(delta function) = constant or wave

FT(constant or wave) = delta function

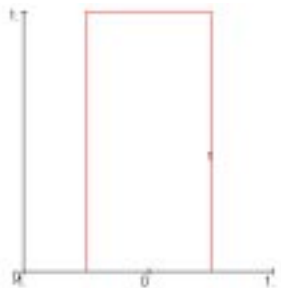
delta function (not quite a function):

for all $x \neq 0$, $\delta(x) = 0$

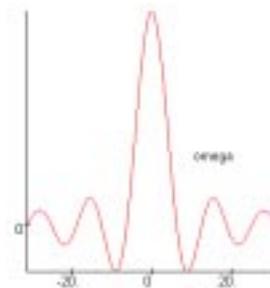
for all ϵ , $\int_{-\epsilon}^{+\epsilon} \delta(x) dx = 1$

© 1999, Denis Zorin

Examples of FT pairs



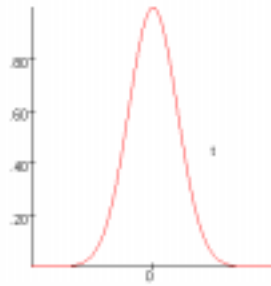
box



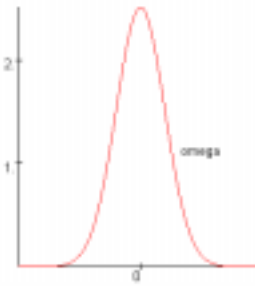
sinc

© 1999, Denis Zorin

Examples of FT pairs



Gaussian

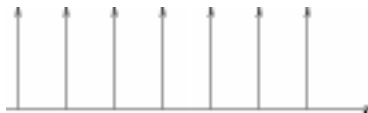


Gaussian

© 1999, Denis Zorin

Comb function

$$comb = \sum_{n=-\infty}^{\infty} \delta(t - nT)$$



FT of a comb is also a comb

© 1999, Denis Zorin

Properties

Name of the property	signals	Fourier transforms
Linearity	$ax(t) + by(t)$	$a\mathcal{F}[x](\omega) + b\mathcal{F}[y](\omega)$
Time shifting (Section 3)	$x(t - t_0)$	$e^{-i\omega t_0} \mathcal{F}[x](\omega)$
Zooming (Section 4.1)	$x(t/a)$	$a\mathcal{F}[x](a\omega)$
Differentiation	$dx(t)/dt$	$i\omega \mathcal{F}[x](\omega)$
Convolution (Section 2.2)	$x(t) * y(t)$	$\mathcal{F}[x](\omega) \mathcal{F}[y](\omega)$

© 1999, Denis Zorin

Shannon theorem

A function can be reconstructed from samples if

- it is bandlimited (Fourier transform is zero for large frequencies)
- sampling frequency is at least twice the max. frequency of the function

© 1999, Denis Zorin

Filtering

solution to aliasing problems: get rid of high frequencies we cannot

reconstruct

In freq. Domain: multiply by a box

what does it mean in the spatial domain?

© 1999, Denis Zorin

Convolution

multiplication in the frequency domain is convolution in the spatial domain

$$U(\omega) = X(\omega)Y(\omega)$$

$$u(t) = \int_{s=-\infty}^{+\infty} x(s)y(t-s)ds$$

© 1999, Denis Zorin