Monte Carlo Ray Tracing V
Metropolis Algorithm and Photon Mapping

April 8, 2005

- Program 10 (monte carlo) questions?
- Internship opportunity

Path tracing wrapup

- Path tracing can handle arbitrary L(D|S)*E light paths
- But, it is SLOW
- Thousands of samples are required to eliminate noise

Importance Sampling Clarification

- Importance sampling is a change of variables in the integration
- Consider the one dimensional integral:

$$I = \int_a^b f(x)dx$$

- Monte Carlo techniques can be used to compute this integral as:

$$I \simeq \frac{1}{N}\frac{1}{(b-a)}\sum f(\xi_i)$$

ξ_i : uniformly distributed random variable in range [a,b]

Importance Sampling Clarification

- We can rewrite the integral as:

$$I = \int_a^b \frac{f(x)}{g(x)} g(x)\,dx$$

- And then perform a change of variables:

$$\text{Let } dy = g(x)\,dx$$

$$I = \int_A^B \frac{f(x(y))}{g(x(y))} dy$$

Importance Sampling Clarification

For a uniformly sampled distribution:

$$I \simeq \frac{1}{N}\frac{1}{(b-a)}\sum f(\xi_i)$$

ξ_i : uniformly distributed random variable in range [a,b]

More generally:

$$I \simeq \frac{1}{N}\frac{1}{(b-a)}\sum \frac{f(\xi_i)}{p(\xi_i)}$$

ξ_i : random variable in range [a,b]

$p(\xi_i)$: probability that distribution will produce value ξ_i

Importance Sampling Clarification

Note that this:

$$I \simeq \frac{1}{N}\frac{1}{(b-a)}\sum\frac{f(\xi_i)}{p(\xi_i)}$$

looks a whole lot like this:

$$I = \int_A^B \frac{f(x(y))}{g(x(y))}dy$$

if g(x) = p(x)

∴ Use a random variable with a probability density that approximates f(x) to minimize variance

Importance Sampling Intuition

- Monte Carlo techniques work very well at integrating straight lines (the average of a constant is always a constant)
- Choose f(x)/p(x) that approximates a straight line
- If could do that exactly we wouldn’t need to integrate!
- But the closer we can make it, the better

Importance Sampling Example

Suppose $f(x) = h(x)\cos x$

We believe that $f(x) \simeq \cos x$

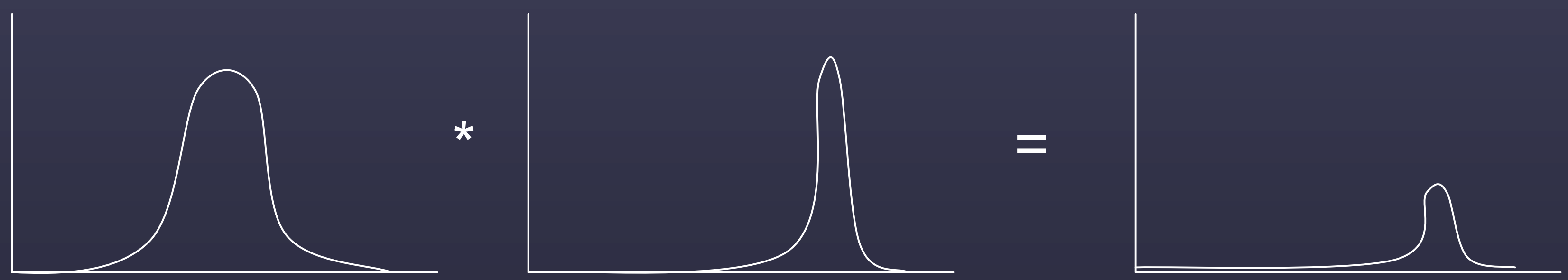
$g(x) = \cos x$

Integrate $\dfrac{f(x)}{g(x)} = h(x)$

with probability distribution cos x

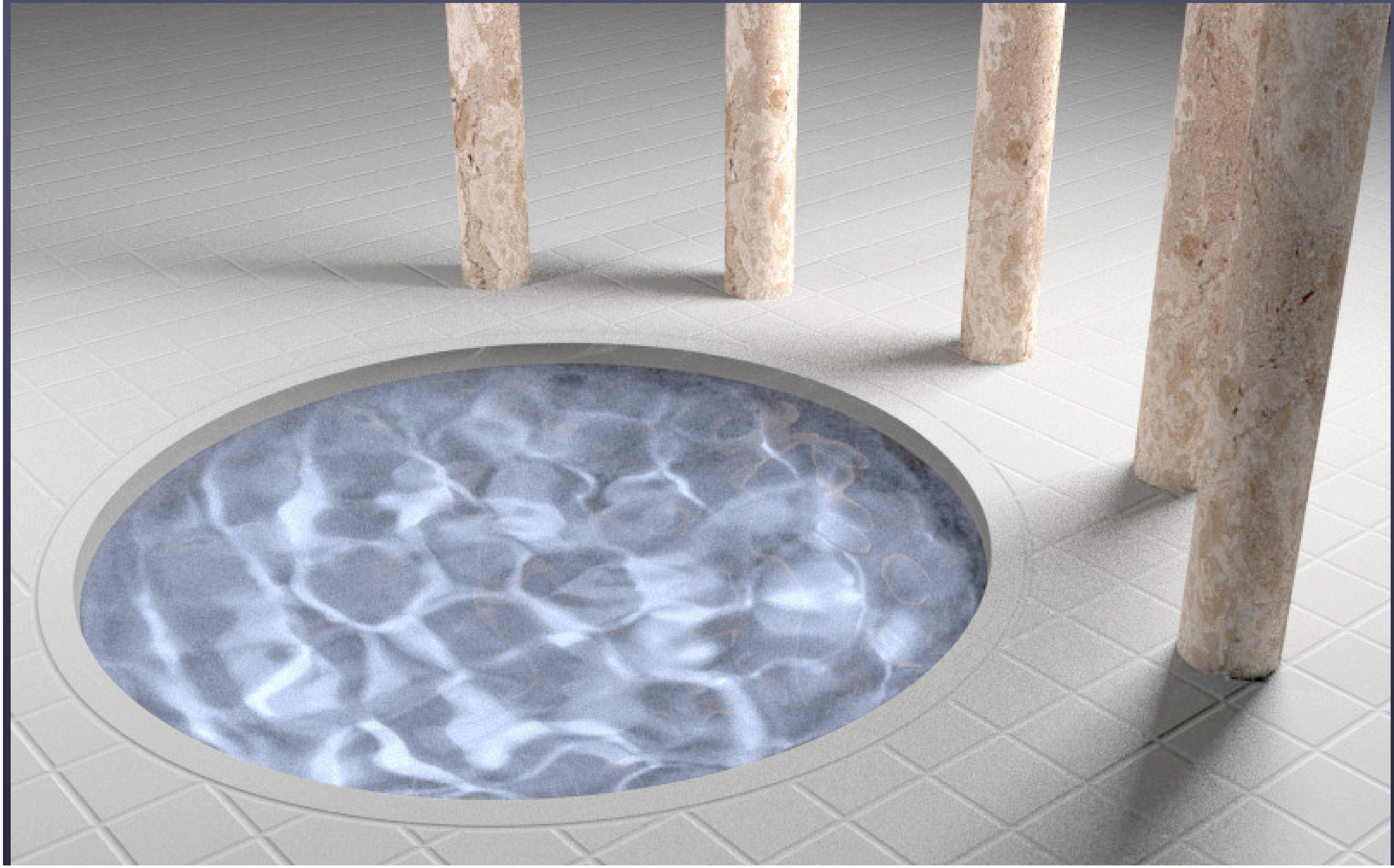
Importance sampling pros/cons

- Importance sampling works by reducing the variance of the sample set
- However, it only works if f(x) ≈ g(x)
- It CAN make it worse if g(x) is a poor estimate!



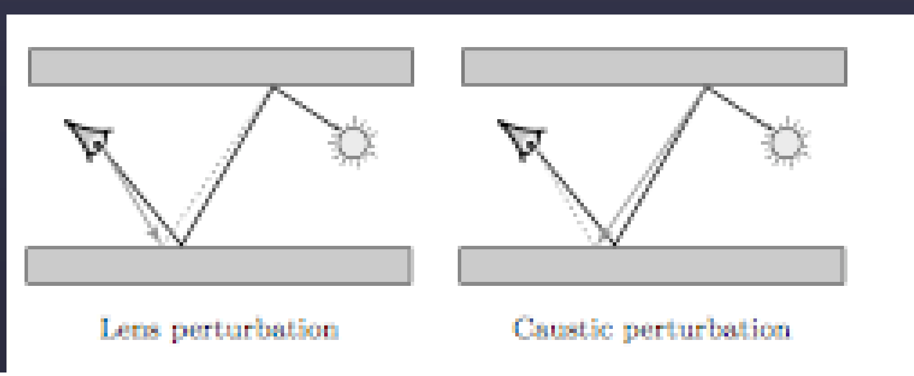
Estimate g(x)   Estimate g(x)

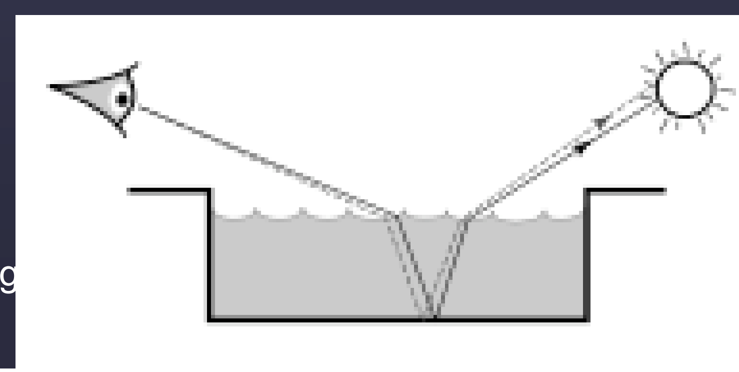
Metropolis algorithm



Metropolis algorithm

- Pure path tracing has a hard time “finding” all of the S* paths
- Small samples may contain a lot of energy (caustics, strong indirect lights)
- Solution:
  - Use a path tracer to find a path to a light source
  - Mutate that path to find other nearby paths
  - Plenty of math to avoid biasing result





Metropolis algorithm

- Math: Rosenbluth/Rosenbluth/Teller (1953)
- Graphics: Veach 1997

http://graphics.stanford.edu/papers/veach_thesis/

Math

- Consider a one-dimensional integral
- Evaluate at a random point
- Propose a nearby point for inclusion
  - Common: use a gaussian distribution
  - Several other more complex models
  - Evaluate f(x)
- Accept or reject the new point based on another random variable:
  - If new value is large: more likely to accept it
  - If old value was large: less likely to accept new one

Equations and demo

X_i : current point

Y : proposed new point

probability of accepting:

$$\alpha\left(X_i, Y\right) = \min\left(1, \frac{f\left(Y\right)}{f\left(X_i\right)}\right)$$

(a tiny bit more complicated for complex walking criteria)

if $\xi < \alpha\left(X_i, Y\right)$

  keep sample $X_{i+1} = Y$

else

  try again

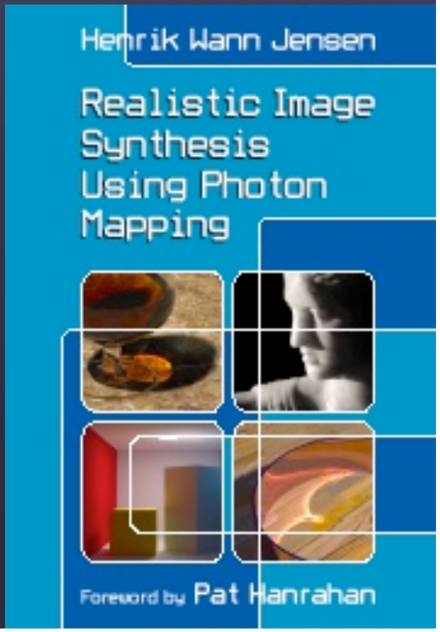
http://www.lbreyer.com/classic.html

Metropolis summary

- Focus samples on important paths
- Not all work contributes to final image (throw away about half of the candidate samples)
- Called Markov Chain or Random Walk algorithm
- Details for graphics are quite complex, must obey properties:
  - Ergodic: Must be able to reach all states via some mutation of the original path
  - Detailed balance: accept/reject ratio leads to correct integral

Photon mapping

- Monte Carlo spends a lot of time computing similar results
- Indirect illumination smooth (except caustics)
- Ward algorithm (Radiance): Cache irradiance values on surfaces
- Photon mapping: Reverse ray tracing to deposit photons on surfaces



Pass 1: Emission

- Send out photons from the light sources
- Deposit energy onto surfaces as they bounce around
- Store photons in a kd-tree
  - Store position, color, incoming direction

- Enhancements:
  - Store “shadow photons” to accelerate shadow queries
  - Two separate maps: one for caustics (high sample densities needed), one for indirect illumination

Pass 2: Reconstruction

- Trace rays from the eye
- Compute direct light the usual way
- Compute indirect light by interpolating nearest samples from the photon map

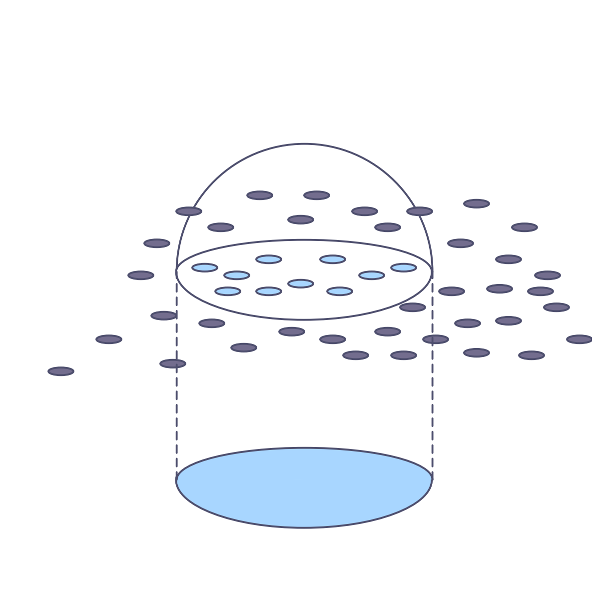


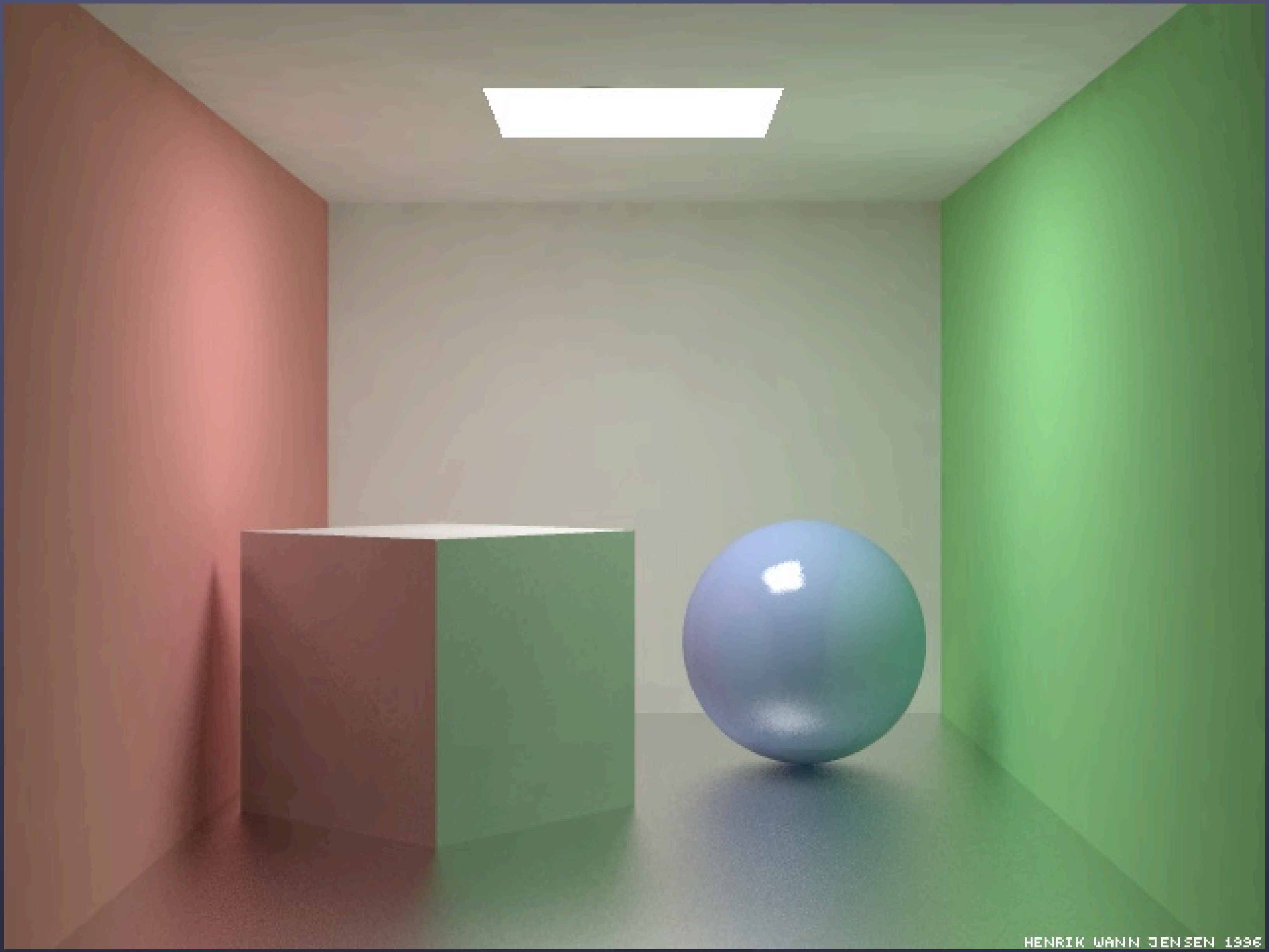
Diagram from John Hart

Photon map

Final image

C

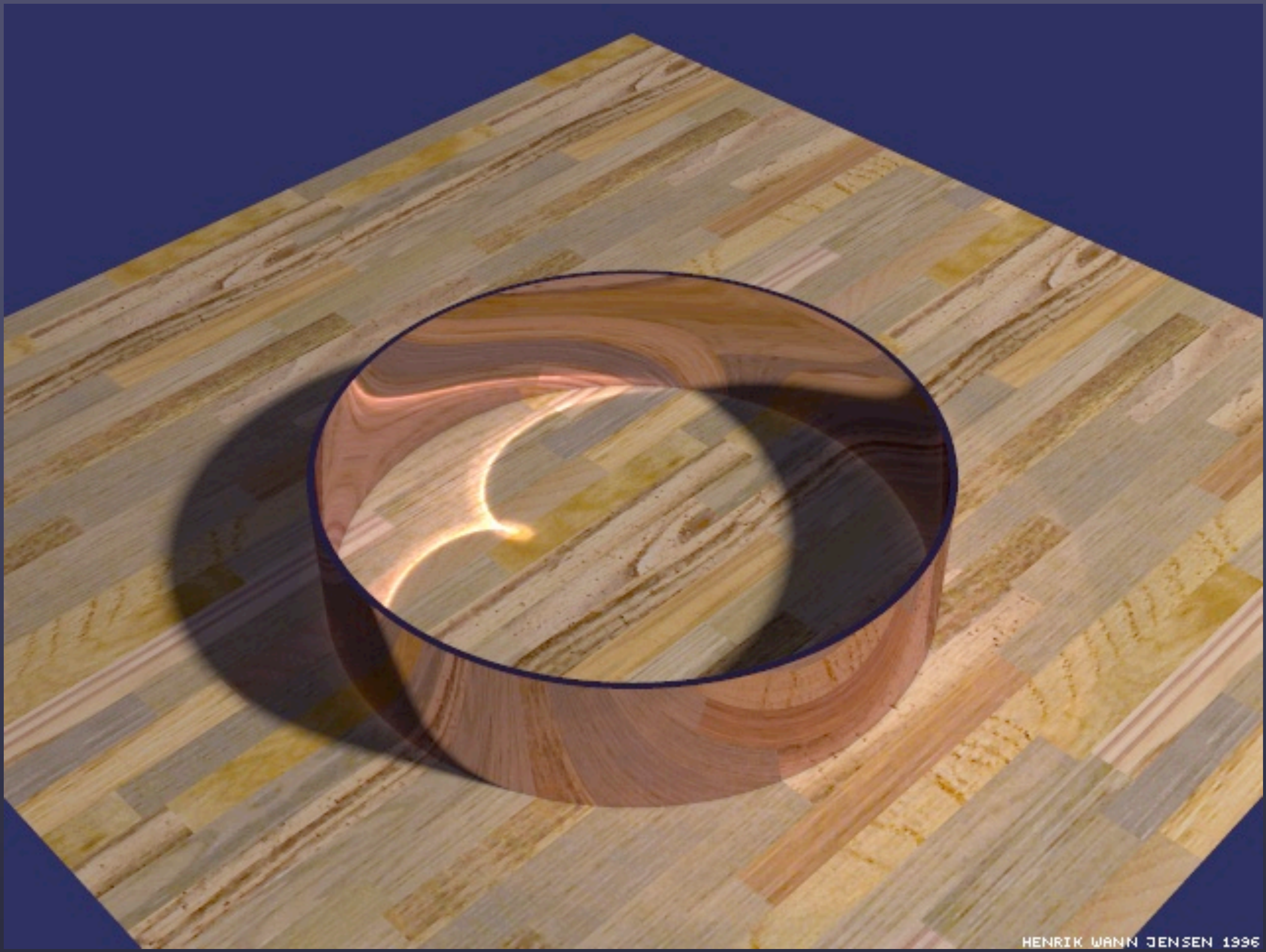
http://www.seanet.com/~myandper/gallery.htm



CS6620 Spring 05

http://graphics.ucsd.edu/~henrik/papers/ewr7/

http://www.cs.kuleuven.ac.be/cwis/research/graphics/RENDERPARK/


HENRIK WANN JENSEN 1996


RENDERED USING DALI - HENRIK WANN JENSEN 2000

Upcoming lectures

- 8 class periods left:
  - Visualization symposium (Monday)
  - Performance programming/tuning
  - Manta architecture
  - Color theory
  - Advanced intersections (CSG, extrusions, etc.)
  - Contest/wrapup
  - Other topics