Real-Time Global Illumination for Dynamic Scenes

Introduction

Carsten Dachsbacher
Visualization Research Center
University of Stuttgart

Jan Kautz
University College London

VECG
Virtual Environments and Computer Graphics

Introduction

We see here an example of a real-world scene which has a lot of visual complexity and richness. Generating synthetic images that come close to this is an extremely challenging problem.

We will discuss some of the issues that need to be addressed to meet this challenge.

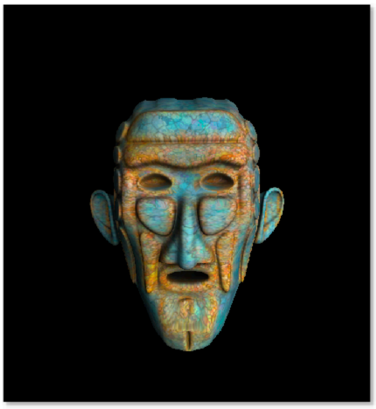
How do we get there?

- Geometric Complexity
- Material Complexity
- Lighting Complexity
- Transport Complexity
- Synergy

There are many types of scene complexity which operate individually and in synergy with each other to generate the visual complexity of the resulting image. We will concentrate on Transport Complexity.

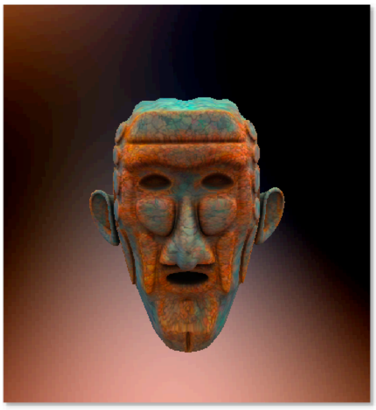
Lighting Complexity

- What kind of lighting environment is an object in?
  - Directional/point lights
  - “Smooth” (low frequency) lighting
  - Environment Map



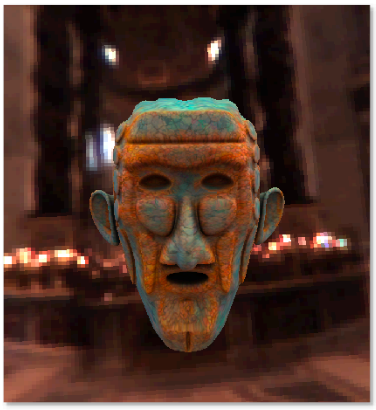
Lighting Complexity

- What kind of lighting environment is an object in?
  - Directional/point lights
  - "Smooth" (low frequency) lighting
  - Environment Map



Lighting Complexity

- What kind of lighting environment is an object in?
  - Directional/point lights
  - "Smooth" (low frequency) lighting
  - Environment Map

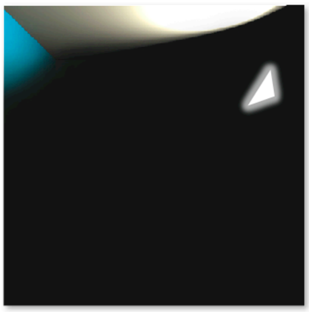


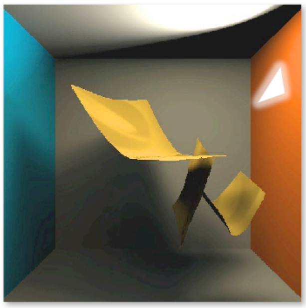
Transport Complexity

- How light interacts with objects/scene
  - Shadows
  - Inter-reflections
  - Caustics

Transport Complexity

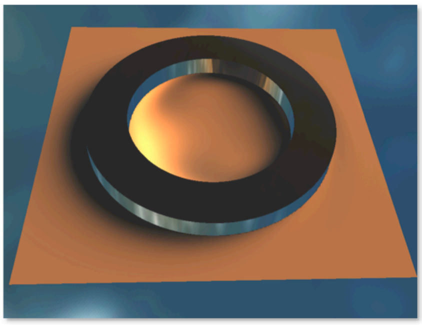
- How light interacts with objects/scene
  - Shadows
  - Inter-reflections
  - Caustics

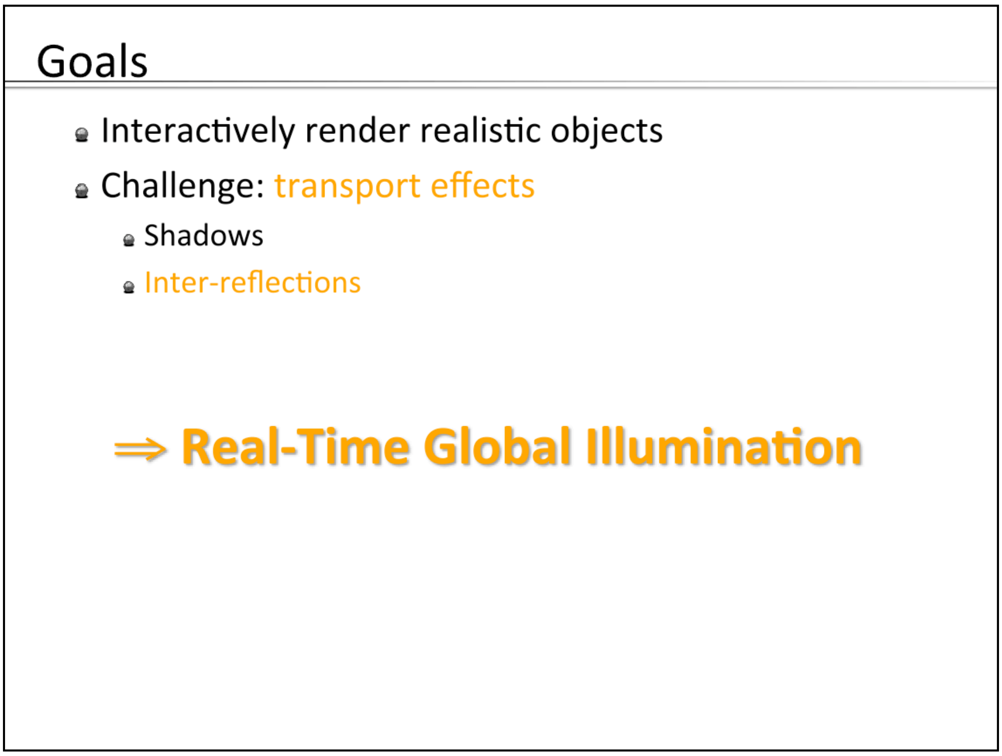




Transport Complexity

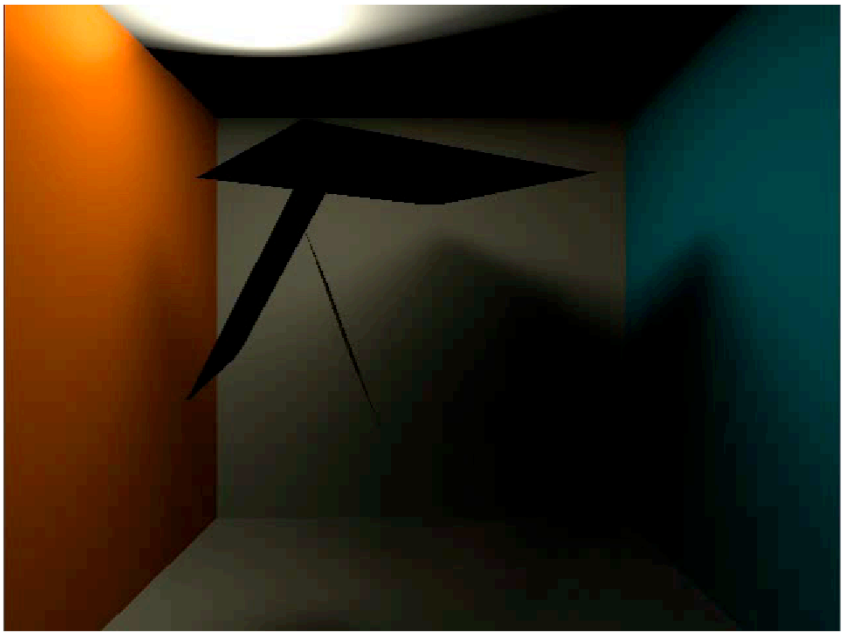
- How light interacts with objects/scene
  - Shadows
  - Inter-reflections
  - Caustics





The primary goal of this course is to accurately render scenes under global illumination at interactive rates.

Example



A first example of a scene under global illumination.

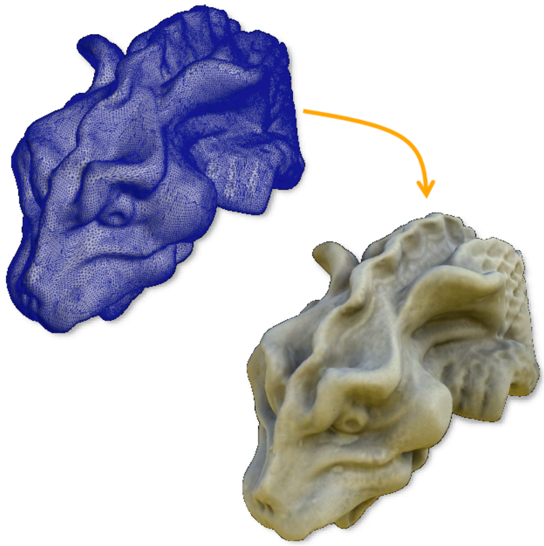
Overview

- Introduction
  - Rendering
  - Rendering Equation
  - Neumann Series
  - Shading Algorithms

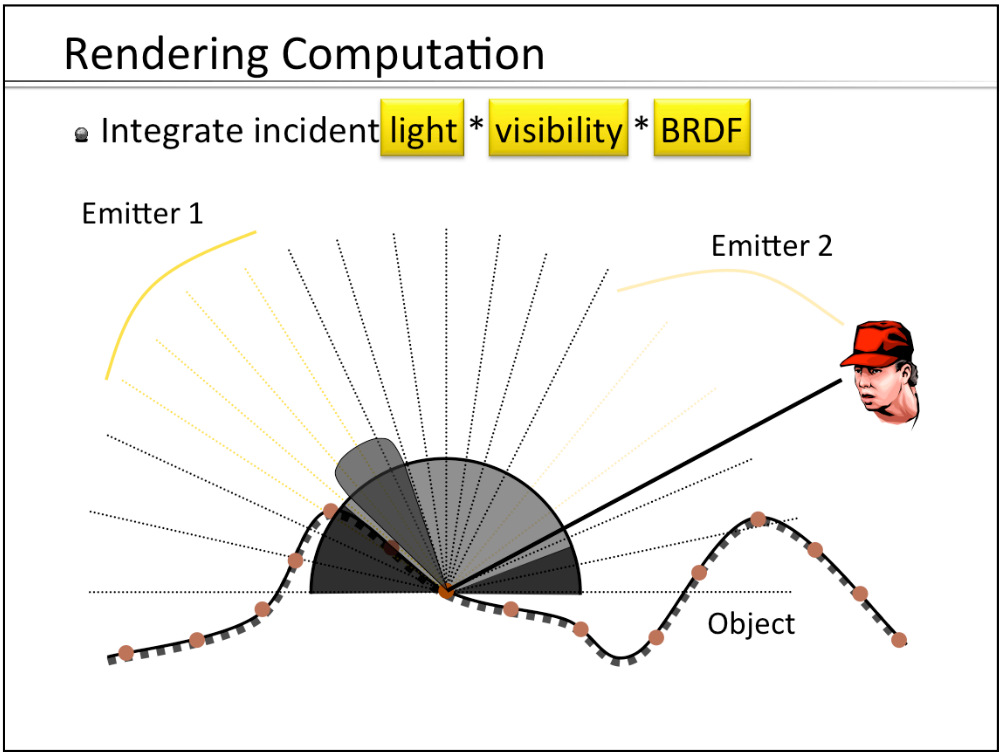
Rendering

- Input:
  - Geometry
  - Material
  - Lighting

- Rendering:
  - How much light is reflected from each point to the viewer?

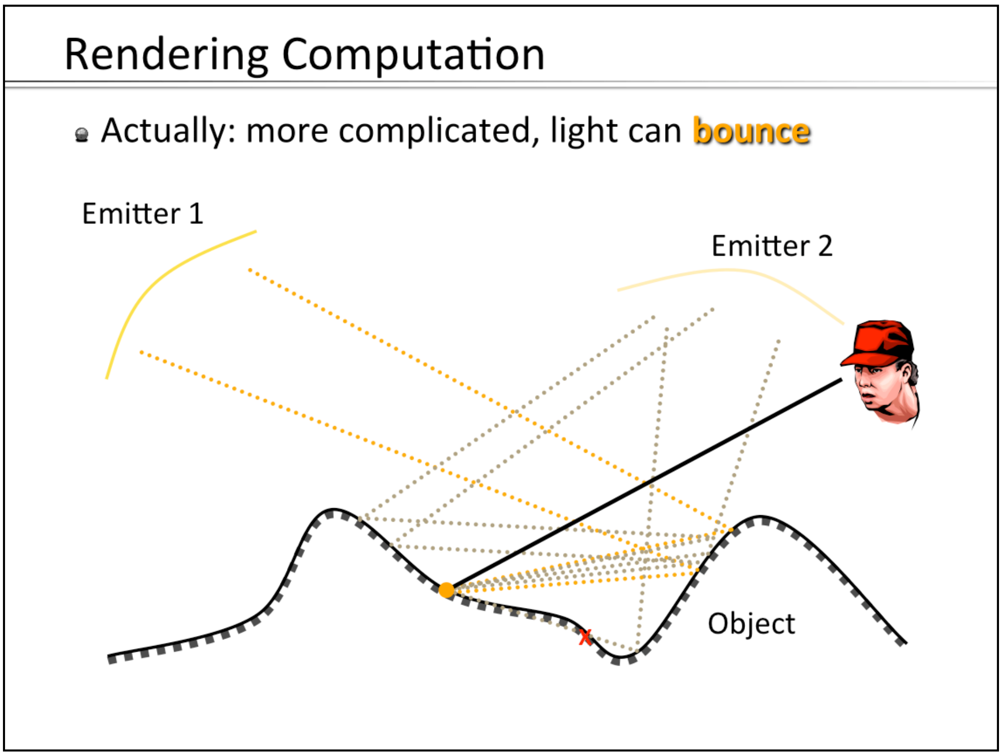


Loosely put, rendering takes input (geometry, materials, lights) and produces an image.



The way we do this is as follows.

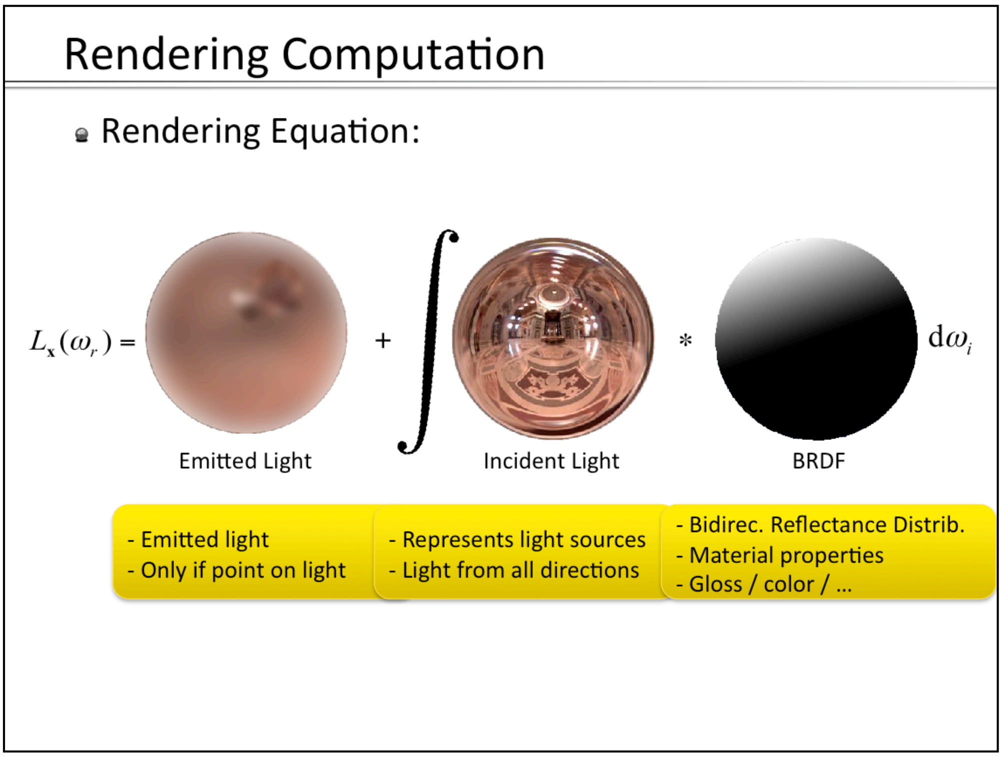
Given an object, two emitters, and a point on the object, we integrate …



Actually, it’s a bit more complicated, because light can bounce.

For instance it can bounce **ones** before it reaches the point,
or can it can bounce twice, before it is reflected towards the viewer.

Of course, we need to make sure that bounced light can actually reach the point and isn’t occluded as in this case here.



A bit more formal, the amount of reflected light at a point **x** in direction omega_o is the emitted light at point **x**, plus the reflected light at **x**.

The amount of reflected light is the integral of all (visible) incident light weighted by the BRDF.

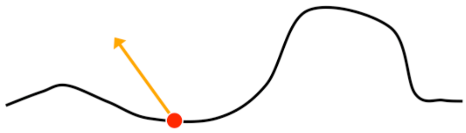
Rendering Equation

$$L(\mathbf{x},\omega_r) = L_e(\mathbf{x},\omega_r) + \int_{\Omega^+} f_r(\omega_i,\mathbf{x},\omega_r) L_i(\mathbf{x},\omega_i)\cos\theta_i \, d\omega_i$$

Given an object illuminated in a lighting environment, the rendering equation models the equilibrium of the flow of light in the scene. It can be used to determine how light a visible point reflects towards the viewer. We will walk through a hemispherical formulation of this equation.

Rendering Equation

$$L(\mathbf{x},\omega_r) = L_e(\mathbf{x},\omega_r) + \int_{\Omega^+} f_r(\omega_i,\mathbf{x},\omega_r) L_i(\mathbf{x},\omega_i)\cos\theta_i \, d\omega_i$$



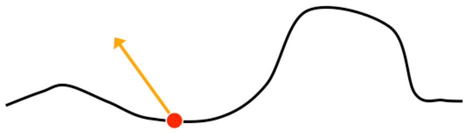
Radiance leaving point $\mathbf{x}$ in direction ω_r

The desired quantity is the radiance leaving a point on the object **x** in a given direction w_r.

Radiance is the intensity of light from a point to a certain direction.

Rendering Equation

$$L(\mathbf{x},\omega_r) = L_e(\mathbf{x},\omega_r) + \int_{\Omega^+} f_r(\omega_i,\mathbf{x},\omega_r)L_i(\mathbf{x},\omega_i)\cos\theta_i\, d\omega_i$$

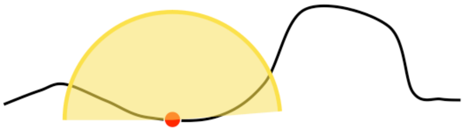


Radiance emitted from point $\mathbf{x}$ in direction ω_r

The first term is the radiance emitted directly from the point in the given direction.

Rendering Equation

$$L(\mathbf{x},\omega_r) = L_e(\mathbf{x},\omega_r) + \int_{\Omega^+} f_r(\omega_i,\mathbf{x},\omega_r) L_i(\mathbf{x},\omega_i)\cos\theta_i \, d\omega_i$$

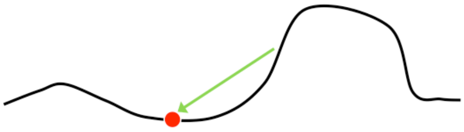


Integral over directions ω_i on the hemisphere around $\mathbf{x}$

This is followed by an integral over the hemisphere around the point, where w_i is used to denote a direction on this hemisphere

Rendering Equation

$$L(\mathbf{x},\omega_r) = L_e(\mathbf{x},\omega_r) + \int_{\Omega^+} f_r(\omega_i,\mathbf{x},\omega_r) L_i(\mathbf{x},\omega_i) \cos\theta_i \, d\omega_i$$

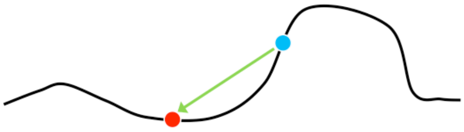


Radiance arriving at point $\mathbf{x}$ from direction ω_i (also LHS)

The second term is the radiance arriving at point **x** from the direction w_i, note that this is also the variable we are solving for so this is an integral equation.

Rendering Equation

$$L(\mathbf{x},\omega_r) = L_e(\mathbf{x},\omega_r) + \int_{\Omega^+} f_r(\omega_i,\mathbf{x},\omega_r) L_i(\mathbf{x},\omega_i) \cos\theta_i \, d\omega_i$$

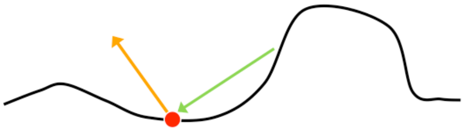


Radiance arriving at point $\mathbf{x}$ from direction ω_i (also LHS)
Note visibility is implicit in $L_i(\mathbf{x},\omega_i)$. It's the closest point $\mathbf{y}$ that emits/reflects along ω_i (i.e., $L_i(\mathbf{x},\omega_i) = L(\mathbf{y},-\omega_i)$)

Also note, that visibility is implicitly defined in L_i(). It is the light arriving along direction w_i, which refers to the closest point **y** along w_i, that emits/reflects light towards **x**.

Rendering Equation

$$L(\mathbf{x},\omega_r) = L_e(\mathbf{x},\omega_r) + \int_{\Omega^+} f_r(\omega_i,\mathbf{x},\omega_r) L_i(\mathbf{x},\omega_i)\cos\theta_i \, d\omega_i$$



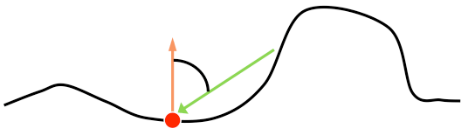
BRDF at point $\mathbf{x}$ evaluated for incident direction ω_i in outgoing direction ω_r

BRDF: defines look of material.

The 1st factor inside the integral is the BRDF of the surface at point **x**, the BRDF is a 4D function that models what percent of light for some input direction w_i leaves in some outgoing direction w_r.

Rendering Equation

$$L(\mathbf{x},\omega_r) = L_e(\mathbf{x},\omega_r) + \int_{\Omega^+} f_r(\omega_i,\mathbf{x},\omega_r) L_i(\mathbf{x},\omega_i) \cos\theta_i \, d\omega_i$$



Lamberts law – cosine between normal and ω_i → dot($\mathbf{n}$, ω_i)

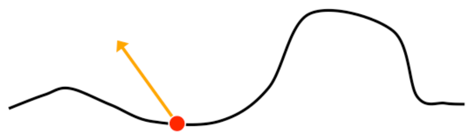
The final term is the cosine term that comes from lamberts law – due to projected area.

Overview

- Introduction
  - Rendering
  - Rendering Equation
  - Neumann Series
  - Shading Algorithms

Neumann Expansion

$$L(\mathbf{x},\omega_r) = L_0(\mathbf{x},\omega_r) + L_1(\mathbf{x},\omega_r) + \ldots$$

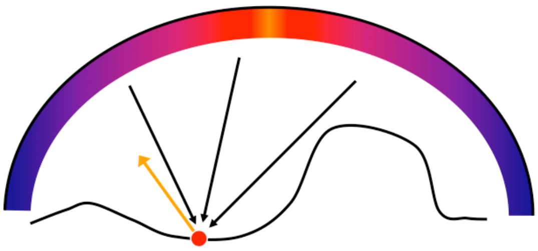


Outgoing radiance expressed as infinite series

One convenient way to reason about the solution to this integral equation is by using a Neumann expansion of this expression, where outgoing radiance is expressed as an infinite series.

Neumann Expansion

$$L(\mathbf{x},\omega_r) = L_0(\mathbf{x},\omega_r) + L_1(\mathbf{x},\omega_r) + \ldots$$

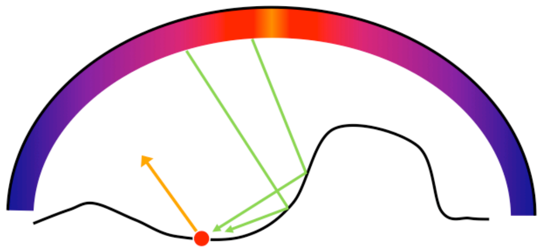


Direct lighting arriving at point **x**

The first term in this series is the direct lighting arriving at point **x**.

Neumann Expansion

$$L(\mathbf{x},\omega_r) = L_0(\mathbf{x},\omega_r) + L_1(\mathbf{x},\omega_r) + \ldots$$



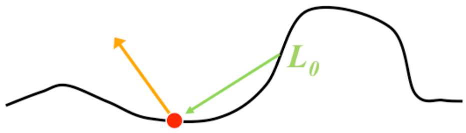
All paths from source that take 1 bounce

The next term in the expansion models all paths from the source radiance function that reach the given point after a single bounce and contribute to outgoing radiance in the given direction.

Neumann Expansion

$$L(\mathbf{x},\omega_r) = L_0(\mathbf{x},\omega_r) + L_1(\mathbf{x},\omega_r) + \ldots$$

$$L_1(\mathbf{x},\omega_r) = \int_{\Omega^+} f_r(\omega_i,\mathbf{x},\omega_r) L_0(\mathbf{x},\omega_i) \cos\theta_i \, d\omega_i$$



All paths from source that take 1 bounce

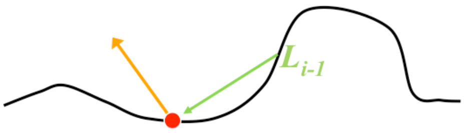
This is also just a conventional integral where the previous term (L_0) is inside of the integral.

As before, visibility is implicit in L_0().

Neumann Expansion

$$L(\mathbf{x},\omega_r) = L_0(\mathbf{x},\omega_r) + L_1(\mathbf{x},\omega_r) + \ldots$$

$$L_i(\mathbf{x},\omega_r) = \int_{\Omega^+} f_r(\omega_i,\mathbf{x},\omega_r) L_{i-1}(\mathbf{x},\omega_i) \cos\theta_i \, d\omega_i$$

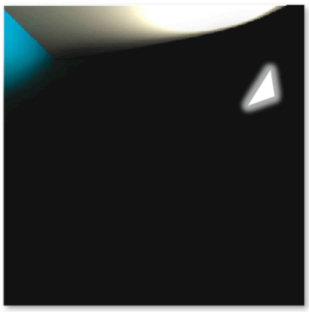


All paths from source that take i bounces

In general the ith bounce models how all of the energy from the “previous bounce” contributes to outgoing radiance in the given direction.

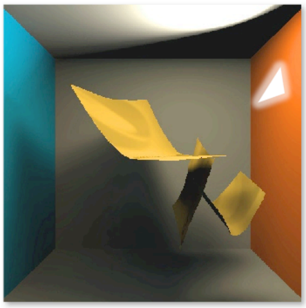
Neumann Expansion

- Direct Illumination: $L_0(\mathbf{x}, \omega_r)$



- Direct + Indirect:

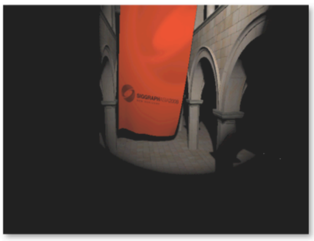
$$L_0(\mathbf{x}, \omega_r) + L_1(\mathbf{x}, \omega_r)$$



Neumann Expansion

Direct + Indirect

$$L_0(\mathbf{x},\omega_r) + L_1(\mathbf{x},\omega_r)$$



Direct only

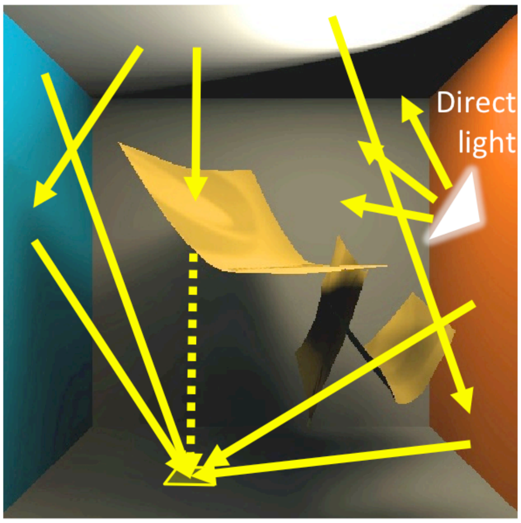
$$L_0(\mathbf{x},\omega_r)$$

Indirect only

$$L_1(\mathbf{x},\omega_r)$$

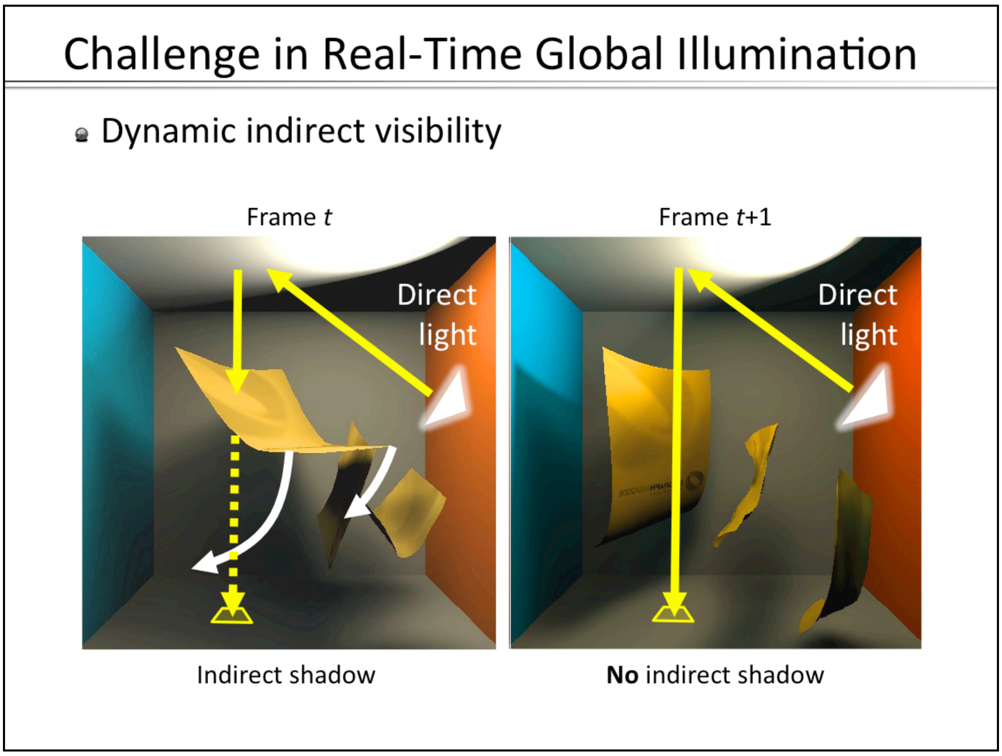
Challenge in Real-Time Global Illumination

- Global illumination is **global**



Light from ***all*** directions

Global Illumination is global!



The most challenging part is the **dynamic indirect visibility**.

Here is a frame from the previous animation and let's look at **a particular light path**.

It starts at the direct light, bounces from a surface and is than blocked by geometry, casting an **indirect shadow**.

In the next frame, this path becomes **unoccluded,** and the indirect shadow turns into **indirect light**.

So this has to be taken into account.

Overview

- Introduction
  - Rendering
  - Rendering Equation
  - Neumann Series
  - Shading Algorithms

Common Shading Algorithms

- Real-time Rendering: Approx. Rendering Equation
  - Direct lighting: many solutions (assume given)
  - Indirect lighting: ambient term 
  - Indirect lighting: no visibility 
  - Indirect lighting: semi-static scenes 
  - Indirect lighting: 1-bounce only 
  - …

In real-time rendering, we usually make approximation to the full rendering equation, as it is too expensive to compute on the fly.

Common approximations are:

- only direct lighting (for the remainder of tutorial: assume we have one)
- trivial indirect lighting (really no indirect illumination at all, just a simple ambient term can make up for it somewhat)
- indirect lighting but without taking visibility (indirect shadowing) into account
- indirect lighting taking visibility into account but only for static scenes
- indirect lighting but only for one indirect bounce
- …

Schedule

8:30 – 8:40 Introduction (Kautz)
- Motivation
- Problem Statement
- Definitions (Rendering Equation, Neumann Series, …)
- Direct Illumination vs. Indirect Illumination

8:40 – 9:10 Screen Space Techniques (Dachsbacher)
- Screen-Space Ambient Occlusion (SSAO)
- Extending SSAO to Indirect Illumination
- Reflective Shadow Maps
- (Multiresolution) Splatting of Indirect Illumination
- Examples, Results, Limitations

9:10 – 9:40 Virtual Point Lights (Kautz)
- Instant Radiosity
- Incremental Instant Radiosity
- Imperfect Shadow Maps
- Examples, Results, Limitations

9:40 – 10:05 Hierarchical Finite Elements (Dachsbacher)
- Dynamic Ambient Occlusion for Indirect Illumination
- Implicit Visibility
- Anti-Radiance
- Examples, Results, Limitations

10:05 – 10:15 Conclusions/Summary (Kautz)
* Comparison of Presented Techniques
* Q&A