UNIVERSITÀ
DEGLI STUDI
FIRENZE

Scuola di Scienze Matematiche, Fisiche e Naturali
Corso di Laurea in Informatica

Box of Sunlight

Riflessione Realistica della Luce nel Rendering di Animazioni 3D

Box of Sunlight

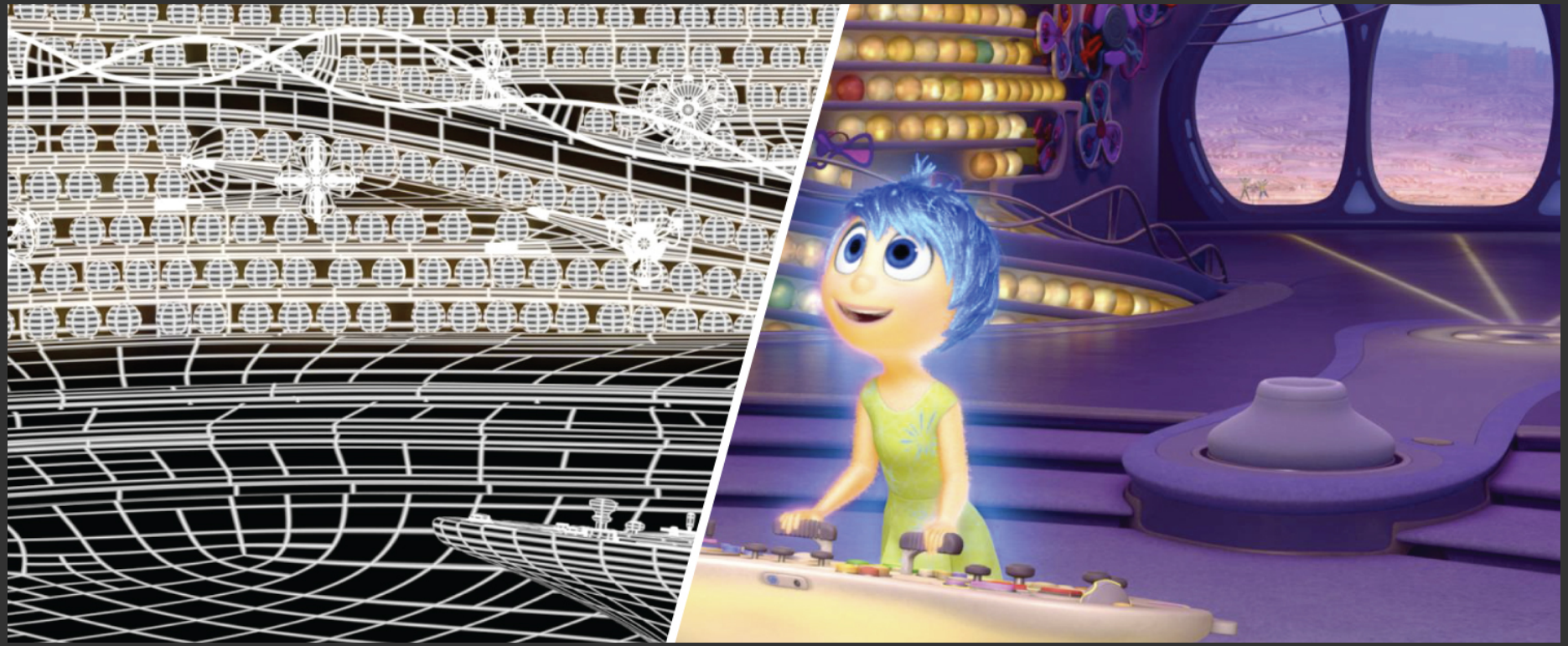
Realistic Reflection of Light in 3D Animation Rendering

Leonardo Zetti

Relatore: *Prof. Stefano Berretti* Correlatore: *Prof. Michele Ginolfi*

Anno Accademico 2023-2024

What is Rendering?



In *rendering*, we convert three-dimensional virtual scenes into a 2D image.

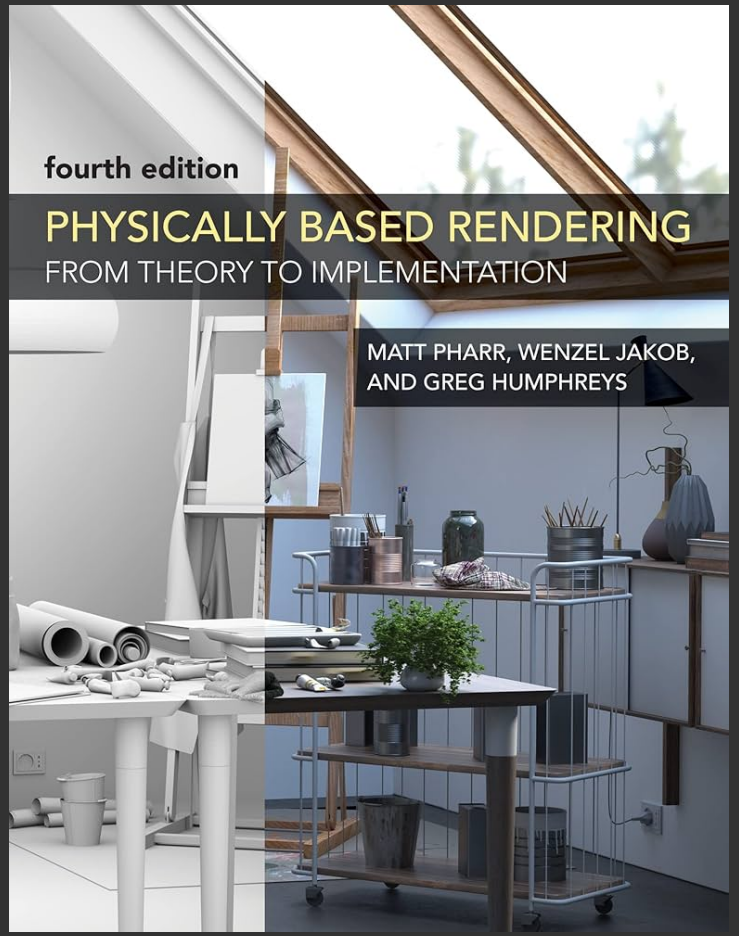
We take a “photo” of the virtual environment, recording light with a *virtual camera*.

Physically-Based Rendering

Physically-based rendering (*PBR*) is based on physical models of real-world light and materials.

In PBR, we employ:

- A *light transport algorithm*, to simulate how light traverses the scene
- *Monte Carlo integration* for approximating integrals (in rendering, it is often necessary to solve integral equations)

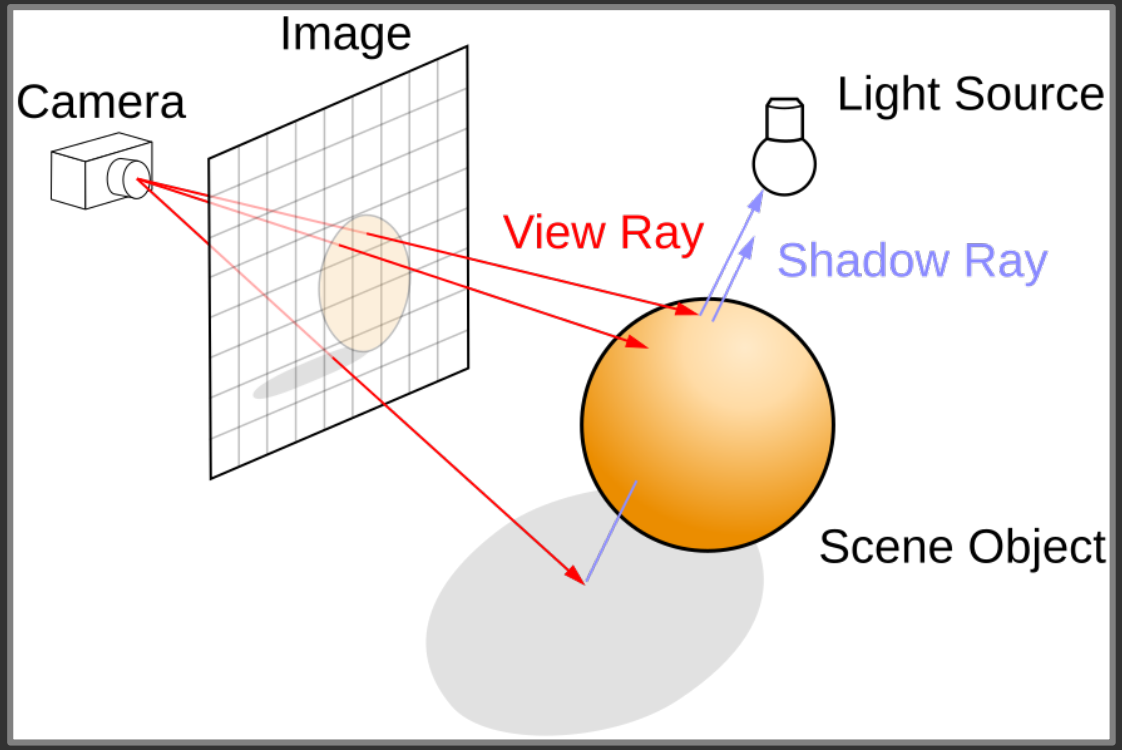


Our Light Transport Algorithm: Ray Tracing

The camera records light.

Main idea of ray tracing: follow the path of light rays in *reverse*.

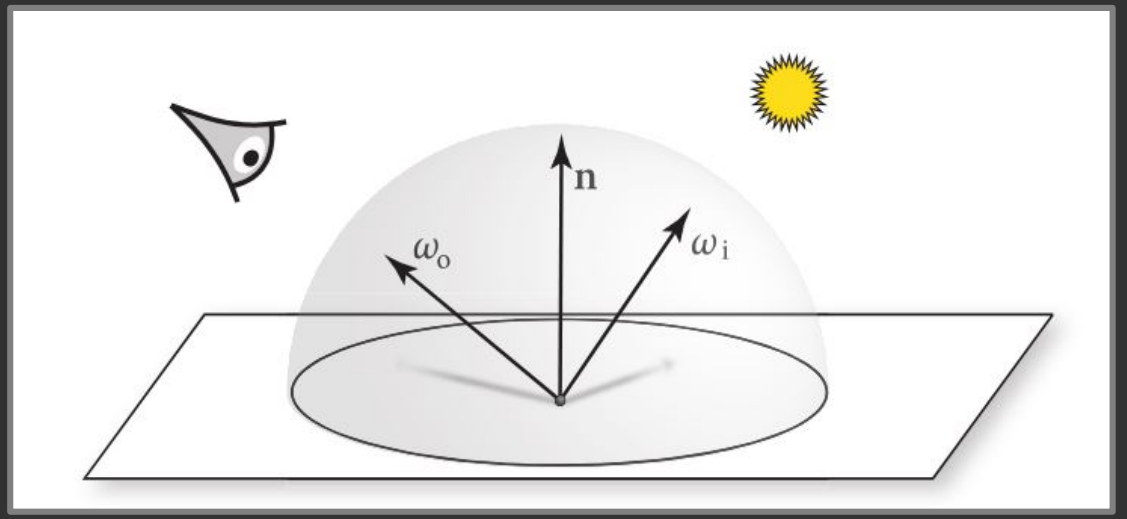
Where did the rays come from?



Some rays represent light that's reflected by objects.

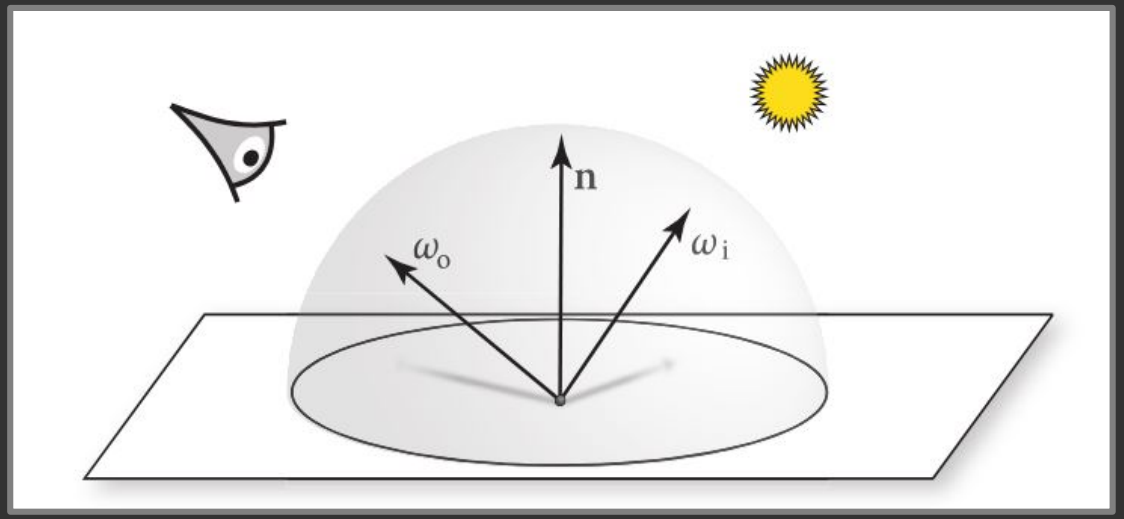
How much light does the object reflect?

The Reflection Equation



$$L_o(\omega_o) = \int_H f_r(\omega_o, \omega_i)\, L_i(\omega_i)\, cos\theta_i \, \mathrm{d}\omega_i$$

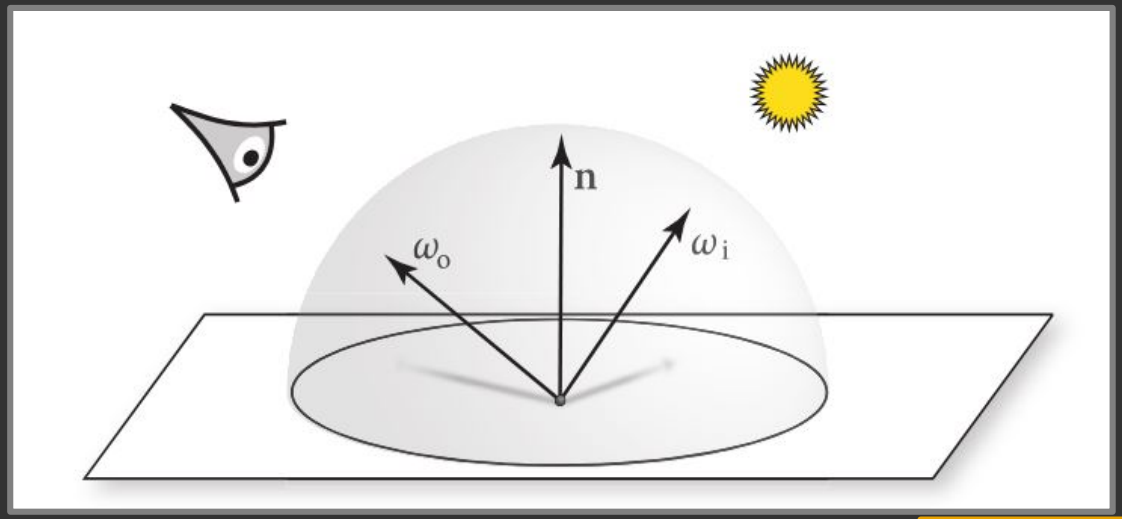
The Reflection Equation



Outgoing light (measured in units of *radiance*)

$$L_o(\omega_o) = \int_H f_r(\omega_o, \omega_i)\, L_i(\omega_i)\, cos\theta_i \, \mathrm{d}\omega_i$$

The Reflection Equation

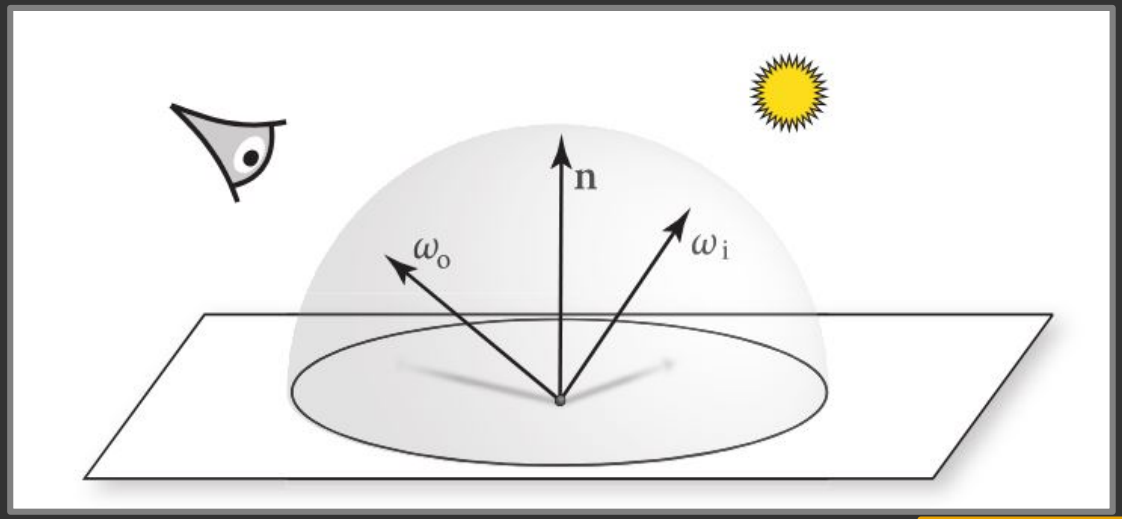


Outgoing light (measured in units of *radiance*)

Incoming light

$$L_o(\omega_o) = \int_H f_r(\omega_o, \omega_i)\, L_i(\omega_i)\, cos\theta_i \, \mathrm{d}\omega_i$$

The Reflection Equation



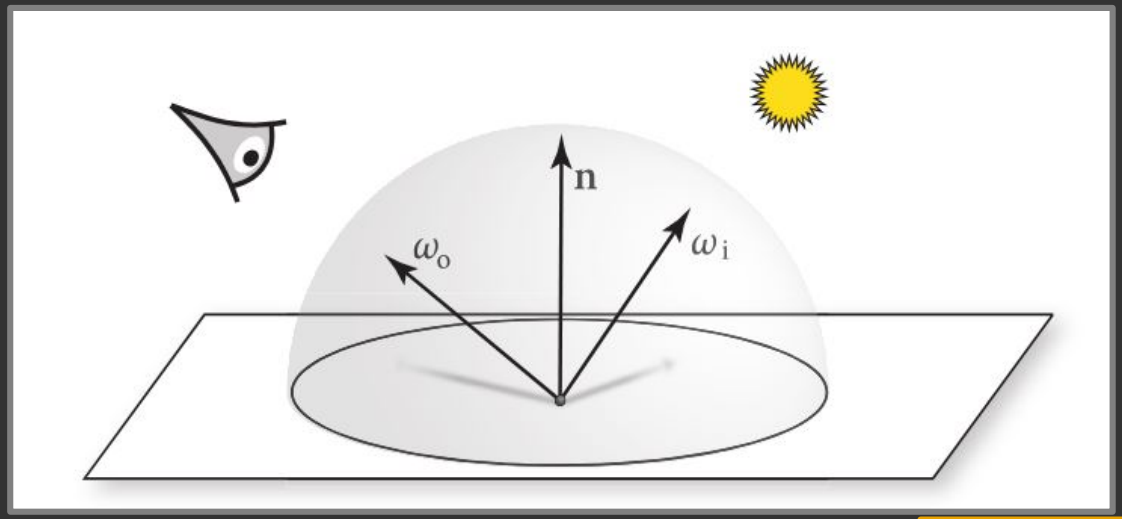
Outgoing light (measured in units of *radiance*)

Incoming light

$$L_o(\omega_o) = \int_H f_r(\omega_o, \omega_i)\, L_i(\omega_i)\, cos\theta_i\, \mathrm{d}\omega_i$$

Bidirectional Reflectance Distribution Function (*BRDF*). The BRDF tells us what portion of the incoming light is reflected.

The Reflection Equation



Outgoing light (measured in units of *radiance*)

Incoming light

$$L_o(\omega_o) = \int_H f_r(\omega_o, \omega_i)\, L_i(\omega_i)\, cos\theta_i\, \mathrm{d}\omega_i$$

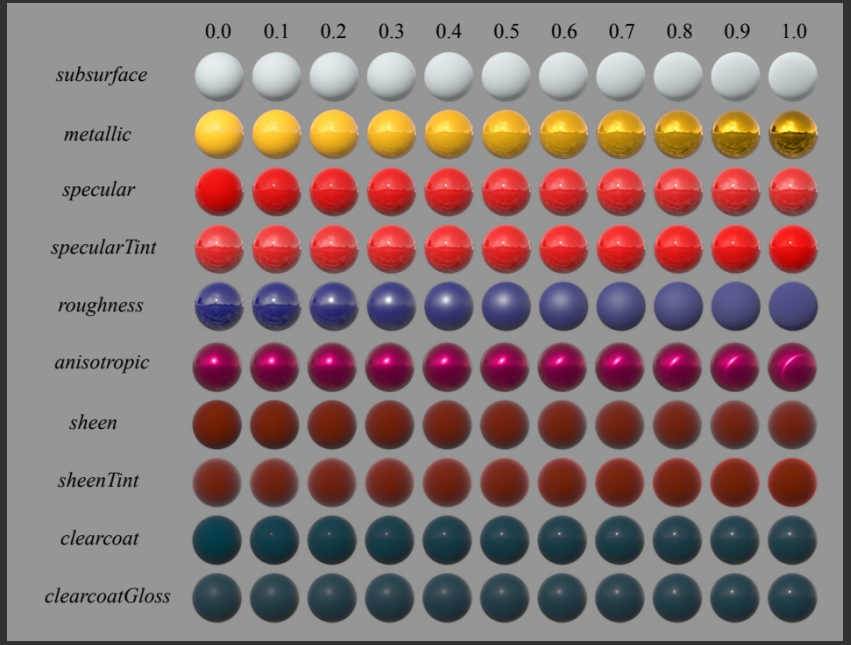
Integral over all incoming light directions (approximated with Monte Carlo integration)

Bidirectional Reflectance Distribution Function (*BRDF*). The BRDF tells us what portion of the incoming light is reflected.

The Disney “Principled” BRDF

Details of the Disney “Principled” BRDF model were made public by Brent Burley in 2012. [1]

The model was first used on the movie *Wreck-it Ralph*.



Parameters of the Disney BRDF

[1] Burley, Brent (2012). *Physically-based shading at Disney*.

BoxOfSunlight

BoxOfSunlight is a rendering application, written in C++.
It is supported by the *OpenGL graphics API*.

It simulates realistic reflection of light by applying
- Ray tracing
- The Disney “Principled” BRDF
- Monte Carlo integration

in a virtual scene illuminated by a *cubemap*.

Our Cubemaps

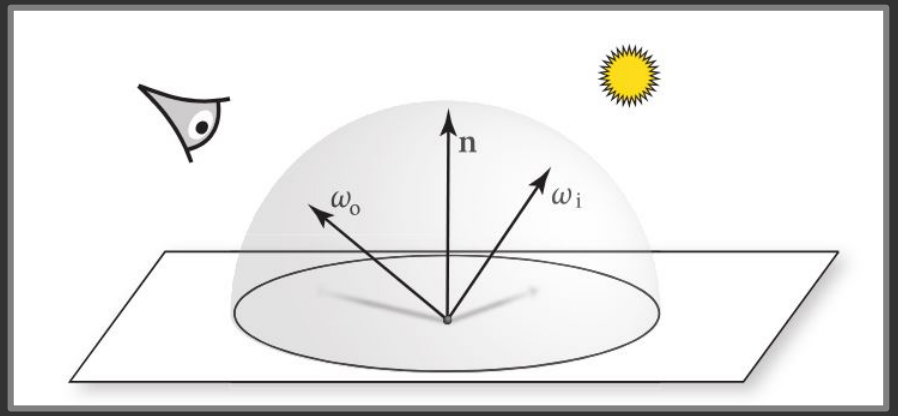
Meadow 2, by Sergej Majboroda (Poly Haven)

Spruit Sunrise, by Greg Zaal (Poly Haven)

Demo

We will generate the following image with BoxOfSunlight.

Monte Carlo integration will be applied to calculate the amount of light reflected by surface points. (*)



(*) with 512 Monte Carlo samples per-iteration

1 iteration

4 iterations

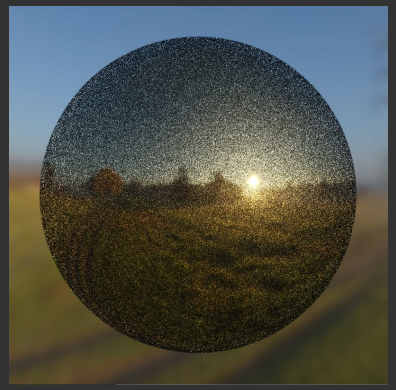
16 iterations

64 iterations

256 iterations

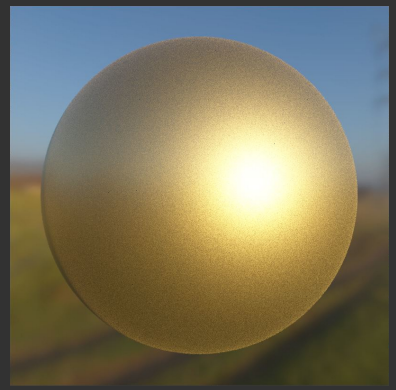
Supported Reflection Effects

metallic surface with various *roughness* values: (*)



roughness = 0

roughness = 0.2

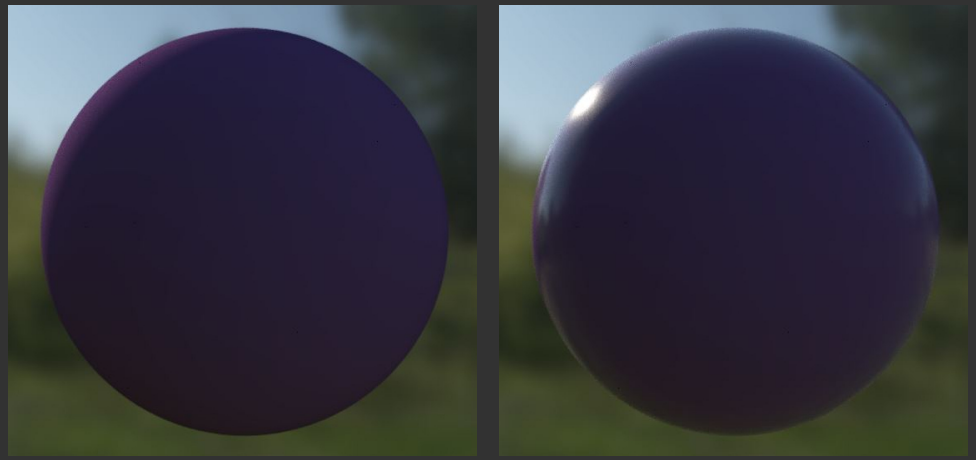


roughness = 0.4

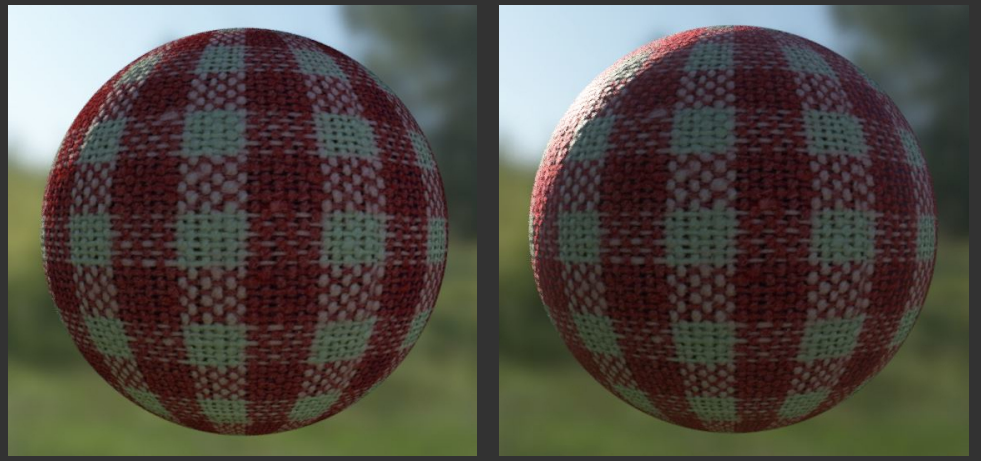
roughness = 0.6

roughness = 1

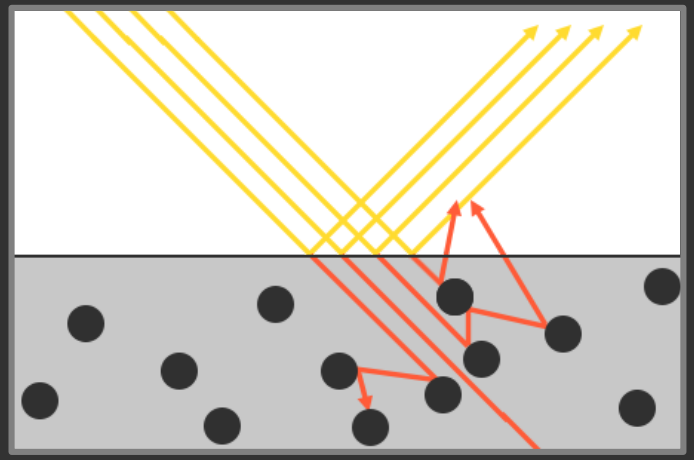
(*) images rendered in 20-25 iterations, with 4096 Monte Carlo samples per-iteration

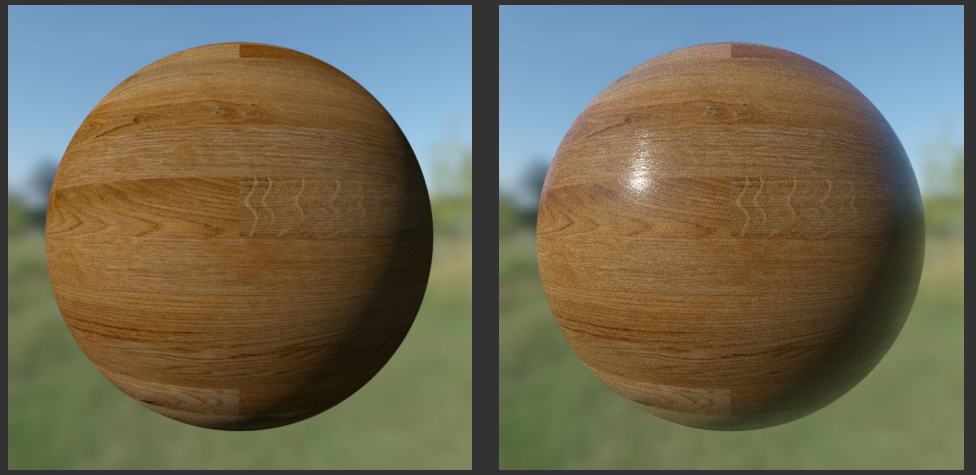


smooth plastic



fabric





varnished wood

Conclusions and Future Perspectives

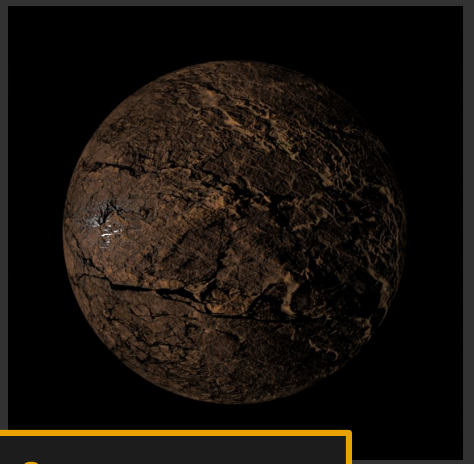
This thesis was a chance to develop some personalized rendering tools.
It offered insights into what it means to create custom programs for artistic purposes.

Possible improvements to BoxOfSunlight:

- Making perfectly specular reflections less noisy by applying *importance sampling* techniques.
- Reducing rendering times by employing acceleration data structures (*bounding volume hierarchies*)

Here, **2752 triangles**, 20 iterations and 4096 Monte Carlo samples per-iteration result in a rendering time of **10 minutes**.

However, when we start to use **tens of thousands** of triangles, rendering times become completely unmanageable.



Thank you for your time