Game Technology

Lecture 7 – 28.11.2014

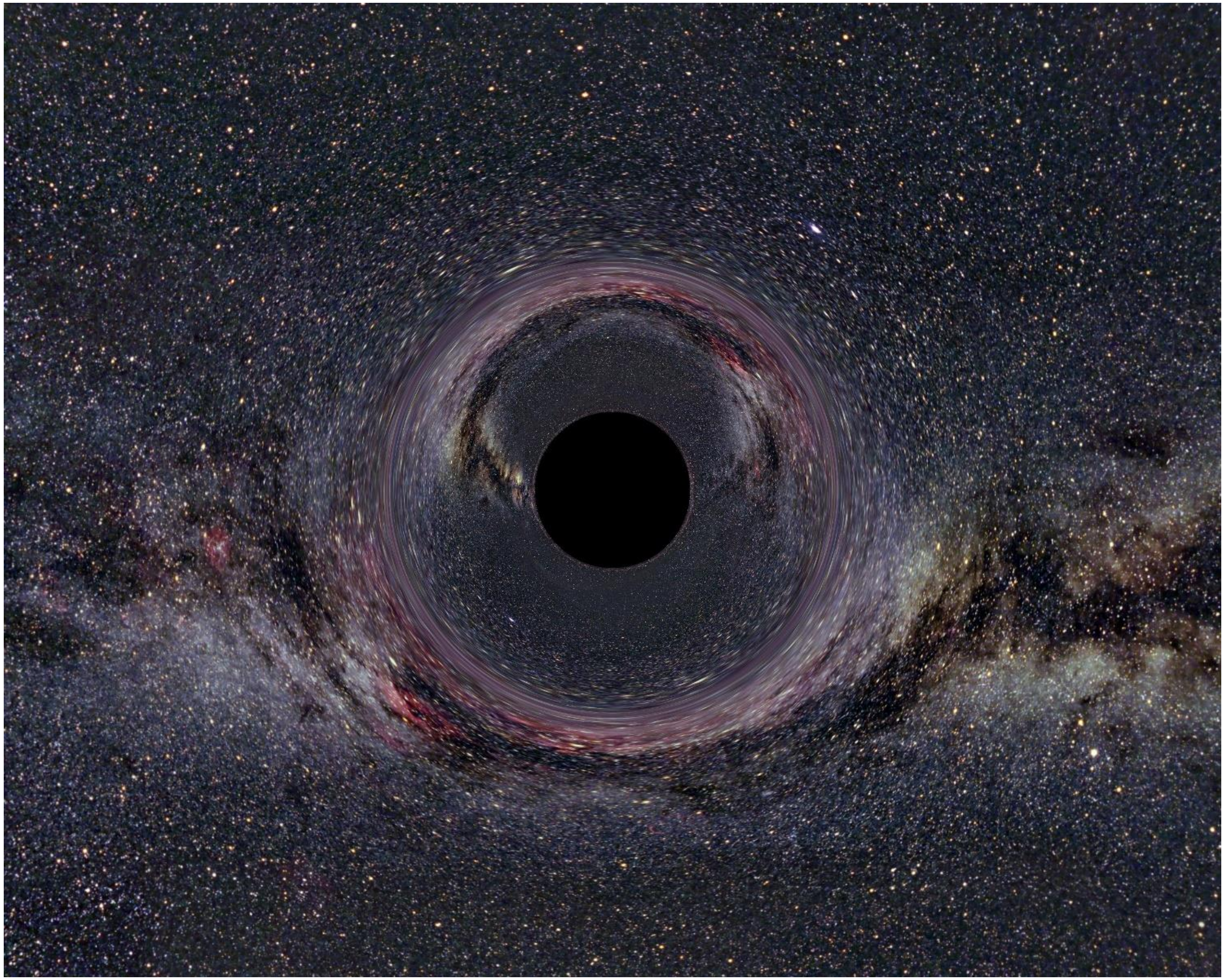
Dipl-Inf. Robert Konrad
Dr.-Ing. Florian Mehm

Prof. Dr.-Ing. Ralf Steinmetz
KOM - Multimedia Communications Lab

Preliminary timetable

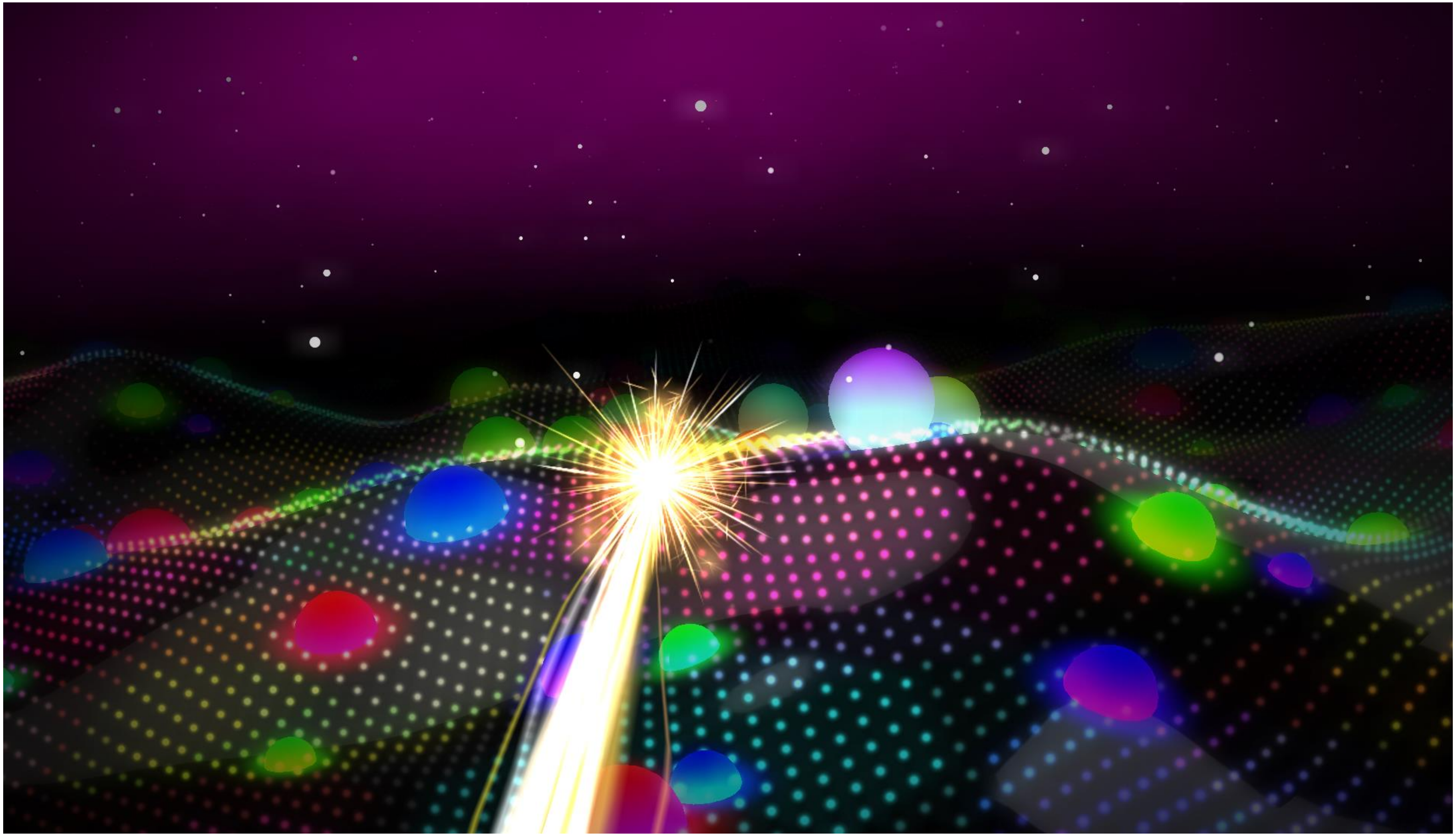
Lecture No.	Date	Topic
1	17.10.2014	Basic Input & Output
2	24.10.2014	Timing & Basic Game Mechanics
3	31.10.2014	Software Rendering 1
4	07.11.2014	Software Rendering 2
5	14.11.2014	Basic Hardware Rendering
6	21.11.2014	Animations
7	**28.11.2014**	**Physically-based Rendering**
8	05.12.2014	Physics 1
9	12.12.2014	Physics 2
10	19.12.2014	Scripting
11	16.01.2015	Compression & Streaming
12	23.01.2015	Multiplayer
13	30.01.2015	Audio
14	06.02.2015	Procedural Content Generation
15	13.02.2015	AI

Physically Based Rendering



- **Light**
  - Starts from light source
  - Bounces around
    - Looses intensity with each collission
  - Eventually reaches the camera

Light Sources



Point Lights

TECHNISCHE UNIVERSITÄT DARMSTADT

- **Defined by a position**
- **and light color/intensity**

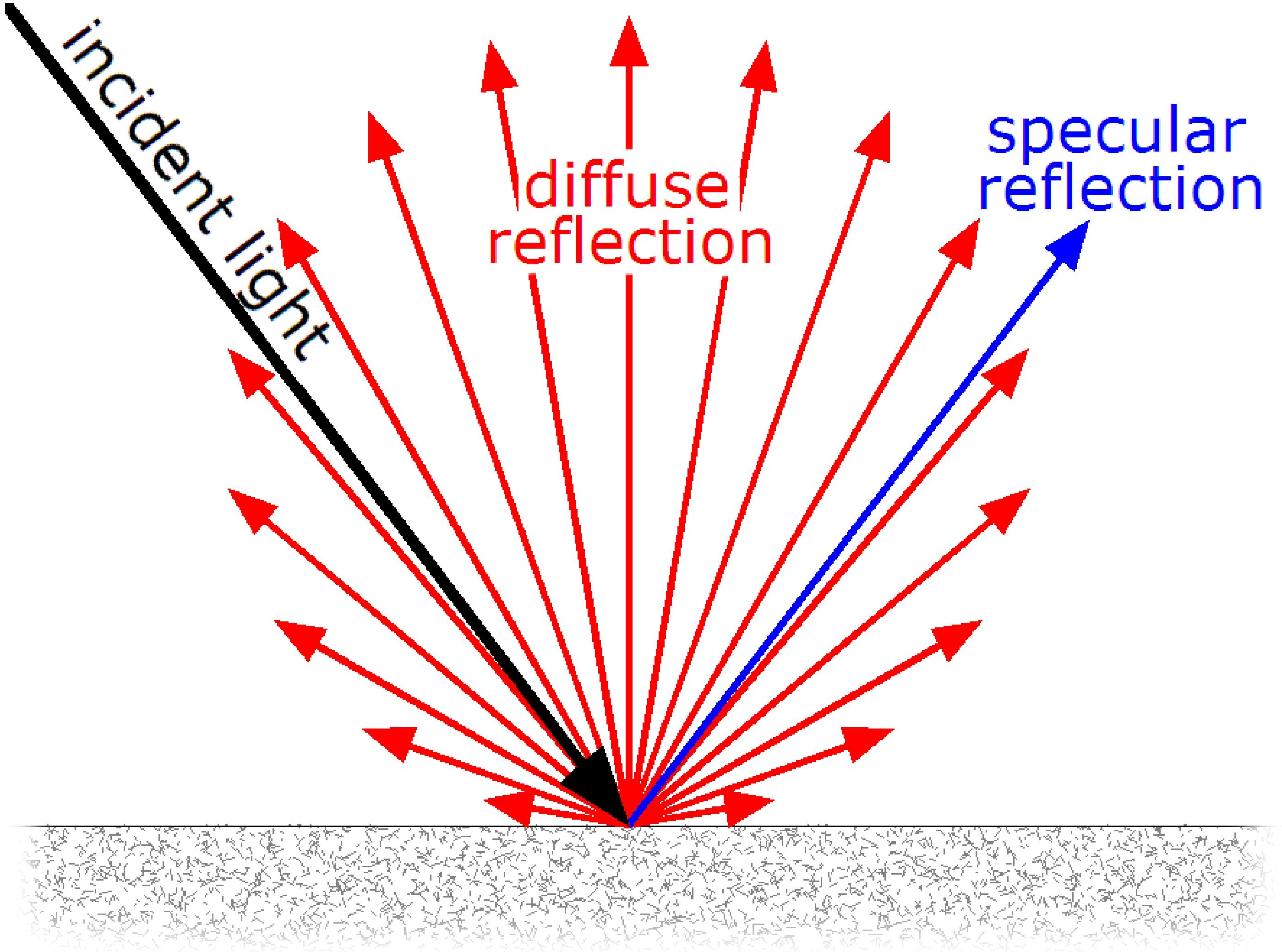
Directional Light

- Just a direction
- and light intensity/color

Area Lights

- More light sources possible
- Not supported by most game engines

Bounces

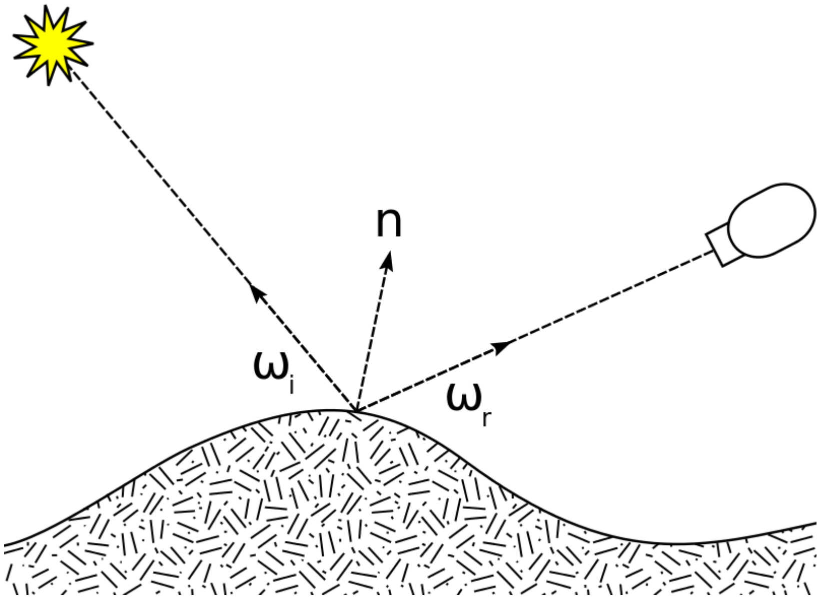


BRDF

- **Bidirectional reflectance distribution function**

  $f_{\mathrm{r}}(\omega_{\mathrm{i}},\, \omega_{\mathrm{r}})$

  - incoming light direction
  - outgoing direction (for example to the camera)
  - returns the ratio of reflected radiance



Path Tracing

- **foreach (pixel)**
  - bounce around a lot

- **Use BRDF at each collision**

- **Very slow**
- **But useful to create reference images**
- **and for prerendered lighting informations**

Realtime Lighting

TECHNISCHE UNIVERSITÄT DARMSTADT

- **Consider only light rays from direct light sources**
  - First bounce
- **Use shadow maps**
  - Second bounce
- **Ignore further light bouncing**
  - No reflections
  - No ambient light

Image Based Lighting

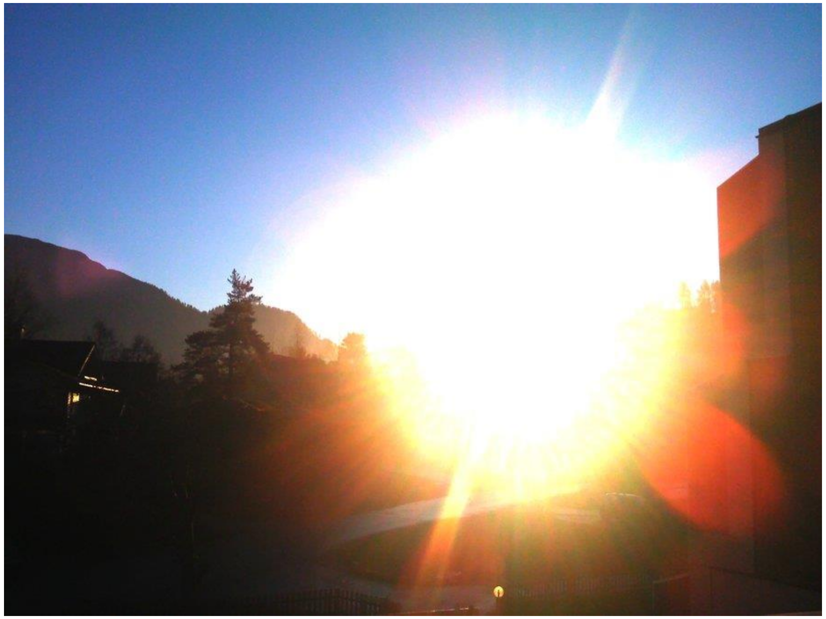
- **Put surroundings in cube map**
  - Use for example path tracing to generate the cube map

- **Ignore lights, instead sample cube map**

- **A cube map is only correct for one position**
- **Ignores dynamic objects**

HDR

- „High dynamic range“
- Use more than 32 bits of data for one pixel

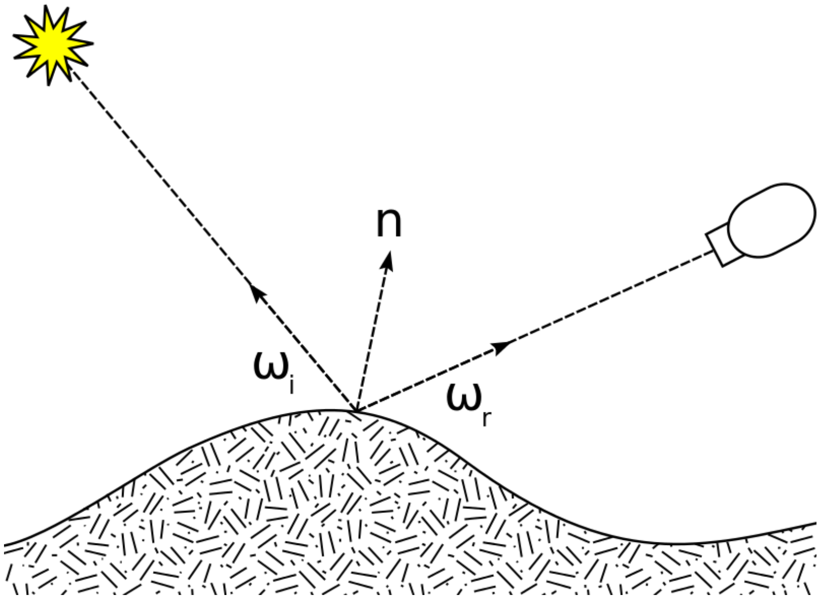


BRDF

- **Bidirectional reflectance distribution function**

  $f_r(\omega_i, \omega_r)$

  - incoming light direction
  - outgoing direction (for example to the camera)
  - returns the ratio of reflected radiance



BRDF Shortcomings

- **Subsurface-Scattering**
- **Wavelength dependence**

BRDF

- **Only positive light**

$$f_r(\omega_i, \omega_r) \geq 0$$

BRDF

- **Inverted**

$$f_{\mathbf{r}}(\omega_{\mathbf{i}}, \omega_{\mathbf{r}}) = f_{\mathbf{r}}(\omega_{\mathbf{r}}, \omega_{\mathbf{i}})$$

BRDF

- **Energy conserving**

$$\forall \omega_{\mathbf{i}},\ \int_{\Omega} f_{\mathrm{r}}(\omega_{\mathrm{i}},\, \omega_{\mathrm{r}})\ \cos\theta_{\mathrm{r}} d\omega_{\mathrm{r}} \leq 1$$

Phong Lighting

TECHNISCHE UNIVERSITÄT DARMSTADT

- **color = ambient + diffuse + specular**

Phong Lighting

TECHNISCHE UNIVERSITÄT DARMSTADT

- color = ~~ambient~~ + diffuse + specular

Gamma

Gamma Service

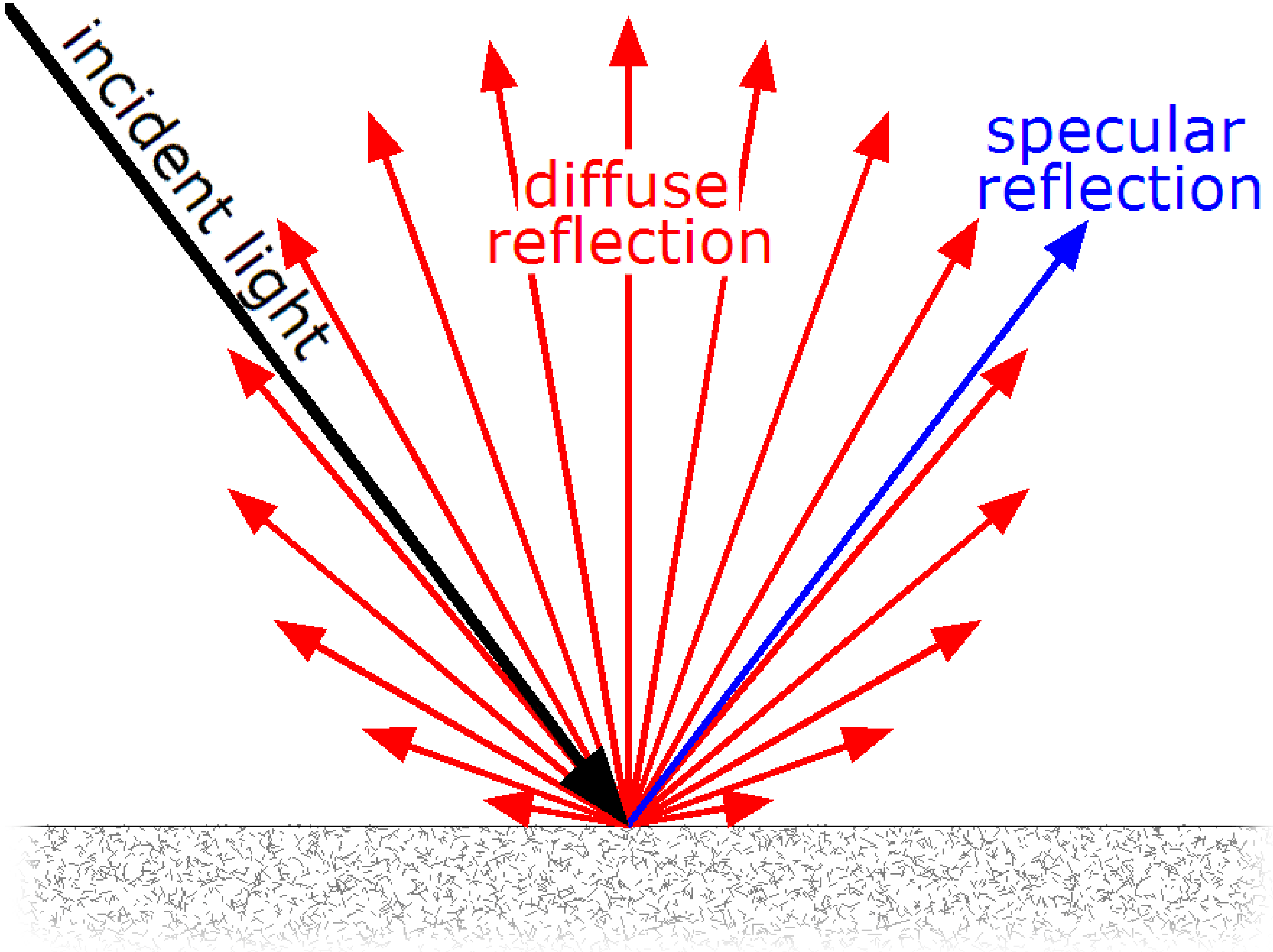
Gesellschaft für zerstörungsfreie Werkstoffprüfung m.b.H

sRGB

- **See Exercise 1**

- **Transform textures to linear (pow 2.2)**
  - Or use sRGB texture reading (also allows proper filtering)
- **Lighting calculations in linear space (gamma 1)**
- **Then transform for sRGB (pow 1 / 2.2)**

Diffuse & Specular



Diffuse

- **Lambertian reflectance / Phong diffuse**
- **I = L*N**

- **Good enough for modern engines**
  - Used for example in Unreal Engine 4

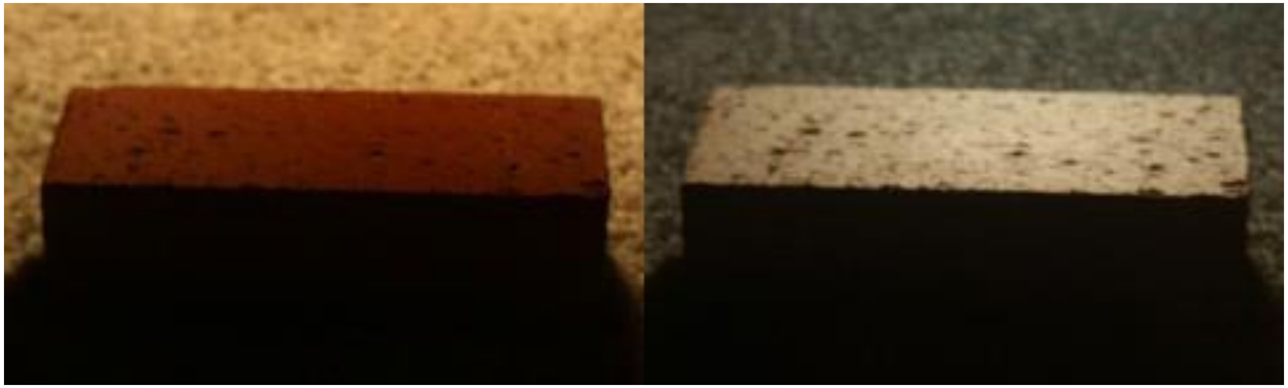
Specular

Brick

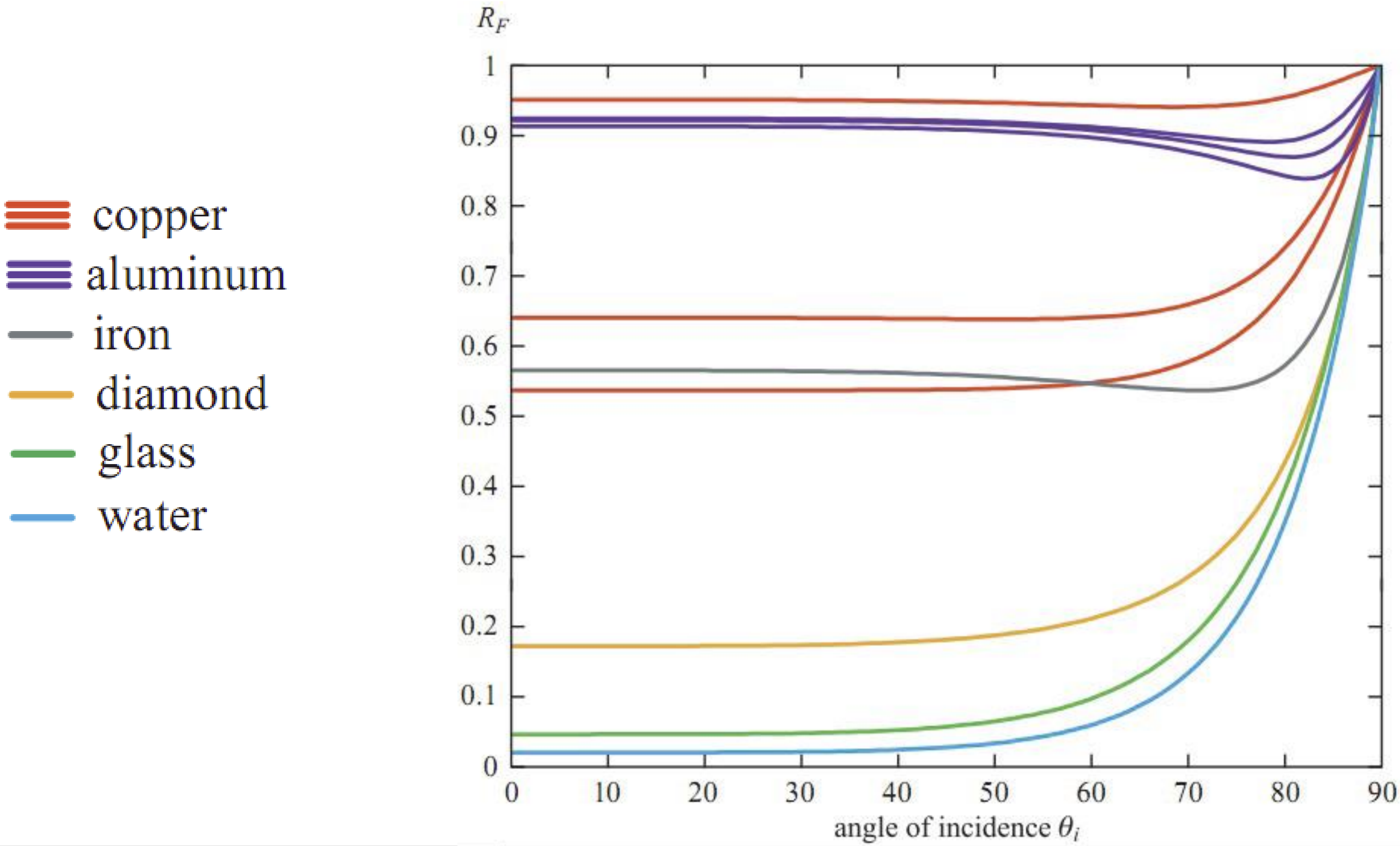
- **angle(normal, light) = angle(normal, camera)**

Brick

- angle(normal, light) = angle(normal, camera)



Fresnel



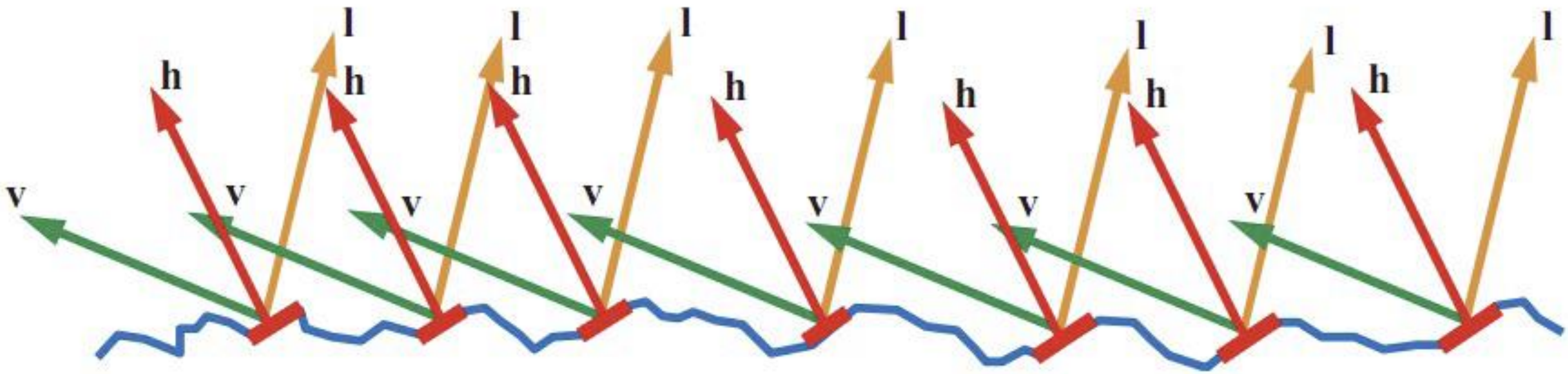
copper
aluminum
iron
diamond
glass
water
R_F
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
10
20
30
40
50
60
70
80
90
angle of incidence θ_i

Fresnel

Schlick Approximation

- Schlick(spec, light, normal) = spec + (1 - spec) (1 - (light*normal))^5

Microfacet Model



Microfacet BRDF

TECHNISCHE UNIVERSITÄT DARMSTADT

$$f(\mathbf{l}, \mathbf{v}) = \frac{F(\mathbf{l}, \mathbf{h}) G(\mathbf{l}, \mathbf{v}, \mathbf{h}) D(\mathbf{h})}{4(\mathbf{n} \cdot \mathbf{l})(\mathbf{n} \cdot \mathbf{v})}$$

Normal Distribution

- **D(h)**
- **Portion of microfacets pointing to h**

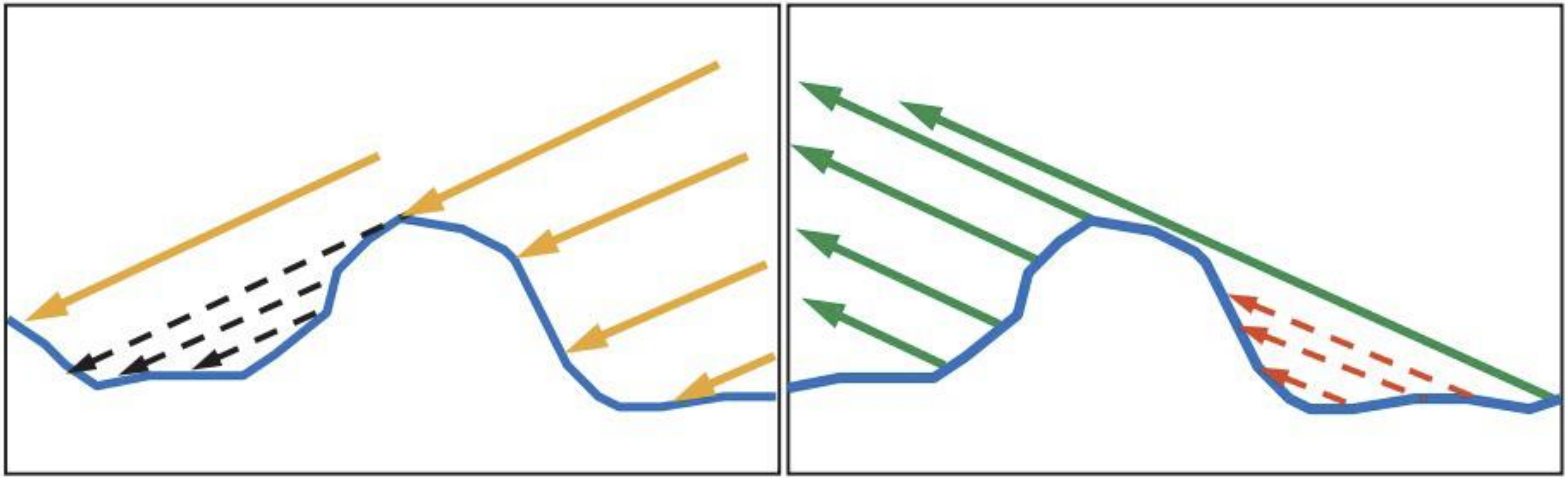
$$D_{tr}(\mathbf{m}) = \frac{\alpha_{tr}^2}{\pi\left((\mathbf{n}\cdot\mathbf{m})^2\left(\alpha_{tr}^2 - 1\right) + 1\right)^2}$$

- **Trowbridge-Reitz (GGX)**
- **α: Roughness**

Geometry Factor

- **G(l, v, h)**
- **Cook-Torrance:**

$$G_{\mathrm{ct}}(\mathbf{l},\mathbf{v},\mathbf{h}) = \min\left(1, \frac{2(\mathbf{n}\cdot\mathbf{h})(\mathbf{n}\cdot\mathbf{v})}{(\mathbf{v}\cdot\mathbf{h})}, \frac{2(\mathbf{n}\cdot\mathbf{h})(\mathbf{n}\cdot\mathbf{l})}{(\mathbf{v}\cdot\mathbf{h})}\right)$$



Physically Based Rendering

- **In practice:**
  - Gamma correction
  - Microfacet BRDF
  - Lots of Cube Maps

Polarization of Reflected Light

- **Specular Reflection**
  - Polarization does not change

- **Diffuse Reflection**
  - Polarization is randomized

Cardboard

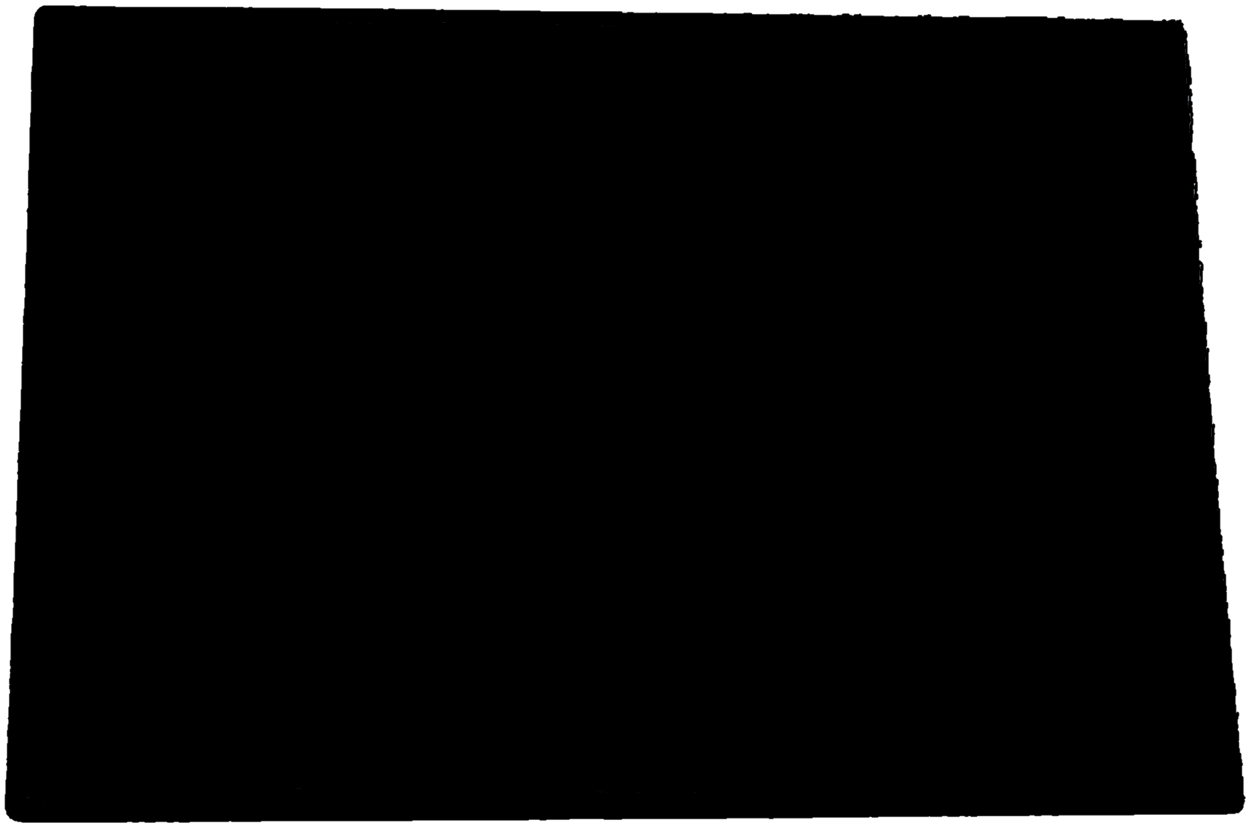
Cardboard Diffuse

Cardboard Specular

Metal

Metal Specular

Metal Diffuse



Metals and Dielectrics

- **Metals:**
  - No diffuse
  - High Specular

- **Dielectrics**
  - Diffuse
  - Low Specular

- **Note: Specular value is specified at low angles**

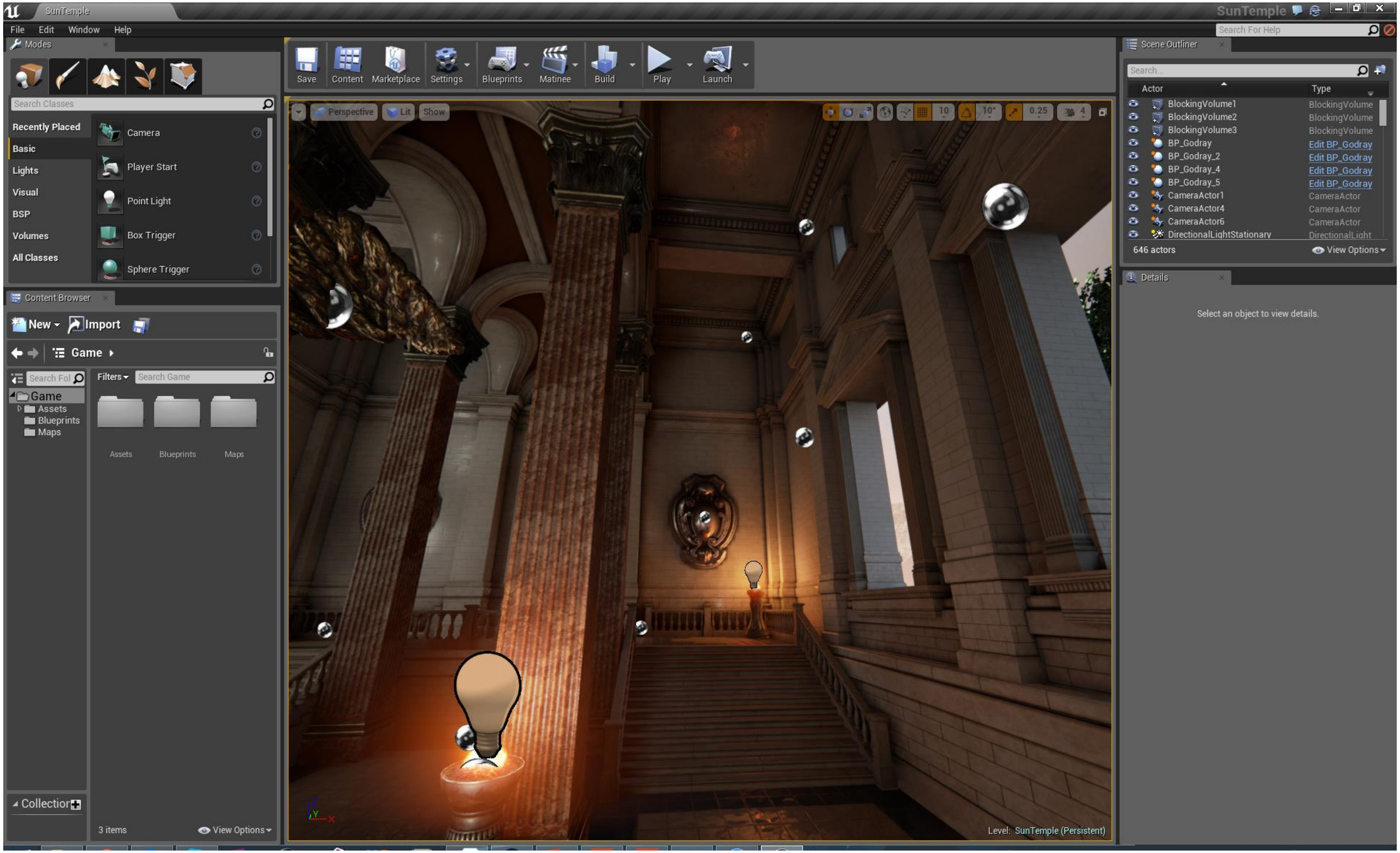
Textures

- **Typical Setup:**
  - Diffuse texture
  - Specular texture
  - Roughness texture
  - Normal Map

Incorporating Image Based Lighting

- **Precalculate Cube Maps**
  - Lots of Cube Maps
  - Manually placed in level editor

Cube Maps



SunTemple
File
Edit
Window
Help
Modes
Search Classes
Recently Placed
Basic
Lights
Visual
BSP
Volumes
All Classes
Camera
Player Start
Point Light
Box Trigger
Sphere Trigger
Content Browser
New
Import
Game
Filters
Search Game
Game
Assets
Blueprints
Maps
Assets
Blueprints
Maps
Collections
3 items
View Options
Save
Content
Marketplace
Settings
Blueprints
Matinee
Build
Play
Launch
Perspective
Lit
Show
Level: SunTemple (Persistent)
Scene Outliner
Search For Help
Actor
Type
BlockingVolume1
BlockingVolume2
BlockingVolume3
BP_Godray
BP_Godray_2
BP_Godray_4
BP_Godray_5
CameraActor1
CameraActor4
CameraActor6
DirectionalLightStationary
BlockingVolume
Edit BP_Godray
CameraActor
DirectionalLight
646 actors
View Options
Details
Select an object to view details.

Cube Maps

- **Can be interpolated**
  - Which is a rough approximation

- **Can not capture dynamic objects**

Reflection Rendering

- **As done in Unreal Engine 4, Killzone Shadow Fall,…**

- **Deferred Rendering Pass**
- **Raytrace depth buffer**
- **If no hit**
  - Interpolate local cube maps
- **If no hit**
  - Use global cube map

Ambient Occlusion

- **Small notches are normally shadowed**
  - Unless lit directly

- **Calculations need very exact light bounces**

Screen Space Ambient Occlusion

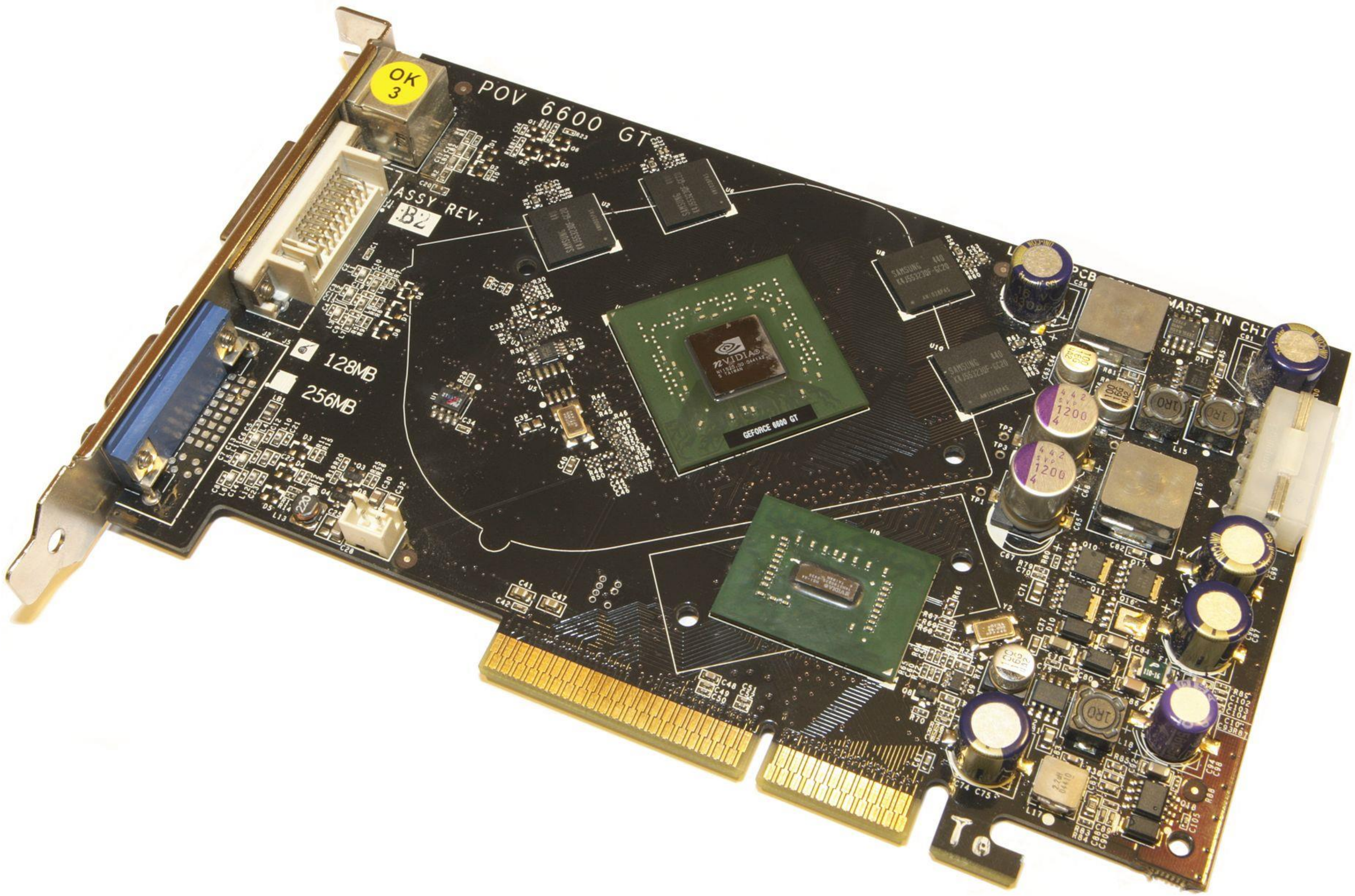
- **Filter after rendering**
- **Darken at sharp normal changes**

Global Illumination

- „Ambient light“

- Spherical Harmonic Lighting
- Voxel Cone Tracing
- …

GPU Internals



Memory

- **Memory bandwidth is extremely important**
  - Textures
  - Framebuffer
  - Depth Buffer

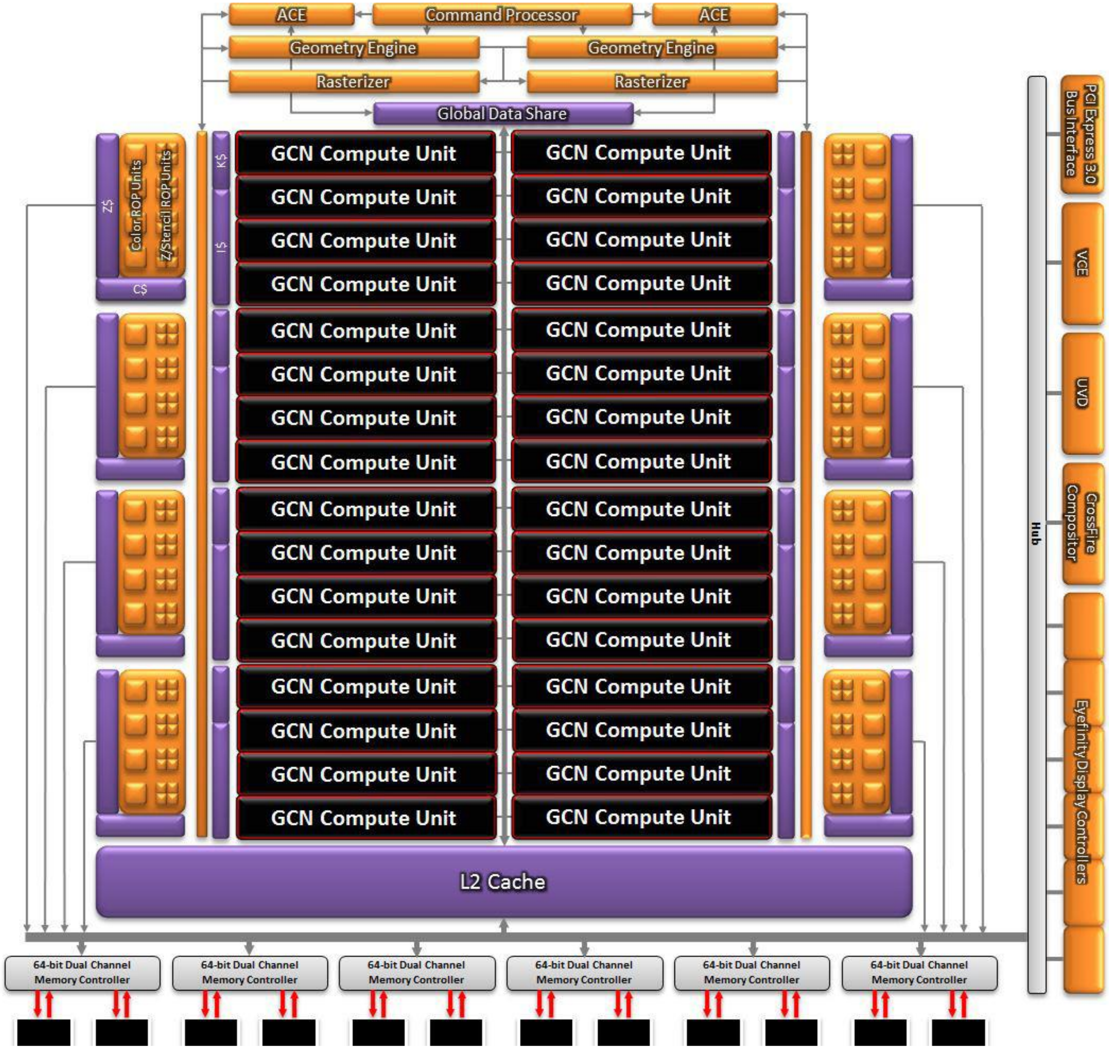
- **Memory access times not very important**
  - Most data is streamed
  - Access times can be hidden by switching tasks

Memory

- **Gigantic discrepancy from low-end to high-end**
  - GeForce 720 starts at 14.4 GB/s
  - PS4: 176 GB/s
  - GeForce 780 Ti: 336 GB/s

Vertex and Fragments Shaders

- **Run on the same hardware**
- **Dynamically scheduled**


ACE
Command Processor
ACE
Geometry Engine
Geometry Engine
Rasterizer
Rasterizer
Global Data Share
GCN Compute Unit
Z$
Color ROP Units
Z/Stencil ROP Units
C$
K$
I$
L2 Cache
Hub
PCI Express 3.0 Bus Interface
VCE
UVD
CrossFire Compositor
Eyefinity Display Controllers
64-bit Dual Channel Memory Controller

SMT

- **Also used in CPUs (Hyperthreading)**
- **Switch to different thread when stalled (for example waiting for memory)**

SIMD

- **Compiler can put calculations on multiple vertices/pixels in one instruction**

- **Problem: Flow control**
  - Wrong paths pseudo-executed
  - Can be efficient when all vertices/pixel in one pack take the same paths

- **Shader variants**
  - „Typically when you create a simple surface shader, it internally expands into 50 or so internal shader variants” (Shader Compilation in Unity 4.5)

Parallelization

- **Small work packages can prevent parallelization**
  - Performance dip for tiny triangles

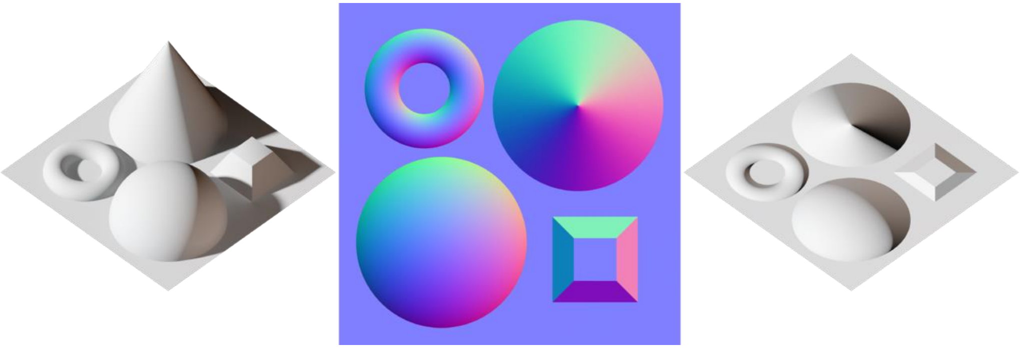
CPU <-> GPU

TECHNISCHE UNIVERSITÄT DARMSTADT

- **Biggest performance trap**

- **Minimize state changes**
- **Minimize draw calls**
- **Send little data to the GPU**
- **If possible never read data from the GPU**

2.1 Normal Maps



2.2 Particles

- **Depth sorting**
  - Particles can not intersect because they are flat
    -> sorting not much of a problem
  - Performance problem: Overdraw

2.3 Skeletal Animations

- **Quaternions because Quaternions**

- **or**

- **Euler angles because sufficient**
  - Human joints are very restrictive