Advanced Computer Graphics
Introduction

Matthias Teschner
Outline

- Organization
- Concepts
- Applications
- History
- Selected variants
- Components
Contact

– Matthias Teschner
  052 / 01-024
  teschner@informatik.uni-freiburg.de
  https://cg.informatik.uni-freiburg.de/
Course Information

– Key course
  – Pattern recognition and computer graphics (rasterization)

– Specialization courses
  – Advanced computer graphics (ray tracing)
  – Simulation in computer graphics (e.g., fluids)

– Master project, lab course, Master thesis
  – Simulation track
  – Rendering track
## Seminars / Projects / Theses

<table>
<thead>
<tr>
<th>Semester</th>
<th>Simulation Track</th>
<th>Rendering Track</th>
</tr>
</thead>
<tbody>
<tr>
<td>Winter</td>
<td>Key Course</td>
<td>Key Course</td>
</tr>
<tr>
<td></td>
<td>Simulation Course</td>
<td></td>
</tr>
<tr>
<td>Summer</td>
<td>Lab Course - Simple fluid solver</td>
<td>Rendering Course</td>
</tr>
<tr>
<td></td>
<td>Simulation Seminar</td>
<td>Lab Course - Simple Ray Tracer</td>
</tr>
<tr>
<td>Winter</td>
<td>Master Project - PPE fluid solver</td>
<td>Master Project - Monte Carlo Ray Tracer</td>
</tr>
<tr>
<td></td>
<td>Rendering Seminar</td>
<td>Rendering Seminar</td>
</tr>
<tr>
<td>Summer</td>
<td>Master Thesis - Research-oriented topic</td>
<td>Master Thesis - Research-oriented topic</td>
</tr>
</tbody>
</table>
Course Goals

– Global illumination concepts
– Ray tracing techniques
– Photorealistic rendering
– Requirements:
  – Key course in graphics and image processing
  – C / C++ / C#
  – Basics in linear algebra
Course Topics

– Aspects for efficient and high-quality rendering
  – Radiometric quantities
  – Rendering equation
  – Monte Carlo integration
  – Primitives (see Key course)
  – Ray traversal / ray shooting
  – Sampling / antialiasing
Material

– Slide sets on
  https://cg.informatik.uni-freiburg.de/teaching.htm
Material

– Matt Pharr, Greg Humphreys
  Physically Based Rendering
  Morgan Kaufmann

– Kevin Suffern
  Ray Tracing from the Ground Up
  A K Peters
Material

– Philip Dutre, Kavita Bala, Philippe Bekaert
  Advanced Global Illumination
  A K Peters

– Peter Shirley, R. Keith Morley
  Realistic Ray Tracing
  A K Peters
Tutorials / Exercises

– Every second Tuesday, starting on April 30
  – Check web page for changes
– Practical exercises
  – Development of ray tracing components
  – Check web page for information and example frameworks
Outline

– Organization
– Concepts
– Applications
– History
– Selected variants
– Components
Ray Tracing

- Computation of light that is transported along rays, in particular towards the sensor
- Tracing rays from a camera through pixel positions of a virtual image plane to compute the light that is transported along these rays

[Wikipedia: Ray Tracing]
Quantifying Light

– Radiance
  – Characterizes strength and direction of radiation / light
  – Is generated / emitted by light sources
  – Travels along lines / rays at infinite speed
  – Is preserved along lines in space
  – Is scattered or absorbed at surfaces
  – Is measured by sensors, i.e. is computed in computer-generated images
Light / Radiance

Light / radiance travels along rays

Light / radiance is emitted at light sources

Incoming light / radiance is absorbed and scattered at surfaces

Cameras capture light / radiance
Ray Tracing - Setting

Path 1
Outgoing radiance from light
Incoming radiance at surface
Direct illumination

Path 2
Outgoing radiance from light
Incoming radiance at surface
Direct illumination

Path 3
Outgoing radiance from surface
Incoming radiance at surface
Indirect illumination

Path 4
Outgoing radiance from surface
Incoming radiance at camera
Ray Tracing - Challenge

Path 4
Computation of outgoing radiance from surface towards camera is the main goal of a ray tracer.

Path 1, 2, 3 ...
Incoming / outgoing radiance at all other paths is required to compute radiance at path 4.

Path 3
Two surfaces illuminate each other. Outgoing radiance from q towards p depends on outgoing radiance from p towards q which depends on …
Reflection properties at surfaces can be described with a function \( f_r \) (BRDF, alternative to Phong model). How much incident light from a particular direction is reflected into a particular direction?

\[
L_i(p, \omega_i) \quad \text{Incoming radiance from direction } \omega_i \\
L_o(p, \omega_o) \approx f_r(p, \omega_i, \omega_o)L_i(p, \omega_i) \quad \text{Outgoing radiance into direction } \omega_o
\]

\( L_o(p, \omega_o) = \ldots \)
Rendered Equation

- The Rendering equation computes reflected radiance into a particular direction given incident radiance from all possible directions.

\[ L_i(p, \omega_i) \]

Incoming radiance from direction \( \omega_i \)

\[ L_i(p, \omega_i) \]

Position \( p \)

\[ L_o(p, \omega_o) \sim \sum \omega_i f_r(p, \omega_i, \omega_o) L_i(p, \omega_i) \]

Rendering equation

Outgoing radiance into direction \( \omega_o \), e.g., towards the camera, is a sum of weighted incident radiances.
Ray Tracing / Rendering Equation

- Ray tracers approximately solve the Rendering equation
  \[ L_o(\mathbf{p}, \omega_o) = L_e(\mathbf{p}, \omega_o) + \int_{2\pi} f_r(\mathbf{p}, \omega_i, \omega_o) L_i(\mathbf{p}, \omega_i) \cos \theta_i d\omega_i \]
- Outgoing radiance is the sum of emitted and reflected radiance
- Incident radiance - weighted with the BRDF \( f_r \) - is integrated over the hemisphere to compute the outgoing radiance
Ray Tracing / Rendering Equation

Path 4
Computation of outgoing radiance from surface towards camera corresponds to solving the Rendering equation

Path 3
Requires solving the Rendering equation

Many paths require solving the Rendering equation

Ray tracer variants differ in the approximation quality of the rendering equation: The more accurate, the more expensive.
Towards Realistic Images

- Capturing global illumination from all directions everywhere in a scene
- Less realistic images consider few and simple light sources
Towards Realistic Images

- Realistic reflection properties of materials
  - Surfaces are not perfectly diffuse or specular

\[ L_i(p, \omega_i) \sim f_r(p, \omega_i, \omega_o) L_i(p, \omega_i) \]

\[
L_o(p, \omega_o) = \ldots
\]

Next Limit / Maxwell Render
http://support.nextlimit.com/display/maxwelldocs/IOR+files
Ray Tracing - Capabilities

- Reflection
- Refraction
- Soft shadows
- Caustics
- Diffuse interreflections
- Specular interreflections
- Depth of field
- Motion blur

[sean.seanie, www.flickr.com]
rendered with POVray 3.7
Ray Tracing - Challenges

- Ray-object intersections
  - Spatial data structures for accelerated ray shooting
  - Dynamic scenes are particularly challenging
- Number of rays (quality vs. costs)
  - At ray-object intersections (solving the Rendering equation)
  - Per pixel (antialiasing)
- Recursion depth (quality vs. costs)
Outline

– Organization
– Concepts
– Applications
– History
– Selected variants
– Components
Areas

- Visual effects in movies and commercials
  - Major software packages have built-in ray tracers, e.g. Maya, 3ds Max (Autodesk), Houdini (Side Effects Software)
- Visualization of architectural design
  - Consideration of realistic indoor and outdoor illumination
- Automotive design
- Flight and car simulators
- Computer games
Software

- Mental ray (NVIDIA ARC)
- Maxwell Render (Next Limit Technologies)
- PreonLab (FIFTY2 Technology)
- Arnold (Solid Angle)
- POV-Ray
- Blender
- Pbtrt
- Mitsuba Renderer (Wenzel Jakob)
Examples

– Mental ray

Mies van der Rohe Farnsworth House
(Artist Alessandro Prodan)

Delta Tracing

[www.mentalimages.com]
Examples

– Mental ray
Examples

– Spellwork Pictures

https://www.youtube.com/watch?v=DMFhM4ZfRRE
Examples

– Spellwork Pictures

Modeling

Rendering
Examples

– Spellwork Pictures

Animation

Animation + Rendering
Examples

– Spellwork Pictures
Outline

– Organization
– Concepts
– Applications
– History
– Selected variants
– Components
Photorealistic Rendering

- Rasterization
  - 1965: rasterized lines (Bresenham)
  - 1967: rasterized flat-shaded polygons (Wylie)
  - 1971: Gouraud shading
  - 1973: Phong illumination model
  - 1974: texture mapping (Blinn)
  - 1974: depth buffer (Catmull)
  - 1975: Phong shading
  - 1977: shadow volumes (Crow)
  - 1978: shadow maps (Williams)
Photorealistic Rendering

- Ray tracing
  - 1968: viewing and shadow rays, non-recursive (Appel)
- Recursive ray tracing
  - 1980: ideal reflection, refraction (Whitted)
- Rendering equation
  - 1986: general description of light distribution (Kajiya)
    - \[ L_o(p, \omega_o) = L_e(p, \omega_o) + \int_{2\pi} f_r(p, \omega_i, \omega_o)L_i(p, \omega_i) \cos \theta_i d\omega_i \]
    - Arbitrary global illumination effects can be considered
- Distribution ray tracing
  - 1986: Monte-Carlo evaluation of integrals (Cook)
  - Approximately solves the Rendering equation
Ray Tracing vs. Rasterization

- Rasterization
  - Given a set of viewing rays and a primitive, efficiently compute the subset of rays hitting the primitive
  - Loop over all primitives
  - Implicit ray representation

- Ray tracing
  - Given a viewing ray and a set of primitives, efficiently compute the subset of primitives hit by the ray
  - Loop over all viewing rays
  - Explicit ray representation
Ray Tracing vs. Rasterization

– Solve the same problem

Ray Tracers
Compute ray-object intersections to estimate $q$ from $p$

Rasterizers
Apply modelview, projection and viewport transform to $p$ in order to estimate $q$
Ray Tracing vs. Rasterization

- Rasterization
  - Well-established, parallelizable algorithms
  - Popular in interactive applications
  - Specialized realizations of global illumination effects
- Ray tracing
  - Natural incorporation of numerous visual effects
  - Unified algorithms for global illumination effects
  - Trade-off between quality and performance
Outline

– Organization
– Concepts
– Applications
– History
– Selected variants
– Components
Ray Tracing

- Ray generation
- Ray traversal
- Intersection
- Shading
- Frame buffer

Viewing rays return a radiance value. Shadow rays return an occlusion value which is also a radiance.
Ray Tracing

Recursive Ray Tracing

– Ray generation
– Ray traversal
– Intersection
– Shading
– Frame buffer

Viewing rays return a radiance.
Shadow rays return an occlusion / radiance.
Reflection and refraction rays return a radiance.
Recursive Ray Tracing

Distribution Ray Tracing (Stochastic Ray Tracing)

Consider more than one randomly perturbed reflection / refraction ray at a surface point, e.g.

\[ L_i(p, \omega_i) \]
Incoming radiance from direction \( \omega_i \)

\[ L_i(p, \omega_i) \]

\[ L_i(p, \omega_i) \]

\[ L_o(p, \omega_o) \sim \sum_{\omega_i} f_r(p, \omega_i, \omega_o) L_i(p, \omega_i) \]

Rendering equation
Outgoing radiance into direction \( \omega_o \), e.g., towards the camera, is a sum of weighted incident radiances
**Distribution Ray Tracing (Stochastic Ray Tracing)**

- **Examples**
  - Distributing rays over the hemisphere to capture the incident radiance (Monte Carlo integration for solving the rendering equation)
  - Shadow rays over an area light source for soft shadows
  - Perturbing ray origins to enable depth-of-field effects
  - Rays per pixel over time to get motion blur effects
Distribution Ray Tracing

Outline

– Organization
– Concepts
– Applications
– History
– Selected variants
– Components
Overview

- Camera generates viewing rays
- Location and radiant intensity of light sources
- Ray-object intersections
- Visibility of light sources
- Surface scattering model
- Recursion
- Participating media
Camera

- Generates viewing rays
- Pinhole camera with a virtual image plane (near plane) in front of the pinhole
- Pinhole is referred to as the eye
- For a position on the image, a camera simulator generates rays along which light is known to contribute to that position, e.g.
  - A ray from the eye through the image position
  - A ray that considers one or multiple lenses
Light Distribution

– Determining the amount of light energy arriving at the differential area around the intersection point
– Therefore, geometric and radiometric distribution of light has to be known
  – For emitted light from point light sources
  – For emitted light from area light sources
  – For reflected light for object surfaces
Ray-Object Intersection

- Determine whether a ray intersects an object
- Determine the first intersection (closest to the ray origin)
- Determine further geometric information, e.g.
  - Surface normal
  - Partial derivatives of position and normal with respect to the local surface parameterization
- Efficient implementations heavily rely on spatial data structures
Visibility

- Determine whether a light source is visible from a surface point to be shaded
- Shadow rays are casted from the object to the light source
- If the distance to the first ray-object intersection along this ray is shorter than the distance to the light source, the surface point is in shadow
Surface Scattering

- Computes radiance scattered back along a viewing ray
- From previous components, we have
  - Ray-object intersection and further geometric information
  - Information on incident lighting
- We further know appearance properties, e.g.
  - A local illumination model
  - A Bidirectional Reflectance Distribution Function BRDF (how much light is reflected from an incoming direction to an outgoing direction)
Recursion

– Recursively invoke the ray-tracing components
– If, e.g., a viewing ray hits a mirror
  – The viewing ray can be reflected at the mirror
  – The ray-tracing routine is applied to the reflected ray
  – The resulting radiance is considered as additional illumination of the mirror
– To approximately solve the rendering equation,
  – Various rays are generated that sample the hemisphere above the surface (Monte Carlo integration)
Participating Media

- E.g., smoke, fog, dust
- In vacuum, radiance along a ray does not change
- In presence of participating media, light can be attenuated or extinguished by scattering it in different directions
- Participating media can be characterized by its transmittance