Advanced Computer Graphics

Summary

Computer Graphics
Computer Science Department
University of Freiburg

SS 11
Outline

- introduction
- transformations
- ray-object intersections
- intersection acceleration
- aliasing
- sampling and reconstruction
- radiometry
- materials
- rendering equation
- path tracing
Ray Tracing - Concept

- tracing rays of light through a scene to compute the radiance that is perceived by a sensor
- tracing a path from a camera through a pixel position of a virtual image plane to compute the color of an object that is visible along the path
Ray Tracing - Motivation

- light is modeled as geometric rays
  - travels in straight lines (e.g., diffraction / bending is not considered)
  - travels at infinite speed (ray tracing computes steady state of light)
  - is emitted by light sources
  - is absorbed or scattered / reflected at surfaces

- radiance
  - characterizes strength and direction of radiation / light
  - is measured by sensors
  - is computed in computer-generated images
  - is preserved along lines in space
  - does not change with distance
Ray Tracing - Capabilities

- reflection
- refraction
- soft shadows
- caustics
- diffuse interreflections
- specular interreflections
- depth of field
- motion blur

[sean.seanie, www.flickr.com]
derived with POVRay 3.7
Ray Tracing - Challenges

- efficient ray shooting
  - ray shooting algorithms build spatial data structures to accelerate ray shooting queries
  - dynamic scenes are more challenging compared to static scenes
- optimal number of rays
  - per pixel
    - for antialiasing
  - at ray-object intersections
    - for interreflections
    - soft shadows
    - approximate evaluation of the rendering equation
- optimal recursion depth
Ray Tracing - Components

- camera
  - generates viewing rays
- light distribution
  - location and radiant intensity of light sources
- ray-object intersection
  - with additional information, e.g., normal
- visibility
  - of light sources
- surface scattering model
  - describes how light interacts with a surface
- recursion
  - important for reflections on shiny surfaces
- ray propagation
  - variation of radiance in, e.g., fog or smoke
Outline

- introduction
- transformations
- ray-object intersections
- intersection acceleration
- aliasing
- sampling and reconstruction
- radiometry
- materials
- rendering equation
- path tracing
Coordinate Spaces

- **object space**
  - coordinate system in which geometric primitives are defined
  - object spaces are object-dependent

- **world space**
  - objects, lights are placed / transformed into world space
  - object-to-world transformations allow to arbitrarily place objects, lights relative to each other

- **camera space**
  - a space with a specific camera setting, e. g. camera at the origin, viewing along z-axis, y-axis is up direction
  - useful for simplified computations (similar to the rendering pipeline)
  - camera is placed in world space with a view transformation
  - inverse view transform is used to get from world to camera space
Homogeneous Coordinates of Points

- \((x, y, z, w)^T\) with \(w \neq 0\) are the homogeneous coordinates of the 3D point \((\frac{x}{w}, \frac{y}{w}, \frac{z}{w})^T\)

- \((\lambda x, \lambda y, \lambda z, \lambda w)^T\) represents the same point \((\frac{\lambda x}{\lambda w}, \frac{\lambda y}{\lambda w}, \frac{\lambda z}{\lambda w})^T = (\frac{x}{w}, \frac{y}{w}, \frac{z}{w})^T\) for all \(\lambda\) with \(\lambda \neq 0\)

- examples
  - \((2, 3, 4, 1) \sim (2, 3, 4)\)
  - \((2, 4, 6, 1) \sim (2, 4, 6)\)
  - \((4, 8, 12, 2) \sim (2, 4, 6)\)
Homogeneous Coordinates of Vectors

- For varying $w$, a point $(x, y, z, w)^T$ is scaled and the points $(\frac{x}{w}, \frac{y}{w}, \frac{z}{w})^T$ represent a line in 3D space.

- The direction of this line is characterized by $(x, y, z)^T$.

- For $w \rightarrow 0$, the point $(\frac{x}{w}, \frac{y}{w}, \frac{z}{w})^T$ moves to infinity in the direction $(x, y, z)^T$.

- $(x, y, z, 0)^T$ is a point at infinity in the direction of $(x, y, z)^T$.

- $(x, y, z, 0)^T$ is a vector in the direction of $(x, y, z)^T$.
Homogeneous Representation of Transformations

- **linear transformation**

  \[
  \begin{pmatrix}
  m_{00} & m_{01} & m_{02} \\
  m_{10} & m_{11} & m_{12} \\
  m_{20} & m_{21} & m_{22}
  \end{pmatrix}
  \begin{pmatrix}
  p_x \\
  p_y \\
  p_z
  \end{pmatrix}
  \sim
  \begin{pmatrix}
  m_{00} & m_{01} & m_{02} & 0 \\
  m_{10} & m_{11} & m_{12} & 0 \\
  m_{20} & m_{21} & m_{22} & 0 \\
  0 & 0 & 0 & 1
  \end{pmatrix}
  \begin{pmatrix}
  p_x \\
  p_y \\
  p_z \\
  1
  \end{pmatrix}
  \]

- **affine transformation**

- representing rotation, scale, shear, translation

- **projective components**

  \( p \) are zero for affine transformations
Planes and Normals

- Planes can be represented by a surface normal \( \mathbf{n} \) and a point \( \mathbf{r} \). All points \( \mathbf{p} \) with \( \mathbf{n} \cdot (\mathbf{p} - \mathbf{r}) = 0 \) form a plane.

\[
\begin{align*}
n_x p_x + n_y p_y + n_z p_z + (-n_x r_x - n_y r_y - n_z r_z) &= 0 \\
n_x p_x + n_y p_y + n_z p_z + d &= 0 \\
(n_x \ n_y \ n_z \ d) (p_x \ p_y \ p_z \ 1)^T &= 0 \\
(n_x \ n_y \ n_z \ d) \mathbf{A}^{-1} \mathbf{A} (p_x \ p_y \ p_z \ 1)^T &= 0
\end{align*}
\]

- The transformed points \( \mathbf{A} (p_x \ p_y \ p_z \ 1)^T \) are on the plane represented by

\[
(n_x \ n_y \ n_z \ d) \mathbf{A}^{-1} = ((\mathbf{A}^{-1})^T (n_x \ n_y \ n_z \ d)^T)^T
\]

- If a surface is transformed by \( \mathbf{A} \), its normal is transformed by \( (\mathbf{A}^{-1})^T \)
Rays

- for ray-object intersections,
  - objects are commonly not transformed
  - instead, rays are transformed with the inverse of the object-to-camera space transformation

- algorithm
  - apply the inverse transform to the ray
  - compute intersection and normal
  - transform the intersection and the normal
Matrix Decomposition

- if keyframe transformations are composed of translation, rotation, and scale, these components have to be decomposed and interpolated independently.
- Projective components are not considered, (but could be extracted easily).
- Translation can be extracted as:

\[
M = \begin{pmatrix}
  m_{11} & m_{12} & m_{13} & t_x \\
  m_{21} & m_{22} & m_{23} & t_y \\
  m_{31} & m_{32} & m_{33} & t_z \\
  0 & 0 & 0 & 1 \\
\end{pmatrix} = \begin{pmatrix}
  1 & 0 & 0 & t_x \\
  0 & 1 & 0 & t_y \\
  0 & 0 & 1 & t_z \\
  0 & 0 & 0 & 1 \\
\end{pmatrix} \times \begin{pmatrix}
  m_{11} & m_{12} & m_{13} & 0 \\
  m_{21} & m_{22} & m_{23} & 0 \\
  m_{31} & m_{32} & m_{33} & 0 \\
  0 & 0 & 0 & 1 \\
\end{pmatrix}
\]
Rotation Extraction

- approaches
  - QR decomposition (extracted rotations are not meaningful)
  - SVD (extracted rotations are not meaningful and rather unstable)
  - Polar decomposition

- Polar decomposition
  - efficient to compute
  - extracts rotation $R$ that is closest to the original transformation $M$
  - Find $R$ minimizing $\|R - M\|_F^2$
    subject to $R^T R - I = 0$
    with $\|R - M\|_F^2 = \sum_{i,j} (r_{i,j} - m_{i,j})^2$
    being the Frobenius matrix norm
  - $M = R(-I)S$
    - $S$ is symmetric, positive definite (scale in a potentially rotated frame)
    - shear cannot be extracted

University of Freiburg - Computer Science Department - Computer Graphics
Outline

- introduction
- transformations
- ray-object intersections
- intersection acceleration
- aliasing
- sampling and reconstruction
- radiometry
- materials
- rendering equation
- path tracing
Motivation

- rays
  - a half-line specified by an origin \( o \) and a direction \( d \)
  - parametric form \( r(t) = o + td \) with \( 0 \leq t \leq \infty \)
- nearest intersection with all objects has to be computed, i.e., intersection with minimal \( t \geq 0 \)
- in implementations, usually \( t \geq \varepsilon \) to avoid self-intersections, e.g., if rays start at object surfaces
Implicit Surfaces

- Implicit functions implicitly define a set of surface points.
- For a surface point \( (x, y, z) \), an implicit function \( f(x, y, z) \) is zero.
- An intersection occurs if a point on a ray satisfies the implicit equation \( f(x, y, z) = f(r(t)) = f(o + td) = 0 \).
- For example, all points \( p \) on a plane with surface normal \( n \) and offset \( r \) satisfy the equation \( n \cdot (p - r) = 0 \).
- The intersection with a ray can be computed based on \( t \):
  \[
  n \cdot (o + td - r) = 0
  \]
  \[
  t = \frac{(r-o) \cdot n}{n \cdot d}
  \]
  if \( d \) is not orthogonal to \( n \).
Implicit Surfaces - Normal

- perpendicular to the surface
- given by the gradient of the implicit function

\[ \mathbf{n} = \nabla f(\mathbf{p}) = \left( \frac{\partial f(\mathbf{p})}{\partial x}, \frac{\partial f(\mathbf{p})}{\partial y}, \frac{\partial f(\mathbf{p})}{\partial z} \right) \]

- e.g., for a point \( \mathbf{p} \) on a plane \( f(\mathbf{p}) = \mathbf{n} \cdot (\mathbf{p} - \mathbf{r}) = 0 \)
  \[ \mathbf{n} = \nabla f(\mathbf{p}) = (n_x, n_y, n_z) \]
Parametric Surfaces

- are represented by functions with 2D parameters
  \[ x = f(u, v) \quad y = g(u, v) \quad z = h(u, v) \]

- intersection is computed using a linear system with three equations and three unknowns \( t, u, v \)
  \[ o_x + td_x = f(u, v) \quad o_y + td_y = g(u, v) \quad o_z + td_z = h(u, v) \]

- normal vector
  \[ n(u, v) = \left( \frac{\partial f}{\partial u}, \frac{\partial g}{\partial u}, \frac{\partial h}{\partial u} \right) \times \left( \frac{\partial f}{\partial v}, \frac{\partial g}{\partial v}, \frac{\partial h}{\partial v} \right) \]
Triangle

- parametric representation (based on barycentric coords)
  \[ p(b_1, b_2) = (1 - b_1 - b_2)p_0 + b_1p_1 + b_2p_2 \]
  \[ b_1 \geq 0 \quad b_2 \geq 0 \quad b_1 + b_2 \leq 1 \]

- intersection is computed using a linear system
  \[ o + td = (1 - b_1 - b_2)p_0 + b_1p_1 + b_2p_2 \]

- solution (for non-degenerated triangles not parallel to the ray)
  \[
  \begin{pmatrix}
  t \\
  b_1 \\
  b_2
  \end{pmatrix}
  = \frac{1}{(d \times e_2) \cdot e_1}
  \begin{pmatrix}
  (s \times e_1) \cdot e_2 \\
  (d \times e_2) \cdot s \\
  (s \times e_1) \cdot d
  \end{pmatrix}
  \]
  \[ e_1 = p_1 - p_0 \]
  \[ e_2 = p_2 - p_0 \]
  \[ s = o - p_0 \]
**Axis-Aligned (Bounding) Box (AABB)**

- boxes are defined by slabs
- intersections of rays with slabs are analyzed to check for ray-box intersection
  - e.g. non-overlapping ray intervals within different slabs indicate that the ray misses the box

The general intersection with an x-slab is given by:

\[
\mathbf{n} \cdot (\mathbf{o} + t\mathbf{d} - \mathbf{r}) = 0
\]

where \(t = \frac{(\mathbf{r} - \mathbf{o}) \cdot \mathbf{n}}{\mathbf{n} \cdot \mathbf{d}}\) and \(\mathbf{n}\) is the normal vector, \(\mathbf{o}\) is the origin of the ray, \(\mathbf{r}\) is the ray direction, and \(\mathbf{d}\) is the direction of the slab.

For the specific case of the x-slab, the equation becomes:

\[
(1, 0, 0)^T \cdot (\mathbf{o} + t\mathbf{d} - (x_{0,1}, 0, 0)^T) = 0
\]

The interval for \(t\) is:

\[
t_{x_{\min}, x_{\max}} = \frac{(x_{0,1} - o_x)}{d_x}
\]
overlapping ray intervals indicate intersections, e. g. $t_{xmin} < t_{ymax} \land t_{xmax} > t_{ymin} \Rightarrow$ intersection (largest entering value $t$ is smaller than the smallest leaving value $t$, only positive values $t$ are considered)
Outline

- introduction
- transformations
- ray-object intersections
- intersection acceleration
- aliasing
- sampling and reconstruction
- radiometry
- materials
- rendering equation
- path tracing
Spatial Data Structures Examples

space subdivision with a uniform grid (space oriented)
bounding volume hierarchy with spheres (object oriented)
Spatial Data Structures

Efficiency

- determined by
  - generation
  - query
  - update

- generation is usually a pre-processing step
- query implements a traversal of the data structure to compute the first intersection of the scene with a ray
- update is only relevant for dynamic scenes
- static scenes: efficiency dominated by the query
- dynamic scenes: efficiency is determined by query and update
Spatial Data Structures Implementation

- depending on the query, different implementations can be appropriate
- e.g. for uniform grids
  - explicit representation
  - spatial hashing
  - compact hashing
  - index sort
  - z-index sort
- e.g. for bounding volume hierarchies
  - priority queues
  - skip lists
Bounding Volume Hierarchy
Ray Traversal - Priority Queue / Heap

- B - bounding volume of the root node
- heap - Min Heap / Priority List

heap = empty; intersection = inf;
if B.intersection(ray)<intersection then heap.add(B);
while heap.notEmpty() and heap.min.intersection(ray)<intersection do
  cand = heap.min; heap.remove (heap.min);
  if cand.leaf() then
    intersection = cand.minIntersection(ray);
  else
    foreach cand.child do
      if cand.child.intersection(ray)<intersection then
        heap.add(cand.child);
    return intersection;

  return intersection;

depth-first, near-to-far
Uniform Grids - Traversal

\( \partial x, \partial y, \partial z \)
parametric distance along the ray between two grid planes perpendicular to \( x, y, z \) (infinite, if a ray is parallel to a principal axis of the grid)

\( dx, dy, dz \)
parametric value of the ray at the next intersection with a grid plane perpendicular to \( x, y, z \) (infinite, if a ray is parallel to a principal axis of the grid)

\( i, j, k \)
indices of the current grid cell

\( px, py, pz \in \{-1, 1\} \)
increments of cell indices if a grid plane is intersected

Simplified 2D algorithm
\[
px = +1 \\
py = -1 \\
\text{initialize } \partial x, \partial y, dx, dy, i, j \\
\text{repeat} \\
\quad \text{if } dx \leq dy \text{ then} \\
\quad \quad \text{begin} \\
\quad \quad \quad i := i + px; \\
\quad \quad \quad dx := dx + \partial x; \\
\quad \quad \text{end;} \\
\quad \text{if } dx \geq dy \text{ then} \\
\quad \quad \text{begin} \\
\quad \quad \quad j := j + py; \\
\quad \quad \quad dy := dy + \partial y; \\
\quad \quad \text{end;} \\
\text{until intersection in cell } i, j;
\]
kd-Trees - Traversal

- check the global bounding box for intersection
- compute the intersection with the plane represented in the root node
- recursively traverse the two children in front-to-back order
- if a node is a leaf, check the primitives
- stop, if a ray-primitive intersection has been found or no further spaces have to be processed
Uniform Grid
Linked List and Dynamic Array

2D grid and linearized representation

Linked list:
- minimal memory overhead
- bad locality
- frequent memory allocations
- single primitives can be inserted

Dynamic array:
- keeps track of capacity and size
- less memory efficient
- improved data locality
- minimized memory allocations
- single primitives can be inserted
Uniform Grid
Compact Grid

2D grid and linearized representation

Compact grid:
- two arrays
- useful for dynamic scenes
- generation rather expensive
- generation + traversal more efficient
  than linked lists or dynamic arrays
- single primitives cannot be inserted or deleted

offset of the corresponding primitive set
concatenation of all primitive sets

L[ C[i] ] - first primitive of cell i
C[i+1]-C[i] - number of primitives in cell i
Uniform Grid
Hashed Grid

- to avoid hash collisions, perfect hashing can be used
- e.g., row displacement compression
  - identify occupied grid cells
  - compute an offset for each row (O) to store all rows with non-overlapping occupied cells in a hash table (H)

  ▪ index into the hash table is $O \ [\text{row}] + \text{column}$
Outline

- introduction
- transformations
- ray-object intersections
- intersection acceleration
- aliasing
- sampling and reconstruction
- radiometry
- materials
- rendering equation
- path tracing
Motivation

- sampling and reconstruction
- inappropriate sampling can cause artifacts in reconstructed functions

[Foley, van Dam, Feiner, Hughes]
if the sampling frequency is too low, the replicated copies of the spectra overlap and the spectrum of the original function cannot be reconstructed.

- insufficient sampling
  - overlapping spectra

- sufficient sampling
  - non-overlapping spectra
  - reconstruction
  - original function in case of sufficient sampling
Antialiasing

- In texturing, textures are filtered according to the given sampling rate
  - Prefiltering

- In ray tracing, the sampling rate and sampling patterns are adapted
  - Nonuniform sampling: tends to turn regular aliasing patterns into noise
  - Adaptive sampling: use more samples in case of large variations between adjacent samples (might still miss high frequencies, small details)

- In ray tracing, radiance at a pixel position is commonly reconstructed from samples within the pixel area and samples in adjacent pixels
Outline

- introduction
- transformations
- ray-object intersections
- intersection acceleration
- aliasing
- sampling and reconstruction
- radiometry
- materials
- rendering equation
- path tracing
Introductory Example

- **regular sampling**
  - subdivide the pixel area into a regular grid
  - trace a ray per grid cell

- **box filter**
  - compute the average incident radiance

- **effect**
  - reduced aliasing due to a higher sampling rate
  - computationally expensive

- **goal**
  - efficient sampling patterns and filter with reduced aliasing
Good Sampling Characteristics

- uniform distribution over the area
- uniform distribution of the projections in x- and y-direction
- maximal minimum distance between samples
  - avoids over- and undersampling of partial areas
- but:
- no regular spacing in x- and / or y-direction
- minimal number of samples with acceptable noise
- number of required samples depends on the application
  - sampling of a pixel area
  - sampling of time for motion blur
  - sampling of the lens area for depth of field
  - sampling of a solid angle for glossy reflection
Random Sampling

- replaces aliasing with noise
- requires less samples compared to regular sampling
- non-uniform sampling of partial areas and projections
Stratified Sampling

- pixel area is subdivided into $n \times n$ strata
- one sample per stratum
- stratified (jittered) sampling reduces clustering of samples, non-uniform sampling of areas and missing of small details
**n-Rooks / Latin Hypercube Sampling (LHS)**

- generate one jittered sample per row and column
- randomly shuffle the samples in x- or y-direction
- uniform distribution of the projections in x- and y-direction
- can generate an arbitrary number of stratified samples
Poisson Disk Sampling

- generate a sequence of random samples
- reject a sample, if it is too close to an existing sample
- rather expensive to compute (dart-throwing)

```plaintext
i = 0
while i < N
    x_i = randfrom [0,1)
    y_i = randfrom [0,1)
    reject = false
    for j=0 to i-1
        if (x_i - x_j)^2 + (y_i - y_j)^2 < d^2
            reject = true
            break
    if not reject
        i = i+1
```
**Best-Candidate Sampling**

- generate a larger number of random candidate samples within the entire sampling area
- choose the candidate farthest to previously computed samples

[Pharr, Humphreys]
Low-Discrepancy Sequences
Hammersley vs. Halton

Hammersley sampling

Halton sampling

[Wong, Luk, Heng]
Filtering Principle

- Pixel values are reconstructed from the radiance values of adjacent samples

\[ I(x, y) = \frac{\sum_i f(x-x_i, y-y_i)L(x_i, y_i)}{\sum_i f(x-x_i, y-y_i)} \]

- \( f \) is a filter function that weights the influence of a sample \((x_i, y_i)\) according to its distance to \((x, y)\)

The extent can be larger than the pixel size.

[Pharr, Humphreys]
**Truncated Sinc Filter**

- reasonably good reconstruction
- introduces blurring

Sinc filter applied to a step function (some ringing)

Sinc filter applied to a sinusoidal function with increasing frequency ⇒ aliasing due to undersampling

[Pharr, Humphreys]

University of Freiburg - Computer Science Department - Computer Graphics
Outline

- introduction
- transformations
- ray-object intersections
- intersection acceleration
- aliasing
- sampling and reconstruction
- radiometry
- materials
- rendering equation
- path tracing
Radiometric Quantities
Energy, Flux, Flux Density

- radiant energy $Q$
  - photons have some radiant energy inversely proportional to their wavelength

- radiant flux $\Phi$, radiant power $P$
  - rate of flow of radiant energy per unit time: $\Phi = \frac{dQ}{dt}$
  - e.g., overall energy of photons emitted by a source per time

- radiant flux density (irradiance, radiant exitance)
  - radiant flux per unit area: $E = \frac{d\Phi}{dA}$
  - rate at which radiation is incident on, or exiting a flat surface area $dA$
  - describes strength of radiation with respect to a surface area (existing or virtual surface)
  - no directional information
Radiometric Quantities

Radiance

- **radiance**
  - radiant flux per unit solid angle per unit projected area incident on, emerging from, passing through a point of a surface in a certain direction:
    \[
    L = \frac{d^2\Phi}{dA \cos \theta d\omega} = \frac{d^2\Phi}{dA d\omega}
    \]
  - describes strength and direction of radiation
  - e.g., a constant radiance in all directions corresponds to a radiant flux in direction \( \theta \) that is proportional to \( \cos \theta \), but constant flux in all directions results in different radiance

[Kevin Boulanger, Ph.D. thesis]
Radiometric Quantities
Properties

- Lambert's Cosine Law
- Inverse Square Law
- Conservation of Radiance
Outline

- introduction
- transformations
- ray-object intersections
- intersection acceleration
- aliasing
- sampling and reconstruction
- radiometry
- materials
- rendering equation
- path tracing
**Bidirectional Reflectance Distribution Function BRDF**

- In its simplest form, a BRDF represents, for each incoming angle of light, the portion of light that is scattered in each outgoing angle.
- Incoming and outgoing angle are 2D.
- BRDF is wavelength-dependent:
  - E.g., RGB or samples.
- BRDF can be spatially invariant:
  - I.e., object surface has constant absorbance spectrum and roughness.
- BRDF can be spatially variant:
  - E.g., textured surface.
BRDF Definition

- differential irradiance at p from a differential cone of directions $d\omega_i$ is $dE(\omega_i) = L_i(\omega_i) \cos \theta_i d\omega_i$
- differential radiance reflected from p into direction $\omega_o$ is proportional to irradiance $dL_o(\omega_o) \propto dE(\omega_i)$
- BRDF $f_r$ is the constant of proportionality $f_r(\omega_i, \omega_o) = \frac{dL_o(\omega_o)}{dE(\omega_i)} = \frac{dL_o(\omega_o)}{L_i(\omega_i) \cos \theta_i d\omega_i}$

In case of RGB, $f_r$ is a three-dimensional vector, operations are performed component-wise

**BRDF Properties**

- should be positive $f_r(\omega_i, \omega_o) \geq 0$

- Helmholtz reciprocity should be obeyed $f_r(\omega_o, \omega_i) = f_r(\omega_i, \omega_o)$
  - incident and reflected light can be reversed
  - important for algorithms that reverse light paths

- energy should be conserved $\forall \omega_i : \int_{2\pi+} f_r(\omega_i, \omega_o) \cos \theta_o d\omega_o \leq 1$

- linearity
  - if a material is defined as a combination of BRDFs, the contributions of the BRDFs are added for the total outgoing radiance
Reflection Equation

- outgoing radiance in direction $\omega_o$ due to incident radiance from direction $\omega_i$
  
  \[ \text{d}L_o(\omega_o) = f_r(\omega_i, \omega_o)L_i(\omega_i)\cos\theta_i\text{d}\omega_i \]

- integration over the hemisphere of incoming directions gives the reflection equation
  
  \[ L_o(\omega_o) = \int_{2\pi} f_r(\omega_i, \omega_o)L_i(\omega_i)\cos\theta_i\text{d}\omega_i \]

- incident radiance - weighted with the BRDF - is integrated over the hemisphere to compute the outgoing radiance

- fundamental equation in rendering
Reflectance Models

- outgoing radiance due to incoming radiance from one direction

perfect diffuse

glossy / specular

perfect specular

retro reflective

University of Freiburg - Computer Science Department - Computer Graphics

[Pharr, Humphreys]
Diffuse BRDF

- outgoing diffuse radiance $L_{o,d}$ is isotropic
  
  $$L_{o,d} = \int_{2\pi} f_r(\omega_i, \omega_o) L_i(\omega_i) \cos \theta_i d\omega_i = \text{const}$$

- $f_r$ has to be constant  
  
  $$L_{o,d} = f_{r,d} \int_{2\pi} L_i(\omega_i) \cos \theta_i d\omega_i = f_{r,d} E$$

- $f_{r,d} = \frac{L_{o,d}}{E}$

- directional-hemispherical reflectance (represents color / absorbance of an diffuse surface)

  $$\rho_d = \int_{2\pi} f_r(\omega_i, \omega_o) \cos \theta_o d\omega_o = f_{r,d} \int_{2\pi} \cos \theta_o d\omega_o = f_{r,d} \pi$$

  $$f_{r,d} = \frac{1}{\pi} \rho_d \quad 0 \leq \rho_d \leq 1 \quad \text{energy conservation}$$
Perfectly Specular Reflectance

- Each ray of incident flux produces one ray of outgoing flux into a specular direction
  \[ L_{o,r}(\omega_o) = f_{r,r}(\omega_i, \omega_o) L_i(\omega_i) \]
- Absorbance of the specular surface is characterized by \(0 \leq \rho_r \leq 1\)
- BRDF
  \[ f_{r,r} = \rho_r \frac{1}{\cos \theta_i \sin \theta_i} \delta(\theta_o - \theta_i) \delta(\phi_o \pm \pi - \phi_i) \]
- Therefore
  \[ L_{o,r}(\omega_o) = \int_{2\pi} \rho_r \frac{1}{\cos \theta_i \sin \theta_i} \delta(\theta_i - \theta_o) \delta(\phi_i \pm \pi - \phi_o) L_i(\omega_i) \cos \theta_i d\omega_i \]
  \[ L_{o,r}(\omega_o) = \rho_r L_i(\omega_i) \]
Glossy / Specular Reflectance (normalized Phong)

- energy conservation requires $\int_{2\pi} f_{r,s}(\omega_i, \omega_o) \cos \theta_o \, d\omega_o \leq 1$
- as $\int_{2\pi} f_{r,s}(\omega_i, \omega_o) \cos \theta_o \, d\omega_o = \rho_s \frac{2\pi}{e+2}$
- the Phong BRDF should be normalized
  $f_{r,s}(\omega_i, \omega_o) = \rho_s \frac{e+2}{2\pi} \left((-\omega_i + 2(\mathbf{n} \cdot \omega_i) \cdot \mathbf{n}) \cdot \omega_o \right)^e$
  with $0 \leq \rho_s \leq 1$
- normalized Phong (diffuse and specular component)
  $f_{r,d} + f_{r,s}(\omega_i, \omega_o) = \rho_d \frac{1}{\pi} + \rho_s \frac{e+2}{2\pi} \left((-\omega_i + 2(\mathbf{n} \cdot \omega_i) \cdot \mathbf{n}) \cdot \omega_o \right)^e$
  with $0 \leq \rho_d + \rho_s \leq 1$
Outline

- introduction
- transformations
- ray-object intersections
- intersection acceleration
- aliasing
- sampling and reconstruction
- radiometry
- materials
- rendering equation
- path tracing
Reflection and Rendering Equation

- reflection equation at point $p$ for reflective surfaces
  - $L_o(p, \omega_o) = \int_{2\pi} f_r(p, \omega_i, \omega_o) L_i(p, \omega_i) \cos \theta_i d\omega_i$
  - incident radiance - weighted with the BRDF - is integrated over the hemisphere to compute the outgoing radiance
  - expresses energy balance between surfaces
  - outgoing radiance from a surface can be incident to another surface

- rendering equation at point $p$ for reflective surfaces
  - $L_o(p, \omega_o) = L_e(p, \omega_o) + \int_{2\pi} f_r(p, \omega_i, \omega_o) L_i(p, \omega_i) \cos \theta_i d\omega_i$
  - adds emissive surfaces to the reflection equation
  - exitant radiance is the sum of emitted and reflected radiance
  - expresses the steady state of radiance in a scene including light sources
Forms of the Rendering Equation

- hemisphere form
  \[ L_o(p, \omega_o) = L_e(p, \omega_o) + \int_{2\pi} f_r(p, \omega_i, \omega_o) L_o(r_c(p, \omega_i), -\omega_i) \cos \theta_i d\omega_i \]

- area form
  - p is a sample point on a surface dA
  - visibility function
    \[ \forall(p, p') : V(p, p') = \begin{cases} 
    1 & \text{if } p \text{ and } p' \text{ see each other} \\
    0 & \text{if } p \text{ and } p' \text{ do not see each other} 
  \end{cases} \]
  - solid angle vs. area
    \[ d\omega_i = \frac{\cos \theta' dA}{\|p' - p\|^2} \]
  - \( \cos \theta' = n' \cdot -\omega_i \)

\[ L_o(p, \omega_o) = L_e(p, \omega_o) + \int_{2\pi} f_r(p, \omega_i, \omega_o) L_o(p', -\omega_i) \frac{\cos \theta_i \cos \theta'}{\|p' - p\|^2} V(p, p') dA \]
Monte Carlo Integration

- approximately evaluate the integral
  \[ \int_{2\pi} \int f_r(p, \omega_i, \omega_o) L_o(r_c(p, \omega_i), -\omega_i) \cos \theta_i d\omega_i \]
  by
  - randomly sampling the hemisphere
  - tracing rays into the sample directions
  - computing the incoming radiance from the sample directions

- challenge
  - approximate the integral as exact as possible
  - trace as few rays as possible
  - trace relevant rays
    - for diffuse surfaces, rays in normal direction are more relevant than rays perpendicular to the normal
    - for specular surfaces, rays in reflection direction are relevant
    - rays to light sources are relevant
Monte Carlo Estimator

Uniform Random Variables

- motivation: approximation of the integral in the rendering equation
- goal: computation of \( \int_{a}^{b} f(x) \, dx \)
- uniformly distributed random variables \( X_i \in [a, b] \)
- probability density function \( p(x) = \frac{1}{b-a} \)
- Monte Carlo estimator \( F_N = \frac{b-a}{N} \sum_{i=1}^{N} f(X_i) \)
- expected value of \( F_N \) is equal to the integral \( \int_{a}^{b} f(x) \, dx \)
  - \( E[F_N] = \int_{a}^{b} f(x) \, dx \)
- variance \( V = \frac{1}{N-1} \sum_{i=1}^{N} [f(X_i) - E[F_N]]^2 \)
- convergence rate of \( O(\sqrt{N}) \)
- independent from the dimensionality
  \( \Rightarrow \) appropriate for high-dimensional integrals
Monte Carlo Estimator
Non-uniform Random Variables

- Monte Carlo estimator \( F_N = \frac{1}{N} \sum_{i=1}^{N} \frac{f(X_i)}{p(X_i)} \) \( \quad p(X_i) \neq 0 \)

- \( E[F_N] = E \left[ \frac{1}{N} \sum_{i=1}^{N} \frac{f(X_i)}{p(X_i)} \right] \)
  
  \[ = \frac{1}{N} \sum_{i=1}^{N} \int_{a}^{b} \frac{f(x)}{p(x)} p(x) \, dx \]
  
  \[ = \frac{1}{N} \sum_{i=1}^{N} \int_{a}^{b} f(x) \, dx \]
  
  \[ = \int_{a}^{b} f(x) \, dx \]
Monte Carlo Estimator

Variance Reduction / Error Reduction

- **importance sampling**
  - motivation: contributions of larger function values are more important
  - PDF should be similar to the shape of the function
  - optimal PDF $p(x) = \frac{f(x)}{\int f(x) \, dx}$
  - e. g., if incident radiance is weighted with $\cos \theta$, the PDF should choose more samples close to the normal direction

- **stratified sampling**
  - subdividing the domain into individual strata does not increase the variance

- **multi-jittered sampling**
  - an alternative to random samples for, e. g., uniform sampling of area lights
Inversion Method

- mapping of a uniform random variable to a goal distribution
- discrete example
  - four outcomes with probabilities $p_1, p_2, p_3, p_4$ and $\sum_i p_i = 1$

- computation of the cumulative distribution function $P(i) = \sum_{j=1}^{i} p_j$
**Inversion Method**

- **discrete example cont.**
  - take a uniform random variable $\xi$
  - $P^{-1}(\xi)$ has the desired distribution
- **continuous case**
  - $P$ and $P^{-1}$ are continuous functions
  - start with the desired PDF $p(x)$
  - compute $P(x) = \int_0^x p(x')dx'$
  - compute the inverse $P^{-1}(x)$
  - obtain a uniformly distributed variable
  - compute $X_i = P^{-1}(\xi)$ which adheres to $p(x)$
2D Sampling

- generation of samples from a 2D joint density function $p(x, y)$
- general case
  - compute the marginal density function $p_x(x) = \int p(x, y) dy$
  - compute the conditional density function $p_y(y|x) = \frac{p(x, y)}{p_x(x)}$
  - generate a sample $X$ according to $p_x(x)$
  - generate a sample $Y$ according to $p_y(y|X) = \frac{p(x, y)}{p_x(X)}$
- marginal density function
  - integral of $p(x, y)$ for a particular $x$ over all $y$-values
- conditional density function
  - density function for $y$ given a particular $x$
Outline

- introduction
- transformations
- ray-object intersections
- intersection acceleration
- aliasing
- sampling and reconstruction
- radiometry
- materials
- rendering equation
- path tracing
Path Tracing - Illustration

- rays are recursively cast along a path to estimate the transported radiance
- recursion stops if
  - a light source is hit
  - a maximum depth / minimum radiance is reached
  - the ray leaves the scene / hits the background
light sources dominate the illumination of surfaces and are rather small

improved efficiency by explicitly sending rays to light sources

\[ L_o(p, \omega_o) = L_e(p, \omega_o) + \int_{2\pi} f_r(p, \omega_i, \omega_o)L_i(p, \omega_i) \cos \theta_i d\omega_i \]

emitted radiance reflected radiance

\[ L_o(p, \omega_o) = L_e(p, \omega_o) + L_r(p, \omega_o) \]

assumption

- no emitted radiance for objects \( L_e(p, \omega_o) = 0 \) \( L_r(p, \omega_o) > 0 \)
- for light sources \( L_e(p, \omega_o) > 0 \) \( L_r(p, \omega_o) > 0 \)
Direct Illumination

- for objects

\[ L_o(p, \omega_o) = \int_{2\pi} f_r(p, \omega_i, \omega_o) L_e(r_c(p, \omega_i), -\omega_i) + L_r(r_c(p, \omega_i), -\omega_i)) \cos \theta_i \, d\omega_i \]

- contribution of direct illumination is computed using the area form of the integral

\[ L_o(p, \omega_o) = \int_{2\pi} f_r(p, \omega_i, \omega_o) L_r(r_c(p, \omega_i), -\omega_i) \cos \theta_i \, d\omega_i + \int_{2\pi} f_r(p, \omega_i, \omega_o) L_e(r_c(p, \omega_i), -\omega_i) \cos \theta_i \, d\omega_i \]

- zero for objects

this second term significantly contributes to \( L_o \)

- requires a sampling of the light sources

V represents the visibility of the light source, estimated by a shadow ray.
Hemisphere vs. Area Form

- both forms can be combined
  - area form with light-source sampling
    plus hemisphere form ignoring the emitted radiance
  - \( L_{o}(p, \omega_{o}) = L_{e}(p, \omega_{o}) + L_{r}(p, \omega_{o}) \)
  - area form considers \( L_{e}(p, \omega_{o}) \)
    - light source sampling excludes directions with \( L_{e}(p, \omega_{o}) = 0 \)
  - hemisphere form considers \( L_{r}(p, \omega_{o}) \)
    - \( L_{e}(p, \omega_{o}) \) is ignored to guarantee that direct illumination is not considered twice
Monte-Carlo estimator for indirect illumination

\[ L_\omega(p, \omega_\omega) = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{p(\omega_i)} f_r(p, \omega_i, \omega_\omega) L_r(r_c(p, \omega_i), -\omega_i) \cos \theta_i \]

- importance sampling
  - uniform sampling (less efficient)
  - proportional to the cosine factor (useful for diffuse surfaces)
  - proportional to the BRDF (useful for glossy surfaces)
  - proportional to the incident radiance (usually unknown, but can be determined by other techniques, e.g. photon mapping)
  - a combination of these factors