Real-Time High Quality Rendering

CSE 291 [Winter 2015], Lecture 2
Graphics Hardware Pipeline, Reflection and Rendering Equations, Taxonomy of Methods
http://www.cs.ucsd.edu/~ravir

Outline of Lecture

- Basics of hardware pipeline
- Reflection and Rendering equations
- Typical Lighting, shading in hardware
- Taxonomy of methods/papers
- Assignment: Sign up for paper presentations

Basic concepts review only. If interested in more background/in depth detail, refer to handouts

Basic Hardware Pipeline

Application

CPU

Create geometry, lights, materials, textures, cubemaps, ... as inputs

Geometry

Transform and lighting calc. Apply per-vertex operations

GPU

Textures, Cubemaps

Per-pixel (per-fragment) operations

Rasterizer

Geometry or Vertex Pipeline

Model, View Transform

Lighting

Projection

Clipping

Screen

These fixed function stages can be replaced by a general per-vertex calculation using vertex shaders in modern programmable hardware

Pixel or Fragment Pipeline

Rasterization (scan conversion)

Texture Mapping

Z-buffering

Framebuffer

These fixed function stages can be replaced by a general per-fragment calculation using fragment shaders in modern programmable hardware

GPU Programmable Shaders

Programmable in Modern GPUs (Vertex Shader)

Geometry Primitive Operations

Scan Conversion (Rasterize)

Programmable in Modern GPUs (Fragment Shader)

Fragment Operations

Vertices

Images

Pixel Operations

Texture Memory

Traditional Approach: Fixed function pipeline (state machine)
New Development (2003-): Programmable pipeline
Outline of Lecture

- Basics of hardware pipeline
- Reflection and Rendering equations
- Typical Lighting, shading in hardware
- Taxonomy of methods/papers
- Assignment: Sign up for paper presentations

Basic concepts/review only. If interested in more background/in depth detail, refer to handouts.

**Reflection Equation**

\[
\begin{align*}
L_r(x, \omega_r) &= L_e(x, \omega_r) + \sum L_i(x, \omega_i) f(x, \omega_i, \omega_r) \cos \theta_i \\
&= L_e(x, \omega_r) + \int L_r'(x', -\omega_i) f(x', \omega_i, \omega_r) \cos \theta_i \, d\omega_i
\end{align*}
\]

- Reflected Light (Output Image)
- Emission (Light from light source)
- BRDF
- Cosine of Incident angle

**Rendering Equation (Kajiya 86)**

\[
L_r(x, \omega_r) = L_e(x, \omega_r) + \int L_r'(x', -\omega_i) f(x', \omega_i, \omega_r) \cos \theta_i \, d\omega_i
\]

- Reflected Light (Output Image)
- Emission (Light from light source)
- BRDF
- Cosine of Incident angle

**Figure 6**: A sample image. All objects are recorded. Color of the objects that is scattered from the green glass block and other blocking from the human body.
Rendering Equation as Integral Equation

\[ L(x, \omega_o) = I(x, \omega_o) + \int K(x', \omega_o, \omega_i) \cos \theta \, d\omega_i \]

Reflected Light (Output Image)
Emission
BRDF
Cosine of Incident angle
\[ L(x, \omega_o) = I(x, \omega_o) + \int K(x', \omega_o, \omega_i) \cos \theta \, d\omega_i \]

Is a Fredholm Integral Equation of second kind [extensively studied numerically] with canonical form
\[ I(u) = e(u) + \int K(u, v) \, dv \]

Kernel of equation

Linear Operator Equation

\[ I(u) = e(u) + \int K(u, v) \, dv \]

Kernel of equation
Light Transport Operator

\[ L = E + KL \]

Can be discretized to a simple matrix equation [or system of simultaneous linear equations] (\(L, E \) are vectors, \(K\) is the light transport matrix)

Ray Tracing and extensions
- General class numerical Monte Carlo methods
- Approximate set of all paths of light in scene

\[ L = E + KL \]
\[ IL - KL = E \]
\[ (I - K)L = E \]
Binomial Theorem
\[ L = (I - K)^{-1}E \]
\[ L = (l + K + K^2 + K^3 + \ldots)E \]
\[ L = E + KE + K^2E + K^3E + \ldots \]

Outline of Lecture

- Basics of hardware pipeline
- Reflection and Rendering equations
- Typical Lighting, shading in hardware
- Taxonomy of methods/papers
- Assignment: Sign up for paper presentations

Basic concepts/review only. If interested in more background/in depth detail, refer to handouts
OpenGL: Sum of Components

Sum over all light sources

\[ L_r(x, \omega_r) = L_e(x, \omega_r) + \sum L_i f(\omega_i, \omega_r)(\omega_i \cdot n) \]

- Ambient
- Emission
- Diffuse
- Specular

Ambient term

Global constant (sometimes per light) added to everything

- In addition to other terms in reflection equation
- Fakes indirect illumination, broad area lights
- Prevents completely black regions etc.
- Hack, no physical basis

Emissive Term (for Light Sources)

Sum over all light sources

\[ L_r(x, \omega_r) = L_e(x, \omega_r) + \sum L_i f(\omega_i, \omega_r)(\omega_i \cdot n) \]

Diffuse Term (Lambertian Reflection)

BRDF constant (Lambertian) ; mult. by diffuse albedo

\[ L_r(x, \omega_r) = L_e(x, \omega_r) + \sum L_i f_d(\omega_i \cdot n) \]

Specular Term (Phong)

Specular Term (Blinn-Phong)

Sum over all light sources

\[ L_r(x, \omega_r) = L_e(x, \omega_r) + \sum L_i f_s(\omega_i \cdot \omega_h)(\omega_i \cdot n)^s \]

\[ \omega_h = \frac{\omega_i + \omega_r}{|\omega_i + \omega_r|} \]

Phong Model

Mirror Diffuse

Blinn-Phong Model

Light (from light source)

\( s \) is shininess
Outline of Lecture

- Basics of hardware pipeline
- Reflection and Rendering equations
- Typical Lighting, shading in hardware
- Taxonomy of methods/papers

Assignment: Sign up for paper presentations

Basic concepts/review only. If interested in more background/in depth detail, refer to handouts

Syllabus / Taxonomy: Basic Hardware

- Basic Hardware Techniques (next wk)
  - Shadow Mapping
  - Environment Mapping

- Graphics Hardware
  - Geometry Engine (82)
  - Reality Engine (83)
  - Realistic Hardware-Accelerated Shading and Lighting
  - Real-Time Procedural Shading (01)
  - Ray Tracing on Graphics Hardware (02)
  - Brook: Stream Computing (04)
  - Photon mapping on hardware (03)
  - GPU-Based Interactive Global Illumination (09)

High Quality Hardware Shading

- Heidrich 99, one of first papers. Hardware tricks (before current programmable graphics, in standard OpenGL)
- Purcell et al. 02, map ray tracing to standard programmable hardware scanline pipeline
- Lindholm et al. 01, first nVidia vertex shaders (GeForce 3)

General Programmable Shading

- First programmable shading systems for hardware
  - Peercy et al. 00: Multipass OpenGL implementation of RenderMan (OpenGL as general SIMD machine)
  - Proudfoot et al. 01: Multiple computation frequencies (precursor to vertex, fragment shaders)
  - Mark et al. 03: Describes nVidia’s Cg: one of first commercial high-level shading languages

Shadow Mapping

- Classic technique to add complex curved shadows
  - Williams 78, oldest paper we read in course
  - Many recent extensions for programmable hardware
    - Soft shadows: Agrawala 00
    - Complex geometry like hair: Lokovic and Veach 00
    - Adaptive techniques: Fernando 01, Staminger 02, Sen 03

Syllabus/Taxonomy

- Hardware Ray Tracing
  - Multi-Level CPU Ray Tracing (05)
  - OptiX (10)
  - Embree (14)

- Image-Based Rendering (background)
  - Light Fields and Lumigraphs (06)
  - Surface Light Fields (00)
  - Reflectance Fields (00)

- Signal Processing (background)
  - Plenoptic Sampling (00)
  - Signal-Processing Framework (01)
  - Frequency Analysis of Light Transport (05)
### New Ideas

- **Precomputed Rendering**
  - Precomputed Radiance Transfer (02)
  - Clustered PCA (03)
  - All-Frequency Shadows (03)
  - Newer Papers (09, 11, 13)

- **Sparse Sampling and Reconstruction**
  - (background) (08,09,12)
  - Axis-Aligned Filtering (12,13)

- **Sampling of Recent Work**
  - Anisotropic Spherical Gaussians (13)
  - Convolution Shadows, Volumes (13)
  - Axis-Aligned Distribution Effects (14)

### Precomputed Transfer

- Precompute on static scenes, followed by real-time relighting, changing view. Can capture most complex shading effects.
- Sloan et al. 02,03: Low-frequency SH, compression
- Ng et al. 03, 04: Wavelets all-frequency, relight, view change
- Sloan 04, Wang 04: Factored BRDF's all-frequency relighting

### Outline of Lecture

- Basics of hardware pipeline
- Reflection and Rendering equations
- Typical Lighting, shading in hardware
- Taxonomy of methods/papers

- **Assignment:** Sign up for paper presentations

Basic concepts/review only. If interested in more background/in depth detail, refer to handouts

### Paper Presentations

- E-mail me the top 3-4 papers you want to present
- Only those that say “presented by students”
- Brief description of projects next week (see schedule)
- Project milestone and final proposal 3 weeks after

### Questions?