Real-Time High Quality Rendering
GAMES202, Lingqi Yan, UC Santa Barbara

Lecture 3:
Real-Time Shadows 1
Announcement

• Enjoying assignment 0?

• Assignment 1 will be released this week
  – Ideally together with Lecture 4

• Adjusted orders of some contents

• The RTR book
Last Lecture

• Recap of CG Basics
  - Basic GPU hardware pipeline
  - OpenGL
  - OpenGL Shading Language (GLSL)
  - The Rendering Equation
  - Calculus
Today

• **Recap: shadow mapping**
  [slides courtesy of Prof. Ravi Ramamoorthi]
  - Issues from shadow mapping and solutions

• The math behind shadow mapping

• Percentage closer soft shadows

• Basic filtering techniques
Shadow Mapping

• A 2-Pass Algorithm
  - The light pass generates the SM
  - The camera pass uses the SM (recall last lecture)

• An image-space algorithm
  - Pro: no knowledge of scene’s geometry is required
  - Con: causing self occlusion and aliasing issues

• Well known shadow rendering technique
  - Basic shadowing technique even for early offline renderings, e.g., Toy Story
Pass 1: Render from Light

• Output a “depth texture” from the light source
Pass 1: Render from Light

• Output a “depth texture” from the light source
Pass 2: Render from Eye

- Render a standard image from the eye
Pass 2: Project to light for shadows

- Project visible points in eye view back to light source

(Reprojected) depths match for light and eye. VISIBLE
Pass 2: Project to light for shadows

- Project visible points in eye view back to light source

(Reprojected) depths from light, eye not the same. BLOCKED!!
Shadow Mapping Results

- A fairly complex scene with shadows

*the point light source*
Shadow Mapping Results

- Compare with and without shadows

**with shadows**

**without shadows**
Visualizing Shadow Mapping

• The scene from the light’s point-of-view

FYI: from the eye’s point-of-view again
Visualizing Shadow Mapping

- The depth buffer from the light’s point-of-view

FYI: from the light’s point-of-view again
Visualizing Shadow Mapping

- Projecting the depth map onto the eye’s view

FYI: depth map for light’s point-of-view again
Visualizing Shadow Mapping

- Scene with shadows

*Notice how specular highlights never appear in shadows*

*Notice how curved surfaces cast shadows on each other*
Issues in Shadow Mapping

- Self occlusion
  - When is it most severe?

[Image from RTR4]
Issues in Shadow Mapping

• Adding a (variable) bias to reduce self occlusion
  - But introducing detached shadow issue

[Image from RTR4]
Issues in Shadow Mapping

• Second-depth shadow mapping*
  - Using the midpoint between first and second depths in SM
  - Unfortunately, requires objects to be watertight
  - And the overhead may not worth it

*Image from RTR4
RTR does not trust in COMPLEXITY
Issues in Shadow Mapping

- Aliasing

[https://developer.download.nvidia.com/books/HTML/gpugems/gpugems_ch11.html]
Questions?
Today

• Recap: shadow mapping
  - Issues from shadow mapping and solutions

• The math behind shadow mapping

• Percentage closer soft shadows

• Basic filtering techniques
Inequalities in Calculus

- There are a lot of useful inequalities in calculus

12. 设 \( f(x) \) 和 \( g(x) \) 在 \([a, b]\) 上都可积，证明不等式:

1. (Schwarz 不等式) \[ \left( \int_a^b f(x) g(x) \, dx \right)^2 \leq \int_a^b f^2(x) \, dx \cdot \int_a^b g^2(x) \, dx ; \]

2. (Minkowski 不等式) \[ \left( \int_a^b \left[ f(x) + g(x) \right]^2 \, dx \right)^{\frac{1}{2}} \leq \left( \int_a^b f^2(x) \, dx \right)^{\frac{1}{2}} + \left( \int_a^b g^2(x) \, dx \right)^{\frac{1}{2}} . \]
Approximation in RTR

• But in RTR, we care more about “approximately equal”

• An important approximation throughout RTR

\[ \int_{\Omega} f(x)g(x) \, dx \approx \frac{\int_{\Omega} f(x) \, dx}{\int_{\Omega} dx} \cdot \int_{\Omega} g(x) \, dx \]

• When is it (more) accurate?
In Shadow Mapping

- Recall: the rendering equation with explicit visibility

\[ L_o(p, \omega_o) = \int_{\Omega^+} L_i(p, \omega_i) f_r(p, \omega_i, \omega_o) \cos \theta_i V(p, \omega_i) d\omega_i \]

- Approximated as

\[ L_o(p, \omega_o) \approx \frac{\int_{\Omega^+} V(p, \omega_i) d\omega_i}{\int_{\Omega^+} d\omega_i} \cdot \int_{\Omega^+} L_i(p, \omega_i) f_r(p, \omega_i, \omega_o) \cos \theta_i d\omega_i \]
In Shadow Mapping

\[ L_o(p, \omega_o) \approx \frac{\int_{\Omega^+} V(p, \omega_i) \, d\omega_i}{\int_{\Omega^+} d\omega_i} \cdot \int_{\Omega^+} L_i(p, \omega_i) f_r(p, \omega_i, \omega_o) \cos \theta_i \, d\omega_i \]

- **When is it accurate?**
  - Small support  
    (point / directional lighting)
  - Smooth integrand  
    (diffuse bsdf / constant radiance area lighting)

- **We’ll see it again in Ambient Occlusions, etc.**
Questions?
Today

- Recap: shadow mapping
  - Issues from shadow mapping and solutions
- The math behind shadow mapping
- Percentage closer soft shadows
- Basic filtering techniques
From Hard Shadows to Soft Shadows

[Image of hard shadows]

[Diagram of solar eclipse with labels for Umbra and Penumbra]

[RenderMan]

[Link: https://www.timeanddate.com/eclipse/umbra-shadow.html]
Percentage Closer Filtering (PCF)

• Provides **anti-aliasing** at shadows’ edges
  - Not for soft shadows (PCSS is, introducing later)
  - Filtering the results of shadow comparisons

• **Why not filtering the shadow map?**
  - Texture filtering just averages color components, i.e. you’ll get blurred shadow map first
  - Averaging depth values, then comparing, you still get a **binary** visibility
Percentage Closer Filtering (PCF)

- Solution [Reeves, SIGGARPH 87]
  - Perform multiple (e.g. 7x7) depth comparisons for each fragment
  - Then, averages results of comparisons
  - e.g. for point P on the floor,
    (1) compare its depth with all pixels in the red box, e.g. 3x3
    (2) get the compared results, e.g.
      1, 0, 1,
      1, 0, 1,
      1, 1, 0,
    (3) take avg. to get visibility, e.g. 0.667
Percentage Closer Filtering

Again, not soft shadows in the umbra/penumbra sense
Percentage Closer Filtering

[https://developer.nvidia.com/gpugems/GPUGems3/gpugems3_ch08.html]
Percentage Closer Filtering

- Does filtering size matter?
  - Small -> sharper
  - Large -> softer

- Can we use PCF to achieve soft shadow effects?

- Key thoughts
  - From hard shadows to soft shadows
  - What’s the correct size to filter?
  - Is it uniform?
Percentage Closer Soft Shadows

• Key observation [Fernando et al.]
  - Where is sharper? Where is softer?

Percentage Closer Soft Shadows

- **Key conclusion**
  - Filter size <-> blocker distance
  - More accurately, relative average projected blocker depth!

- **A mathematical “translation”**

\[
w_{Penumbra} = (d_{Receiver} - d_{Blocker}) \cdot \frac{w_{Light}}{d_{Blocker}}
\]

[Fernando et al.]
Percentage Closer Soft Shadows

• Now the only question:
  - What’s the blocker depth $d_{\text{Blocker}}$

• The complete algorithm of PCSS
  - Step 1: Blocker search
    (getting the average blocker depth in a certain region)
  - Step 2: Penumbra estimation
    (use the average blocker depth to determine filter size)
  - Step 3: Percentage Closer Filtering

• Which region to perform blocker search?
  - Can be set constant (e.g. 5x5), but can be better with heuristics
Percentage Closer Soft Shadows

- Which region (on the shadow map) to perform blocker search?
  - depends on the light size
  - and receiver’s distance from the light

[Fernando et al.]
Percentage Closer Soft Shadows

Video game: Dying Light
Questions?
Next Lecture

- Basic filtering techniques
- Variance soft shadow mapping
- MIPMAP and Summed-Area Variance Shadow Maps
- Moment shadow mapping
Thank you!