Real-Time High Quality Rendering

GAMES202, Lingqi Yan, UC Santa Barbara

Lecture 12:
Real-Time Ray Tracing 1

Announcements

- GAMES101 resubmission
  - Starting next Monday!
- GAMES202 homework 3
  - Will be released soon hopefully
  - Your understanding is greatly appreciated

Last Lectures

- Real-Time Physically-Based Materials
  - Microfacet BRDF, NDF, shadowing-masking
  - Kulla-Conty Approximation for multiple bounces
  - Disney principled BRDF
- Shading with microfacet BRDFs under polygonal lighting
  - Linearly Transformed Cosines (LTC)
- Non-photorealistic rendering (NPR)

Some Arrangements

- Volumetric / scattering materials will not be covered in this course
  - Too many dependencies (RTE, BSSRDF, single/multiple scattering, etc.)
  - Will be fully covered in offline rendering, together with RTR techniques (delta tracking, dual scattering, layered materials, etc.)

[Final Fantasy VII Remake]

[Black Myth: Wukong]

Some Arrangements

- Unreal Engine 5 early access is available now!
  - Again, both Nanite and Lumen are **TECHNICAL** breakthroughs
  - The underlying science is already understandable after learning this course
  - Will briefly analyze (or rather, guess) possible approaches in the last lecture

[UE5 Early Access Trailer (weakest boss ever)]

Today

- New Topic: Real-Time Ray Tracing (RTRT)
  - Basic idea
  - Motion vector
  - Temporal accumulation / filtering
  - Failure cases
- Filtering techniques and implementation (next lecture)
  - Joint bilateral filtering
  - Spatiotemporal Variance-Guided Filtering (SVGF)

RTRT is the Future

- In the real-time industry, people claim that

> "Ray tracing is the future **and ever will be**."
>
> — The real-time industry

RTRT is Happening

- In 2018, NVIDIA announced GeForce **RTX** series (Turing architecture)
  - Opening a $250 billion market

RTRT is Happening

- What does RTX do?

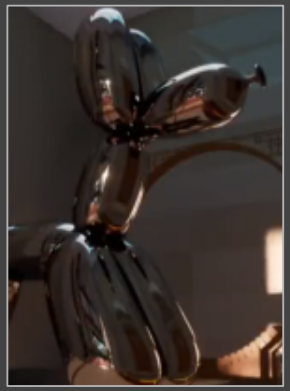
Impressive demos of RTRT

Star Wars Reflections

RTX Demo

Porsche 70 Trailer

SOL

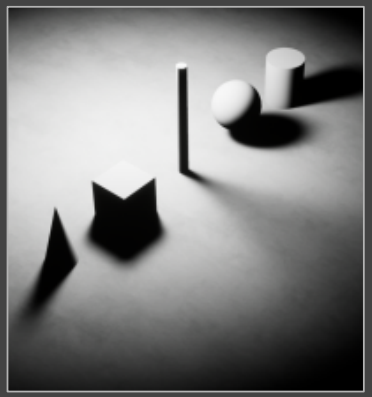


Rosewood Bangkok

RTRT is Happening

- What does RTX actually do?

Advanced **ray traced** effects



Shadows

Reflections & Specular

Ambient Occlusion

Global Illumination

RTRT is Happening

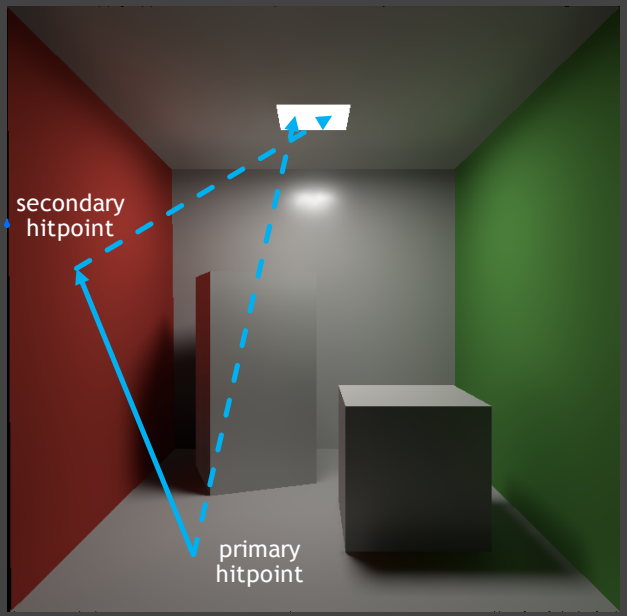
- What does RTX really do?

10 Giga rays per second == **1 sample per pixel**

(for real time applications)

RTRT is Happening

- What does RTX actually do?
- 1 SPP path tracing =
  - 1 rasterization (primary) +
  - 1 ray (primary visibility) +
  - 1 ray (secondary bounce) +
  - 1 ray (secondary vis.)



RTRT is Happening

- 1 SPP = Extremely noisy results
- Key technology
  - Denoising

Fun image on Twitter

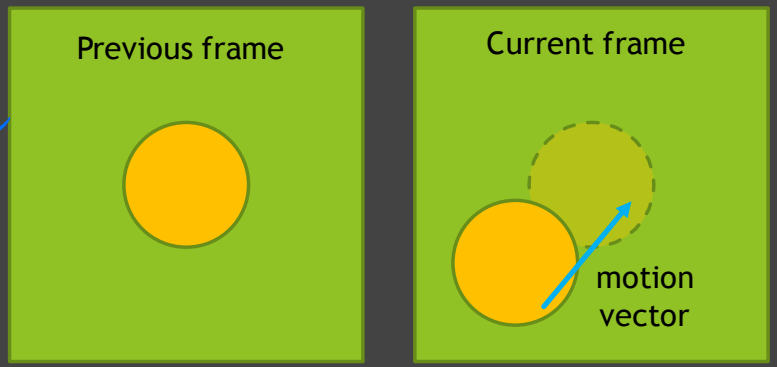
State of the Art* Denoising Solution

Before we proceed…

- Goals (**with 1 SPP**)
  - Quality (no overblur, no artifacts, keep all details…)
  - Speed (< 2 ms to denoise one frame)
- **Mission impossible**
  - Sheared filtering series (SF, AAF, FSF, MAAF, …)
  - Other offline filtering methods (IPP, BM3D, APR, …)
  - Deep learning series (CNN, Autoencoder, …)

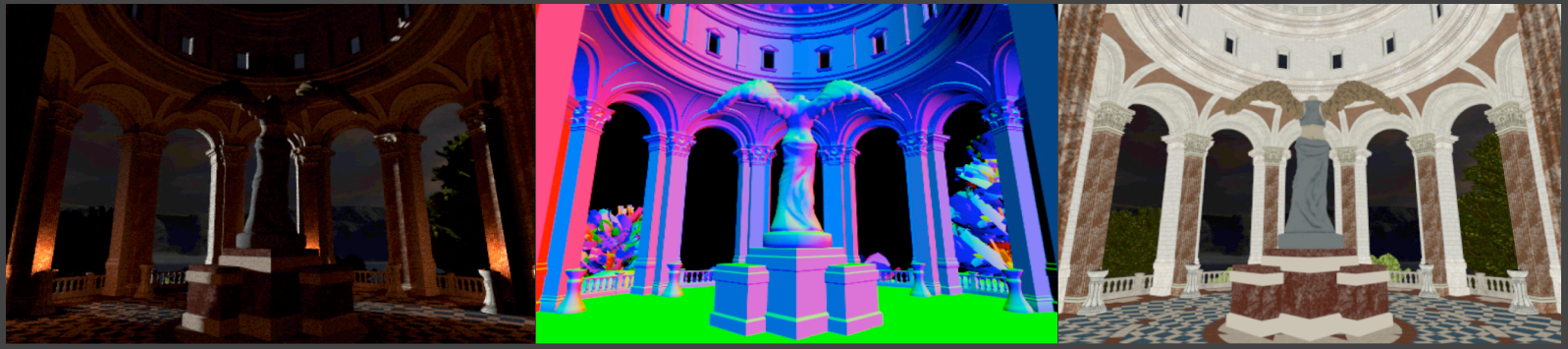
Industrial Solution

- 3 most important ideas
  - Temporal! 时间上的复用
  - **Temporal!!**
  - **Temporal!!!**
- Key idea
  - Suppose the previous frame is denoised and reuse it
  - Use **motion vectors** to find previous locations
  - Essentially increased SPP
  - Spatial?



The G-Buffers

- Geometry buffer
  - The auxiliary information acquired **FOR FREE*** during rendering
  - Usually, per pixel depth, normal, world coordinate, etc.
  - Therefore, only **screen space** info



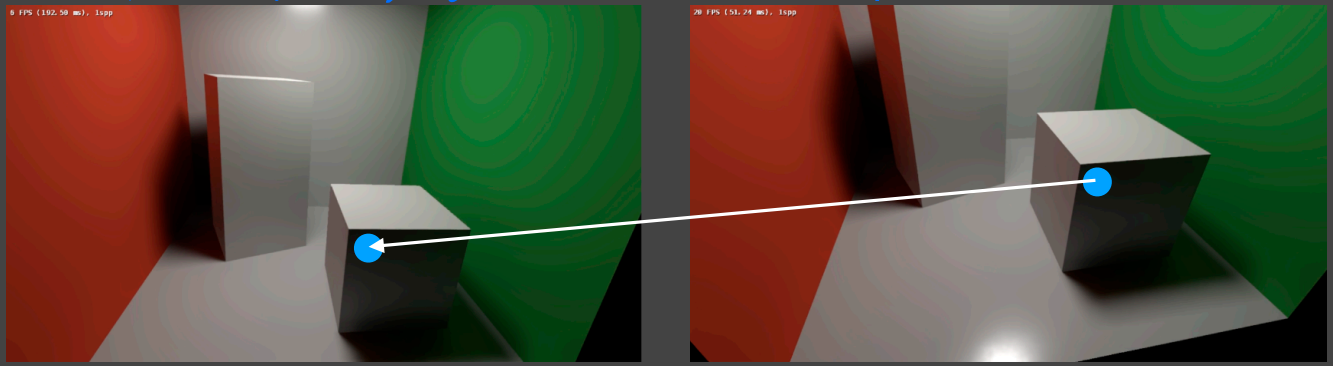
Direct illumination | Normal | Albedo

Back Projection

反投影

- Pixel x in the current frame i
  - ~~Where was **it** in the last frame $i-1$?~~
  - What pixel in frame $i-1$ contains **the same place/point that you see though pixel x in frame i**?

帧i-1中哪个像素经过帧i中的像素看到的相同的位置/点.



frame i-1　　frame i

Back Projection

- Pixel x in the current frame i
  - Where was it in the last frame $i-1$?
- Back projection
  1. If world coord s is available as a G-buffer, just take it
  2. Otherwise, $s = M^{-1}V^{-1}P^{-1}E^{-1}x$ (still require z value)
     - Motion is known: $s' \xrightarrow{T} s$, thus $s' = T^{-1}s$
     - Project world coord in frame $i-1$ to its screen: $x' = P'V'M's'$