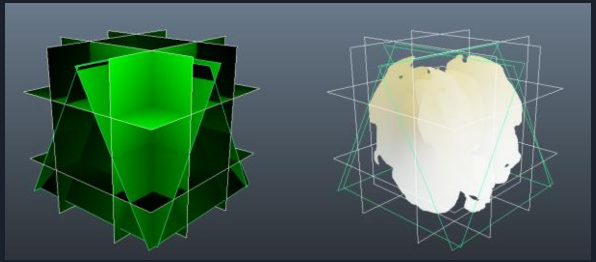
Real-Time Volumetric Cloud Rendering with Ray-Marching Technique

Victor Shu, Yun Jiang

Introduction

Geometry Planes – Unrealistic, Hard to Shade [WITNESS]

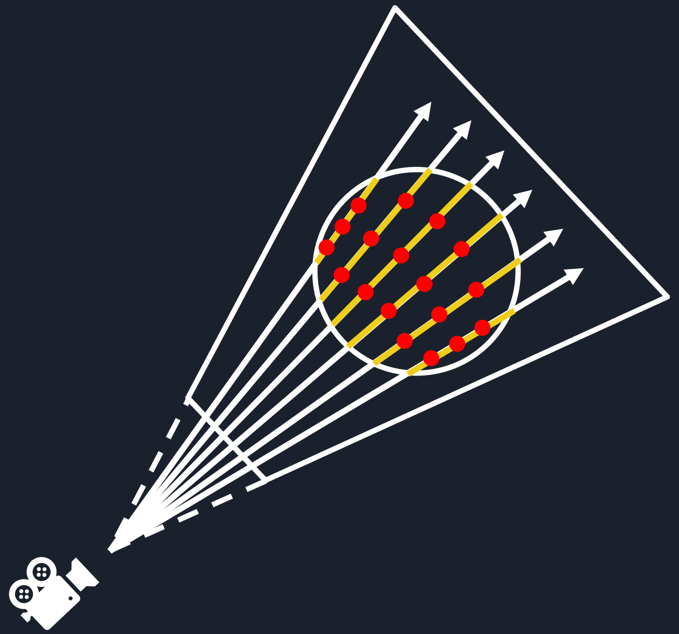
Voxel Clouds – Poor Performance [GUERRILLA]

Introduction

Volumetric Cloud Rendering with Ray Marching
in Battlefield 4 and Star Wars Battlefront[EA]

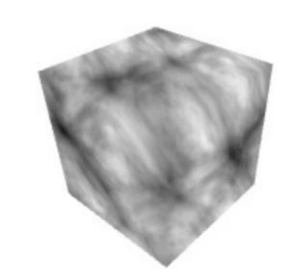
Ray Marching

- Similar to ray tracing
  - Interact with volume
  - Only primary ray
- Steps:
  1. Ray Casting
  2. Sampling
  3. Shading
  4. Composition

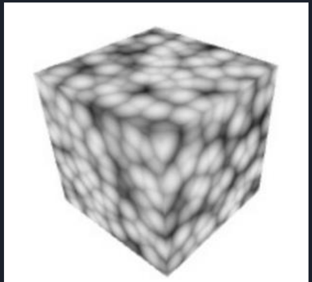


Cloud Shape

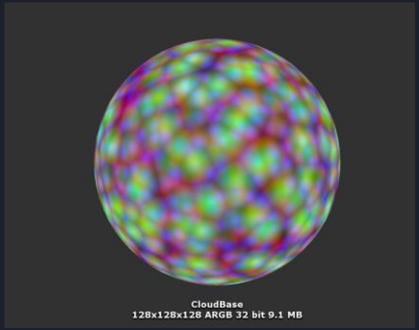
Texture	Size	R	G	B	A
Shape	128*128*128	Perlin	Worley	Worley	Worley
Detail	32*32*32	Worley	Worley	Worley	-



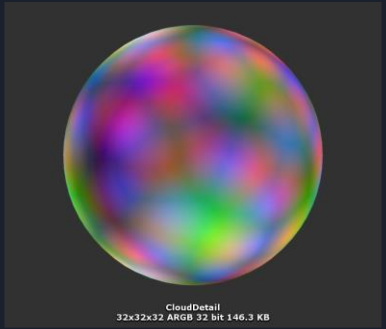
3D Perlin Noise [RURIK]



3D Worley Noise [RURIK]



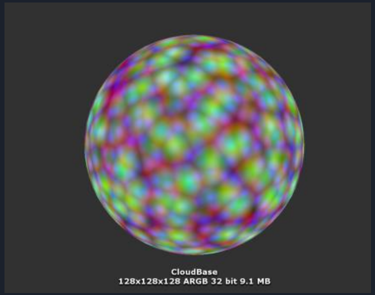
Shape Texture



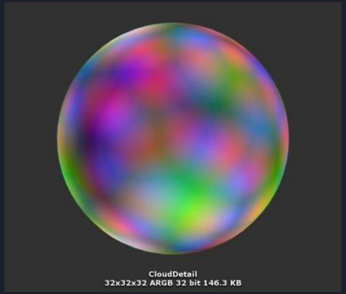
Detail Texture

Cloud Shape

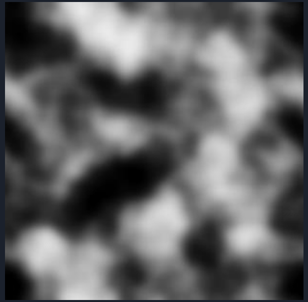
1. Basic Shape
2. Detail

Shape = (

Shape Texture

\- 

Detail Texture

) * 

Coverage Texture

Lighting

What We Want:

Final Color = (ShadowColor + ShadowStrength + Scattering) * Dark Edges * Transmittance

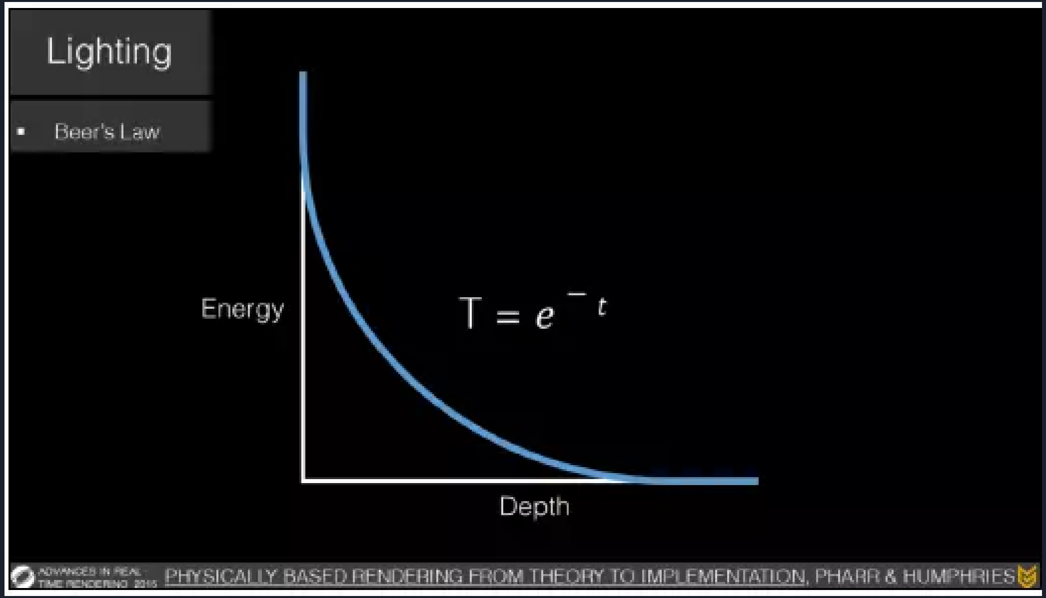
ShadowColor = Base color for the shadow part of the clouds

Scattering = pow(dot(rayDir, lightDir), ScatteringStrength);

Transmittance = Beer's Law

Shadow Color

Lighting



Beer’s Law: Transmittance along with the distance [GUERRILLA]

Lighting

What We Want:

Final Color = (ShadowColor + ShadowStrength + Scattering) * Dark Edges * Transmittance

ShadowColor = Base color for the shadow part of the clouds

Scattering = pow(dot(rayDir, lightDir), ScatteringStrength);

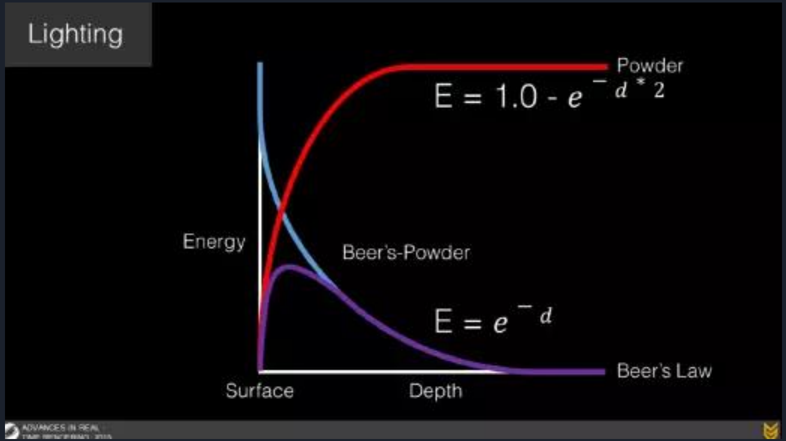
Transmittance = Beer's Law

Dark Edges = Powder Effect

Shadow Color

Lighting

Dark Edge [GUERRILLA]



Powder Effect [GUERRILLA]

Lighting

What We Want:

Final Color = (ShadowColor + ShadowStrength + Scattering) * Dark Edges * Transmittance

ShadowColor = Base color for the shadow part of the clouds

Scattering = pow(dot(rayDir, lightDir), ScatteringStrength);

Transmittance = Beer's Law

Dark Edges = Powder Effect

ShadowStrength = Sample towards sunlight

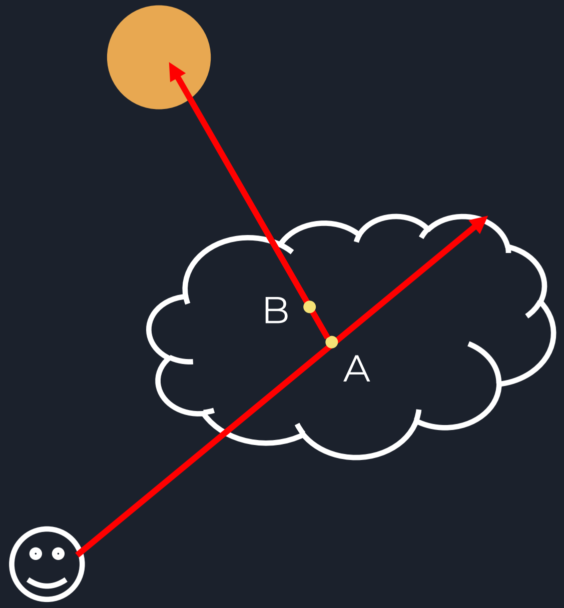
Shadow Color

Rendering

Steps:

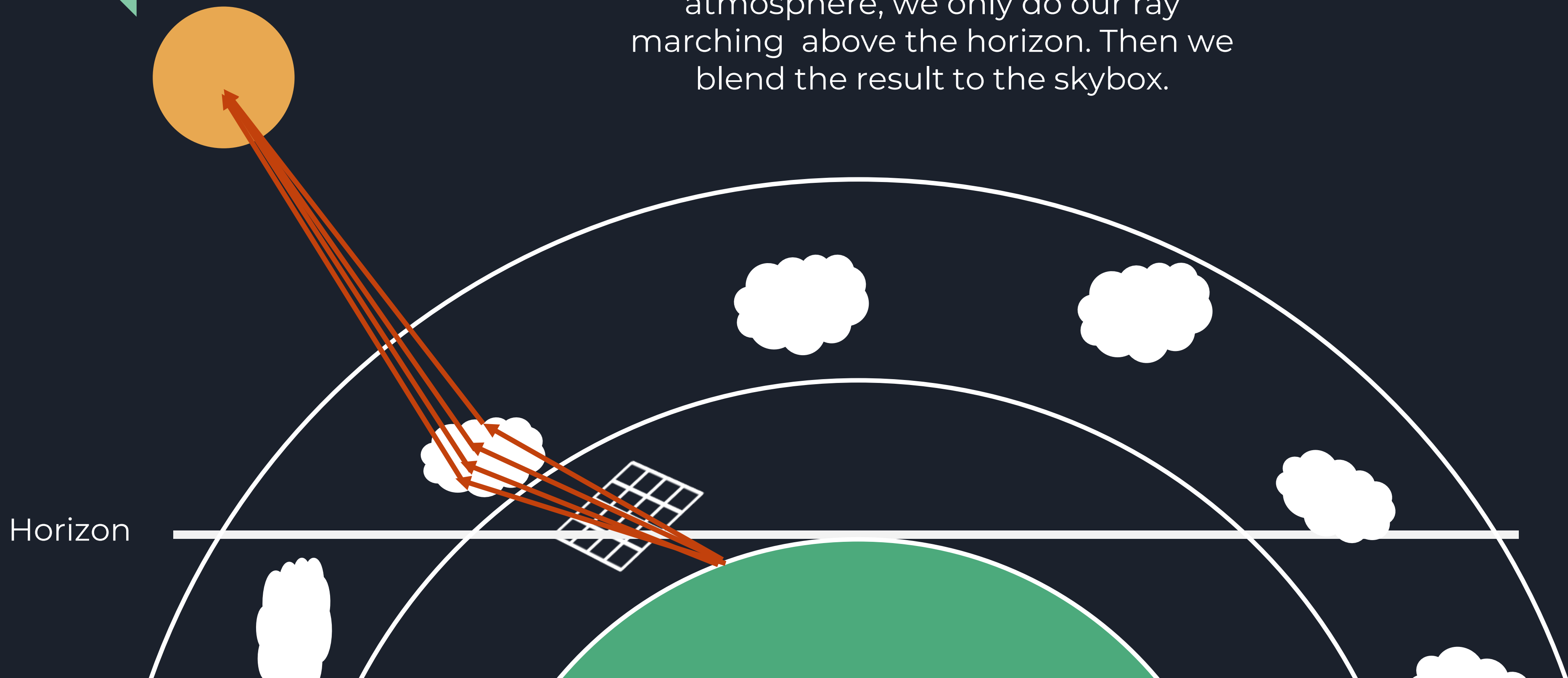
1. Get density1 of A from sampling the shape texture along camera ray
2. Get density2 of B from sampling the shape texture along shadow ray
3. Use the difference (density1 – density2) to represent the shadow strength

How to get the sample UV?
Use ray marching, find the intersection points of the ray, sample along the ray in the atmosphere.



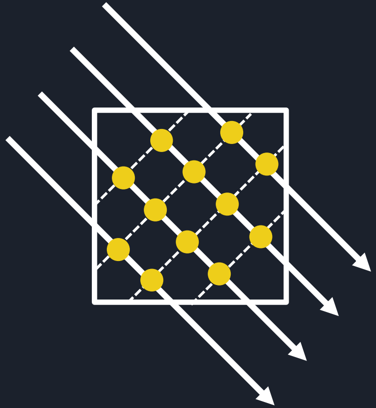
Rendering

Remember that it's a donut-like atmosphere, we only do our ray marching above the horizon. Then we blend the result to the skybox.



Temporal Upsampling

- Banding issue
- Randomized Ray Marching Step Size
  - Noise issue
- Temporal Blending
  - Blend the current frame with last frame



Atmosphere Scattering

GPU Gems 2 – Chapter 16 [NVIDIA]

Atmosphere Scattering

- Rayleigh Scattering
- Mie Scattering

Sky Color with Rayleigh and Mie Scattering[EA]

Atmosphere Scattering

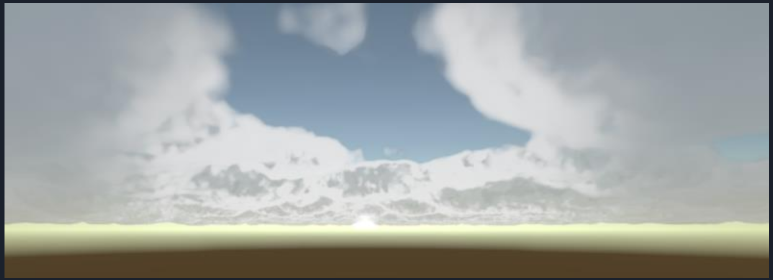
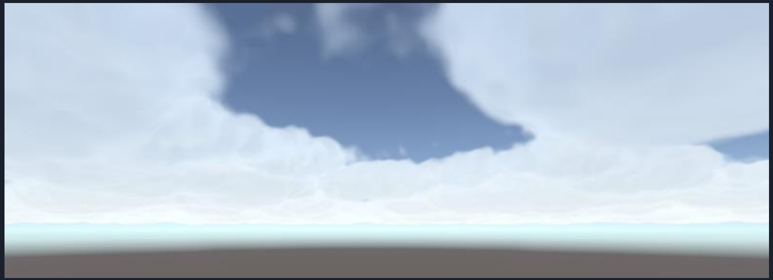
- Rayleigh Scattering
- Mie Scattering
- Phase Function
- Out-Scattering Equation
- In-Scattering Equation

$$F(\theta, g) = \frac{3 \times \left(1 - g^2\right)}{2 \times (2 + g^2)} \times \frac{1 + \cos^2 \theta}{(1 + g^2 - 2 \cdot g \cdot \cos \theta)^{\frac{3}{2}}}$$

$$t(P_a P_b, \lambda) = 4\pi \times K(\lambda) \times \int_{P_a}^{P_b} \exp -\frac{h}{H_0} ds$$

$$I_v(\lambda) = I_s(\lambda) \times K(\lambda) \times F(\theta, g) \times \int_{P_a}^{P_b} \left(\exp -\frac{h}{H_0} \times \exp\left(-t(PP_c, \lambda) - t(PP_a, \lambda)\right)\right) ds$$

Atmosphere Scattering

Optimizations

- Temporal Upsampling
- Early Exit

Results

1024

512

256

128

64

32

16

Results

1024


512


256


128


64


32


16


Results

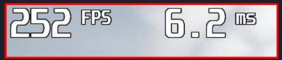
1024

512

256

128

64



32

16

Results

1024

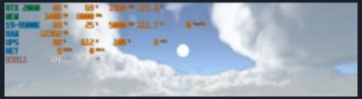
512

256

128

64

32

16


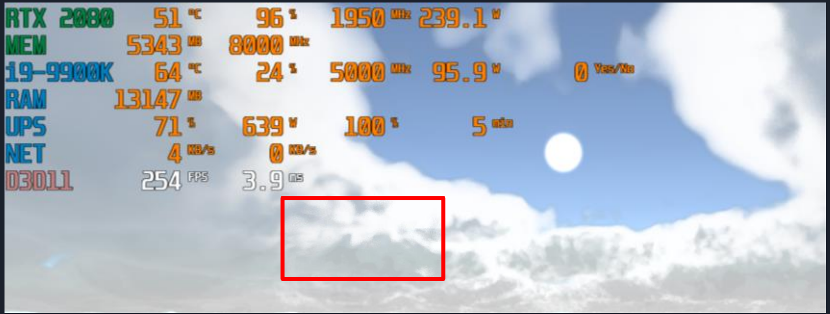

Results

With Early Exit

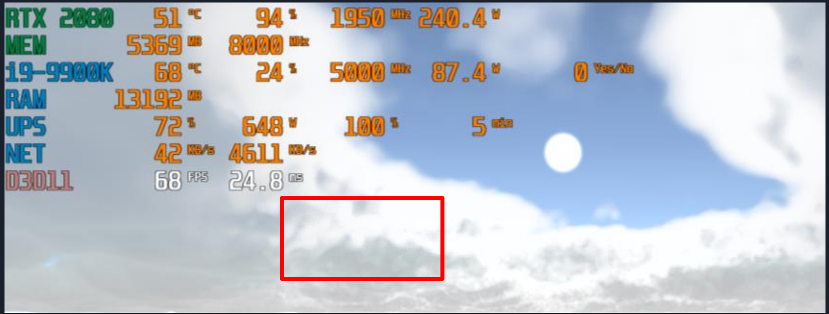
Without Early Exit

Results

With Early Exit



Without Early Exit



Results

With Early Exit

Without Early Exit

Contribution

- Yun Jiang
  - Basic Cloud Rendering
    - Noise Generation
    - Ray Marching
    - Shading
- Victor Shu
  - Enhancements
    - Temporal Upsampling
    - Atmospheric Scattering

References

- The Art of *The Witness* – Clouds [WITNESS]
  http://www.artofluis.com/3d-work/the-art-of-the-witness/clouds/
- The Real-Time Volumetric Cloudscapes of *Horizon Zero Dawn* [GUERRILLA]
  http://killzone.dl.playstation.net/killzone/horizonzerodawn/presentations/Siggraph15_Schneider_Real-Time_Volumetric_Cloudscapes_of_Horizon_Zero_Dawn.pdf
- Physically Based Sky, Atmosphere and Cloud Rendering in *Frostbite* [EA]
  https://media.contentapi.ea.com/content/dam/eacom/frostbite/files/s2016-pbs-frostbite-sky-clouds-new.pdf
- Convincing Cloud Rendering: An Implementation of Real-Time Dynamic Volumetric Clouds in *Frostbite* [RURIK]
  http://publications.lib.chalmers.se/records/fulltext/241770/241770.pdf
- GPU Gems 2, Chapter 16: Accurate Atmospheric Scattering [NVIDIA]
  https://developer.nvidia.com/gpugems/GPUGems2/gpugems2_chapter16.html

DEMO