Milestone 1: DDGI Minecraft

Helen Liu, Janine Liu, Spencer Webster-Bass

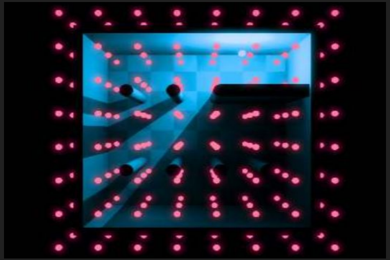
Overview

Goal: Generate a real-time dynamic Minecraft-inspired scene illuminated by Dynamic Diffuse Global Illumination (DDGI) Probes as described in this paper.

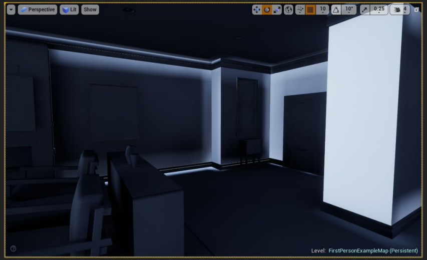
- Sample light info at probe locations to approximate indirect lighting at any point in scene
- Generate and trace rays from light probes, shade the resulting buffer, and update the buffer every frame.

- **Pros**: handles dynamic lighting and geometry and prevents light leaking

Example Dynamic Scene with Diffuse Global Illumination (Source: paper)



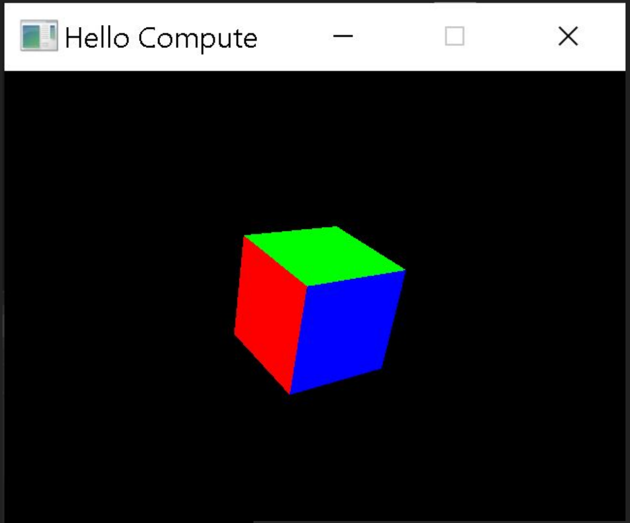
Placement of probes in scene (Source: paper)

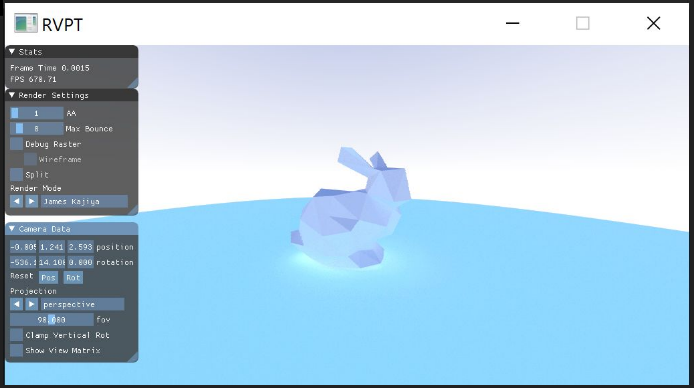


Light leaking problem in Unreal Engine 4. (Source: online)

Vulkan API

1. Started with CIS565 Vulkan Examples: experimented with implementing path tracer from scratch
2. Looked into using Vulkan ray-tracing extension
3. Started with forward rendering compute-based Vulkan pathtracer source code

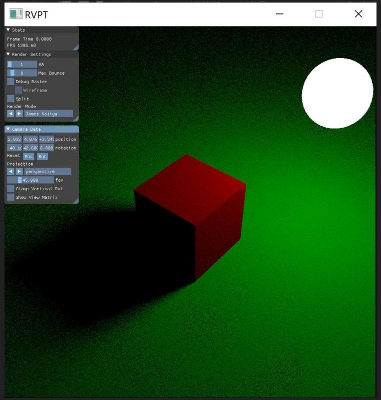




```
▼ Stats
Frame Time 0.0015
FPS 670.71
▼ Render Settings
1 AA
8 Max Bounce
□ Debug Raster
□ Wireframe
□ Split
Render Mode
◀ ▶ James Kajiya
▼ Camera Data
-0.005 1.241 2.593 position
-536.1 14.100 0.000 rotation
Reset Pos Rot
Projection
◀ ▶ perspective
90.000 fov
□ Clamp Vertical Rot
□ Show View Matrix
```

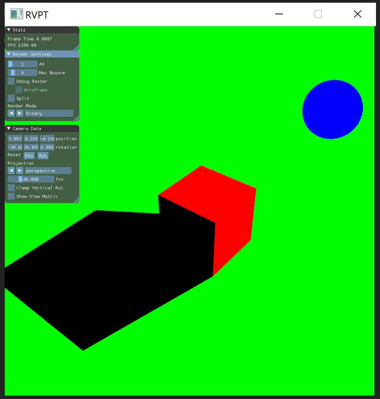
Base Code

Rendering Modes



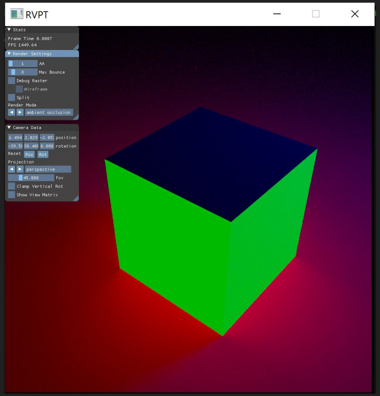
```
▼ Stats
Frame Time 0.0008
FPS 1305.60
▼ Render Settings
1 AA
8 Max Bounce
□ Debug Raster
□ Wireframe
□ Split
Render Mode
◀ ▶ James Kajiya
▼ Camera Data
2.832 4.076 -3.34 position
-40.16 42.60 0.000 rotation
Reset Pos Rot
Projection
◀ ▶ perspective
45.000 fov
□ Clamp Vertical Rot
□ Show View Matrix
```

Direct Lighting
(multiple samples)



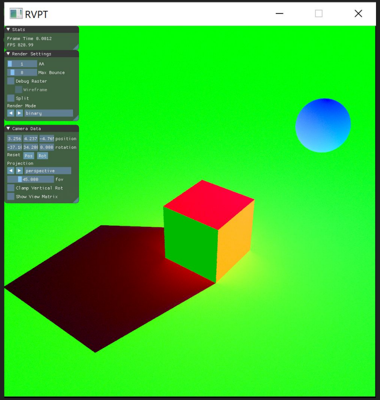
```
▼ Stats
Frame Time 0.0007
FPS 1398.00
▼ Render Settings
1 AA
8 Max Bounce
□ Debug Raster
□ Wireframe
□ Split
Render Mode
◀ ▶ binary
▼ Camera Data
3.083 4.229 -4.241 position
-38.66 38.09 0.000 rotation
Reset Pos Rot
Projection
◀ ▶ perspective
45.000 fov
□ Clamp Vertical Rot
□ Show View Matrix
```

Direct Lighting
(unsampled)



```
▼ Stats
Frame Time 0.0007
FPS 1449.64
▼ Render Settings
1 AA
8 Max Bounce
□ Debug Raster
□ Wireframe
□ Split
Render Mode
◀ ▶ ambient occlusion
▼ Camera Data
1.494 2.029 -2.05 position
-39.58 38.40 0.000 rotation
Reset Pos Rot
Projection
◀ ▶ perspective
45.000 fov
□ Clamp Vertical Rot
□ Show View Matrix
```

Indirect Lighting
(one sample, does
not yet account for
visibility)



```
▼ Stats
Frame Time 0.0012
FPS 820.99
▼ Render Settings
1 AA
8 Max Bounce
□ Debug Raster
□ Wireframe
□ Split
Render Mode
◀ ▶ binary
▼ Camera Data
3.256 4.237 -4.765 position
-37.16 34.20 0.000 rotation
Reset Pos Rot
Projection
◀ ▶ perspective
45.000 fov
□ Clamp Vertical Rot
□ Show View Matrix
```

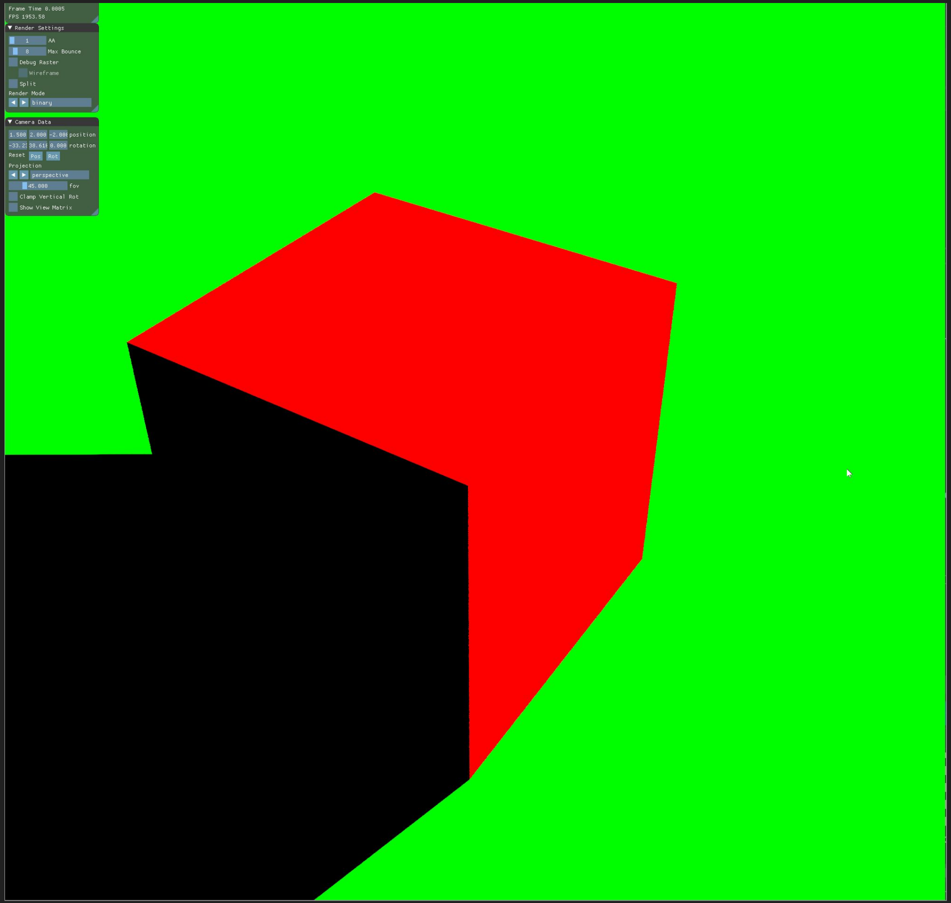
Combined

Camera Movement

- Movement controls with WASD

- Rotate with Mouse

- Standard FPS camera



Frame Time 0.0005
FPS 1953.58

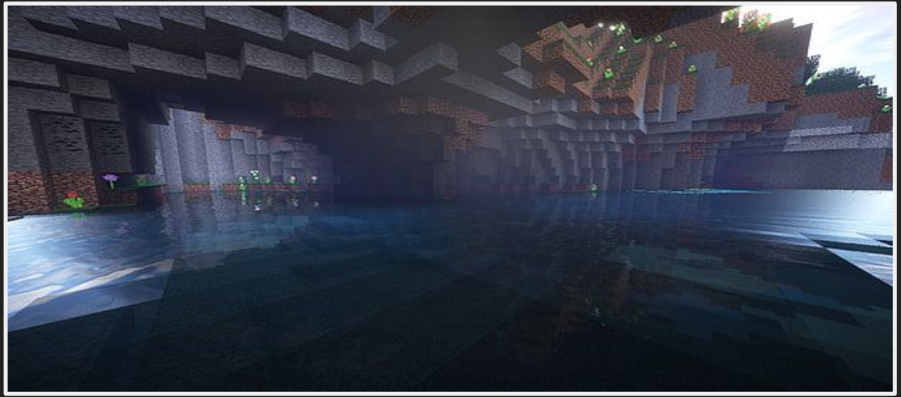
1 AA
8 Max Bounce
□ Debug Raster
□ Wireframe
□ Split
Render Mode ◀ ▶ binary

1.500 2.000 -2.000 position
-33.23 38.616 0.000 rotation

Projection ◀ ▶ perspective
45.000 fov
□ Clamp Vertical Rot
□ Show View Matrix

Goals for Milestone 2

- Probe Implementation
  - Generate rays from singular probe and save info to a texture
  - Disperse and visualize probes in the scene
  - Implement an algorithm to find surrounding probes for every pixel
- Scene generation
  - darkly lit cave
  - sparse internal lights
  - external light (holes in ceiling)
- Try RTX extension again :(



Example of cave scene we want to achieve
(Source: Website)