PHYSICALLY BASED RENDERING

ARDEN RASMUSSEN

1. Introduction

This paper describes the implementation of a simplistic physically based renderer TRM[1]. TRM is a minimal physically based renderer that utilizes ray marching.

2. Shading

Currently to simplify the implementation, the renderer only supports three bidirectional shading functions:

(1) Specular reflection
(2) Specular transmission
(3) Lambertian diffuse

2.1. **Specular Reflection.** Specular reflection is the shading for objects that are perfictly reflective, such as mirrors.

2.2. **Specular Transmission.** Specular transmission is the shading method for objects that are transmissive, such as glass or water.

2.3. **Lambertian Diffuse.** Lambertian diffuse shading is for all other objects, at the moment there are no material shaders that combine some combination of these three.

3. SDF

3.1. **Objects.** There are a number which can be used to construct a scene, and more objects can easily be added, by simply defining their signed distance functions.

(1) Sphere
(2) Box
(3) Cylinder
(4) Torus
(5) Plane
(6) Pyramid
(7) MengerSponge
(8) SerpinskiTetrahedron

Date: April 21, 2020.

[1] `https://github.com/LuxAter/trm`

3.2. **Operations.** There are also a number of operations that can be done to each SDF, which in turn defined a new SDF. This means that operations can easily be concatonated.

(1) Elongate
(2) Round
(3) Onion
(4) Union
(5) Subtraction
(6) Intersection
(7) SmoothUnion
(8) SmoothSubtraction
(9) SmoothIntersection

4. Optimization

Optimization is fairly easy to implement by using OpenMP[2]

```
#pragma omp parallel for schedule(dynamic, 256) shared(buffer)
```

This line is placed directly priore to the main render loop. It splits the loop into chunks of 256 steps, and dynamically allocates the chunks to each of the available threads. The final part of the line declares that the image buffer is shared by each thread.

For more information on how OpenMP is used, visit the OpenMP documentation.

5. Results

Here are some results of the renderer.

[2] https://www.openmp.org/