CS 488 - Spring 2025

# Final Project Proposal

Benjamin Colussi - 2046973

## Introduction and purpose:

Welcome to Mirrodin, a planet made entirely of metal. Five moons orbit the planet, each emitting a different wavelength of light corresponding to the five colours of mana, red, green, white, blue, and black. Glimmervoid is Mirrodin's equivalent of a desert. It is a landscape of rolling hills made of hexagonal metal plates stretching infinitely into the horizon.

This project is inspired by the original artwork of Lars Grant-West depicting Glimmervoid for the trading card game, Magic: The Gathering. The scene makes for an interesting simulation of light transport due to the combination of reflective surfaces, multiple distant light sources of various wavelengths, and atmospheric scattering effects. The simulation of light in this scene will be difficult to implement because of the purely reflective objects that make up the landscape. As rays are traced and reflected, the colour that is shown in each pixel will ultimately correspond to the colours of the atmosphere and light sources themselves. The light sources are also in the distance, and will take some creative modelling and simulation to get the desired effect of being giant, distant objects in the sky.

Reference art depicting Glimmervoid, as well as the coloured moons and hazy atmosphere of Mirrodin:


## Technical outline:

In order to simulate the scene in a physically-realistic (as physically realistic as an alien atmosphere can be simulated) manner, unidirectional path tracing will be used to model light transport. Path tracing is good at modelling the reflection of light rays, and the scene will produce many reflections due to the smooth metallic materials. The existing ray tracer can be extended to a path tracer in a logical manner. A BVH has already been defined and I have implemented SAH-based sorting to improve performance.

Ray marching will be used to simulate the volumetric scattering of the participating media in the atmosphere. Since the entire scene will be from within the atmosphere of the planet, ray marching will be used for all rays. Physical models for light scattering will be used, such as Beer-Lambert, Rayleigh, and Mie. Constant values similar to those corresponding to Earth, its moon, and its atmosphere will be used as a starting point, from which artistic experimentation may expand in order to get the desired atmospheric effect of the alien planet. Since moons are much closer than suns, we can use metres as a unit without worrying too much about floating point inaccuracies. Further optimizations may be pursued to speed up the rendering process, as path tracing and ray marching together will be computationally expensive.

We will have to define new structures to model the atmosphere, as well as the distant light sources. Using spheres will make implementation simpler, because the scene will be static and the atmosphere and sky will be fixed, so we will not require actual object files. We will be able to load in a landscape object to the renderer, similar to functionality of the base code. This will allow us to experiment with different objects and scenes, as well as test objects more easily. I found a simple way to model a hexagonal landscape in Blender, so I will be able to build the landscape object file myself.

Photon mapping will be used in a first pass to enhance the visual quality of the metallic surfaces under the intense lighting, rendering caustics. We will implement a photon structure, which contains location and direction data of photon hits on surfaces in the scene. We will also store these photon hits in a KD-tree, which makes sense because of the spatial nature of the data. Photons will be traced from all light sources towards the scene, in our case we will only trace photons from the moons towards the atmosphere, as other directions would be pointless. After the first pass, we will then use the path tracer to retrieve photon data and render the image, while also ray marching to calculate the atmospheric effects.

**Objectives:**

The objectives of my project are listed in a rough order of implementation, with the meaty objectives being unidirectional path tracing, atmospheric scattering, and photon mapping.

1. Refactor existing renderer:
    a. Create an offline rendering process that writes to a single .ppm file.
    b. Implement parallelization using OpenMP.
2. Unidirectional path tracing:
    a. Monte Carlo integration of PDFs.
    b. Multiple importance sampling.
    c. Recursive ray tracing with Russian roulette.
3. Atmospheric scattering:
    a. Ray marching: Must march rays at all times, because our camera view will be from within the atmosphere.
    b. Beer-Lambert law: Models the attenuation in intensity of light passing through a homogeneous medium. The intensity decays exponentially in the absorbance of the medium, and absorbance is proportional to the length of beam passing through the medium. Quantifies extinction of photons.
    c. Rayleigh scattering: Models scattering of light by particles much smaller than the wavelength.
    d. Mie scattering: Models scattering of light by particles comparable to the wavelength.
4. Photon mapping:
    a. Photon structure.
    b. KD-tree to store traced photon spatial hits.
5. Modelling:
    a. Spherical interfaces:
        i. Necessary for modelling the static atmosphere and moons.
        ii. Must implement ray-sphere intersections.
    b. Light sources:
        i. Moons must be modelled as distant, emissive light sources to get a glowing effect with different wavelengths of light.
    c. Metallic surfaces:
        i. Implement accurate models for smooth reflective metallic surfaces. This is already implemented in Assignment 1.
6. Object files:
    a. Use Blender to model hexagonal landscape.

Please note that I have already received credit for implementing SAH-based BVH in Assignment 1.

**Bibliography:**

Used as reference for implementing path tracing, volumetric scattering, and pretty much everything else too:

- Matt Pharr, Wenzel Jakob, and Greg Humphreys. 2016. Physically Based Rendering: From Theory to Implementation (4th. ed.). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Atmospheric scattering:

- Display of The Earth Taking into Account Atmospheric Scattering, Nishita et al., Siggraph 1993.
- Display Method of the Sky Color Taking into Account Multiple Scattering, Nishita et al., Siggraph 1996.
- Preetham, A. J., P. S. Shirley, and B. E. Smits. 1999. A practical analytic model for daylight. In Proceedings of SIGGRAPH '99, Computer Graphics Proceedings, Annual Conference Series, 91–100.

Used as supplementary reference. Good article on atmospheric scattering implemented with ray marching.

- Scratchapixel 4.0, founded by Jean-Colas Prunier around 2009.

Photon mapping:

- Jensen, H. (1996). Global Illumination using Photon Maps.