Exercise 1

Classes: week 10 (April 28th) and week 11 (May 5th and May 8th)

Introduction

You will implement part of the Turner Whitted Ray Tracing algorithm covered in the theoretical class. The algorithm consists in tracing primary and shadow rays to calculate local colors and shadows. Students will use the P3F (P3D file Format) scene files.
P3F language is designed as a minimal scene description language for the scenes used to benchmark the ray tracers developed by the students in P3D Course. **Read carefully the P3F format description in attachment.**

To implement the Ray Tracing algorithm, you should use the provided Visual Studio 2022 **DistributionRayTracer** solution for Windows OS.

IMPORTANT: Download and install the free Visual Studio 2022 Community edition. The free version provides all that you need.

Compiling and running

Download the zip file **DistributionRayTracer.zip** to your computer, extract it and run the solution. Make sure you have selected "Release" configuration and "x64" on the platform. In the Build tab, select Clean Solution and then select Build Solution. The application will ask for a scene name. Provide any file from the P3D Scenes directory, for instance, **balls_low.p3f**. It will just draw blueish points in a running OpenGL window. By disabling the global boolean variable **drawModeEnabled** in the main.cpp file, the program just creates an output image file named RT_Output.png.
The project has a Dependencies folder, which includes the **freeglut**, **glew** and **Devil** packages, within the project. Also notice that, in the renderscene() function in the main.cpp file the main loops that calculate the pixels (samples) colors are parallelized by exploting the OMP (Open Multicore Processing) library.

Tasks

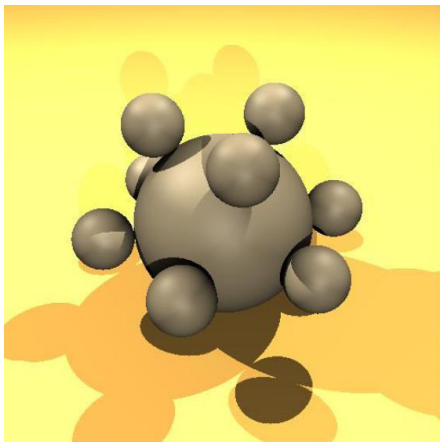
Now, you should go to the renderscene() function in the main.cpp file. Notice that the renderscene() function is divided in two main zones: progressive (zone A) and not progressive (zone B). Within zone B, it is subdivided in other two subzones: zone B.1 for Distribution RayTracer and zone B.2 for Whitted RayTracer. The task for the first 3 lab classes will be to implement the Turner Whitted Tracer. So, go to zone B.2 and complete the implementation of rayTracing(ray, 1, 1.0, dummy) function as well as the scene->GetCamera()->PrimaryRay(pixel) in the camera.h file.

The students should program the following features:

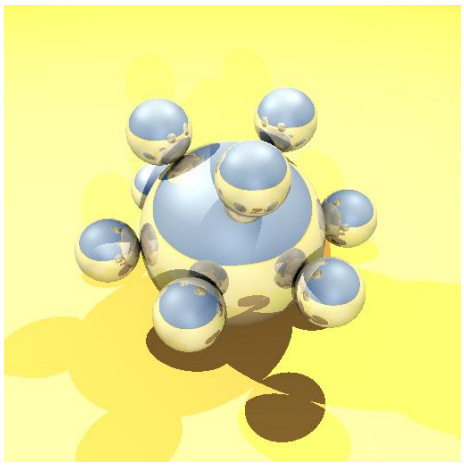
- Local color component (Blinn-Phong model illumination)
- Multiple source lights
- Hard Shadows
- Global color component by implementing the mirror reflection (and refraction with Schlick approximation of Fresnel Equations for dieletric materials.
- Ray intersections with spheres, triangles, and axis-aligned boxes.

The code of the provided project has already implemented several C++ classes like scene, camera, ray, geometric objects, material, lights, bounding box and more. It also provides some math functions, a P3F parser function, support for loading cubic textures (skybox) as well as getting the environment color from a light ray. You are free to use these features, to change and/or extend them.

I**MPORTANT**: this code comes with no guarantee, so use it at your own risk. Using the balls_low.p3f scene test file, your algorithm, in the first lab class, just with local color (or direct illumination) and shadows, should produce the following image :

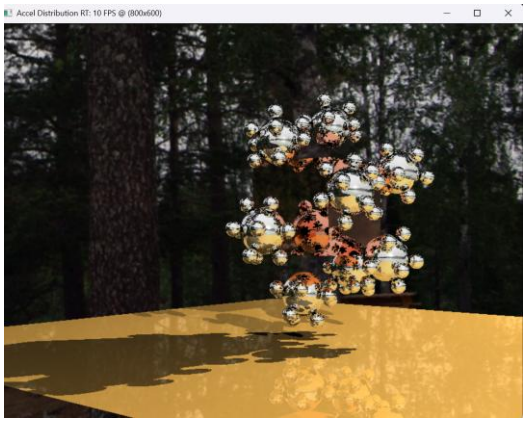


In the second and third lab classes, by implementing the indirect illumination, you should get these images:



balls_low

teste



Balls_box