Introduction to Raytracing and Shading

The goal of this assignment is to implement your first ray tracer and get familiar with different shading. In the first exercise, you will implement a raytracer and use it to render spheres and planes with different projections. In the second exercise, you will extend the ray tracer to support different shading.

Using Eigen

In all exercises, you will need to do operations with vectors and matrices. To simplify the code, you will use Eigen. Have a look at the Getting Started page of Eigen as well as the Quick Reference page for a reference of the basic matrix operations supported.

Preparing the Environment and Submission

Follow the instructions on the general instructions page to set up what you need for the assignment.

Ex.1: Basic Ray Tracing

Description

In this exercise, you will modify the provided code to launch primary rays from the camera following a perspective projection model, and implement the intersection with a parallelogram.

Tasks

Each `TODO` in the provided code corresponds to a part that you need to fill to complete the assignment.

Ray Tracing a Sphere

1. Replace the ad-hoc ray sphere intersection algorithm with the generic one derived in class.

Ray Tracing a Parallelogram

1. Set up the parameters of the parallelogram (position of the corner, plus one vector for each side of the parallelogram)
2. Create a function to check if a ray intersects with an arbitrary parallelogram.
3. Calculate the intersection between the ray and the parallelogram using the function defined above.
4. Compute the normal of that intersection point.

Ray Tracing with Perspective Projection

5. Modify the ray equation for perspective projection.
6. Compare the difference in the result for a sphere, for a parallelogram (you can also create a scene with multiple objects).

Ex.2: Shading

Description

In this second exercise, you will implement the shading equations introduced in class.

Tasks

1. Implement the basic shading components discussed in class: ambient, specular, and diffuse.
2. Add RGB components instead of the current grey-scale one.
3. Experiment with the different parameters and observe their effect on the ray-traced shaped.

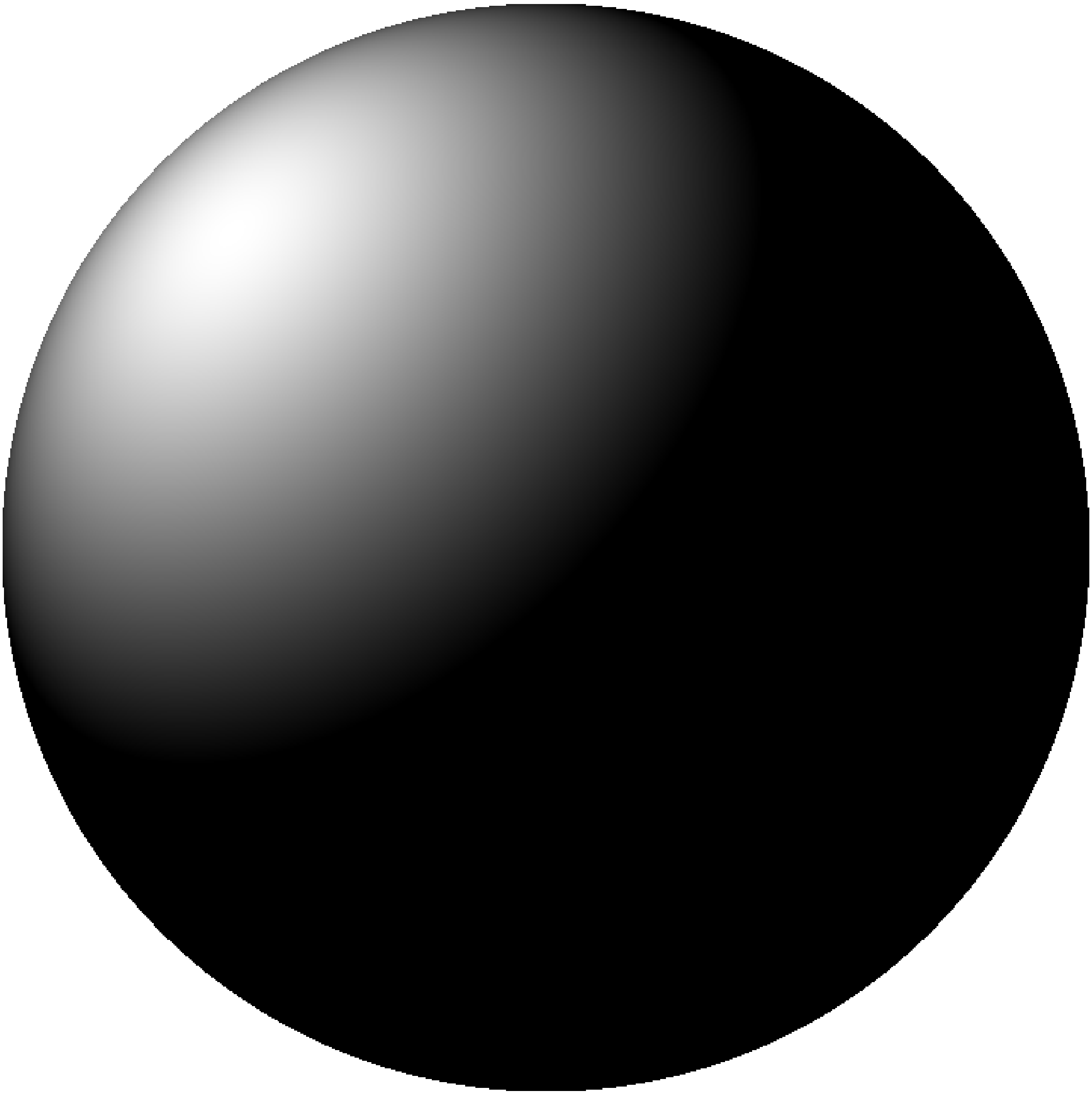
Starting Code

After compiling the code following the process described in the general instructions page, you can launch the program from the command-line as follows:

```
mkdir build; cd build; cmake ..; make
./assignment2
```

Once you complete the assignment, you should see result pictures generated in your folder.

If you run the code provided in this assignment, it should produce the following image in your current working directory:



Tip: if you are using VSCode you can open the png in a tab, and it will automatically refresh every time the png is updated.