UNIVERSITÄT BERN

Introduction to Computer Graphics

Assignment 1 – Ray-tracing Planes and Cylinders

Handout date: 27.09.2019

Submission deadline: 04.10.2019, 12:00

Late submissions are not accepted

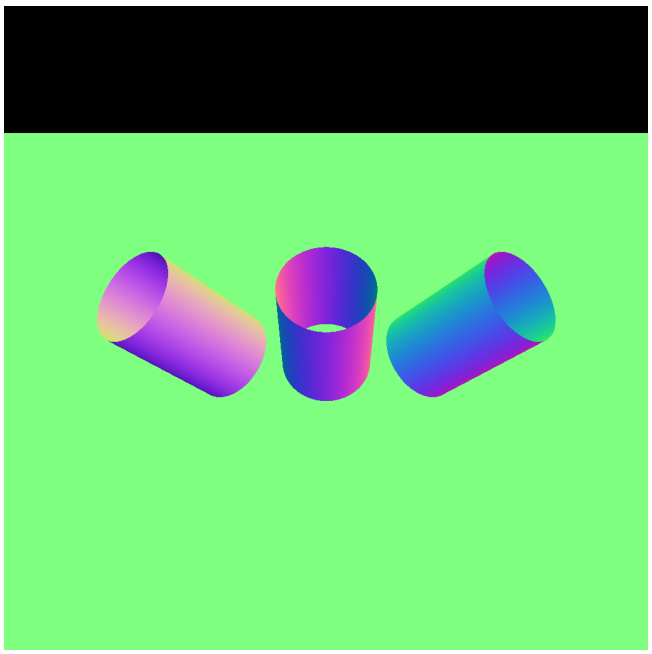

Figure 1: Expected results for `scenes/cylinders/cylinders.sce`

In this assignment, you will implement ray intersections with planes and cylinders and compute surface normals at the intersection points. The framework code provided this week is identical to last week's, except "`todo`" comments have been inserted in `Plane.cpp` and `Cylinder.cpp` to indicate where you need to add your implementations. If you already set up a GitHub repository last week to collaborate with your fellow group members, you can copy these TODO comments over to your repository (or just note where your implementation needs to go and get started).

In the `expected_results` directory, we provide the images you should expect your finished code to produce for a subset of the provided scenes. One such result is shown in Figure 1. The framework is configured to visualize the surface normals in false color (the normal XYZ components in $[-1, 1]$ are mapped linearly to RGB values in $[0, 1]$).

Notes on Implementing Cylinder Intersections

You are asked to compute the intersection with an *open* cylinder (i.e. without end caps). The approach we recommend you take is to first determine where the ray intersects a version of the cylinder that extends

infinitely in each direction (as if `height` $= \infty$). Then, from this list of intersection candidates, discard those that fall outside the cylinder's actual extent. Finally, choose the first remaining intersection that appears in front of the viewer.

Hint: you may find it helpful to read over `Sphere::intersect` in `Sphere.cpp`.

Theory Exercise

Please give a derivation of the formulas you use to solve for the cylinder intersections and **compute cylinder normals**.

Grading

Each part of this assignment is weighted as follows:

- Ray-plane intersection: 20%
- Ray-cylinder intersection + normal derivations (theory exercise): 15%
- Ray-cylinder intersection implementation: 45%
- Cylinder normal implementation: 20%

What to hand in

A .zip compressed file with the following contents:

- Hand in **only** the files you changed (in this case, `Plane.cpp` and `Cylinder.cpp`) and the requested program output. It is up to you to make sure that all files that you have changed are in the zip.
- A `readme.txt` file containing a description on how you solved each exercise and the encountered problems.
- Other files that are required by your `readme.txt` file. For example, if you mention some screenshot images in `readme.txt`, these images need to be submitted too.
- A `TheoryExercise.pdf` containing your cylinder-ray intersection derivations. **Make sure this file is well-readable, i.e. decent resolution, contrast etc.** – You will not receive points for what we cannot decipher.

Submit solutions to ILIAS before the deadline. Late submissions receive 0 points!