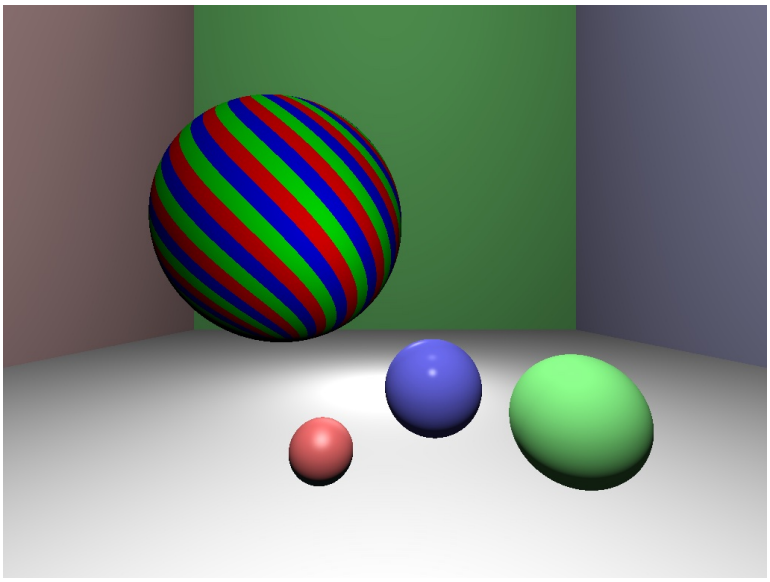
Università
della
Svizzera
italiana

Faculty
of Informatics

Computer Graphics (Fall 2023)

Assignment 3:
Ray-plane intersection, texturing, tone mapping

October 5, 2022



In this assignment, you will extend our current raytracer by adding planes, texturing, and more realistic illumination handling. You will be given an updated framework with clearly marked places for implementing the required functionality. The updated framework is also a solution for the previous assignment. Upon completing all the exercises, you should be able to generate an image similar to the one above.

Updated framework

We updated the framework by:

- adding *Textures.h* file, which contains functions generating procedural textures,
- extending the definition of the *Material* structure by adding variable `texture` which is the reference to the function for computing the procedural texture; by default set to `NULL` - no texture,
- extending the definition of *Hit* structure, which now can store texture coordinates, `uv`, for objects that can be textured,
- adding a new class *Plane*,
- extending function `PhongModel`, such that it takes also texture coordinates as a parameter,
- adding dummy function `toneMapping`.

The code is commented. Before continuing with the assignment, please familiarize yourself with the updates and check the code for comments indicating the places which should be modified.

Exercise 1 [5 points]

In this exercise, your task is to implement a plane-ray intersection routine and add six planes to the scene, forming a box around the spheres we already have. More specifically, the template already contains the class `Plane`. Your tasks is to implement the `intersection` function. Like what was discussed during the lecture, we define a plane using one point and the normal. Additionally, the constructor of the class can take a material structure as an argument. After implementing the intersection routine, please add to the scene six planes such that they form a box extending from -15 to 15 along x direction, from -3 to 27 along y direction, and from -0.01 to 30 along z direction. You can specify the material of the planes according to your preference, but they should not be completely black.

Exercise 2 [3+2 points]

Now, we will add a textured sphere to the scene. You will find the code for declaring and adding the sphere in function `sceneDefinition`. You will first need to uncomment it. Next, you need to make sure that the `intersection` function for the sphere correctly sets the texture coordinates, `uv`, for the intersection point and the function `PhongModel` uses the texture function in the `Material structure` to set the diffuse color of the object. You can distinguish between objects with and without texture by checking whether the pointer to the texture function in the `Material` structure given as an argument to `PhongModel` function is `NULL`. Finally, you have to implement a procedural texture function in file *Textures.h*. You can choose between implementing the checkerboard pattern discussed during the lecture or tilted red, green, blue stripes as shown in the image above. You will get 3 points for the correct realization of the checkerboard pattern and 5 for the color pattern. Again the image does not need to look identical to the one in this document, but the sphere should have a similar red, green, blue stripe pattern which is tilted, i.e., neither horizon nor vertical.

Exercise 3 [5 points]

Finally, make your illumination more realistic. Introduce the attenuation of the light due to distance in `PhongModel` function. You should also take care of the tone mapping and gamma correction by implementing a simple tone mapping routine. To make the image look nice, you will most likely need to tweak the intensities of the lights and coefficients in the material definition. Since we were not considering gamma correction before, our current raytracer has exaggerated ambient illumination. Try reducing both the intensity of the ambient light and the ambient coefficient for all materials. You are encouraged to play around with all the settings related to the light computation to create an image you like. However, please do not change the geometry of the scene, i.e., the positions and sizes of the objects.

Submission

Your submission **must** contain only one ZIP-file containing:

- a readme file with information which exercises you solved, the authors of the solutions, and an explanation of encountered problems, if any,
- one `result.ppm` file generated by your code after solving the exercises,
- a folder named `code` which contains all the files required to run your code, in particular there should be `main.cpp` file which will be used to compile your ratracer with the command `g++ main.cpp`.

The ZIP-file you submit **must** be of a form `surname1_surname2_Assignment1.zip` for team of two, and `surname_Assignment1.zip` for a single submission. Only one person from the team should submit the solution. **You MUST follow the above rules when submitting the assignment.**

Solutions must be returned on October 11, 2022 via iCorsi3