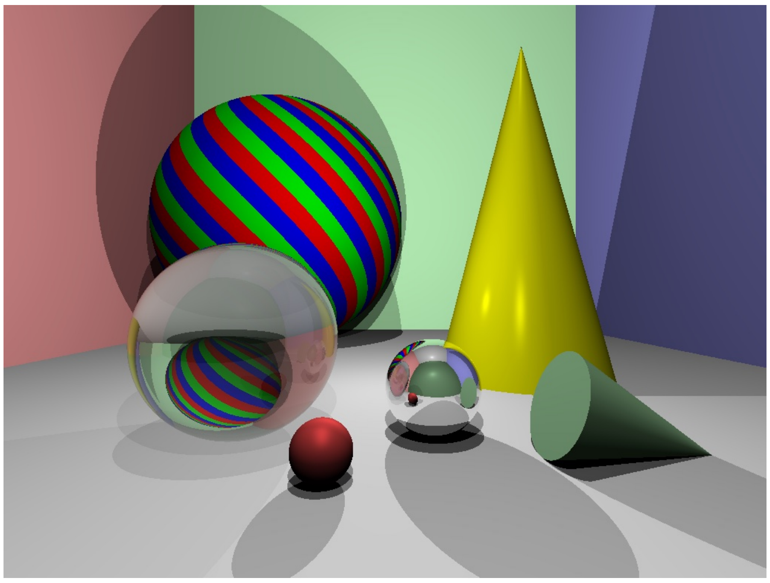
Università
della
Svizzera
italiana

Faculty
of Informatics

Computer Graphics (Fall 2022)

Assignment 6: Shadows, Reflection, and Refraction

October 27, 2022



... and here we are, the final assignment on raytracing, in which you are challenged with implementing the remaining functionalities of every basic raytracer, i.e., shadows, reflections, and refraction. The image above shows what you can expect after correctly solving all the exercises (including the bonus one). The exact result, as usual, will depend on the precise settings of material parameters, illumination, and tone mapping. For all exercises, remember to take care of the numerical issues when computing secondary rays.

Updated framework

By now, you should be familiar with the framework and understand how the raytracer is implemented. The updated framework is the solution to the previous assignment and does not include any additional comments. You can modify it the way you want to implement the different functionalities. You can also take your solution to the assignment 5 and continue with it.

Useful functionality of the GLM library

`glm::reflect` and `glm::refract` are your potential good friends for this assignment. Make sure that you check how they work before using them.

Exercise 1 [5 points]

Add to your raytracer shadows. Probably the most convenient way of doing it is to include shadow computation into the function `PhongModel`. Simply, make sure that you add a contribution of a particular light source to the color of a point, only if the light source illuminates the point. If it does not, discard this light source from the light computation.

Exercise 2 [5 points]

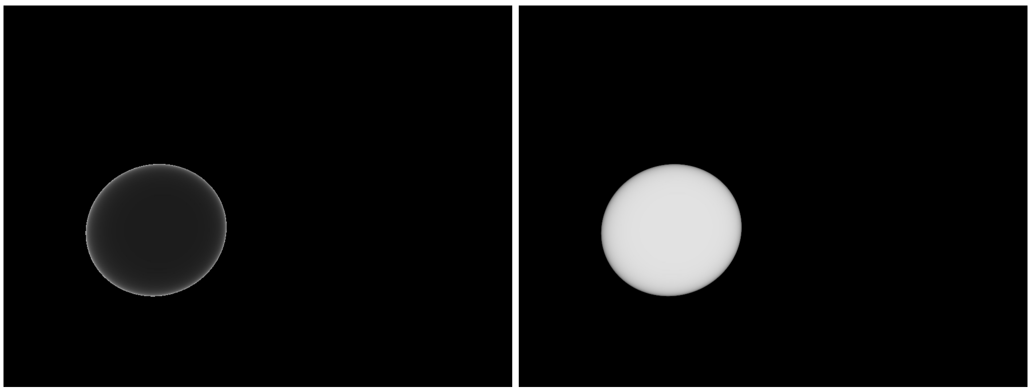
Make your raytracer handle objects made of reflective materials. To this end, you will first need to extend the `Material` structure to keep the information about the reflectivity. The simplest solution is to store a boolean flag indicating whether the material reflects the light or not. For more refined control over the material properties, you can store a real number which indicates the portion of the light undergoing perfect reflection. The rest can come directly from the Phong model of the material. To demonstrate the implemented capabilities, make the sphere in the middle (originally blue, see the image above) reflect the light. Do not change the size of the objects and their positions so you can easily verify whether the reflection matches the image above.

Exercise 3 [5 points]

Implement the refraction effect. Similar to the previous exercise, you need to make sure that the `Material` structure can keep the information whether the object refracts the light or not. Additionally, you have to store the refractive index of the material. When implementing the refraction, be careful about proper handling of ray intersections, coming both from outside and inside of the object. Note that if you consider a glass object, sometimes the rays will be going from air to glass, and in other cases, from glass to air. You have to detect these cases as the refraction computation is different in both cases. One possibility is to keep at each execution of the `trace_ray` function a flag whether you are inside or outside of the object. The other possibility is to determine whether you go into or out of the object based on the angle between the normal vector and the ray direction at the intersection point. To get the full amount of points, it is enough if you get the effect right for a sphere made of solid material. To demonstrate the effect, add a refractive sphere to the scene. The center of the sphere should be at $(-3, -1, 8)$, and the radius 2. The index of refraction for the material of the sphere should be 2.0.

Bonus exercise 4 [2 points]

Extend handling of the refraction by considering the Fresnel effect. In case you run into problems and need to debug your code, it might be useful to visualize for the transparent sphere the two coefficients for the Fresnel effect, i.e., the portion of the light reflected and the portion of the light refracted. Below you can find two images that visualize them for our reference raytracer, reflection coefficient on the left and refraction on the right. Note that to visualize the coefficients you need to disable the tone mapping and the gamma correction. The coefficients should be in range $(0, 1)$. The correct behaviour is that more light should undergo reflection as the primary rays approach the sphere's boundary.



Bonus exercise 5 [2 points]

Animate the reflecting and refracting spheres so they appear jumping, similar to the sample video you can find with this assignment, also shown during the first lecture. Make sure at least one of the spheres deforms as it touches the ground plane.

Submission

You should submit one ZIP-file via iCorsi containing:

- one modified *main.cpp*, and any other files you modified to complete the assignment,
- the final image generated using your code,
- 5-second video clip for the Exercise 5.

Additionally, in the comment to your submission please indicate which exercises you solved and what problems you encountered, if any.

Solutions must be returned on November 3, 2022 via iCorsi3