GPU Programming 2016/17

Assignment 1

In this assignment you will complete the implementation of a ray caster and parallelize it on the CPU.

0.) Download the skeleton code and generate the build system using `cmake`.[1]

1.) Complete the implementation of the ray caster. The default scene currently implemented in `Scene` is shown in Fig. 1.

   i.) Implement `Renderer::render()` that iterates over the pixels in the image, for each pixel generates a ray, and shades the pixel when an intersection point has been found.

   ii.) Implement `Scene::traceRay()` that iterates over all objects and finds the intersection point closest to the camera.

   iii.) Implement `Sphere::intersect()` and `Plane::intersect()`.

   iv.) Implement `Renderer::shade()` that iterates over all lights and calls `Intersection::shade()`.

   v.) Implement the Phong shading model (as discussed in the tutorial) in `Phong::shade()`.

2.) Parallelize the ray caster using C++ threads.

   i.) Implement `ParallelRenderer::init()` that determines the number of threads as a function of the hardware capabilities.

   ii.) Implement `ParallelRenderer::render()` that sets up and spawns the given number of threads to render the image. Ensure that the image plane is tiled in an approximately equal manner in the x and y direction.

   iii.) Implement `ParallelRenderer::renderTile()` that computes a tile of the image.

3.) Analysis of execution time.

[1] Under Linux and MacOS you can use a package manager (apt, brew, port,...) to install `cmake`. Then type `cmake ./build` on the command line to generate the make file. On Windows `cmake -G ''Visual Studio 12'' .\build` generates a Visual Studio solution.

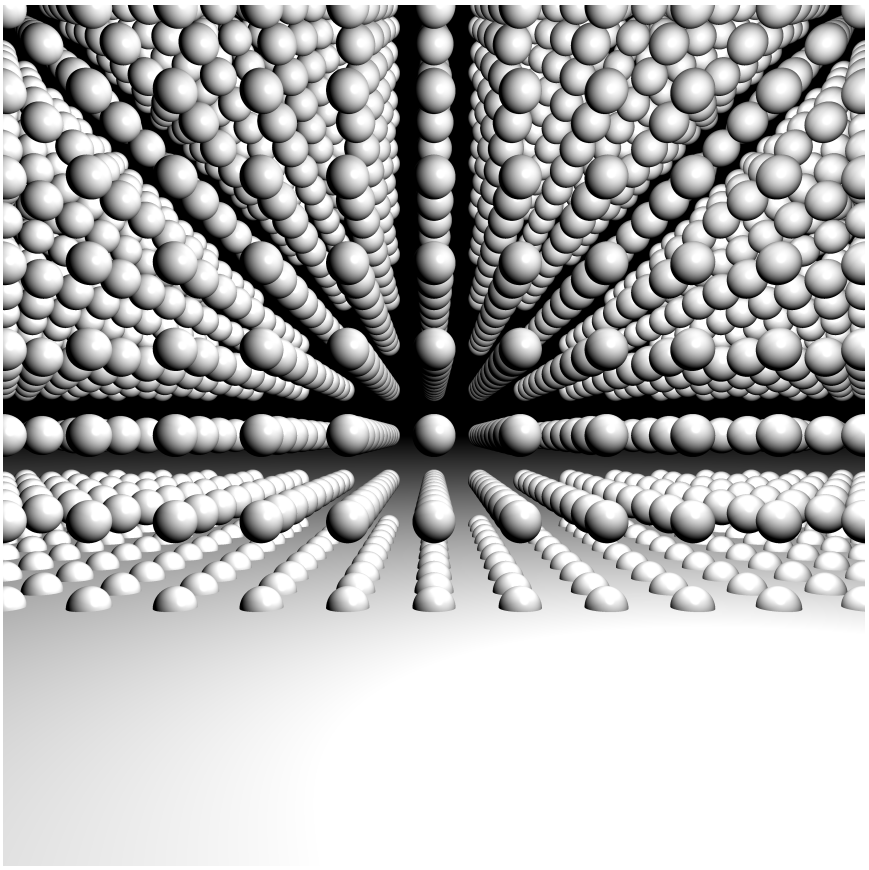

Figure 1: Default scene

i.) Generate a plot with the execution time as a function of the number of threads (see Fig. 2 for an example).

ii.) Analyse and explain the observed performance in the plot. Suggest (and / or implement) ways to improve the efficiency.

Notes:

- Please finish the implemation until the week of 18/11/2016 and sign up for a time slot to discuss your implementation.
- Please make sure that your code compiles and executes on the machines in the lab or bring your own machine.
- Please email gpucourse@isg.cs.ovgu.de for bug reports (with diff if possible) or comments.

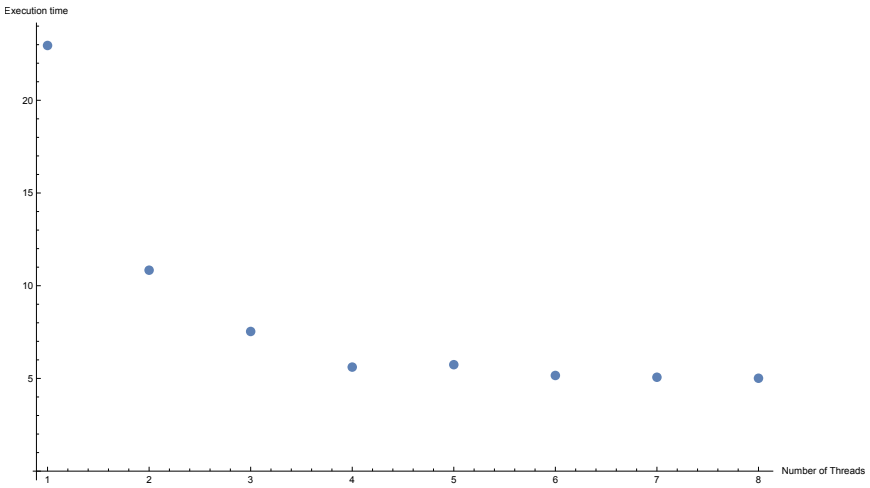

Figure 2: Execution time as a function of the number of threads.

- Plagiarism policy: you can discuss the assignment but everybody has to implement their own solution.