Global Illumination Methods

Practical Course

14 November 2018
Till Niese, Jochen Görtler

Universität
Konstanz

graphics.uni.kn

CMake

Adding Files

To add new file to your project you have to modify the `SOURCES` and `HEADERS` variables in the *CMakeLists.txt* file:

```
set(SOURCES main.cpp)
set(HEADERS include/gui.h
            include/image.h
            include/ray.h
            # more files ...
)
```

The files will be automatically added to your project after a rebuild in your IDE.

GLM

Documentation

The official documentation can be found on:
https://glm.g-truc.net
(The site is not reachable from the university network.)

The documentation can also be downloaded on our website.

GLM - Data-Types

To use `GLM` you need to include `<glm/glm.hpp>`.
This header will include all common data types and utility functions.

Single precisions (float)

```
glm::vec3, glm::vec4
glm::mat4
```

Double precision (double)

```
glm::dvec3, glm::dvec4
glm::dmat4
```

GLM

`GLM` types itself do not have methods to calculate *length*, *angles*, ...
Those operations are provided by utility functions.

```cpp
#include <glm/glm.hpp>

auto v1 = glm::normalize(glm::dvec3(1.4, 0, 0));
auto v2 = glm::normalize(glm::dvec3(0, 1.4, 0));

auto angle = std::acos(glm::dot(v1, v2));
```

Work Package I

Tasks

1. Make classes for various entities.
2. Come up with a basic scene that contains objects from these entities.
3. Implement intersection test for these entities.
4. Setup the view frustum for the raytracer.
5. Perform a simple raytracing and assign a solid color to each object.

Date

This assignment is due **December, 5th**. Please bring your Laptop to class.
If you have any questions regarding the assignment, just write us an email.

Task 1

Implement classes for the following entities.

Implicit representation

- Sphere
- Triangle

Explicit representation

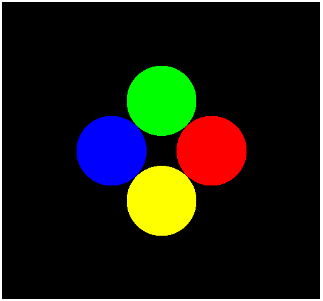
- Sphere
- Quad
- Cone

For each entity you need to implement the `intersect` and `boundingBox` methods.

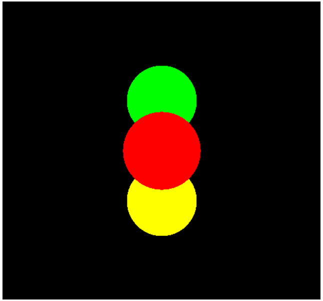
Task 2

Scene

Create a simple test scene with all object types and representation.
Assign a hard coded color to each object.



Front view



Right view

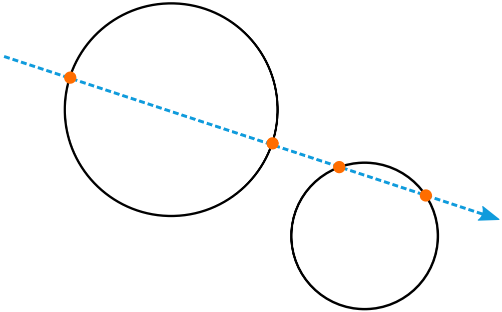
Task 3

Intersection test

For implicit representations there are analytical ways to calculate the intersection points.

Explicit representations are usually implemented using lists of triangles.

If you get multiple intersection points, choose the right one.



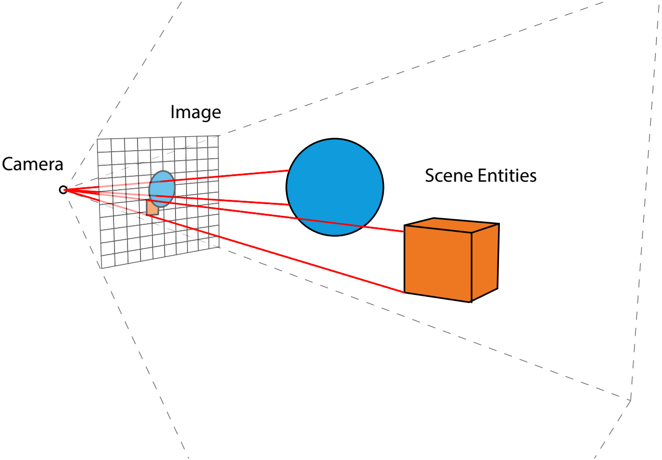
Task 4 & 5

Raycasting

Implement `run()`, which shoots a `Ray` though each pixel.

For each `Ray` determine the closest object being hit and set the pixel to the color defined in the `Material`.

Be careful to leave the general structure of the nested `for`-loops intact.



Literature

- P. Shirley, S. Marschner.
  Fundamentals of Computer Graphics.
  A K Peters, Ltd., 2009.
- T. Möller, B. Trumbore.
  Fast, minimum storage ray-triangle intersection.
  Journal of Graphics Tools, 2(1):21-28, 1997.
- P. Shirley.
  Realistic Ray Tracing.
  A K Peters, Ltd., 2000.