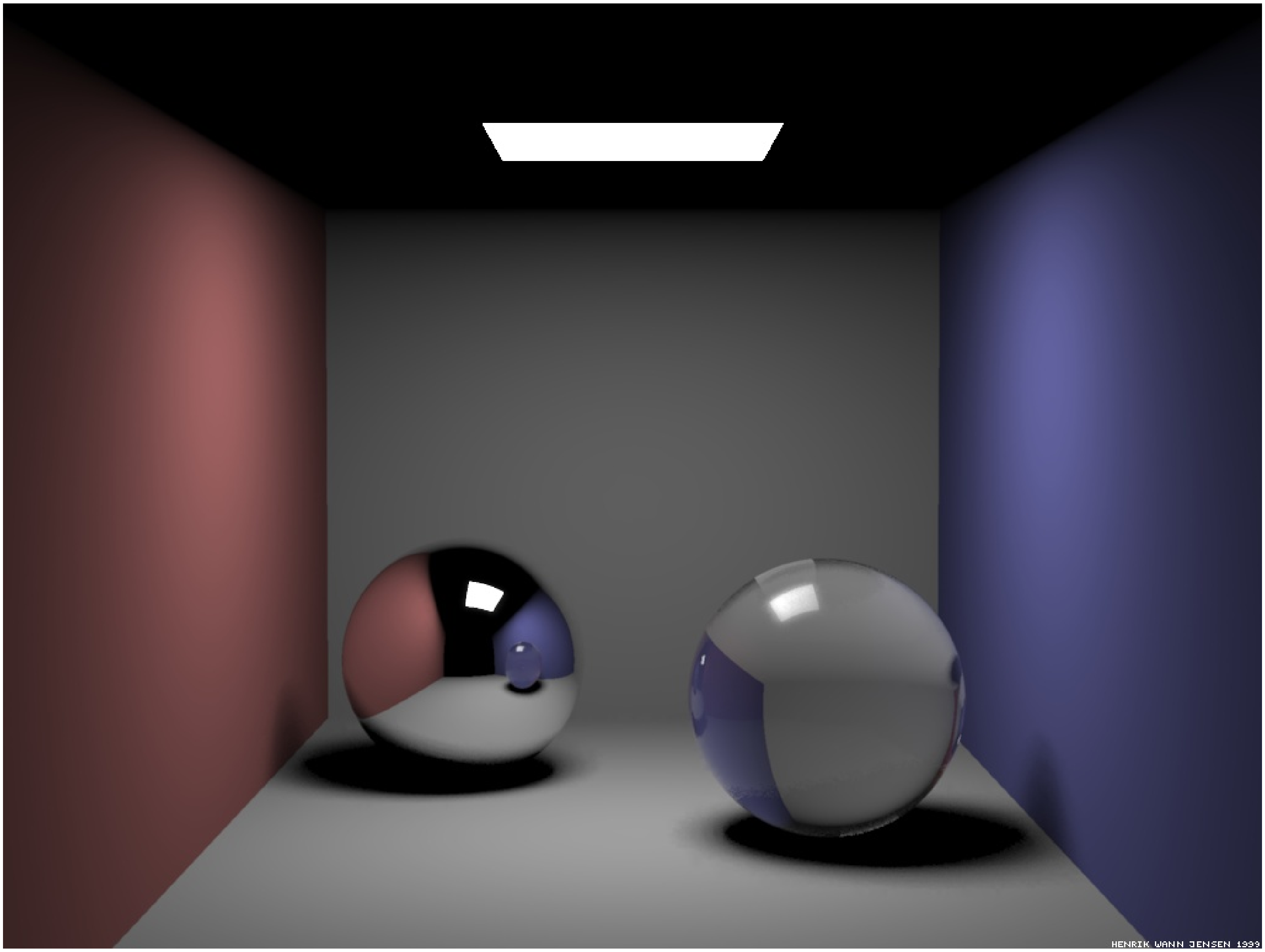
CSC 305 Assignment 1 - Ray Tracer

Due 11:55 p.m. Monday, February 1st, 2016



(An example of a high-quality ray traced image. To render such an image will require much more knowledge than our assignment. Image credit: Dr. Henrik Wann Jensen, UCSD. 1999.)

In this assignment you will write a simple ray tracer using C++. The goal of this assignment is to familiarize yourself with the development environment with C++ & CMAKE, and to learn the fundamental aspects of synthesizing a digital image from a three dimensional scene.

The *minimum requirement* of this assignment is to write a ray tracer that renders a sphere from a fixed point of view. The sphere should cast a hard shadow to a floor plane beneath it. To improve your grade for this assignment, you can add *advanced features* such as mirror reflections, shadows, anti-aliasing, etc. A list of possible improvements is listed below. You are encouraged to propose and implement your own advanced rendering features, and marks will be given according to the difficulty of your enhancements. But please come to talk to your instructor/TA before you start to do anything that is not on the list!

We will provide a code framework for display / save images (see Resources/ImageKit) You are not required to use it for this assignment, but it is highly recommended to start with this framework if you are not experienced in C++ graphics programming.

Minimum Requirements (70%)

Features listed below are worth **70%** of the mark for the first assignment. Everyone in the class is expected to submit this part.

- The program compiles and runs without crashing.
- The program outputs an image that renders
  - one **sphere**,
  - a **floor (plane)** beneath the sphere,
  - with **diffuse** surface shading on both the sphere and the floor,
  - illuminated by a single **point light source**,
  - rendered from a **fixed point of view**,
  - and the sphere casts **hard shadow** onto the floor.

- Quality and style of your submitted code will also be marked, including
  - Data structures used
  - Efficiency of the implementation (running time)
  - Documentation.
  - Please provide a README file explaining features implemented in your program with your Connex submission.

Advanced Requirements (30%)

Features list below are worth **30%** of the mark for the first assignment. Therefore if you implement all features listed in the minimum requirements and the advanced requirements section, you can get up to a full mark in this assignment. If you want to substitute some features of your own proposal, please consult your instructor/TA first and get some advices.

- Modelling:

  - Render multiple spheres in the scene. **(2%)**
  - Perform simple object-space transforms of your object, moving the spheres around with transform

    - matrix calculations, instead of modifying their centre coordinates. **(3%)**
    - Render a simple triangle mesh, for example, a cube box or a Cornell box enclosing the scene (see the teaser image). **(8%)**

- Rendering:

    - Render the image with super-sampling antialiasing. Casting more than one ray per pixel to improve the jagging, pixelated edge of objects. **(2%)**
    - Light the scene with multiple point light sources and cast shadows from all light sources. **(5%)**
    - Render a reflective surface, for example a mirror ball (see the teaser image). **(10%)**

Submission

Please submit your assignment through Connex. In your submission, upload a zip file that contains all codes of your assignment project, with a README file that very briefly explain all features you have implemented.

Do not submit any build files, such as object files, .pdb files or binary executables in your Connex submission.

Marking

The marking of all assignments in CSC305 will be **presentation based**. We will have a demo and presentation session for each of our 3 assignments. During a demo session, every student is expected to give **a short (~3 min), face-to-face presentation** of his/her work to the TA (talking in front of your screen to the TA, not in front of the whole class). At the demo session please bring with you a list of features that you think you have completed correctly, so we can go through it. The TA may ask a few questions to test a student's understanding of his/her solution, also may change parameters of the student's program to test its correctness and robustness. The presentation will take place in the teaching lab (ECS354), but the student is free to use his/her own laptop to demonstrate the assignment.

The instructor/TA will read the student's submission on Connex, looking at the coding style and comments, code correctness in parts that contains the key math, and documentation files. *The instructor/TA will not attempt to compile or run the student's code on his own after the presentation session: the major portion of the grade is marked at the time of face-to-face presentation.*