Prof. Dr. Reinhard Klein
Max Hermann, Alexander Gress, Christopher Schwartz

University of Bonn
Institute for Computer Science II
November 12, 2013

Winter Term 2011/2012

Introduction to Computer Graphics - Assignment Sheet 3

The practical tasks on this sheet will be assessed in the exercise on November 19 and November 20. You will need to show the running program and explain the source code. Both, theoretical and practical tasks have to be delivered until 8:00 am on November 19. Therefore, e-mail it to `hermann@cs.uni-bonn.de`. A *clear* scan or photo of the theoretical solution in handwritten form is possible.

Practical Part

Task 1 (Transformations & Scene Graph, *7 points*)

In this task, a very simplified model of the planetary system is to be implemented and visualised. For this application where transformations relative between the objects in the scene have to be respected, a scene graph paradigm shall be used, such that each "satellite" has to be transformed only with respect to the corresponding "planet" (e.g. the moon rotates around the earth which in turn rotates itself around the sun). As preparation for this exercise, please read chapter 3 of the OpenGL Redbook, that you can find under `http://www.glprogramming.com/red/chapter03.html`. In particular, take a look at Example 3-6 (Building a Solar System).

- Download the framework from the homepage. It contains a class `COrb` representing an orb/satellite with members for the objects radius, distance to its "father", color and a list of satellites (its "children"). Read also the class description in `COrb`.
- Implement the `COrb` member function `draw()` in `COrb.cpp`. The `draw()` function should render the orb with respect to its local coordinate system and according to the data stored in its members. The `draw()` member function should also call the `draw()` member functions of its satellites. Assume for now that the translation is given in the x-direction.
- Generate and render a scene graph with the Sun in the origin and the Earth at a corresponding position. Adjust the distance and radii such that each rendered orb is clearly visible on screen.
- Use the function `glViewport()` to split your window and render a side view in the left part and a top view in the right part.
- Animate your system by rotating Earth on a circular orbit in dependance of some global "time"-parameter (this requires modification of the class `COrb`). The angular velocity should be adjustable via keyboard input.
- Insert the following orbs into your scene graph *and animate them*:
  - Mars rotating around the Sun.
  - Moon rotating around the Earth.
  - Sputnik rotating around the Earth.
  - Apollo11 rotating around the Moon.

  Choose different angular velocities for each orb.

Note: The distances, radii and angular velocities of the orbs can be chosen freely to produce a visually pleasing result. It is sufficient to represent all orbs by spheres.

Hints:

- To control the speed of the animation the function `glutGet(GLUT_ELAPSED_TIME)` may be usefull which returns the time in milliseconds since the start of the program.

Theoretical Part

Task 2 (Quaternions, *6 points*)

a) Specify a quaternion that performs a rotation about the z-axis by 80°.

b) Specify a quaternion that performs a rotation about the axis $(1, 1, 1)^t$ by 120°.

c) Suppose you want to travel from some place on this planet (say, 65°E 32°N) to some other place (say, 98°W 32°N)) along the shortest possible path (while staying on the surface, of course).
   - Assuming that the earth is a perfect sphere, what is the quaternion corresponding to this travel (i.e. the quaternion that rotates the start point of this travel to the end point along the route)?
   - Suppose you use a plane for this travel, and further suppose the time of flight is eight hours. Where is the plane after four hours? Where after five? Assume for simplicity that the plane travels at constant speed. Indicate these positions and the (approximate) route on the worldmap provided in Figure **??**.
   - What is the last country crossed by the plane before reaching the destination country? Also mark this on the map.

Please indicate also the intermediate steps of your solution. Note that the shortest route on the sphere does in general not correspond to the shortest route on a map. In fact, the shortest route between two points on the sphere is an arc of a great circle (which are circles on the sphere whose centers are coincident with the center of the sphere).[1]

Good luck!

[1] A nice illustration of great circle arcs is provided by the following applet: `http://demonstrations.wolfram.com/ShortestPathBetweenTwoPointsOnASphere/`

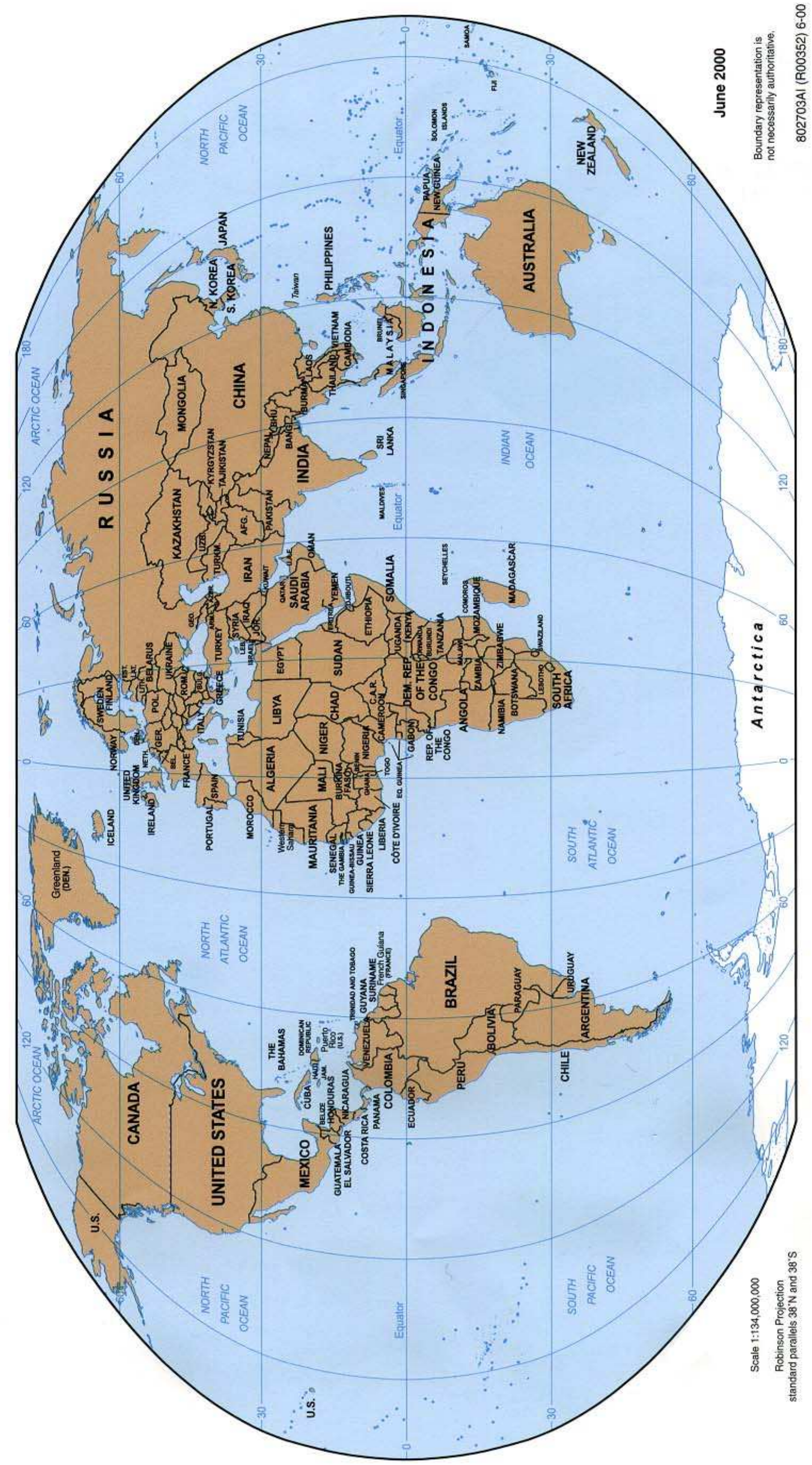

Figure 1: Worldmap