TNM086 – VR-technology
5. VR Workbench

October 11, 2012

1 Introduction

The topic of this lab exercise is Virtual Reality rendering, interaction and navigation. The aim is for you to gain insight in how tracking and display hardware interact with VR software to implement an efficient interface to the virtual environment.

1.1 Examination

The tasks of this exercise must be presented by showing the running program and the code to a lab assistant. You must also be able to answer questions regarding the mathematical and Human/Computer interaction principles as well as the tasks and the implementation of these. Complete all tasks before engaging a lab assistant for examination.

1.2 Equipment

The most important part of this exercise will be performed on the VR workbench in the VR laboratory. Much of the programming, however, can also be done on any computer with the required software installed. Navigation and interaction experiments must be conducted in the lab.

1.3 Software

In this exercise you will be using the Simple Graphics Cluster Toolkit (SGCT) to configure VR display and handle data synchronization. SGCT includes functionality for cluster data and frame synchronization so you final application will run also in the VR Dome and on any other similar supported platform.

The tracking information from head tracker and wand are extracted using VRPN (Virtual Reality Peripheral Network). There are code examples available on how to use VRPN together will SGCT. SGCT also supports keyboard input, which can be used in place of tracking hardware during the initial development.

The implementation of 3D graphics will be done using OpenGL or OpenSceneGraph, so the completion of the associated exercises is encouraged before starting this one. You are also encouraged to read through the instructions before booking the VR equipment and prepare all tasks on other computers, for example in the Linux lab.

1.4 Documentation

In this exercise you will be using both OpenGL and OpenSceneGraph so the documentation associated with the associated exercises may be very useful. Here are listed some sources of documentation for SGCT.

- SGCT Wiki
  https://c-student.itn.liu.se/wiki
  - Getting Started Guide
  - Programmer Reference
  - Code samples
- OpenGL Mathematics (GLM)
  http://glm.g-truc.net
  - Reference manual
  - Code samples

2 VR Programming Using OpenGL

We start by using OpenGL to implement the virtual environment.

Task 1 — Port Your OpenGL Application:
Use the VR software to display an OpenGL application on the VR system. You may use your code from an earlier OpenGL exercise. The display should be rendered in stereo and use the head tracking.

The VR software takes care of both window handling and human computer interfaces, so all such functions performed by GLUT must be removed. Mouse interaction through GLUT should also be removed. You may replace it with VR devices in the following task.

Task 2 — Interaction with OpenGL:
Use the VR software to extract wand tracker data and use this to implement a pointer with which you can select at least one object. The selection should be visually indicated in some way.

Selecting objects is most simply done by calculating the intersection or closeness between a line pointing from your wand and spherical objects, such as planets. There are competent linear algebra packages available that you may use for that.

3 Advanced VR Programming

Now that you know the basic principles of programming interaction in VR it is time to move on to more complicated things. This part of the exercise may be performed with the assistance of a scene graph API, for example OpenSceneGraph. You will try different things so it is recommended that you document how to select between the different functionalities you have implemented.

Task 3 — Port Your Application: (optional)
Use SGCT to display an OpenSceneGraph application on the VR system, or any other scene graph system of your choice. You may use your code from the OpenSceneGraph exercise. The display should be rendered in stereo and use the head tracking.

Task 4 — Implement Navigation:
Use the wand information you can get from VRPN to control the transform of your scene so that you can navigation in your virtual environment. Allow for the switch between "gaze direction" and "cross hair" mode. Start by navigating with constant speed. With OpenGL you will explicitly manipulate the model transform and with a scene graph API you will modify the properties of a transform node that controls the position of the objects in your scene.

Task 5 — Control Flying Speed:
Implement hand controlled acceleration and hand controlled speed for the navigation modes from the previous task. It is not necessary, but recommended, to try out a "dead zone".

Task 6 — Alternative Navigation: (optional)
You may also try implementing any other mode of navigation of your choice. Alternatives are, but not limited to, virtual controls, attractors/repellers, dynamic scaling and goal driven navigation.

Task 7 — Interaction with Objects:
Enable selecting objects on distance in your virtual environment. Indicate the selection in some way, for example by making the bounding box or sphere visible or changing the colour of the object. With OpenGL you have to explicitly calculate or know in advance the position of your objects and use linear algebra to determine intersection. These calculations may be performed using GLM. With OpenSceneGraph you may use the `Intersector` feature.

Task 8 — Manipulation of Objects:
Implement so that you can manipulate at least two different objects in your scene. The manipulation should be based on wand movements and include both translation and rotation of the object. How do you move an object at distance? What is the centre of rotation?

Task 9 — Object Scaling:
Implement a means for scaling the selected object. You may choose any metaphor for this manipulation.

4 Putting All Together

Task 10 — Implement a Game: (optional)
Select OpenGL or OpenSceneGraph of your own preferences to implement a VR game that makes use of both head tracking to provide natural navigation and the wand for absolutely positioned interaction.