Interactive Visual Data Analysis

Assignment 1 – *Camera and Bounding Box*

In this first assignment, starting from a given template project, we will equip our visualization tool with a user-controlled 3D camera and render a simple bounding box.

1.1. SVN Checkout and Template Project

The top-level repository for this practical course is `https://cgsr1.informatik.tu-muenchen.de:8443/svn/pra1314`. Do a SVN-Checkout of the subfolders `pra1314/GROUP` and `pra1314/external` into the same folder, where `GROUP` is the name of your group's repository. Note: you do not have sufficient permissions to checkout `pra1314` directly. Note2: If you work in the computer room create a folder on a local drive (e.g. C:), not on a network drive.

Create a folder called `solution` within your `GROUP` folder, and copy the contents of the template project from `external/TemplateProject` to this folder. Also, create a `readme.txt` in your `solution` folder for future use. The correct folder structure should look like this:

```
.../yourlocaldir/external/
.../yourlocaldir/GROUP/solution/
.../yourlocaldir/GROUP/solution/Template.sln
.../yourlocaldir/GROUP/solution/readme.txt
...
```

Hint: SVN-Commit the current state of your group folder **before** continuing. This will later make it easier to identify temporary files generated by Visual Studio.

Open `Template.sln` with Visual Studio 2012 and hit *F5*. Everything should be compiled without errors or warnings, and on program start a window called *VisTool* should open and render a simple triangle (test both Debug/x64 and Release/x64 configurations). Now you can finally start coding…

1.2. Camera

Every interactive 3D visualization tool should enable the user to explore what he is currently looking at by moving around the camera. DXUT already offers camera classes, so you don't have to implement your own camera. The template project already has a DXUT `CModelViewerCamera`. It is centered at the origin and enables the user to rotate around the scene and zoom the camera using mouse input.

Add a (second) `CFirstPersonCamera` to your visualization tool, so that the camera can also be moved around in first person style. Make it possible to toggle between both cameras.

- Use a global variable of type `CFirstPersonCamera`.
- Forward the required DXUT callbacks to this camera.
  (i.e. call `CFirstPersonCamera::FrameMove()` in `OnFrameMove()` etc.)
- In each frame, get the current world/view/projection matrices from the active camera, combine them and upload the result to the GPU (= set the existing effect matrix variable).
- Bind a hotkey for toggling between the two cameras.

Hint: Have a look at the *DirectX* / *DXUT* / *Effects 11* / *DirectXMath* samples in the Microsoft sample database (http://code.msdn.microsoft.com).

1.3. Bounding Box

When visualizing datasets given on regular grids, it is very common to indicate the boundaries of the domain by rendering a bounding box.

Replace the triangle with a bounding box that has a fixed world-space extent of $[-1;1]^3$. Render only the edges of the box using line primitives.

- You can hardcode the box vertices in the vertex shader just like it is done for the simple triangle in the template project.
- Use `D3D11_PRIMITIVE_TOPOLOGY_LINELIST` in order to draw lines instead of triangles.
- **Optional (advanced):** Realize a fancier bounding box. E.g. use actual geometry instead of lines (cylinders, spheres, …). Note: Transparent bounding boxes (using alpha blending) will make your life harder later on, so e.g. avoid anti-aliased lines.

1.4. SVN Submission

When committing your final solution to your repository **always** refer to the following checklist. Violations of these simple rules will result in minuses!

- Your code compiles and runs **out-of-the-box**. Checkout everything on another computer and test it. Opening `solution.sln` and pressing F5 should produce no errors, for both Debug and Release configurations!
- Your **repository folder does not contain any temporary files or folders** that are created by Visual Studio or during the build process (cf. template project).
  Incomplete blacklist of file extensions/folder names: `exe, lib, obj, fxo, exp, pdb, ilk, idb, lastbuildstate, log, tlog, sdf, suo, ipch, x64, …`
- Your **repository folder does not contain any copies of files** that can be found in the shared repository folders (`pra1314/external`, `/data`, `/bigdata`, …) – except for the template project, of course. Always use relative paths to refer to these files from within your project (for source files, you can "Add Existing Item" to your project in Visual Studio).
- Update your `readme.txt` (if necessary)
  - Should we rate a specific SVN-Revision?
  - Information on the usage of your tool (e.g. key bindings)
  - …
- Your code is commented sufficiently for us to understand it, and important parts belonging to specific exercises are marked appropriately (e.g. "`// assignment 1.2`")

The working solution must be committed till **October 23, 09:00am**. If anything is not working as described here or if you want a specific SVN-Revision to be rated, explain yourself in the `readme.txt` file in your `solution` directory.