MoSIG1 - ICG 25/03/2009

OpenGL - Lab Session 4

Textures

In this lab session, the commands to map textures will be tested and applied to the primitives created in the previous lab sessions.

The result of your first 4 lab sessions must be handed in before Tuesday, March 31st, 2009, 10am at this e-mail address :

`estelle.duveau@inria.fr`

It must contain :

- the code (`main.cpp`)
- the corresponding Visual Studio project (`.vcproj`)
- the images used for texture mapping.

1 Introduction

In the folder of this lab session, there are four images (in BMP format) to copy in the folder of your project. In your current program, copy the new functions (`loadImage()` and `loadGLTexture()`) written in `to_copy.cpp` and replace your `drawSphere()` by the one in `to_Copy.cpp`.

The function

```
int loadImage(char *filename, Image *image)
```

reads a BMP-file of file-name `filename`, stores it in `image` and return 1 if the operation is successful, 0 otherwise.

The function

```
GLvoid loadGLTexture(char* textureName, GLuint* textureIdx)
```

reads the image stored in the BMP-file `textureName` (by calling `loadImage()`) and stores it in the texture object of name `textureIdx`. The mapping parameters are also defined in this function. Analyse this function and make sure you understand what happens in it.

2 Texture mapping

In `initScene()`, load the 4 BMP-files (`textureX.bmp` in the `GLuint textureX` for each primitive `X`), enable texture mapping and define the texture function. Your spheres should now be displayed textured.

We are now going to study the mapping to the objects. In `drawSphere()`, the functions `glBindTexture()` and `glTexCoord()` are used respectively to select the texture used from now on (remember, OpenGL is a state machine...) and to define the texture coordinates of each vertex.

Analyse how the texture is mapped on the sphere and, only then, texture the other primitives so that :

- Cube : the whole image is displayed once on each face ;
- Cylinder : the whole image is displayed once on the tube ;
- Cone : the apex vertex has $(0.5, 1.0)$ as texture coordinates and the vertices of the apex base have from $(0.0, 0.0)$ to $(1.0, 0.0)$ as texture coordinates.

3 Texture filtering

1. Modify the repeat/clamp parameters and observe the results. Don't forget to try texture coordinates greater than 1.
2. Modify the filtering parameters (magnification, minification) and observe the results.
   `Tip :` Zoom on the sphere to see the results of magnification filtering and observe the top of the cube to see the results of minification filtering.
3. Even with linear filtering, the textures sometimes look weird when we zoom in/out (especially the top of the cube). Improve the visual results by using mip-mapping.
4. Try different texture functions.

4 Application

Apply what you've learned to 'paint' (either by materials or textures or both) your model.
`Tip :` For more flexibility, make the texture object's name a parameter of the functions `drawX()`.