CS 550000 Computer Graphics
Homework3 Vertex Lighting

CS 101062337 Salas, Hung-Jin LIN

PROJECT ABSTRACT

Based on homework two, I first set up the shader program environment and variables binding on data exchanging between main program and shader. With little changes in building up vertex information buffer traversal and updating display function, the most work on main program are done. And finally add light equations to calculate new color in vertex shader and fragment shader for Gouraud and Phong shadings.

WORKS

```
struct SubModel {
    GLfloat *vertices, *normals;
    GLMmaterial material;
    unsigned int num_points;
};
```

First thing to do is adding **uniform and attribute variables location bindings** between main program and shader program. The **l**ocations are stored in a nested data struct `GLResource` (shown in right), and won't mess up the program with tons of global variables in code, and that structure is maintained by **class `World` which acted as scene manager of world in Blender**. Next job is to **update the traversal in object** when loading them to `buffers` which are in the right vertex order and will later be passing into graphic display directly. With just little changes on array pointer changes and make them traverse by groups, and with proper structure `SubModel` to store them.

```
struct GLResource {

    struct LightResource {
        GLint ambient;
        GLint diffuse;
        GLint specular;
        ...
        GLint spotCosCutoff;
        GLint is_on;
    };

    GLint Position;
    GLint MVP;
    GLint Normal;
    ...
    GLint EyePosition;
    LightResource Material;
    LightResource LightSource[3];
};
```

Another part of project is **vertex shader**. In shader, transform the vertex positions and normal vectors first, and then follow the light equation to do the vertex lighting calculation; then assign the final result to fragment shader which will forward it to *gl_FragColor* that is the final color for model on displaying. The bonus part on **fragment shader** is very **similar to vertex shader with only altering few input in equation**.

NEW COMMANDS

Add control on light sources: switch on/off, select focused #light then can do position translation on them with J/L/I/K/O/M like previous work. With **`R`** can still go into rotation mode and **also make model spin**.

```
7 : Switch control to previous light
9 : Switch control to next light
```

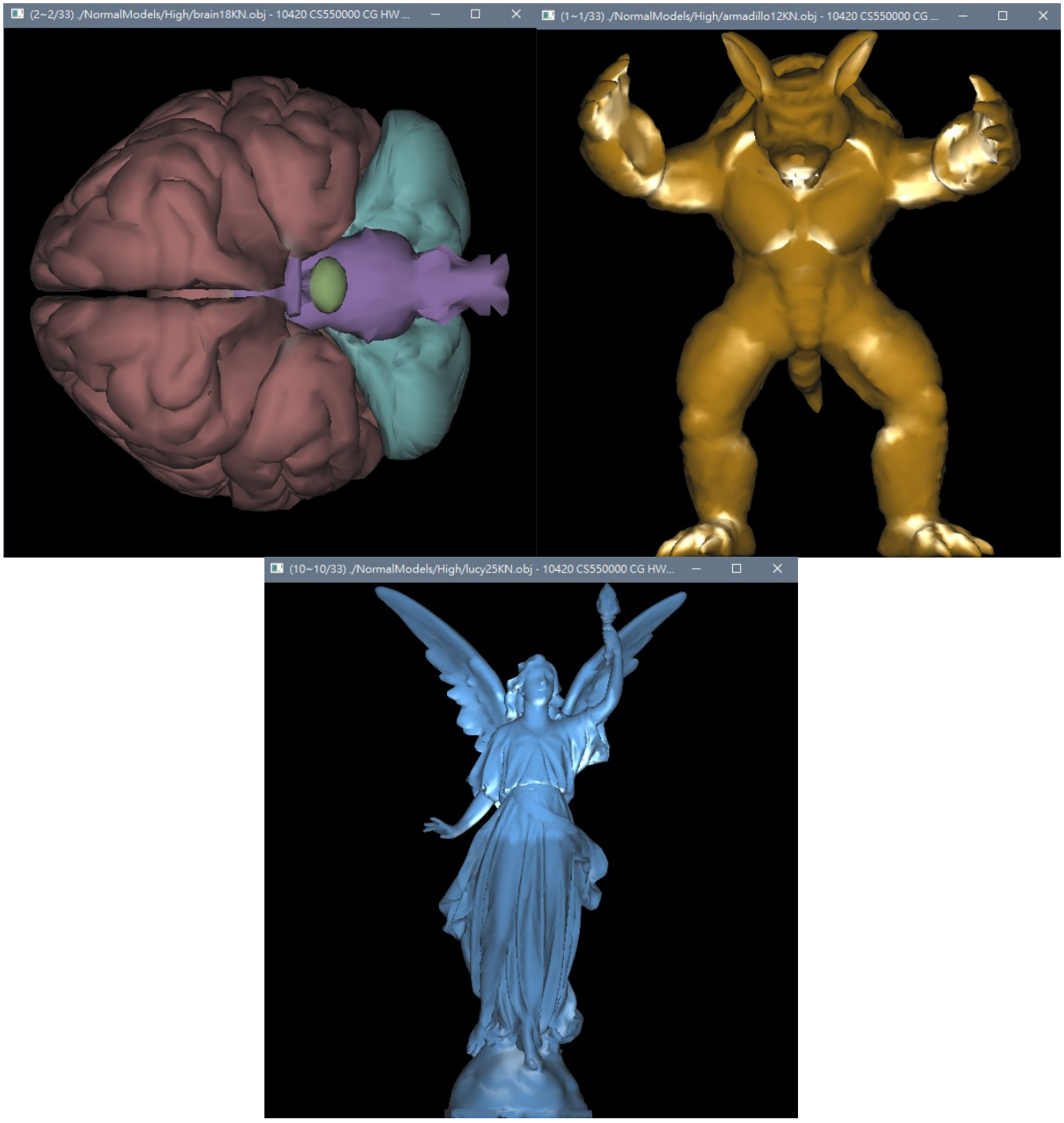
```
1 : Toggle Directional light on/off
2 : Toggle Point light on/off
3 : Toggle Spotlight on/off

5 : Toggle Gouraud/Phong shading
```

SNAPSHOTS

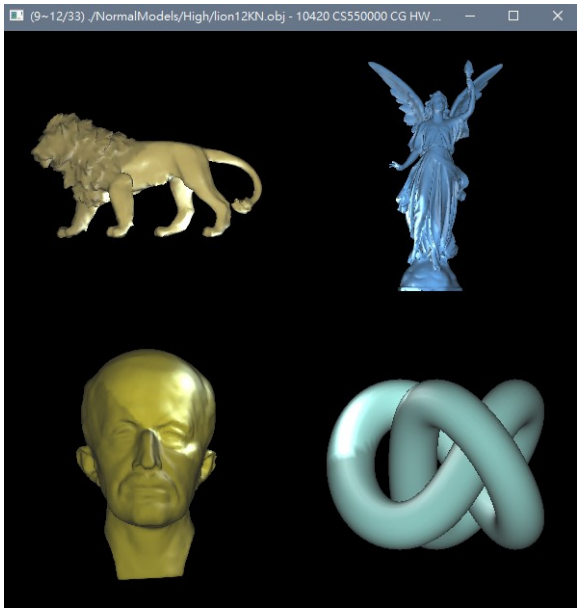
1.1 DIFFERENT LIGHT SOURCE

- **(Top-Left) Directional light; (Top-Tight) Point light; (Bottom) Spotlight (with directional light on).**



1.2 DISPLAY MULTI-MODELS

- **It can still display any four of models and keep in the order when loading with <u>global lighting</u>.**



1.3 PHONG SHADING

- **Display four models with <u>global lighting</u> (use Phong instead of Gouraud shading).**