C++ Exercises

NYCU IDS 310554012
Bo-Cyuan Lin (林柏全) [Date: 2021/7/16]

There are three project files and .exe files respectively. All of them are coded in VS2019 in WIN10.

First project: contains all the exercises in exercise01.

Second project: contains only read file class, since reading the obj also be done in exercise03,04,05

Third project: contains exercise 02-2,03,04,05.

Exercise01, The tables of the headers/classes/functions/variables use in the [Exercise01.cpp]:

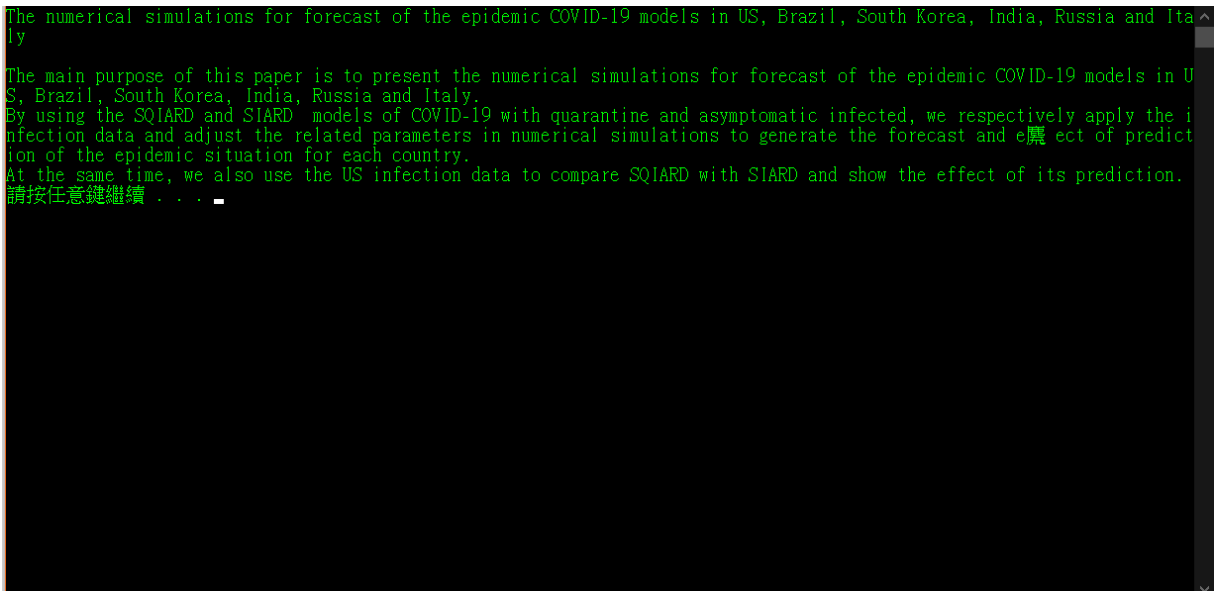
Class/Functions/Variables	**Explanations**
1_swap_exe.h	
Class swap_exe	**The class of different swap functions**
Public:	
void swap_int(int &, int &);	Swap the inputs with type integer
void swap_float(float &, float &);	Swap the inputs with type float
void swap_double(double &, double &);	Swap the inputs with type double
void swap_char(char &, char &);	Swap the inputs with type char
template<class T> void swap_template(T &, T &);	Swap the inputs with type with all types by template
template<class T> void print_status(T &, T &);	Print the status
2_outmessage_exe.h	
void out_message(const char* msg)	Print the input message
3_2D_distance_exe.h	
class distance2D	**The class calculate the 2D distance**
Public:	
float computeDistance(float *, float *);	Calculate the 2D distance
4_3D_distance_exe.h	
class distance3D	**The class calculate the 3D distance**
Public:	
float computeDistance(float *, float *, float *);	Calculate the 3D distance

Execution [Exercise01.exe]

Exercise02-1, The tables of the headers/classes/functions/variables use in the [Exercise02.cpp]:

Class/Functions/Variables	Explanations
1_fileloader.h	
class MyFirstClass	**The class to the read the file**
Public:	
void readFile(const char*);	Read the file with input path
void printf() const;	Print the reading result
Private:	
std::vector<std::string> filebuffer;	Buffer of the getline

Execution [Exercise02.exe]



Exercise02-2-05,

The tables of the headers/classes/functions/variables use in the [Obj_file_loader.cpp]:

Class/Functions/Variables	Explanations
objloader.h	
Class ReadObjfile	**The class to read the .obj**
Public:	
struct vec2 { float x = 0, y = 0;};	Struct for 2D elements
struct vec3 {float x = 0, y = 0, z = 0;};	Struct for 3D elements
struct face {std::vector<int> vertices; std::vector<int> textures; std::vector<int> normals; vec3 fnormal;};	Struct for faces
struct fnormals{std::vector<vec3> fnormal;};	Struct for computing vetex normals
bool readfile(const char* filename);	Read the .obj
bool texture_loader(const char* path);	Load the texture
void object_status() const;	Print the detial of the object
void draw(int default_mode);	Draw the model
void point_mode();	Point mode
void line_mode();	Line mode
void solid_mode();	Solid mode
void keyboard_control(unsigned char key, int x, int y);	Control keyboard
void mouse_control(int button, int state, int x, int y);	Control mouse
void display_texture(bool);	Use texture or not
void display_light(bool);	Turn on light or not
void rotate_deform_idle();	Idle function for deform and rotate
void set_screenshot_path(const char *);	Set the screenshot path
void init_glut(int argc, char* argv[], int, int, float, float);	Initialize the glut
Private:	
void multiple_light();	Set light for glut
void rescale_position();	Rescale the object by dividing maximum distance
void rotate();	Rotate angle
void saveImage();	Save the image use stb
vec3 face_normal(vec3, vec3, vec3);	Compute the face normal base on input three points
vec2 UV_map(vec3);	Compute UV map by formula
std::string mpath;	Store the path of object
int mwindows_width = 500, mwindows_height = 500, mpng_channel = 4; //	Windows width and height and channel's number of screenshot
bool mdis_texture = 0, mset_light = 0, mcompute_uv = 1;	Use texture, light and uv map or not
int maxis = 0, mtheta[3] = { 0, 0, 0};	Rotate parameters
int mMode = 0;	Variable to change displaying mode

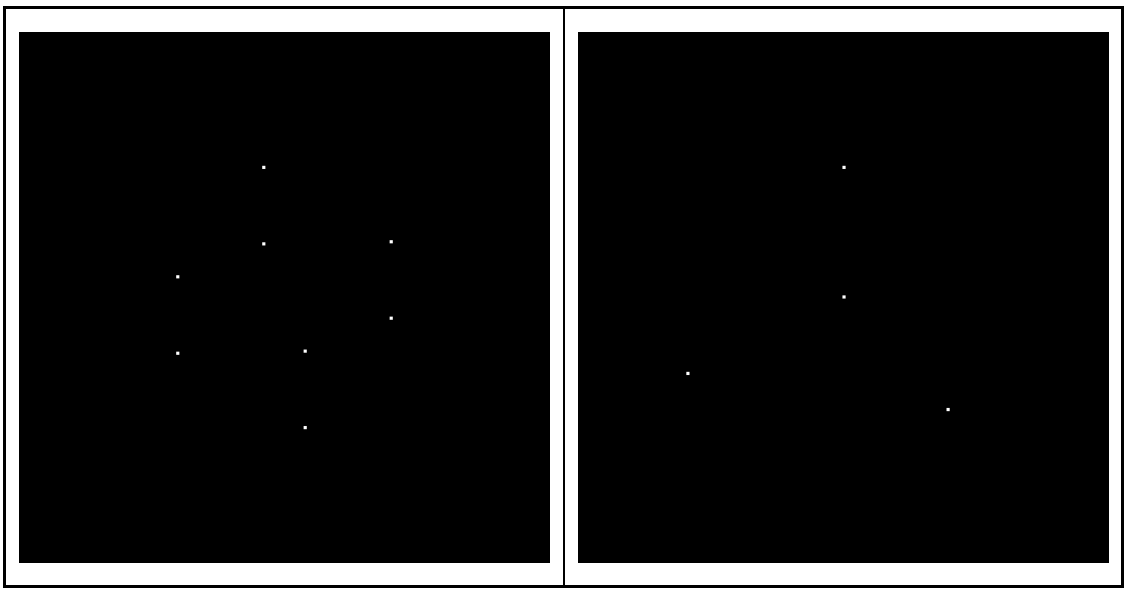
std::string mscreenshot_path = "screenshot.png";	Store the path of screenshot
float mscale_obj_windows = 1;	Scale the object
int msolid_normal = 0;	Variable to switch vertex normal or face normal
float mdeform_t = 0.05, mdeform_a = acos(-1);	Deforming parameters
GLuint mtextureID;	Texture ID
std::vector<vec3> vertex_positions;	Positions of obj data
std::vector<vec2> vertex_textures;	Textures of obj data
std::vector<vec3> vertex_normals;	Normals of obj data
//std::vector<vec3> vertex_colors;	Colors of obj data
std::vector<face> faces;	Faces with face normal, positions, textures, normals

Execution [Obj_file_loader.exe]

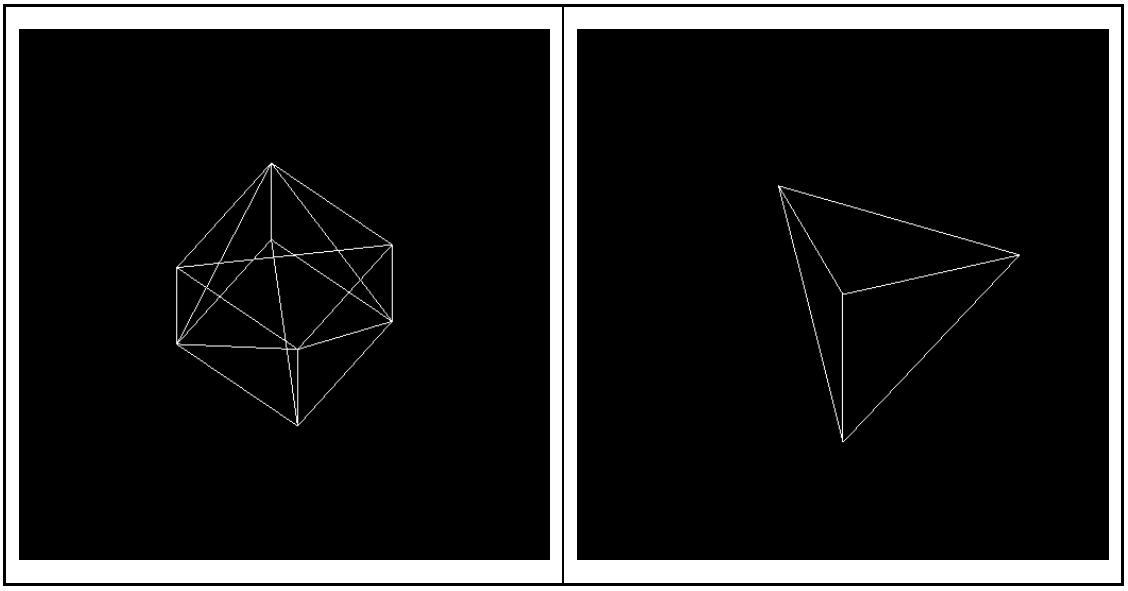
Exercise 03:

(Note: the displaying image followings are obtained by saveImage() in my code. Press P(or p) can conduct the screenshot and the S(or s) can switch the three modes. Moreover, left mouse button can rotate x-axis, right mouse button can rotate y-axis, middle mouse button can rotate y-axis.)

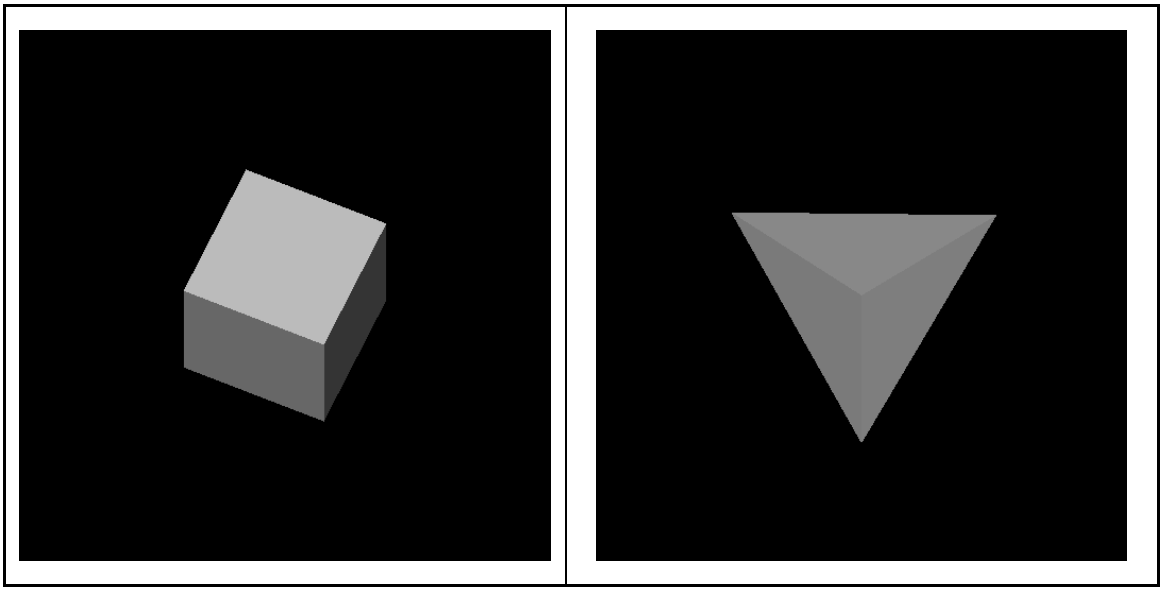
MODE POINTS:



MODE LINES:



MODE SOLID:



Short Questions:

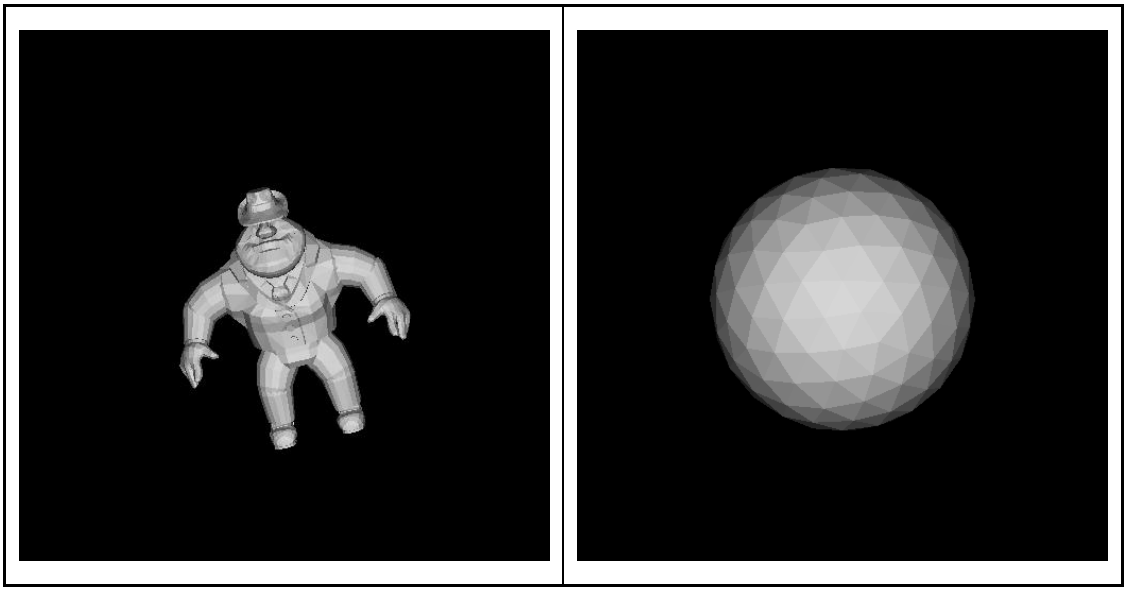
1. If the dimension of the mesh is very large, what happens and what should you do in order to render it and see it on screen?
   ANS: Find out the maximum distance in the obj file, then use it to scale the all the positions.

2. What happens if the camera is placed “incorrectly”? What does it mean “incorrectly”? State your assumptions if any.
   ANS: Actually, I don’t understand what the problem means.

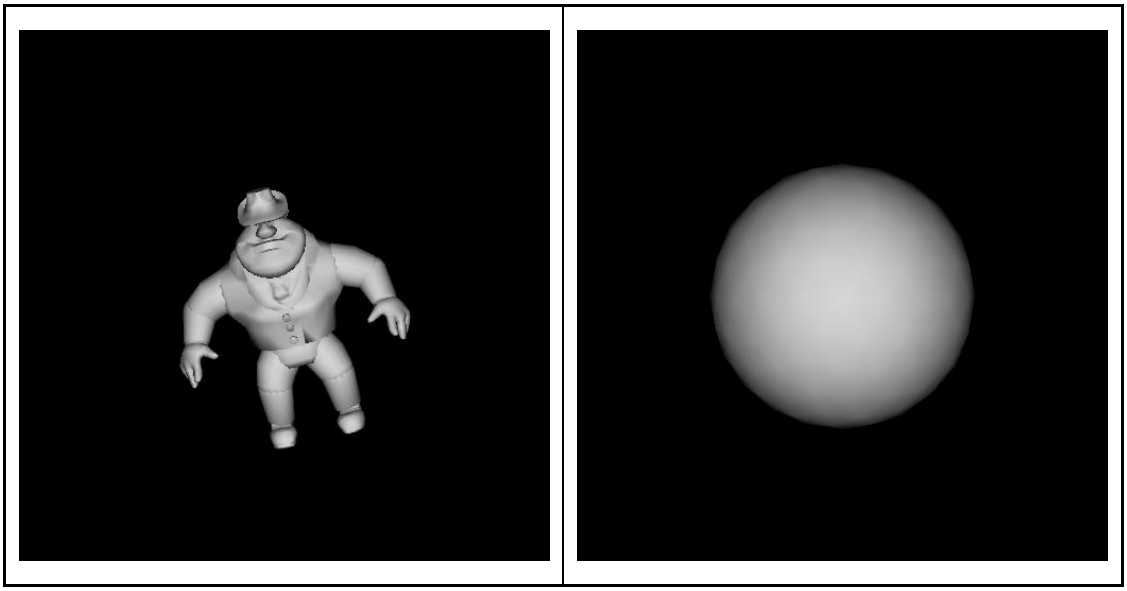
3. What should be done in order to view the mesh at any direction?
   ANS: I use mouse left, middle, right button to rotate the 3D object in x-axis, y-axis, z-axis respectively. So that I can view the whole object.

Exercise 04 and Exercise 05:

FACE NORMALS:



VERTEX NORMALS:



Short Questions:

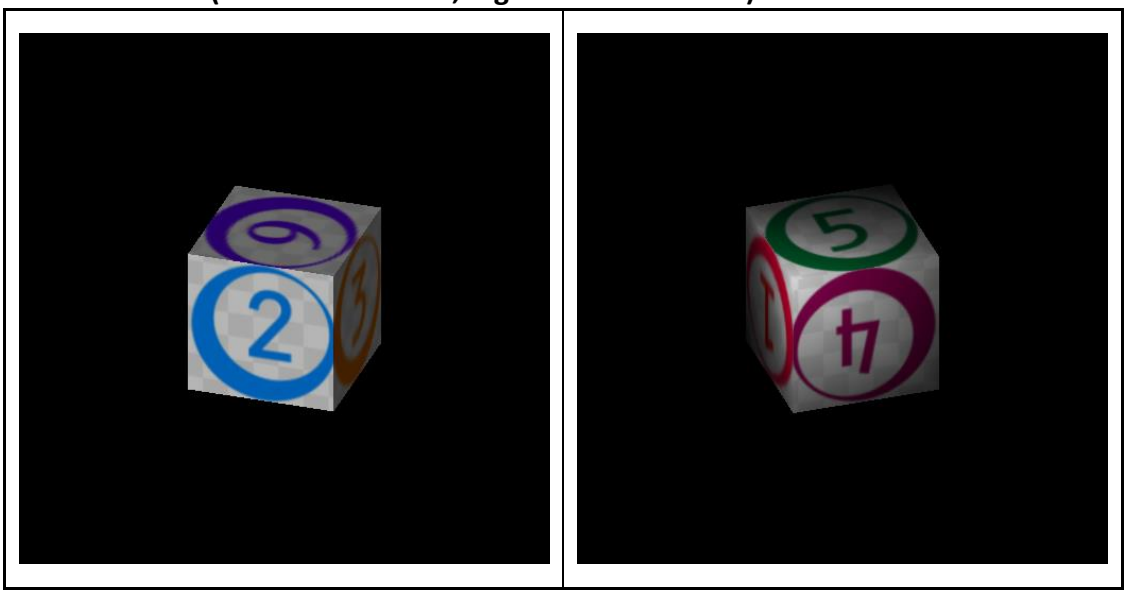
1. What are the advantages of maintaining the faces attached at a vertex?
   ANS: I construct a struct of face to save the positions, vertex normals, face normal, texture coordinates with vector method (like the linked list). Under this method, can maintaining the faces attached at a vertex. So that, it's more quick and convenient to code.

2. What should be done if the position of a vertex changes? Give an efficient method if any.
3. What should be done if the positions of some vertices change? Give an efficient method and state your assumptions.
   ANS: Since all of the vertices attach the faces, so if the position changes, we can through faces to update the normal or the other features.

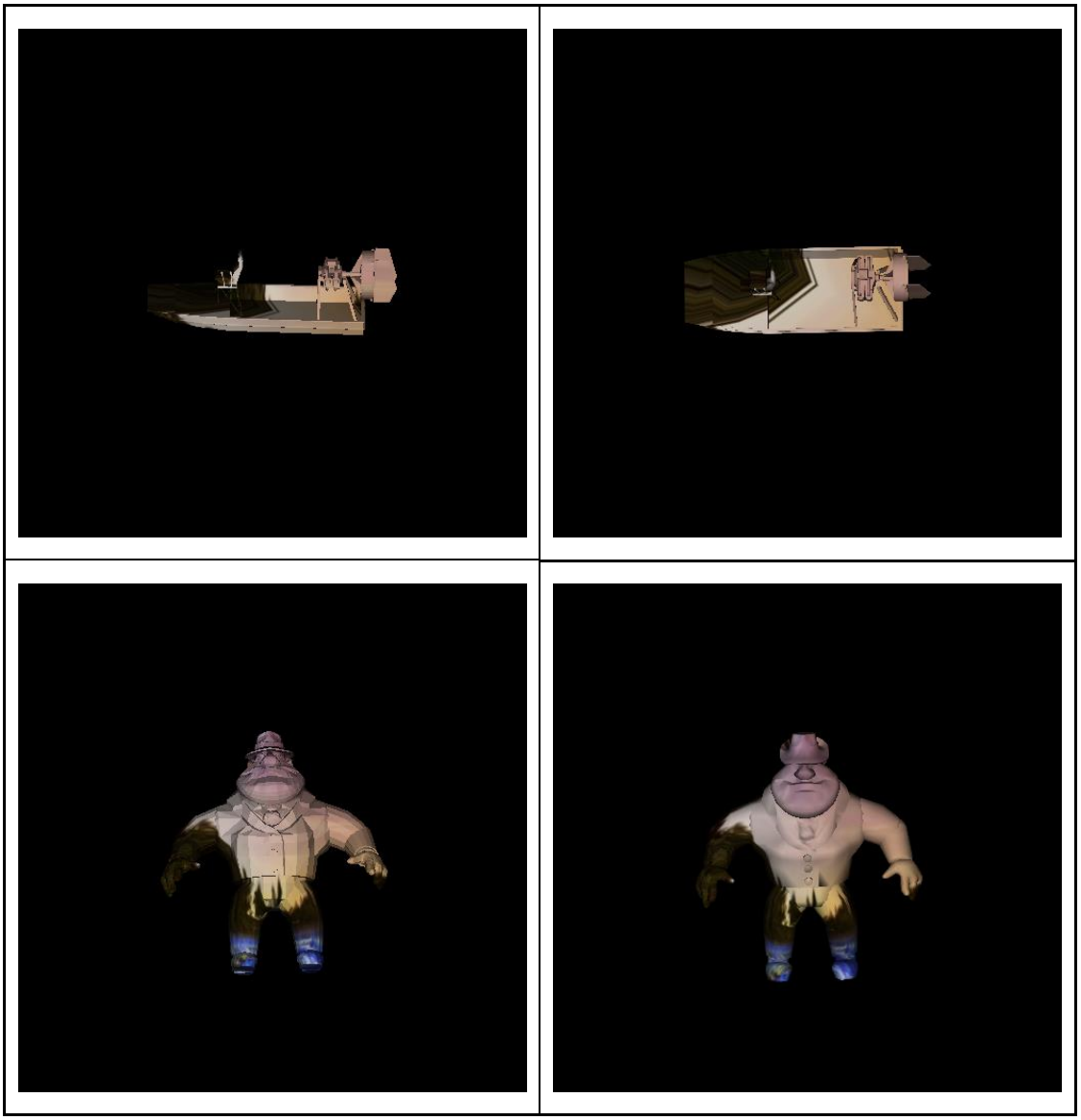
4. What happen if the data type of a vertex changes from float to double or vice versa? What should be done?
   ANS: My code is use float, I will to change the type of them to maintain the same in code.

In my code, I use the UV map the compute the texture for all positions globally and set it as default, if the input obj file contains v/vt or v/vt/vn forms, then the default will be closed and use the coordinates in obj file. The normal can be done with the same method.

TEXTURE IN OBJ FILE (Left: Face Normal, Right: Vertex Normal):



DEFAULT USE GLOBAL UV MAP:



Short Questions:

1. How to do multi-textures?
   ANS: In my code now only can use one texture since I declare GLuint textureID. If using multi-textures, I will change the declaration to vector<GLuint> textureID to save multi-textures, and arrange them in the format in obj file.

2. Do you know anything about Cg programming? Look for information about Cg programming and summarize it.
   ANS: Cg programming is a high level shader language and can enable profound speed and detail increases as well as many special effects in computer graphics, moreover, Cg also keep the style of C language.

References,

[1] http://www.martinreddy.net/gfx/ (obj format)
[2] https://people.sc.fsu.edu/~jburkardt/data/obj/obj.html (obj files)
[3] http://www.opengl-tutorial.org/beginners-tutorials/tutorial-1-opening-a-window/ (opengl Tutorial)
[4] https://filesamples.com/formats/bmp (bmp files)
[5] https://github.com/nothings/stb [stb library]
[6] https://www.opengl.org// [opengl]