COSC 4370: Interactive Computer Graphics

Instructor: Prof. Zhigang Deng

Name: NHAT NGUYEN

UID: 1652765

HW3 REPORT

Assignment problem:

Practice 3D viewing and implementing Phong shader.

Sub-problems:

- Install GLEW, GLFW and GLM
- Read, understand, and figure out how to run the given code and get an output.
- Find out which function to use and how they work.
- Figure out how to get GetViewMatrix() function to work.
- Figure out how to get the projection matrix.
- Figure out how to get phong.vs and phong.frag to work

First thought:

All the dependencies are relatively easy to install since we already have experience with those in HW2.

Very overwhelming amount of code. We have not been explained how the codes work. Instructions given not much to work off on.

Process:

After many googling, I found learnopengl.com, which has all I need to work on this problem.

GetViewMatrix()

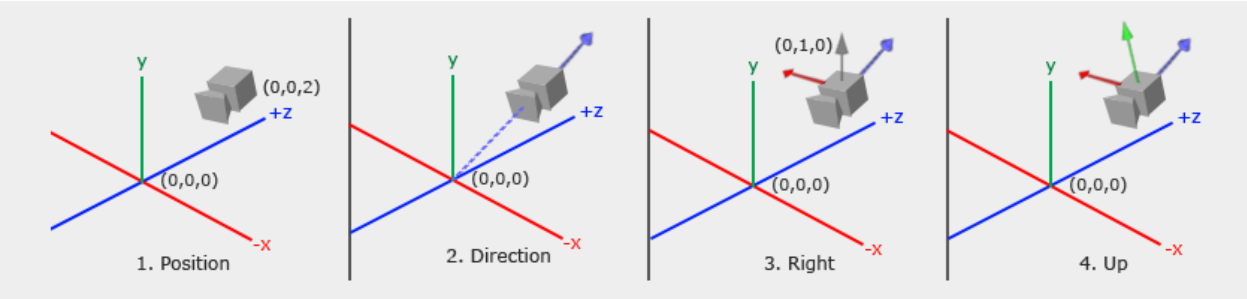
First I tried to figure out how lookAt() function works to get the view matrix. lookAt() takes 3 arguments: the position of the camera, the vector direction of the camera and the vector up in world space. All 3 arguments are given in the starter code.

+ The position of the Camera is the given Position = 0, 0, 3

+ The vector direction indicates where the camera is looking at = Position + Front, Front is a point in front of the camera, ‘ + ’ because the camera is moving in reverse relative with space, so we want the front to be reverse, hence ‘ + ’ and not ‘ - ’ to get the direction vector.

+ Up vector is the vector indicate the up direction of the camera. We can get this by getting the normalizing the cross product of y axis and direction of the camera, to get a right direction of the camera, then do a cross product of that right vector and the direction of the camera.

As demonstrate here:



(Image from https://learnopengl.com/Getting-started/Camera)

Project matrix:

Can be calculated using glm::perspective(view angle, aspect ratio, near clipping plane, far clipping plane) to get a perspective view like the given screenshot suggested.

I can understand how this perspective function works to get the project matrix, but I really don't remember it has ever been mentioned in the lecture.

phong.vs:

gl_Position, vertex position. Calculated using Projection location * camera location * object location * scale (determine the zoom and speed of the camera)

FragPos is the fragment's position, the current segment where light is calculated.

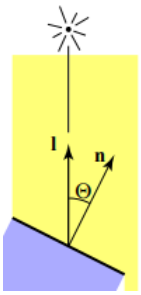
Normal, normal vector of that fragment.

phong.frag:

We must work on 3 major components to get the lighting to be convincing: ambient, diffuse and specular.

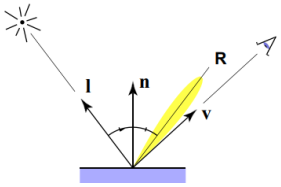
+ Ambient is the simulate reflective light from the environment, calculated using ambientStrength * lightColor

+ Diffuse simulate the directional light on an object. This is what give the viewers the illusional of shadows. Basically, the max dot product of the Normal vector and the light direction vector.



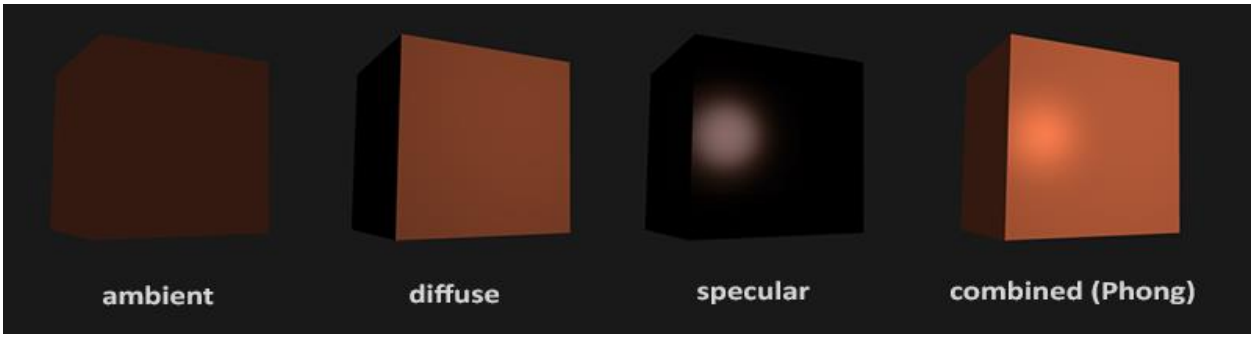
Vector n is the diffuse vector.

+ Specular is the actual reflection of the light object on the object, a glare. Calculate by first find the view vector, then get the reflect direction. Then specify the "spread", or the intensity of the glare. Then we can get the specular by multiplying these variable



R is the specular vector.

Finally, we combine all these components, multiply by the color, and convert to vec4 to generate a 3D model view.



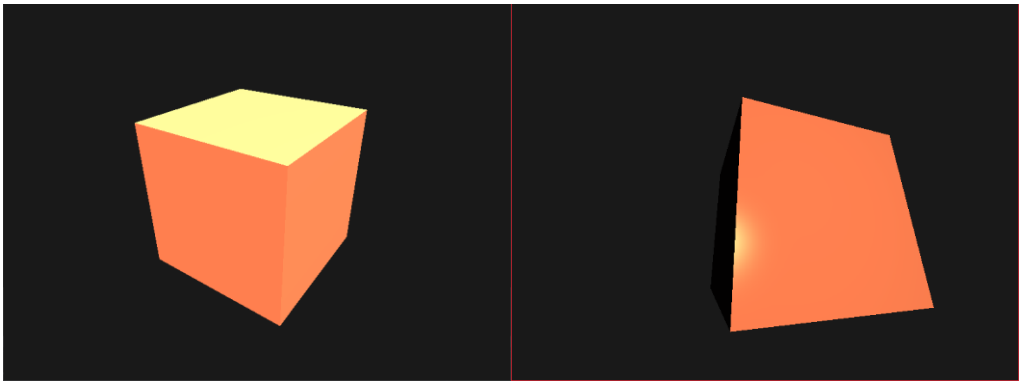
(Image from https://learnopengl.com/Lighting/Basic-Lighting)

Note: The light source must be moved to generate an image similar to the screenshot given. Testing with ambient strength, pov, and many others variable gave some interesting result.

Method using:

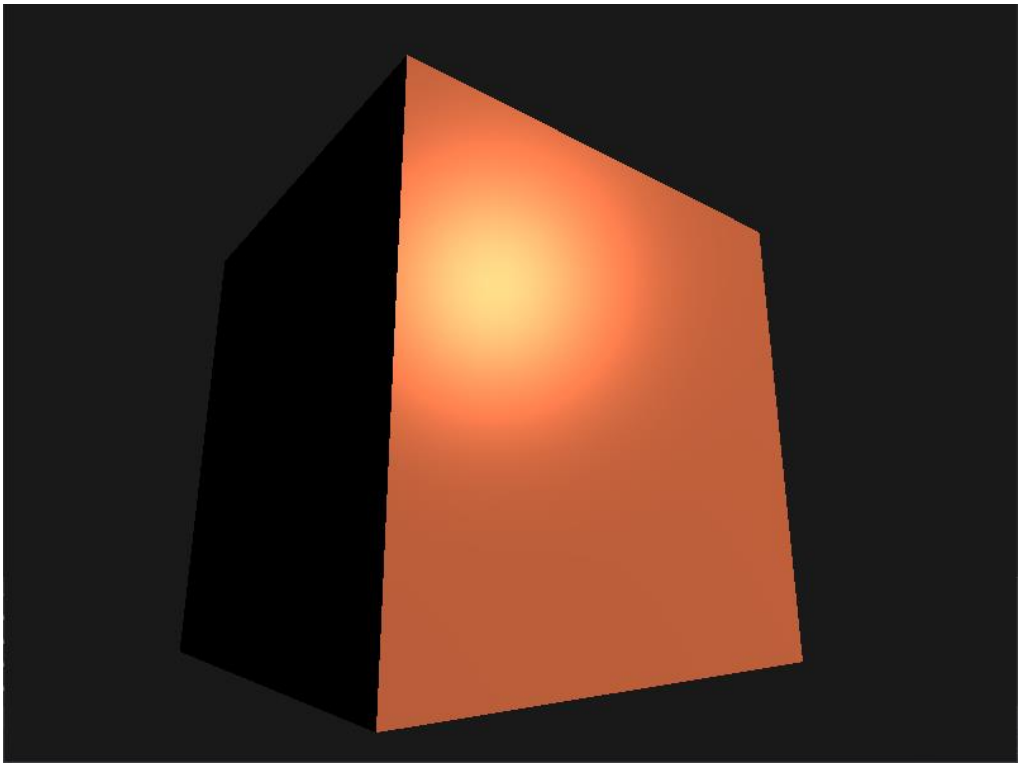
Perspective projection and Blinn-Phong Reflectance Model shading.

Result:



ambientStrength = 1

POV = 90



Conclusion:

This assignment is meant to familiarize us with shading method. Give a deeper insight on how shading and lighting work in openGL. I tried my best to understand how these codes work and I believe that I understand most if not all of them, but in the end I am not very confident that I can create these by myself had I not discovered learnopengl.com.

Note: If I recall correctly, there's one lecture where one of the TA explain how shading works, but that's weeks before this assignment. It would be great if we got these starter codes explained to us before hand and went over how openGL works.