TP00 — OpenGL

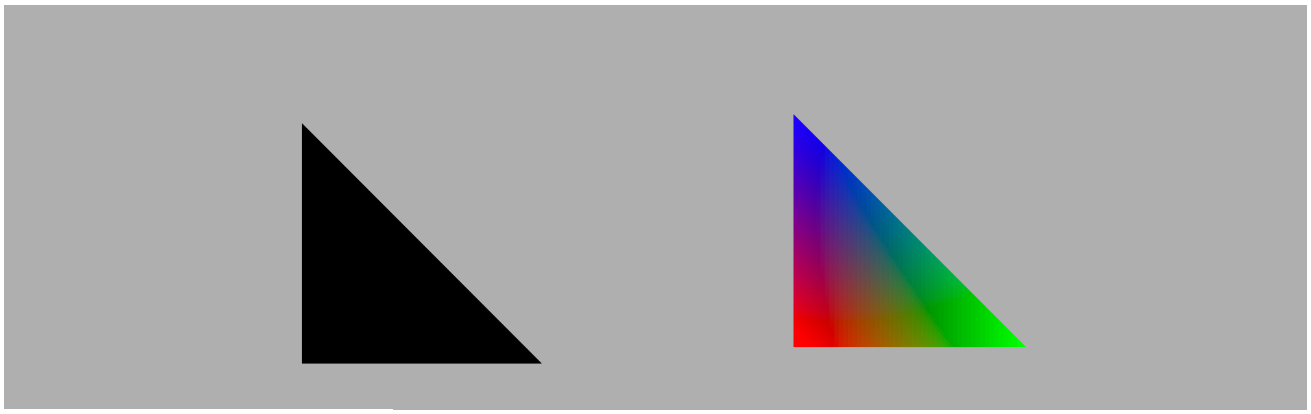
Xiang WEI

1 Codebase

It's important to run under ./src instead of ./build, otherwise it couldn't load the two shader files well and cause no output.

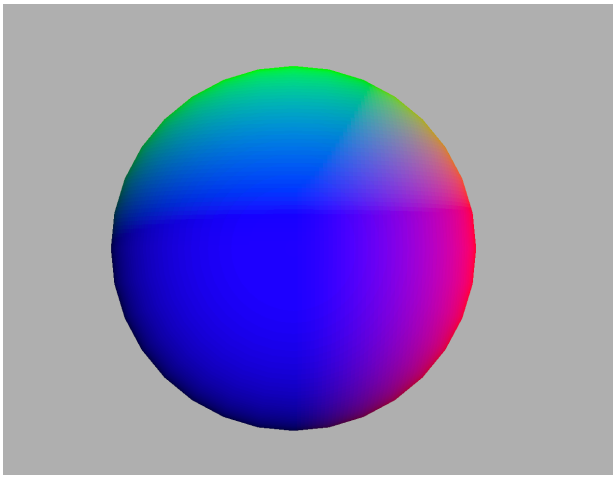
2 A single Triangle

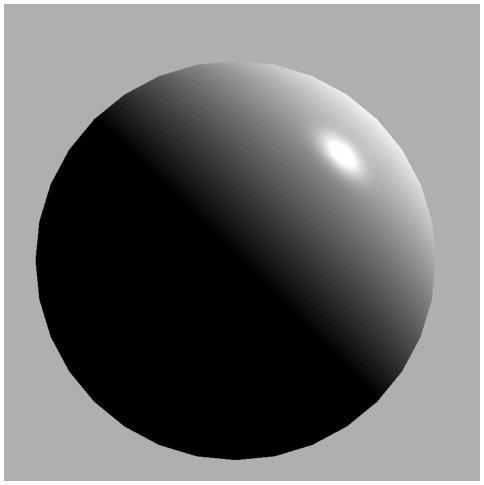
To draw a single triangle, we need to set the vertex position and indices (use push_back for older version) and bind them with GPU. If we want to apply colors, we also need make vertexShader to receive the value.



3 Mesh sphere

To generate a sphere, I based on the formulas converting coordinate system from cartesian coordinates to spherical coordinates and give the same value as the vertex position to vertex normal.



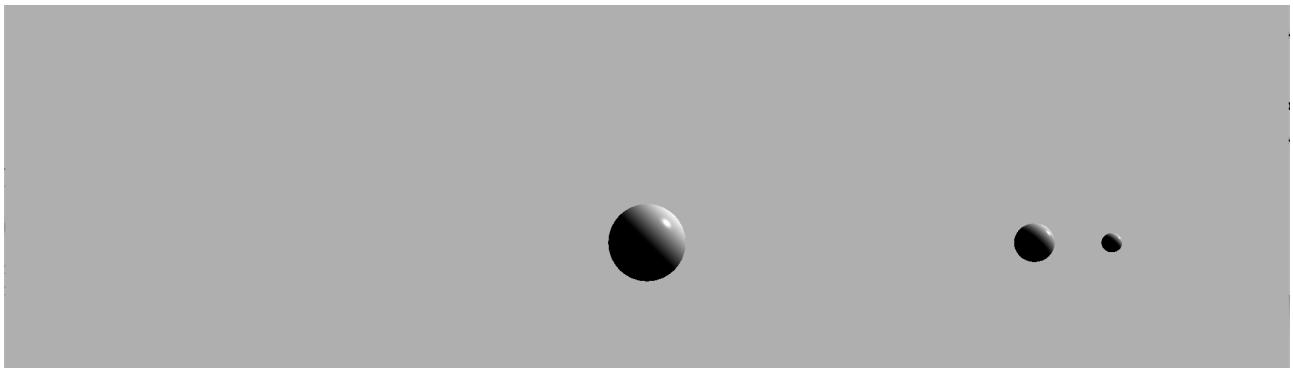


3.1 Shading

To apply Phong lighting model, we need to get the camera position and use it to calculate view vector. Also, we need to calculate the reflection vector, diffuse lighting and specular lighting. (Result shown as picture above)

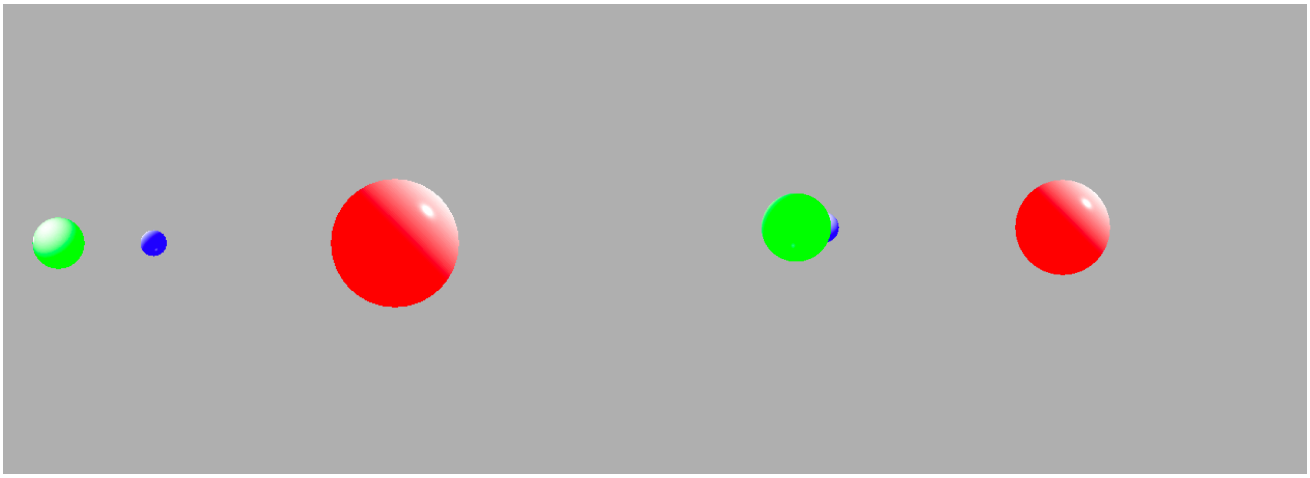
4 Three planets

To generate three sphere, we just need to init and generate one sphere and then render it three times by setting different transform matrix. Pay attention to the *glClear()* function in case it erase your previous drawing.



5 Animation

Put the transform matrix into update function and apply the currentTime variable to the angle and coordinate. Don't call *render()* directly in the update function.



6 Planet Textures

When loading the texture images, pay attention to the path and also to the unbind texture operation. When sending the value to vertexShader, remember that our vertexTexCoords is a *vec2* not *vec3*. And don't forget to multiply the TexCoord with the shading model in our fragmentShader.