CS7GV6 Computer Graphics Assignment 1

Van Allen Bruns Jr - 19329560

Compulsory Elements

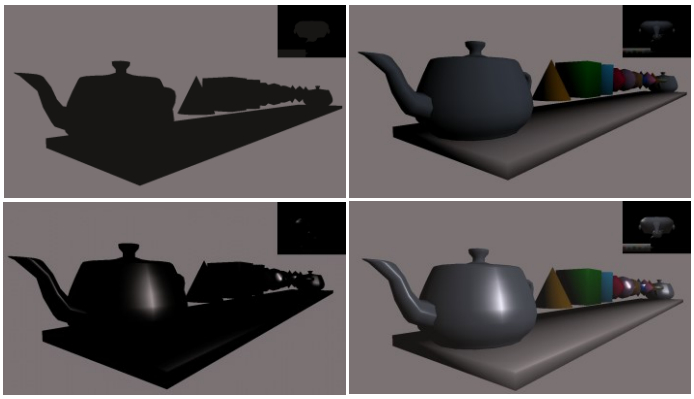
All compulsory elements were implemented in labs prior to this assignment except for lighting and shading, so the focus of this report is covering lighting and shading. Before we begin, here's an outline of the keyboard controls for this assignment:

Viewport Toggle F5: Top Left F6: Top Right F7: Bottom Left F8: Bottom Right	**Object Selection** 0-9: Individual selection Ctrl-a: select all	**Transformation** Ctrl-xyz: Light move toggle Shift-xyz: Translate xyz: Rotate u: Scale +alt: Reverse (except light)
Movement w: Forward (based on eye) a: Left s: Backward (based on eye) d: Right	**Left Arm Rotate (+alt)** f: Forward shoulder g: Side shoulder h: Upper arm j: Elbow k: Forearm	**Right Arm Rotate (+alt)** c: Forward shoulder v: Side shoulder b: Upper arm n: Elbow m: Forearm

Lighting and Shading

Initially, I had trouble with the slides, so I consulted (DISQUS, 2019). Then, I turned to (John Kessenich, 2017) for inspiration. The result looked great, but reflection didn't follow the eye of the viewer. Finally, I combined all of them together to create a highly-customizable shading scheme.

You can see here the ambient, diffuse, specular, and combined results. For the shaders, notice all the uniform variables, meaning we can tweak the light source and object material properties until we get exactly what we want!



```
#version 330

in vec3 vertexPosition;
in vec3 vertexNormal;

out vec3 positionEye;
out vec3 normal;
out vec3 lightDirection;
out float attenuation;

uniform mat4 view;
uniform mat4 projection;
uniform mat4 model;

uniform vec3 lightPosition;
uniform float attenuationConst;
uniform float attenuationLinear;
uniform float attenuationQuadratic;

void main(){
	vec3 normalEye = vec3 (view * model * vec4 (vertexNormal, 0.0));
	vec3 lightEye = vec3 (view * vec4 (lightPosition, 1.0));
	positionEye = vec3 (view * model * vec4 (vertexPosition, 1.0));
	normal = normalize (normalEye);
	lightDirection = lightEye - positionEye;

	float lightDistance = length (lightDirection);

	lightDirection = lightDirection / lightDistance;
	attenuation = 1.0 / (attenuationConst
				+ attenuationLinear * lightDistance
				+ attenuationQuadratic * lightDistance *
lightDistance);

	gl_Position =  projection * vec4 (positionEye, 1.0);
}
```

```
#version 330

in vec3 positionEye;
in vec3 normal;
in vec3 lightDirection;
in float attenuation;

out vec4 fragColour;

uniform vec3 lightDiffuse;
uniform vec3 lightSpecular;
uniform vec3 lightAmbient;

uniform vec3 reflectDiffuse;
uniform vec3 reflectSpecular;
uniform vec3 reflectAmbient;

uniform float spotSize;
uniform float shininess;

void main () {
	// ambient intensity
	vec3 Ia = lightAmbient * reflectAmbient;

	float diffuse = max (dot (lightDirection, normal), 0.0);

	// diffuse intensity
	vec3 Id = lightDiffuse * reflectDiffuse * diffuse;

	vec3 reflectionEye = reflect (-lightDirection, normal);
	float specular = dot (reflectionEye, normalize (-positionEye));
	specular = clamp (specular, 0.0, 1.0);
	specular = pow (specular, spotSize);

	// specular intensity
	vec3 Is = lightSpecular * reflectSpecular * specular * shininess / attenuation;

	// final colour
	fragColour = vec4 (Is + Id + Ia, 1.0);
}
```

References

DISQUS. (2019, October 25). *Basic Lighting*. Retrieved from Learn OpenGL: https://learnopengl.com/Lighting/Basic-Lighting

John Kessenich, G. S. (2017). *OpenGL Programming Guide Ninth Edition.* Crawfordsville, Indiana: Pearson Education, Inc.