컴퓨터애니메이션실습(CAL)

HW3_Phong_Shading

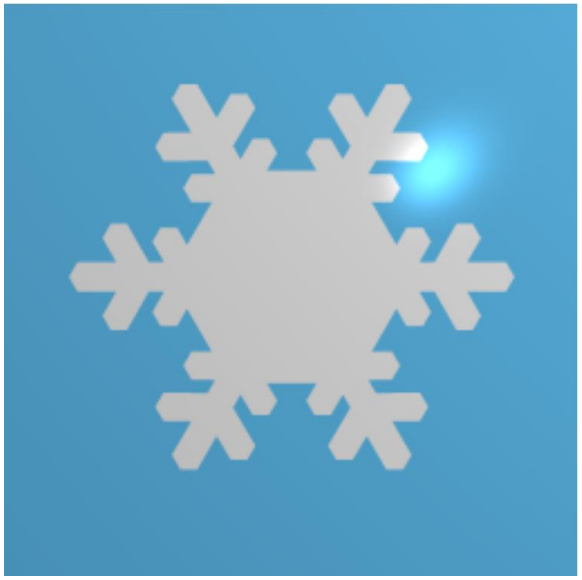
Self-scoring table

	P1	P2	P3	P4	E1	Total
Score	**1**	**1**	**1**	**1**	**1**	**5**

2018707068 김경환

KwangWoon University

Practice 01. Simple texturing:



Simple Texturing의 경우에는 먼저 glTexImage2D() 함수를 매번 호출해야하는 부담을 해결하고, texture 객체의 고유한 이름을 생성하기 위해 glGenTextures()을 사용하여 texId에 texture 객체를 부여한다.

그리고 glActiveTexture()와 glBindTexture()에 의해 texture unit을 활성화하고 texId에 binding을 한다.

또, loadRawTexture()에서 "m02_snow_color_map.raw" file을 read하여 rgb channel의 모든 texel을 읽어 raw에 저장하고, 이를 glTexImage2D()를 통해 texture를 생성한다.

그 후에 Rendering은 render함수의 if (simpleTexturing && !normalMapping)에 의해 실행되고 내용은 먼저 setUniformMVP()에 의해 Model, View, Projection matrices를 shader에 set uniform한다. 그리고 tex와 LightPosition, Ka, Kd, Ks, Shininess를 shader에 set uniform하고, glUseProgram(), drawVBO()에 program와 vertex buffer object를 지정하여 mesh를 그려낸다.

여기서 사용된 vertex buffer object의 경우에는 사전에 quad와 tri로 그릴 경우를 대비하여 저장해뒀다.

Practice 02. Alpha texturing:

Alpha texturing의 경우에는 Simple texturing과는 대부분은 유사하지만 loadAlphaTexture()에서 raw에 texel 정보를 저장할 때 RGBA로 한 texel당 4개를 사용한다. 그리고 RGBA 중 A에 Alpha Blending할

data를 저장하고, texture 상태 설정으로 GL_CLAMP_EDGE와 GL_NEAREST를 사용한다.

Rendering은 render()의 if (alphaTexturing)에서 실행되며 Simple texturing과 다른 점은 glEnable(GL_BLEND)와 glBlendFunc(GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA)를 통해서 Alpha Blending을 가능하게 하고 식을 설정한다.

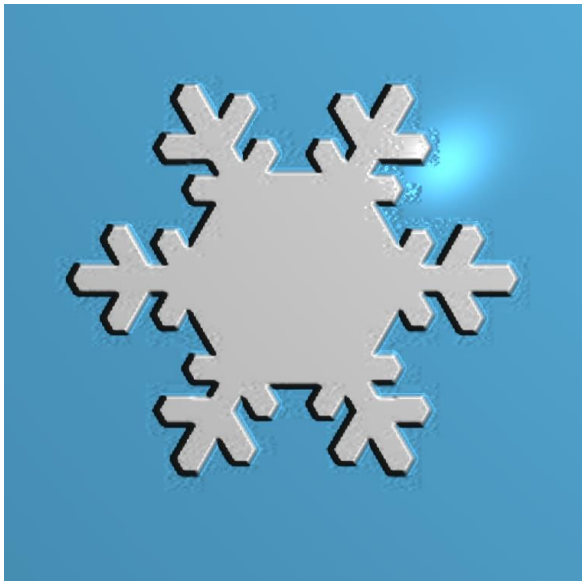
Practice 03. Double vision:

Double vision의 경우에는 위 두 개의 예제와 거의 동일하나 texture 정보가 저장되어 있는 file이 .raw file이 아니고 header file이며 vertex, fragment shader가 기존과는 다르다.

그리하여 demonHeadImage의 경우에는 #include "m02_demon_image.h"를 통해서 texture 정보를 저장하고 texture를 생성하는 것은 이전과 동일하다.

Vertex shader는 leftSeparation과 rightSeparation이 존재하여 기존 VertexTexcoord에서 Separation만큼 변화를 줘서 framgent shader에 전달하고 frament shader에서는 이를 통해 color를 얻어내 mix하여 결과를 반환한다.

Practice 04. Normal mapping



Normal mapping의 경우도 practice 1, 2와 거의 유사하나 texture를 diffuse와 normal 두 개를 사용하는 것과 fragment shader에서 phong reflection에 사용될 normal vector를 계산하는 과정에서 차이가 있다. 이러한 fragment shader는 normal mapping의 Height scale을 위한 uniform으로 scale이 있고, normal map이 저장되어 있는 texcoord에서 rgb값을 얻어 decompression을 수행한 후 x, y에 저장된 값을 scaling한다. 여기서 x, y에 저장된 값은 height fields로부터 얻어진 noraml map의 x, y 값으로 이는 H_g, H_r, H_a와 연관이 있으므로 height가 더 컸던 것 같은 효과를 준다.
그리고 나머지는 다른 fragment shader와 동일하다.

Exercise 01. Draw a color-, alpha-, normal-mapped snow:

color-, alpha-, normal-mapped snow은 color 정보가 저장된 "m02_snow_color_map.raw" file과 alpha 정보가 저장된 "m02_snow_alpha_map.raw" file을 읽어 color 정보는 RGBA중 RGB에 alpha 정보는 RGBA 중 A에 저장한다. 그리고 normal mapping의 효과를 주기 위해 rendering 하는 함수에서 RGBA가 저장된 texDIffuse와 normal map 정보가 저장된 texNormal을 set uniform하고, frament shader에서 scale과 normal map이 저장된 texcoord를 통해 phong reflection의 normal vector를 변경하여 해당 효과를 얻어낸다.