2023학년도 1학기

컴퓨터애니메이션실습

컴퓨터 애니메이션

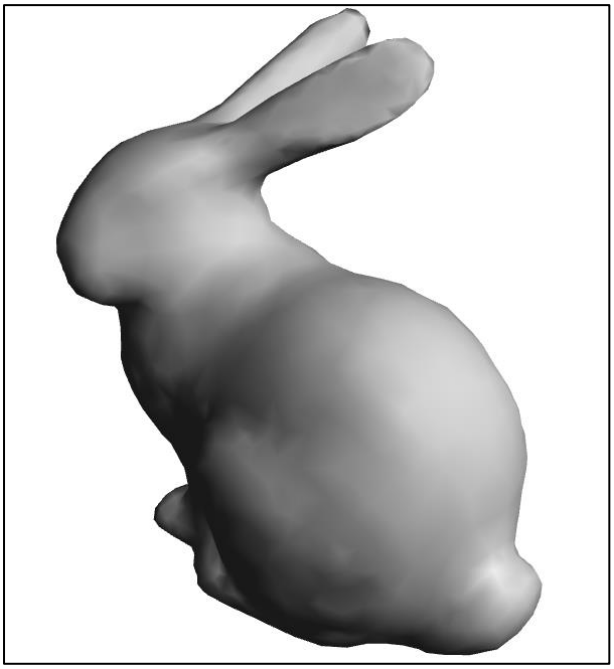
실습 보고서

Self-Scoring Table

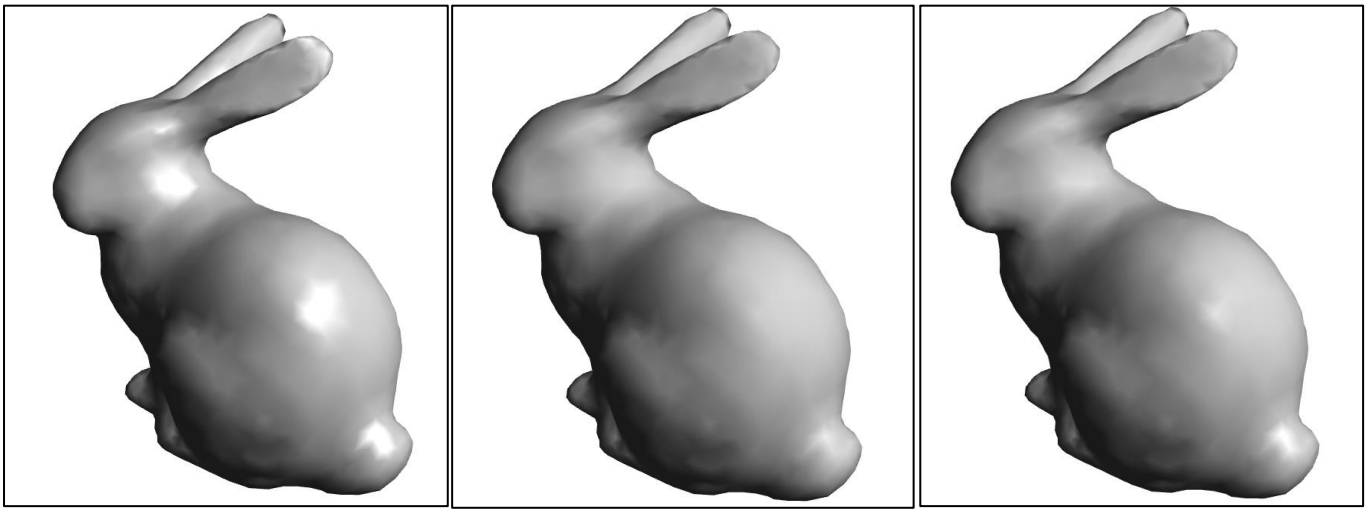
	P1	P2	E1	Total
Score	1	1	1	3

정보융합학부 2018204058 김민교

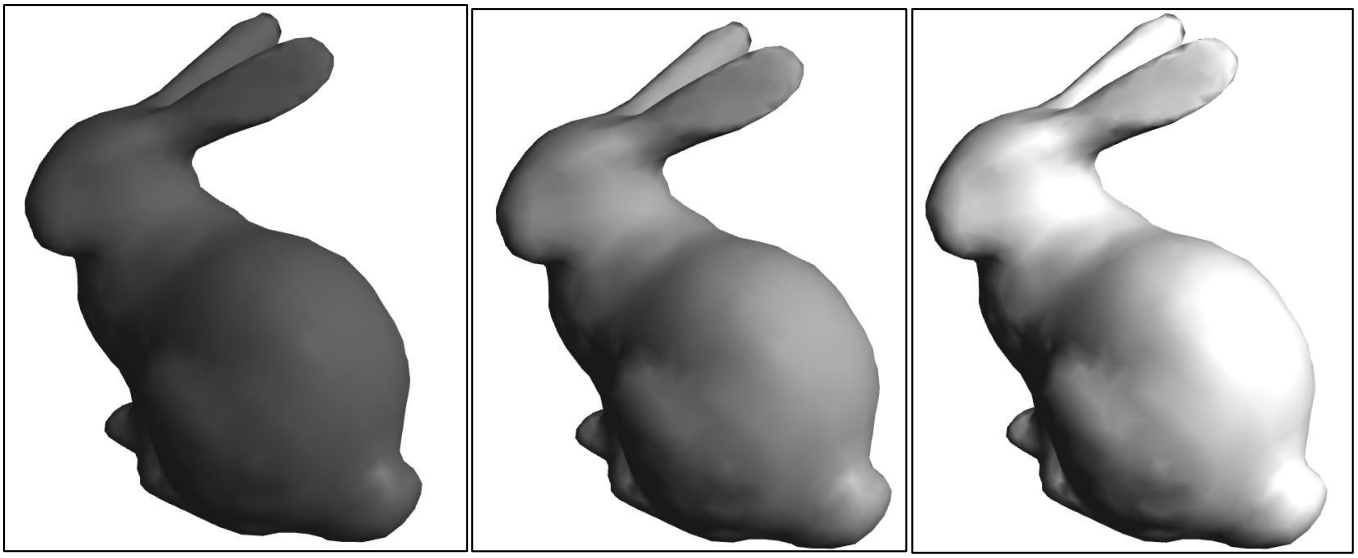
P1 – Gouraud shading with varying material properties



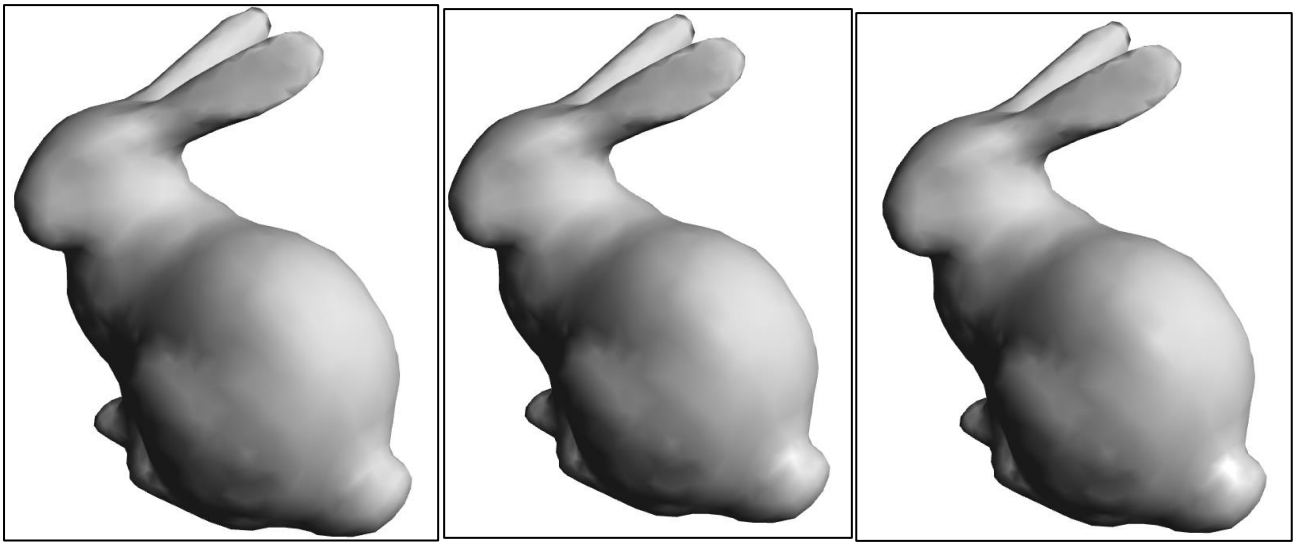
Gouraud shading을 적용한 결과이다.



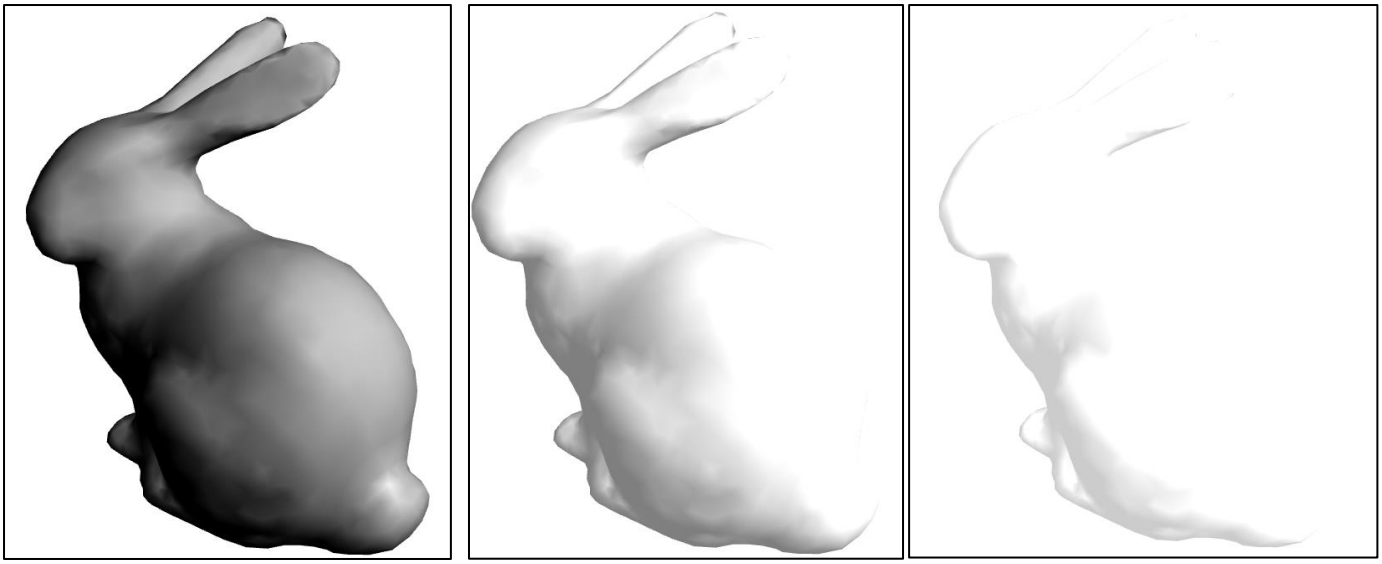
순서대로, Shininess 15, 128, 64 값을 준 결과이다. Shininess값이 작아질수록 하이라이트가 퍼진다.



diffuse 계수를 (0.3, 0.3, 0.3) , (0.6, 0.6, 0.6) , (1, 1, 1) 값을 준 결과이다. 값이 낮을수록 물체가 빛을 난반사하는 정도가 낮아 어두운 것을 확인하였다.

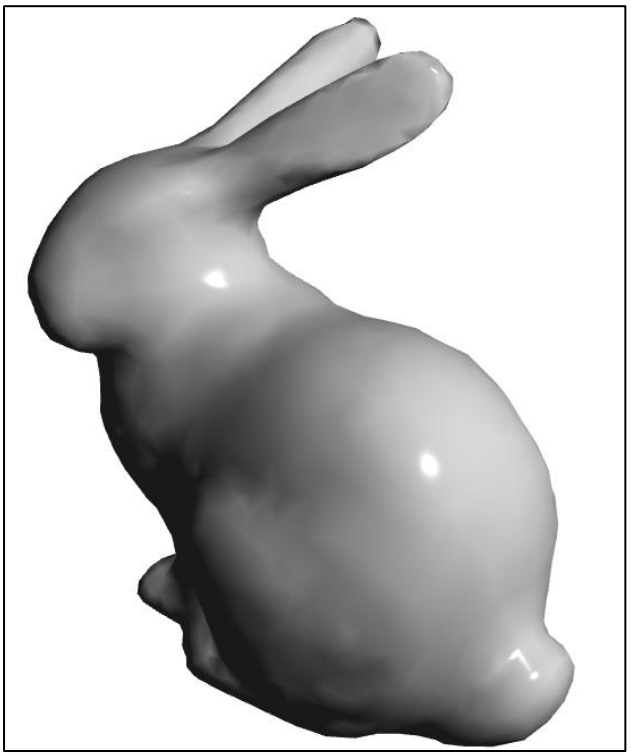


specular 계수를 (0,0,0) , (0.5, 0.5, 0.5) (1, 1, 1) 값을 준 결과이다. 값이 낮을수록 물질이 광택정도가 낮아 하이라이트가 없어지는 것을 확인하였다.

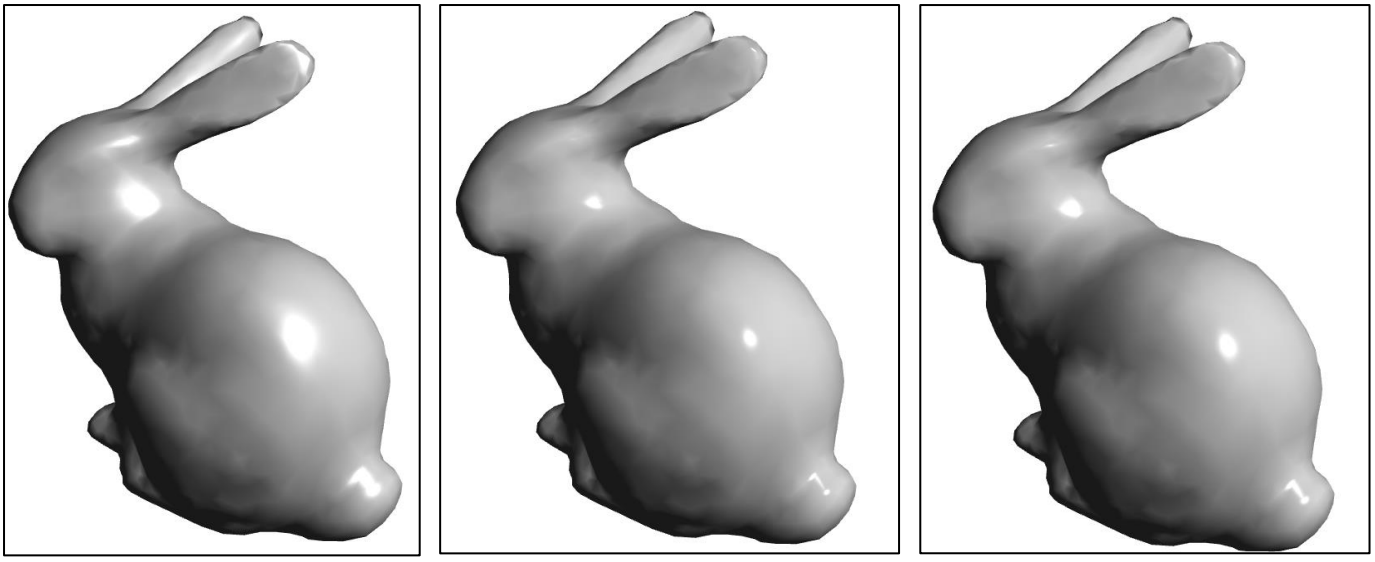


ambient 계수를 (0,0,0) (0.5, 0.5, 0.5) (0.8, 0.8, 0.8) 을 준 결과이다. ambient는 주변광으로써, 값이 높아질수록 빛이 너무 밝아 토끼의 형태가 거의 안보이는 것을 확인하였다.

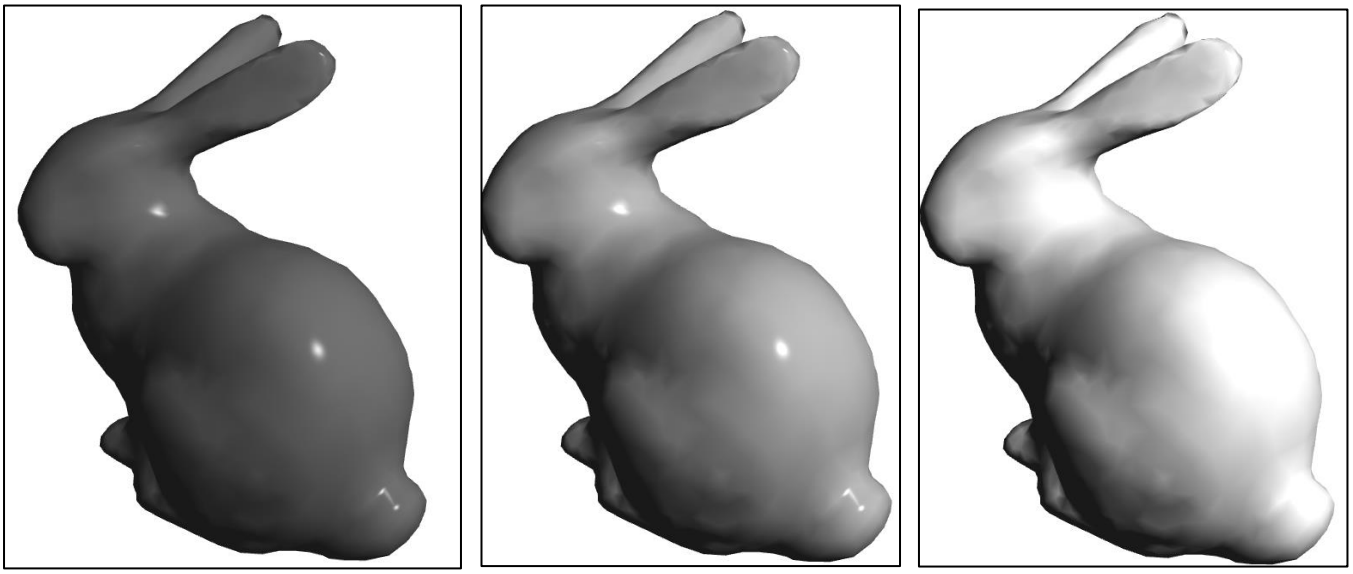
P2 – Phong shading with varying material properties



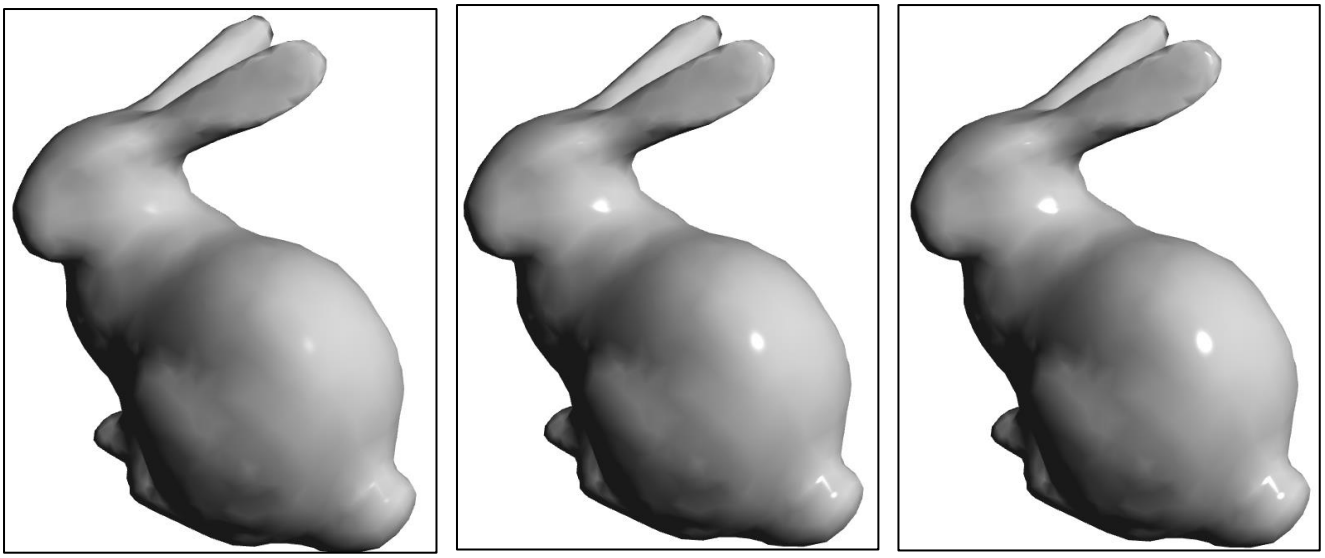
Phong shading을 적용한 결과이다. Gouraud shading을 한 것보다 더 사실적이고 하이라이트가 잘 표현되었다.



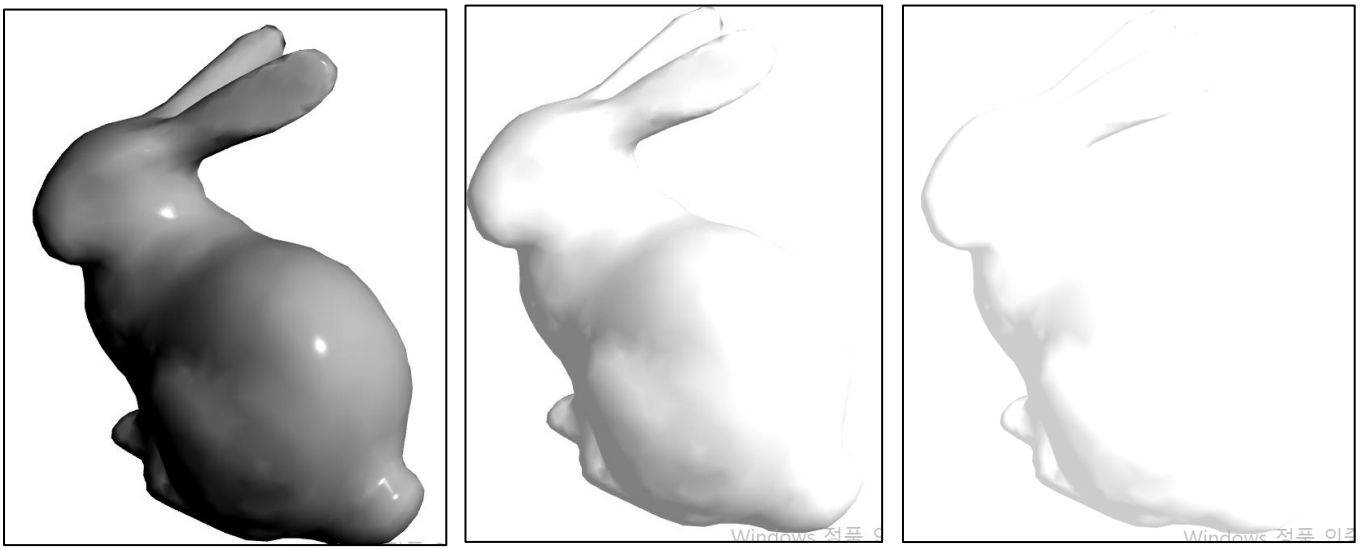
순서대로, Shininess 15, 128, 64 값을 준 결과이다. Shininess값이 작아질수록 하이라이트가 퍼진다.



diffuse 계수를 (0.35, 0.35, 0.35) , (0.65, 0.65, 0.65) , (1, 1, 1) 값을 준 결과이다. 값이 낮을수록 물체가 빛을 난반사하는 정도가 낮아 어두운 것을 확인하였다.

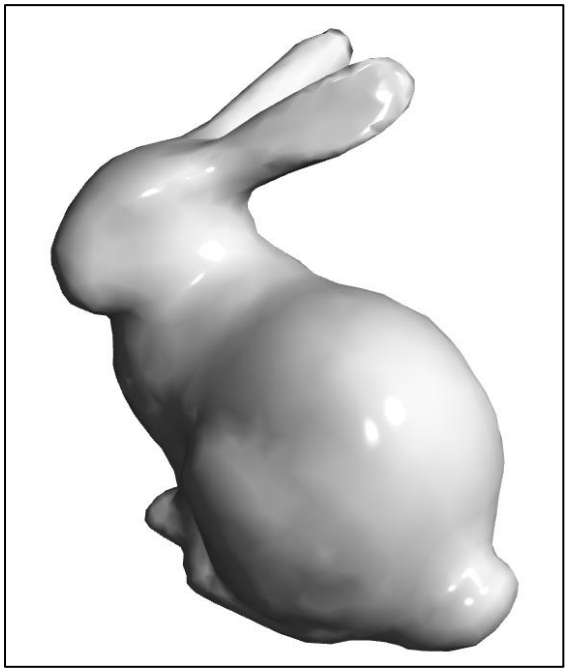


specular 계수를 (0,0,0) , (0.5, 0.5, 0.5) (1, 1, 1) 값을 준 결과이다. 값이 낮을수록 물질이 광택정도가 낮아 하이라이트가 없어지는 것을 확인하였다.

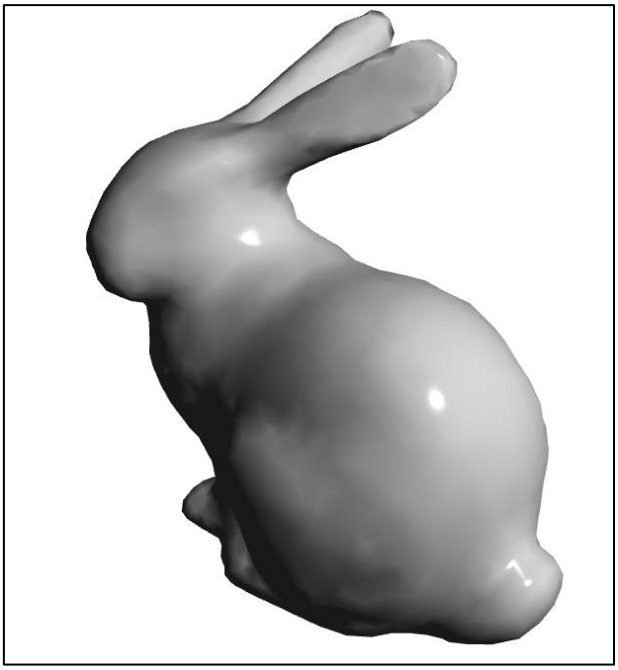


ambient 계수를 (0,0,0) (0.5, 0.5, 0.5) (0.8, 0.8, 0.8) 을 준 결과이다. ambient는 주변광으로써, 값이 높아질수록 빛이 너무 밝아 토끼의 형태가 거의 안보이는 것을 확인하였다.

E1 – Place two lights and then show two highlights using Phong shading



< Multiple Light>



< Single Light>

Exercise 1의 결과이다.

- 코드

빛의 위치 :

```
28      // Light configuration
29      Vector3f    light0(5.0, 5.0, 0.0);
30      Vector3f    light1(5.0,5.0, 5.0); //Light 하나 더 추가
```

Light0은 (5, 5, 0)에, Light1은 (5,5,5)에 위치시켰다.

```
230     // Light0 position in the fragment shader, represented in the view coordinate system
231     Vector3f    l0 = ViewMatrix.block<3, 3>(0, 0) * light0 + ViewMatrix.block<3, 1>(0, 3);
232     setUniform(program, "LightPosition0", l0);
233
234     // Light1 position in the fragment shader, represented in the view coordinate system
235     Vector3f    l1 = ViewMatrix.block<3, 3>(0, 0) * light1 + ViewMatrix.block<3, 1>(0, 3);
236     setUniform(program, "LightPosition1", l1);
```

setUniform 함수를 통해서 LightPosition0, LightPosition 변수의 위치를 찾는다. 그리고 전에 계산해 놓은 각 빛의 카메라 좌표계에서의 위치를 fragment shader에 보낸다.

```
5   uniform vec3    LightPosition0;             // In the view coordinate system
6   uniform vec3    LightPosition1;             // In the view coordinate system
```

이렇게 보낸 값은 fragment shader 프로그램의 변수에 저장될 것이다.

```
77  void
78  main(void)
79  {
80      // Lighting
81
82      vec3 color0 = phongReflectionModel(L0, position, normal);
83      vec3 color1 = phongReflectionModel(L1, position, normal);
84
85      outColor = color0+color1;
86
87  }
88
```

fragment shader 프로그램의 메인 함수이다. L0과 L1은 각 Light0과 Light1의 정보를 담고있다. phongReflectionModel 함수를 통해 각각의 빛에 따른 픽셀들의 색깔을 결정한다. 이 색깔들을 더하여 최종 픽셀의 색깔을 결정한다.