リスト7.9 noise3D1C.frag

```
varying vec3 P;
varying vec3 N;
varying vec3 objPos;//オブジェクト座標
uniform float thUpper;//上方しきい値
uniform float thLower;//下方しきい値
uniform float amp;   //ノイズ振幅
uniform sampler3D smpl3D;
vec4 col0 = vec4(0.7, 0.9, 0.1, 1.0);
vec4 col1 = vec4(1.0, 0.6, 0.0, 1.0);
vec4 col2 = vec4(0.8, 0.3, 0.1, 1.0);
//年輪の中心
float x0 = 1.0;
float y0 = 0.0;
//波長
float lambda0 = 0.2;

void main(void)
{
  vec3 L = normalize(gl_LightSource[0].position.xyz - P);
  N = normalize(N);

  vec4 ambient = gl_FrontLightProduct[0].ambient;
  float dotNL = dot(N, L);
  vec4 diffuse = gl_FrontLightProduct[0].diffuse * max(0.0, dotNL);
  vec3 V = normalize(-P);
  vec3 H = normalize(L + V);
  float powNH = pow(max(dot(N, H), 0.0), gl_FrontMaterial.shininess);
  if(dotNL <= 0.0) powNH = 0.0;
  vec4 specular = gl_FrontLightProduct[0].specular * powNH;
  //統合
  float intensity0 = texture3D(smpl3D,gl_TexCoord[0].stp).r;
  float d = sqrt((objPos.x-x0)*(objPos.x-x0) +(objPos.y-y0)*(objPos.y-y0));
  float PI = 3.141592;
  float lambda = lambda0 * (1.0 - 0.02 * objPos.z);
  float intensity = (1.0 + sin( (2.0*PI* d + amp * intensity0)/lambda) ) / 2.0;
  if(intensity >= thUpper)
    gl_FragColor = (ambient, diffuse) * col0 * intensity + specular;
  else if( intensity >= thLower)
    gl_FragColor = (ambient, diffuse) * col1 * intensity + specular;
  else
    gl_FragColor = (ambient, diffuse) * col2 * intensity + specular;
}
```