リスト 9.3 | anisotropic.frag

```
varying vec3 P;
varying vec3 N;
varying vec3 T;
uniform float roughX;//X方向粗さ係数
uniform float roughY;//Y方向粗さ係数

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

  vec4 ambient = gl_FrontLightProduct[0].ambient;
  float dotNL = dot(N, L);
  vec4 diffuse = gl_FrontLightProduct[0].diffuse * max(0.0, dotNL);
  vec4 specular = vec4(0.0);
  if(dotNL > 0.0)
  {
    vec3 V = normalize(-P);
    vec3 H = normalize(L + V);
    float dotNH = max(dot(N, H), 0.0);
    float d = dotNH * dotNH;
    float b = d * d;
    vec3 Q = normalize(H - dotNH * N);
    float dotTQ2 = dot(T, Q) * dot(T, Q);
    float c = ((d - 1.0) / d) * (dotTQ2 / (roughX*roughX) + (1.0 - dotTQ2) / (roughY
* roughY));
    float dotNV = dot(N, V);
    //分布関数
    float D = min(1.0, exp(c) /(4.0 * 3.14 * roughX * roughY * b));
    //フレネル係数(垂直反射係数)
    vec3 F0 = gl_FrontLightProduct[0].specular.rgb;
    //Schlickの近似式
    vec3 F = F0 + (vec3(1.0) - F0) * pow(1.0 - dot(L, H), 5.0);
    //幾何減衰係数
    float dotVH = dot(V, H);
    float Gout = 2.0 * dotNH * dotNV / dotVH;
    float Gin = 2.0 * dotNH * dotNL / dotVH;
    float G = min(1.0, min(Gout, Gin));
    specular.rgb = D * F * G / dotNV;
  }
  //統合
  gl_FragColor = ambient + diffuse + specular;
  if(gl_FragColor.r > 1.0) gl_FragColor.rgb /= gl_FragColor.r;
}
```