Laborator 8

1 Obiective

Obiectivul acestui laborator este de a descrie succint câteva tehnici avansate de iluminare în OpenGL 4.

2 Modelul de iluminare Blinn-Phong

Modelul de iluminare Blinn-Phong este o versiune ușor modificată a modelului Phong și a fost dezvoltat de către Jim Blinn. Scopul său este de a îmbunătăți reflexiile speculare în anumite condiții, cum ar fi un coeficient foarte scăzut de strălucire (rezultând astfel o zonă specular mare). Reflexiile speculare ale modelului Phong tind să fie întrerupte imediat ce unghiul dintre direcția de vizionare și reflexia luminii crește peste 90.0 grade (cosinusul devine negativ și astfel este fixat la 0.0).

Soluția propusă de Blinn este de a folosi un așa-zis **semi-vector** în locul vectorului de reflexie pentru a calcula componenta speculară. Semi vectorul **(H)** este un vector unitar la jumătatea distanței dintre direcția de vizionare și direcția luminii (Figura 1).

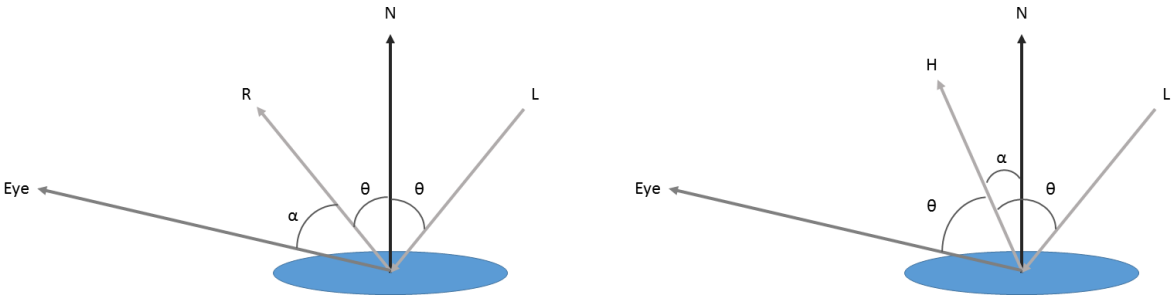

Figura 1 - Modelul specular Phong (stânga) versus modelul specular Blinn (dreapta)

Semi-vectorul este calculat ca:

$$\boldsymbol{H} = \frac{\boldsymbol{L} + \boldsymbol{Eye}}{\|\boldsymbol{L} + \boldsymbol{Eye}\|}$$

Acest lucru se traduce în codul GLSL după cum urmează:

```
//normalize light direction
vec3 lightDirN = normalize(lightDir);

//compute view direction
vec3 viewDirN = normalize(viewPosEye - fragPosEye.xyz);

//compute half vector
vec3 halfVector = normalize(lightDirN + viewDirN);
```

Componenta specular este apoi calculată ca fiind cosinusul unghiului dintre normală și semi-vector:

```
//compute specular light
float specCoeff = pow(max(dot(normalEye, halfVector), 0.0f), shininess);
specular = specularStrength * specCoeff * lightColor;
```

Modelul Blinn-Phong îmbunătățește performanța, evitând calculul costisitor al vectorului de reflexie.

3 Lumini punctiforme

Laboratorul 8 a introdus conceptul de **lumini direcționale**, care sunt surse de lumină infinit de îndepărtate. Toate razele de lumină care provin dintr-o sursă de lumină direcțională sunt paralele și astfel direcția luminii implicate în calculul iluminării este constantă între obiecte și toate fragmentele acestora. Astfel, aceste surse de lumină sunt ideale pentru iluminarea globală din cadrul scenelor noastre.

Luminile punctiforme sunt surse de lumină cu o poziție dată care iluminează radial și uniform în toate direcțiile. Spre deosebire de luminile direcționale, razele generate de luminile punctiforme se estompează în funcție de distanță, făcând astfel obiectele mai apropiate de sursă să pară mai iluminate decât obiectele mai îndepartate.

Atunci când se utilizează luminile punctiforme, direcția luminii nu este constantă între fragmente și trebuie calculată pentru fiecare fragment diferit:

```
//compute light direction
vec3 lightDirN = normalize(lightPosEye - fragPosEye.xyz);
```

Pentru a reduce intensitatea luminii punctiforme cu distanța, trebuie să aplicăm un coeficient de atenuare. Atenuarea poate fi calculată ca o funcție liniară sau patratică dependentă de distanță. Atenuarea liniară necesită un calcul mai puțin complex, dar oferă mai puțin realism, deoarece, în lumea reală, sursele de lumină sunt strălucitoare atunci când stau aproape de obiecte, dar își diminuează rapid strălucirea, cu cât ne aflăm mai departe de ele. Realismul este considerabil crescut atunci când se încorporează atenuarea patratică în funcție de distanță. Astfel, atenuarea poate fi calculată folosind:

$$Att = \frac{1.0}{K_c + K_l * d + K_q * d^2}$$

unde K_c este un termen constant, de obicei menținut la 1.0, K_l este un termen liniar, K_q este un termen pătratic și d este distanța.

Alegerea valorilor potrivite pentru aceste constante depinde de mai mulți factori cum ar fi mediul, distanța necesară pentru călătoria luminii, tipul de lumină etc. Puteți folosi Tabelul 1 ca ghid pentru specificarea coeficienților de atenuare pe baza distanței pe care lumina trebuie să o parcurgă.

Pentru a implementa atenuarea, ar trebui să definim coeficienții de atenuare în shader sau să îi trimitem din aplicație prin variabile **uniform**:

```
float constant = 1.0f;
```

```
float linear = 0.0045f;
float quadratic = 0.0075f;
```

Distance	Constant	Linear	Quadratic
7	1.0	0.7	1.8
13	1.0	0.35	0.44
20	1.0	0.22	0.20
32	1.0	0.14	0.07
50	1.0	0.09	0.032
65	1.0	0.07	0.017
100	1.0	0.045	0.0075
160	1.0	0.027	0.0028
200	1.0	0.022	0.0019
325	1.0	0.014	0.0007
600	1.0	0.007	0.0002
3250	1.0	0.0014	0.000007

Tabel 1 – Valori comune pentru coeficienții de atenuare (sursa: Ogre3D wiki)

Ar trebui să calculam atenuarea folosind ecuația de mai sus:

```
//compute distance to light
float dist = length(lightPosEye - fragPosEye.xyz);
//compute attenuation
float att = 1.0f / (constant + linear * dist + quadratic * (dist * dist));
```

În cele din urmă, ar trebui să aplicăm atenuarea componentelor ambientale, difuze și speculare:

```
//compute ambient light
ambient = att * ambientStrength * lightColor;

//compute diffuse light
diffuse = att * max(dot(normalEye, lightDirN), 0.0f) * lightColor;

specular = att * specularStrength * specCoeff * lightColor;
```

4 Hărți de iluminare

Până acum, când era vorba de iluminare, am considerat obiectele ca fiind compuse dintr-un singur material (având o singură culoare de bază) (Figura 2). Cu toate acestea, această abordare nu oferă prea multă flexibilitate pentru reprezentarea vizuală a unui obiect. Vom extinde acum modelul de iluminare pentru a include **hărți de iluminare**.

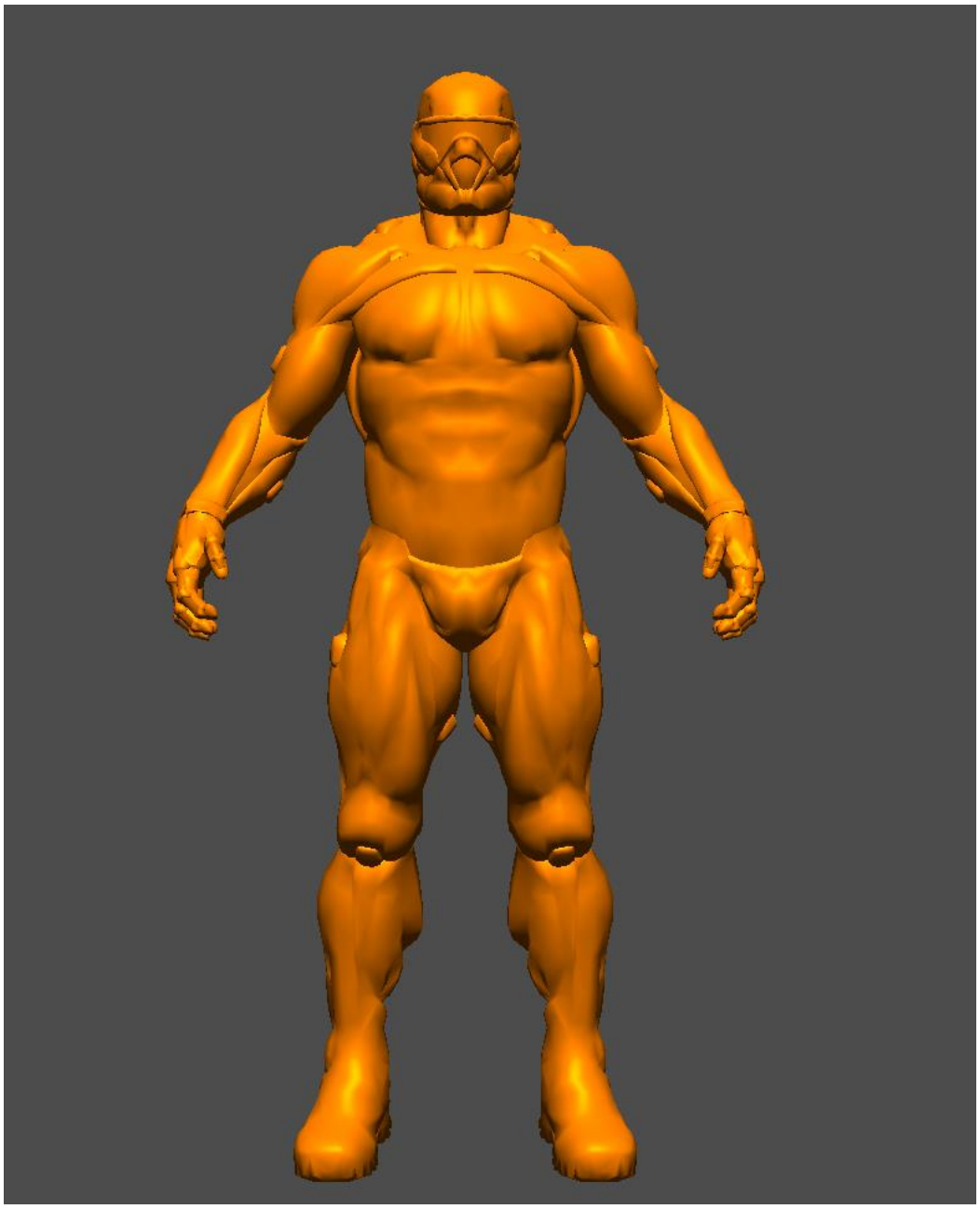

Figura 2 – Crysis 2 nanosuit rasterizat cu lumină punctiformă și culoare de bază portocalie

Hărțile de iluminare ne furnizează un mijloc de eșantionare a culorii de bază pentru fiecare fragment dintr-o textură și pentru fiecare componentă a luminii (ambientală, difuză și specular). Ele sunt încorporate în proprietățile materialelor asociate modelelor OBJ (specificate in fișierul MTL).

Hărțile difuze definesc culorile unui obiect. Informațiile despre culoare se eșantionează dintr-o textură și se combină cu efectele iluminării pentru a determina culoarea finală a fragmentelor (Figura 3). Hărțile hibride folosesc de obicei și culoarea de bază pentru componenta ambientală. Astfel, calculul componentelor ambientale și difuze trebuie modificate:

```
    ambient *= texture(diffuseTexture, fragTexCoords);
    diffuse *= texture(diffuseTexture, fragTexCoords);
```

Figure 3 – Crysis 2 nanosuit rasterizat cu hărți de iluminare difuze

De obicei, nu toate materialele care compun un obiect reflectă aceeași cantitate de strălucire specular. Pentru a diferenția între reflexia diferitelor materiale, putem folosi **hărți speculare** care atribuie un coeficient specular mai mare materialelor cu grad ridicat de reflexie și unul inferior materialelor care nu prezintă în mod normal componente speculare (Figura 4). Hărțile speculare pot fi eșantionate exact ca hărțile difuze, dar sunt aplicate la calculul componentelor speculare:

```
specular *= texture(specularTexture, fragTexCoords);
```

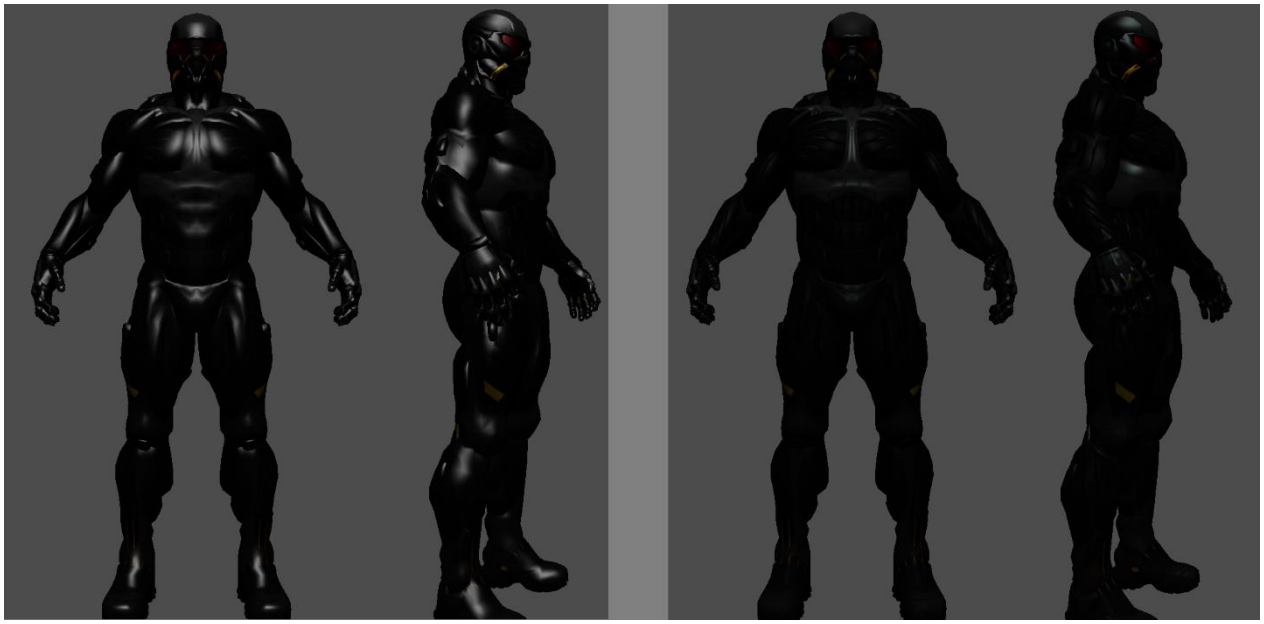

Figure 4 – Crysis 2 nanosuit rasterizat fără hărți speculare (stânga) și cu hărți speculare (dreapta)

Biblioteca de încărcare a obiextelor OBJ pe care o utilizati include deja funcționalitate pentru încărcarea hărților difuze și speculare (texturi) și legarea corespunzătoare a acestor obiecte de textură în cadrul claselor Model3D și Mesh. Variabilele **uniform** "diffuseTexture" și "specularTexture" trebuie să fie definite în programul shader pentru a permite eșantionarea hărților de iluminare ale obiectului.

5 Lectură suplimentară

- OpenGL Programming Guide 8th Edition – Capitolul 7
- Tutoriale OpenGL – Light casters, Lighting maps, Multiple lights, Advanced lighting - http://www.learnopengl.com/

6 Temă

1. Descărcați și explorați mostrele de cod furnizate ca resurse pentru acest laborator
2. Implementați modelul de iluminare Blinn-Phong pentru aplicația demo dată
3. Transformați lumina direcțională într-o lumină punctiformă pentru aplicația demonstrativă
4. Afișati mai multe obiecte împrăștiate prin toată scena pentru a observa efectul luminilor punctiforme
5. Incorporați hărți de iluminare în aplicația demo
6. Definiți o lumină direcțională și un al doilea punct (locație) de lumină în aplicația demonstrativă