

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 6241

**Fizikalno temeljen model
ostvarivanja prikaza**

Antonio Junaković

Zagreb, lipanj 2019.

SADRŽAJ

1. Uvod	1
2. OpenGL	2
3. Sjenčanje	3
3.1. Sjenčari	3
3.2. Sjenčar vrhova	4
3.3. Teselacijski sjenčari	6
3.4. Geometrijski sjenčar	6
3.5. Sjenčar fragmenata	7
3.6. Računski sjenčar	8
4. Interakcija svjetlosti s različitim površinama	9
4.1. Model mikrozrcala	9
4.2. Očuvanje energije	10
4.3. Radiometrijske mjere	11
4.4. Jednadžba ostvarivanja prikaza i jednadžba reflektivnosti	12
5. Funkcija distribucije dvosmjerne refleksije	14
5.1. Formalna definicija	14
5.2. Cook-Torranceov model	15
5.3. Funkcija distribucije normala	16
5.4. Fresnelov koeficijent	16
5.5. Geometrijska funkcija	18
6. Osvjetljavanje temeljeno na slici	20
6.1. Uzorkovanje kubne teksture	20
6.2. Ravnopravna projekcija	21
6.3. Difuzna obasjanost	22

6.4. Spekularno osvjetljavanje	25
7. Kvaliteta slike	28
7.1. Slike visokog dinamičkog raspona	28
7.2. Mapiranje tona slike	29
7.3. Gama ispravljanje	29
8. Programsko rješenje	30
9. Zaključak	32
Literatura	33

1. Uvod

Već od samih početaka računalne grafike, brojni istraživači su radili na definiranju teorijske baze za ostvarivanje prikaza temeljene na fizikalnim principima iz stvarnog svijeta. S obzirom na ograničenja ranog računalnog sklopovlja, takvi pristupi nisu bili izvedivi u stvarnom vremenu, te su računalne igrice uglavnom koristile pojednostavljene modele prikaza. Nagli i ubrzani napredak performansi grafičkih kartica, te sklopovlja općenito, omogućio je adaptiranje aproksimacija teorijske baze za izvođenje u stvarnom vremenu, stvarajući kompromis između fizikalne točnosti prikaza i složenosti izračuna. Dan danas gotovo sve igrice implementiraju nekakav oblik fotorealističnog modela prikazivanja, što je ujedno i jedan od glavnih razloga koji potiče razvoj tog područja računalne grafike.

Predmet ovog rada je uvod u jedan od takvih modela prikazivanja, uz funkcionalnu implementaciju u modernom grafičkom aplikacijskom sučelju OpenGL. Cilj je razlučiti konačnu dobivenu sliku na komponente koje ju sastavljaju, uz detaljno objašnjenje izračun vrijednosti svake od tih komponenata. Nadalje, generirani prikaz će biti uspoređen sa svojstvima ekvivalentnog materijala iz stvarnog svijeta, naglašavajući sličnosti i razlike pri ponašanju svjetlosti u različitim uvjetima.

2. OpenGL

OpenGL (engl. Open Graphics Library) je otvoreno, platformski i jezični nezavisno aplikacijsko programsko sučelje za generiranje 2D i 3D vektorske grafike. Razvijeno je iz biblioteke IRIS GL tada vodećeg proizvođača 3D grafičkih radnih stanica Silicon Graphics zbog potrebe za otvorenim standardom. OpenGL od konkurencije je izdvajala jednostavnost korištenja i izravan način rada (engl. immediate mode). Jedini značajniji suparnik koji je opstao jest Microsoftov Direct3D, koji je zbog raširenosti operacijskog sustava Windows postao popularnija opcija.

Zbog povećanja složenosti scene i broja vrhova došlo je do potrebe za uvodom novog načina rada sa sjenčarima. U početku je bilo moguće pisati samo sjenčare fragmenata, no ubrzo nakon što se počela očitovati moć sjenčanja su uvedeni i sjenčari vrhova. Danas je način rada sa sjenčarima postao standardan u većini grafičkih biblioteka, a neke uopće niti nemaju izravan način rada.

Khronos Grupa je američki neprofitni konzorcij financiran od strane tvrtki članica koji se bavi održavanjem otvorenih standarda među koje ubrajamo i OpenGL. Također je bitno napomenuti da Khronos Grupa održava još i noviji grafički standard Vulkan koji je još niže razine nego što je OpenGL tako da omogućuje izravniji pristup grafičkom sklopovlju.

OpenGL ES je varijanta aplikacijskog sučelja OpenGL osmišljena za prijenosne uređaje kao što su mobiteli, tableti i igraće konzole. Verzije 2 i novije ne podržavaju izravni način rada nego samo način rada sa sjenčarima. Varijanta WebGL je dostupna unutar web-preglednika zahvaljujući biblioteci ANGLE¹ koja prevodi OpenGL ES pozive u pozive Direct3D-a 9 ili 11, OpenGL-a te Vulkanu.

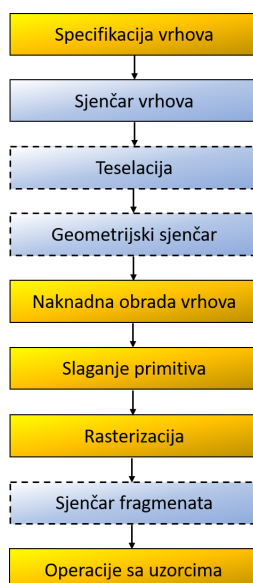
¹ANGLE (engl. Almost Native Graphics Layer Engine) je otvorena Googleova biblioteka koja služi kao apstrakcijski sloj između grafičkih pogona koji je ugrađen u većinu poznatijih web-preglednika.

3. Sjenčanje

3.1. Sjenčari

Sjenčar (engl. shader) je općeniti naziv za program koji se izvršava na grafičkoj kartici. Prva moderna upotreba riječi se pojavila unutar specifikacije sučelja alata RenderMan animacijskog studija Pixar.

GLSL (engl. GL Shading Language) je programski jezik koji se koristi za pisanje programa sjenčara OpenGL aplikacijskog sučelja. Po sintaksi je sličan jezicima C i C++. Upravljačkom programu (engl. driver) se izvorni kod sjenčara predaje u tekstualnom obliku koji se onda prevodi u strojni jezik specifičan prisutnom sklopovlju. Također je u novijim verzijama OpenGL-a moguće predati već preveden kod sjenčara u posredničkom strojnom jeziku SPIR-V (engl. Standard, Portable Intermediate Representation - V). Posebnost strojnog jezika SPIR-V je da, osim što se GLSL može prevesti u njega, također postoje i prevoditelji Microsoftovog jezika HLSL (engl. High-Level Shading Language) korištenog unutar Direct3D aplikacijskog sučelja.

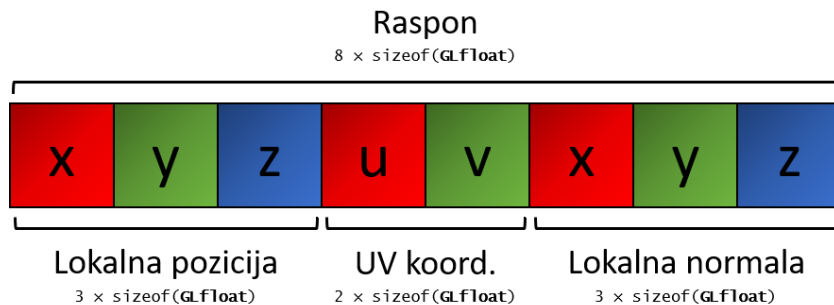


Slika 3.1: Grafički protočni sustav [5]

Na slici 3.1 je prikazan grafički protočni sustav koji je iniciran izdavanjem naredbe za crtanje objekta. U prvom koraku se dohvaća tok vrhova iz objekta niza vrhova (engl. Vertex Array Object) te se šalje u sjenčar vrhova koji nad njima obavlja željene operacije. Ako su teselacijski sjenčari zadani, transformirani vrhovi ulaze u teselacijsku obradu. Isto vrijedi i za geometrijski sjenčar. U sljedećim koracima se odvija odsijecanje prema volumenu pogleda, odsijecanje prema korisnikovim specifikacijama (ako ih je zadao u sjenčaru), dijeljenje perspektive te transformacija u prostor pogleda (engl. viewport). Konačno, pred kraj se odvija rasterizacija vrhova u fragmente, te se za svaki zasebni fragment aktivira sjenčar fragmenata. Na izlazu iz sjenčara fragmenata se provodi niz ispitivanja nad fragmentima ako su ona omogućena u OpenGL kontekstu.

3.2. Sjenčar vrhova

Sjenčar vrhova (engl. vertex shader) je dio programa sjenčanja koji obavlja transformacije nad vrhovima. Tok atributa vrhova je učitani iz međuspremnika postavljenog objekta niza vrhova OpenGL konteksta.



Slika 3.2: Primjer raspona atributa vrhova

Na slici 3.2 je prikazan primjer raspona atributa vrhova (engl. stride). Sastoji se od osam vrijednosti s pomičnim zarezom jednostruke preciznosti, od kojih prve tri čine lokalnu poziciju vrha, sljedeće dvije čine UV teksturne koordinate, a zadnje tri čine lokalnu normalu. Upravo ova raspodjela je korištena u implementaciji, no naravno po potrebi je moguće dodati još atributa kao što su primjerice tangenta, bitangenta, stvarna normala itd.

Za omogućavanje atributa se poziva funkcija `glEnableVertexAttribArray` s parametrom indeksa atributa. Mapiranje atributa se ostvaruje pozivom funkcije `glVertexAttribPointer`, kojoj se kao parametar predaje indeks atributa, tip

te duljina podataka, ukupna duljina raspona i odmak od početka raspona. Po potrebi se može omogućiti normalizacija atributa.

```
#version 410 core
layout (location = 0) in vec3 local_pos;
layout (location = 1) in vec2 local_uv;
layout (location = 2) in vec3 local_norm;
uniform mat4 proj_view;
uniform mat4 model;
out vec3 world_pos;
out vec2 uv;
out vec3 world_norm;
void main() {
    world_pos = (model * vec4(local_pos, 1.0)).xyz;
    uv = local_uv;
    world_norm = normalize(mat3(model) * local_norm);
    gl_Position = proj_view * vec4(world_pos, 1.0);
}
```

Slika 3.3: Primjer jednostavnog sjenčara vrhova u jeziku GLSL

Na slici 3.3 je prikazan primjer relativno jednostavnog sjenčara vrhova napisanog u jeziku za sjenčanje GLSL. Direktivom `#version` je deklarirana željena verzija jezika za sjenčanje. Tipovi deklarirani s prefiksom `in` definiraju ulazne tipove podataka u sjenčar, u ovom slučaju attribute vrhova locirane na indeksu parametra `location`. Tipovi deklarirani s prefiksom `uniform` predstavljaju vrijednosti zadane sjenčaru iz vanjskog programa. Oni su prisutni unutar svih stadija sjenčanja, a vrijednost im je jednaka za sve vrhove. Konačno, tipovi deklarirani s prefiksom `out` definiraju izlazne tipove podataka koji se propagiraju u sljedeći stadij grafičkog protočnog sustava.

Unutar ulazne funkcije `main` provodimo transformaciju lokalne pozicije vrha u poziciju unutar svijeta. Koordinate teksture se samo propagiraju u sljedeći stadij s obzirom da za uzorkovanje nije potrebno provesti nikakvu transformaciju. Na lokalnu normalu primjenjujemo transformacijsku matricu `model` skraćenu na tri dimenzije konstruktorom `mat3`. Time izbacujemo translaciju ostavljajući preostale transformacije - pretpostavka je da će skaliranje biti uniformno i da neće utjecati na smjer normale. Također ju još i normaliziramo tako da dobijemo točniji rezultat nakon što se primjeni interpolacija nad njom. U implicitnu izlaznu varijablu `gl_Position` stav-

ljamo poziciju transformiranog vrha u sustavu promatrača (dakle, nakon primjene matrice pogleda i projekcije). Varijabla je u homogenom prostoru, te sadrži informaciju o 2D poziciji na ekranu i dubini vrha. Ta se varijabla interpolira i za svaki fragment će biti aktiviran sjenčar fragmenata s također interpoliranim ostalim vrijednostima izlaza.

3.3. Teselacijski sjenčari

Teselacija ili popločavanje je postupak u kojemu se primitiv rastavlja na veći broj manjih primitiva. Postupak je noviji dodatak grafičkom protočnom sustavu, a njegova prisutnost se može provjeriti ispitivanjem `GL_ARB_tessellation_shader` [5]. Ovaj sjenčar nije korišten u implementaciji, no veoma je koristan za dodavanje detalja objektima koji se nalaze bliže kameri tj. oku uzorkovanjem visinske mape (engl. height map).

Postoje dva tipa teselacijskih sjenčara: kontrolni teselacijski sjenčar (engl. tessellation control shader) i evaluacijski teselacijski sjenčar (engl. tessellation evaluation shader). Kontrolni se koristi za zadavanje parametara i uvjeta raspodjele vrhova primitiva, dok evaluacijski se koristi za izračun konačne pozicije novodobivenih vrhova. Primjena teselacije je prikazana na slici 3.4.



Slika 3.4: Primjena teselacije

3.4. Geometrijski sjenčar

Geometrijski sjenčar (engl. geometry shader) je tip sjenčara koji upravlja procesuiranjem primitiva. Kao i teselacijski sjenčari, geometrijski sjenčar je noviji dodatak grafičkom protočnom sustavu čija se prisutnost može provjeriti ispitivanjem prisutnosti proširenja `GL_ARB_geometry_shader4` [5]. Također ni ovaj sjenčar nije korišten u implementaciji, no veoma je koristan pri prikazivanju čestica ili većeg broja

istih primitiva čijom je pozicijom lakše upravljati samom transformacijom nad jednim vrhom nego nad čitavim primitivom.

Geometrijski sjenčar može pretvarati primitive iz jednog tipa u drugi, a najčešće je korišten pri pretvorbi vrhova u trokute ili četverokute. Na slici 3.5 je prikazana primjena geometrijskog sjenčara.



Slika 3.5: Primjena geometrijskog sjenčara

3.5. Sjenčar fragmenata

Sjenčar fragmenata (engl. fragment shader) je tip sjenčara koji obrađuje svaki individualni fragment generiran rasterizacijskim stadijem grafičkog protočnog sustava. Ovaj sjenčar je konačni korak u kojemu se određuje boja, prozirnost i dubina fragmenta. Kako će se fragment ugraditi u prikazanu sliku određuju postavke konteksta vezane uz ispitivanje dubine (engl. depth testing), miješanje boja (engl. blending), odsijecanje (engl. scissor testing) i sl.

Na slici 3.6 je prikazan primjer jednostavnog sjenčara fragmenata napisanog u jeziku GLSL. Unutar ulazne funkcije `main` je implementiran naivni model izračuna intenziteta boje. Vektor od pozicije fragmenta u svjetskom prostoru do pozicije točkastog izvora svjetla je izračunat i pretvoren u jedinični vektor. Intenzitet svjetla je onda izračunat kao kosinus kuta (to jest skalarni produkt) između tog vektora i vektora normale fragmenata, te je zaokružen na vrijednost veću ili jednaku 0.1. Boja se onda određuje uzorkovanjem difuzne teksture i množenjem s prethodno izračunatim intenzitetom. Varijabla `color` je izlazna varijabla koja opisuje vrijednosti RGBA komponenti boje fragmenata.

```

#version 410 core
in vec3 world_pos;
in vec2 uv;
in vec3 world_norm;
uniform vec3 light_pos;
uniform sampler2D diffuse;
out vec4 color;
void main() {
    vec3 to_light = normalize(light_pos - world_pos);
    float light_int =
        max(0.1, dot(to_light, normalize(world_norm)));
    color = vec4(light_int * texture(diffuse, uv).rgb, 1.0);
}

```

Slika 3.6: Primjer jednostavnog sjenčara fragmenata u jeziku GLSL

Ovaj pristup je jednostavna demonstracija sjenčara fragmenata i služi samo kao uvodni korak. U sljedećim poglavljima će biti opisane neke od jednažbi fizikalno temeljenog modela prikazivanja koje se mogu koristiti umjesto ove za dobivanje uvjerljivije i realističnije slike.

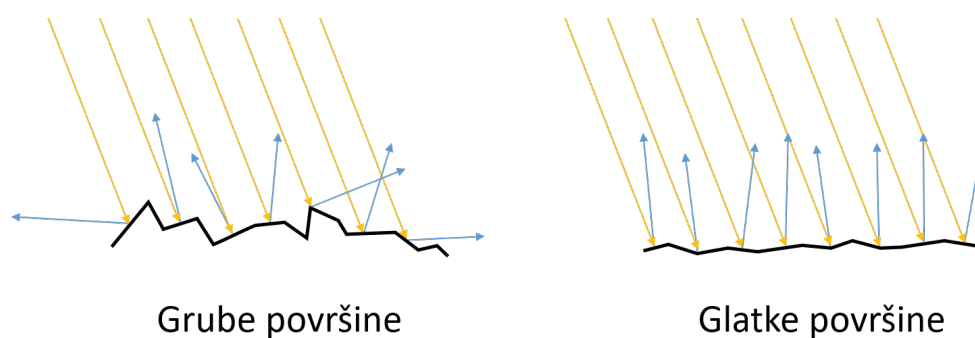
3.6. Računski sjenčar

Računski sjenčar (engl. compute shader) je tip sjenčara koji nije vezan za grafički protočni sustav. Može biti korišten za generiranje prikaza, no njegova češća uporaba je za zadaće koje nisu izravno povezane s prikazivanjem poligona. Po ideji je srodan aplikacijskom programskom sučelju OpenCL. Prisutnost računskog sjenčara se može provjeriti ispitivanjem `GL_ARB_compute_shader` [5] proširenja.

4. Interakcija svjetlosti s različitim površinama

4.1. Model mikrozrcala

Svi fizikalno temeljeni modeli ostvarivanja prikaza se baziraju na teoriji modela mikrozrcala (engl. microfacets model theory). Teorija opisuje da se bilo koja površina na mikroskopskoj razini može opisati kao nakupina malih savršeno reflektivnih zrcala nazvanih mikrozrcala (engl. microfacets). Što je grublja površina, to su mikrozrcala kaotičnije poravnata u odnosu na ravninu. Učinak toga jest da za grublju površinu reflektirana svjetlost će biti rasprostranjena raspršeno, dok u kontrastu za glatku površinu će reflektirana svjetlost biti rasprostranjena otprilike u istom smjeru.



Slika 4.1: Reflektirane zrake svjetlosti modela mikrozrcala

Na slici 4.1 je s lijeve strane prikazana gruba površina, a s desne glatka površina. Žutom bojom su nacrtane upadne zrake iz izvora svjetlosti, dok su s plavom bojom nacrtane reflektirane zrake. Sa slike se može zaključiti da će za grube površine spe-

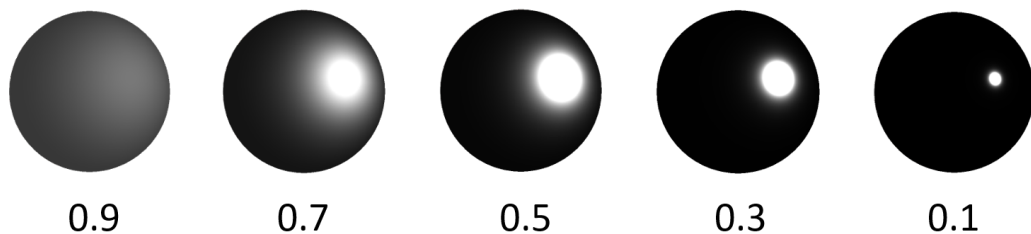
kularna refleksija biti slabija ali raširenija, dok će za glatke površine biti oštrija ali sažetija.

Uvedimo sad izraz polovičnog vektora \hat{h} (engl. halfway vector) kao vektor koji se nalazi “na pola puta” između vektora smjera izvora svjetla \hat{l} i vektora smjera pogleda \hat{v} . Izraz za polovični vektor je dan na slici 4.2.

$$\hat{h} = \frac{\hat{l} + \hat{v}}{\|\hat{l} + \hat{v}\|}$$

Slika 4.2: Izraz za polovični vektor

Nadalje možemo definirati faktor gruboće (engl. roughness) kao vrijednost iz domene $[0, 1]$. Određen tim faktorom će biti udio mikrozrcala čija je normala poravnata s polovičnim vektorom \hat{h} .



Slika 4.3: Intenzitet spekularne refleksije kugle po različitim faktorima gruboće

Na slici 4.3 je na kugli prikazan udio mikrozrcala poravnatih s polovičnim vektorom \hat{h} po različitim faktorima gruboće. Bijela boja odgovara većem udjelu, a crna manjem.

4.2. Očuvanje energije

Aproksimacija modela mikrozrcala uključuje i pravilo očuvanja energije, to jest, energija izlazne svjetlosti ne smije biti veća od energije ulazne svjetlosti (osim kod emisivnih površina). Kad se svjetlost sudari s površinom dijeli se na dio koji je reflektiran i dio koji je refraktiran. Nakon dodatnih sudara refraktirane svjetlosti unutar površine tijela može doći do naknadnog izviranja dijela iste zrake koja će kontribuirati difuznoj boji površine. Taj efekt se naziva potpovršinsko raspršivanje (engl. subsurface scattering) i sjenčari koji to uzimaju u obzir realističnije prikazuju materijale kao što su ljudska koža, mramor i vosak. Jednostavniji modeli zanemaruju taj učinak i pretpostavljaju da će čitava refraktirana svjetlost biti apsorbirana u površini tijela. Nadalje,

metalne površine slijede ista pravila refleksije i refrakcije, no za razliku od dielektrika (nemetallnih materijala) u potpunosti apsorbiraju čitavu refraktiranu svjetlost bez raspršivanja, što znači da nemaju difuznu boju. Upravo zbog toga se uvodi faktor metalnosti (engl. metalness) koji nam omogućuje da tretiramo različite metalne površine i dielektrike. Raspon faktora metalnosti je $[0, 1]$, no uobičajeno je da za većinu materijala postavljamo vrijednost na 1 (u potpunosti metalna površina) ili na 0 (u potpunosti nemetalna površina).

Sljedeća važna observacija jest da su reflektirana i refraktirana svjetlost međusobno ekskluzivne. To znači da za udio difuzne svjetlosti k_d i udio spekularne svjetlosti k_s vrijedi jednačina na slici 4.4.

$$k_d + k_s = 1$$

Slika 4.4: Izraz za očuvanje energije svjetlosti

4.3. Radiometrijske mjere

Prostorni kut (engl. solid angle) je mjera površine oblika projiciranog na kuglu radijusa 1. Oznaka za mjernu jedinicu je sr (steradian). Energija isijavanja (engl. radiant energy) je radiometrijska mjera elektromagnetske radijacije. Korištena mjerna jedinica je J (Joule). Tok isijavanja (engl. radiant flux) je radiometrijska mjera za energiju isijavanja emitiranu, reflektiranu, transmitiranu ili primljenu po jedinici vremena. Korištena mjerna jedinica je W (Watt). Sjajnost (engl. radiance) je radiometrijska mjera za tok isijavanja emitiran, reflektiran, transmitiran ili primljen po jedinici steradiana i jedinici projicirane površine.

4.4. Jednadžba ostvarivanja prikaza i jednadžba reflektivnosti

Jednadžba ostvarivanja prikaza (engl. rendering equation) je integralna jednadžba u kojoj se prikazuje izlazna sjajnost točke kao suma emitirane i reflektirane sjajnosti.

$$L_o(\mathbf{x}, \omega_o, \lambda, t) = L_e(\mathbf{x}, \omega_o, \lambda, t) + \int_{\Omega} f_r(\mathbf{x}, \omega_i, \omega_o, \lambda, t) L_i(\mathbf{x}, \omega_i, \lambda, t) (\omega_i \cdot \mathbf{n}) d\omega_i$$

Slika 4.5: Jednadžba ostvarivanja prikaza

Na slici 4.5 je prikazana jednadžba ostvarivanja prikaza. Značenje svih oznaka je objašnjeno u nastavku:

- L_o je ukupna sjajnost valne duljine λ usmjerena izvan prema ω_o smjeru u trenutku t iz točke \mathbf{x} .
- \mathbf{x} je točka u prostoru.
- ω_o je smjer svjetlosti na izlasku.
- λ je valna duljina svjetlosti.
- t je trenutak u vremenu.
- L_e je emitirana sjajnost.
- Ω je jedinična polukugla centrirana oko normale \mathbf{n} koja sadrži sve moguće vrijednosti za ω_i .
- f_r je funkcija distribucije dvosmjerne refleksije - omjer svjetlosti reflektiran od ω_i do ω_o u točki \mathbf{x} , vremenu t i pri valnoj duljini λ .
- ω_i je negiran smjer svjetlosti na ulasku.
- L_i je sjajnost valne duljine λ nadolazeće prema točki \mathbf{x} iz smjera ω_i u vremenu t .
- \mathbf{n} je površinska normala u točki \mathbf{x} .
- $(\omega_i \cdot \mathbf{n})$ je faktor oslabljivanja, često zapisan kao $\cos \theta_i$ gdje je θ_i kut između ω_i i normale \mathbf{n} .

$$L_o(\mathbf{x}, \omega_o, \lambda, t) = \int_{\Omega} f_r(\mathbf{x}, \omega_i, \omega_o, \lambda, t) L_i(\mathbf{x}, \omega_i, \lambda, t) (\omega_i \cdot \mathbf{n}) d\omega_i$$

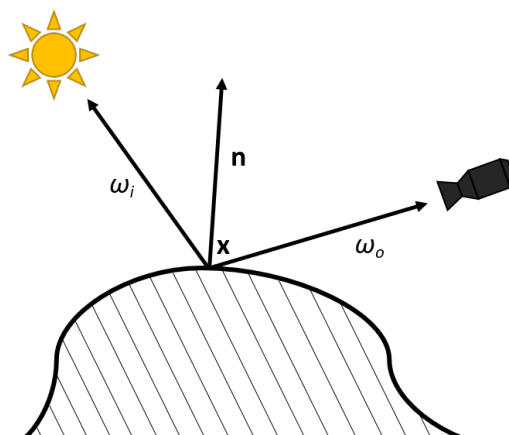
Slika 4.6: Jednadžba reflektivnosti

Izrazom na slici 4.6 je prikazan specijalizirani oblik jednadžbe ostvarivanja prikaza nazvan jednadžba reflektivnosti (engl. reflectance equation). Jednadžba vrijedi za površine koje ne emitiraju svjetlost, to jest, za površine čija je emitirana sjajnost L_e jednaka nuli. Unutar te jednadžbe jedina nepoznanica u ovom trenutku nam je funkcija distribucije dvosmjerne refleksije f_r - funkcija koja skalira ulaznu sjajnost na temelju svojstava materijala površine. Naglasak na nju će biti u idućem poglavlju.

5. Funkcija distribucije dvosmjerne refleksije

5.1. Formalna definicija

Funkcija distribucije dvosmjerne refleksije (engl. bidirectional reflectance distribution function, BRDF) je funkcija koja definira kako se svjetlost reflektira o neprozirnu površinu. Često se koristi u optičkim proračunima u stvarnom svijetu, te u algoritmima računalne grafike i računalnog vida. Funkcija ovisi o smjeru svjetlosti pri ulasku ω_i , smjeru svjetlosti na izlasku ω_o , položaju točke u prostoru x , boji ili valnoj duljini svjetlosti λ i trenutku u vremenu t . Na slici 5.1 je prikazana ilustracija koja predstavlja parametre funkcije prikazane u perspektivi prostora.



Slika 5.1: Parametri funkcije distribucije dvosmjerne refleksije

5.2. Cook-Torranceov model

Cook-Torranceov model je jedan od najčešćih modela korištenih za izračun funkcije distribucije dvosmjerne refleksije. Ponašanje svjetlosti u modelu razlikuje difuznu i spekularnu komponentu. Na slici 5.2 je prikazan izraz za funkciju distribucije dvosmjerne refleksije. Izraz $f_{lambert}$ opisuje difuznu komponentu, izraz $f_{cook-torrance}$ opisuje spekularnu komponentu, a izrazi k_d i k_s opisuju udio svake od komponenti.

$$f_r = k_d f_{lambert} + k_s f_{cook-torrance}$$

Slika 5.2: Funkcija distribucije dvosmjerne refleksije Cook-Torranceovog modela [1]

Difuznu komponentu postavljamo na konstantu kao što se može uočiti na slici 5.3. Izraz c predstavlja albedo boju materijala. Boja je podijeljena sa π jer, ako se prisjetimo, čitav izraz za f_r se nalazi pod integralom koji će nakon integracije čitav rezultat skalirati sa π . Također, difuznu komponentu nije potrebno množiti s $(\omega_i \cdot \mathbf{n})$ s obzirom da se taj izraz već nalazi unutar integrala.

$$f_{lambert} = \frac{c}{\pi}$$

Slika 5.3: Izraz Lambertove difuzne komponente [1]

Izraz za spekularnu komponentu je nešto kompliciraniji kao što vidimo na slici 5.4. U izrazu D predstavlja funkciju distribucije normala, F predstavlja Fresnelov koeficijent, a G predstavlja geometrijsku funkciju.

$$f_{cook-torrance} = \frac{DFG}{4(\omega_o \cdot \mathbf{n})(\omega_i \cdot \mathbf{n})}$$

Slika 5.4: Izraz Cook-Torranceove spekularne komponente [1]

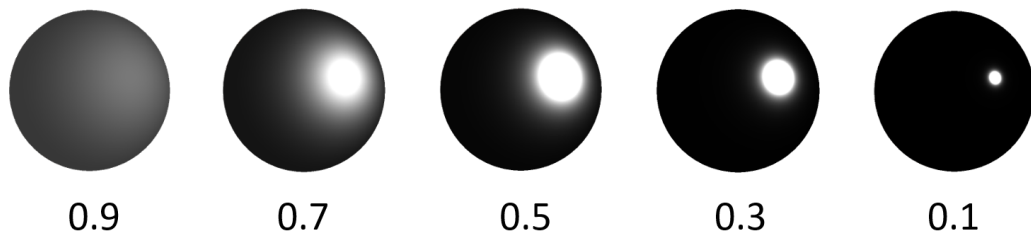
5.3. Funkcija distribucije normala

Funkcija distribucije normala D (engl. normal distribution function) statistički aproksimira omjer mikrozrcala koji su poravnati s polovičnim vektorom \hat{h} . Parametri funkcije su normala površine \mathbf{n} , polovični vektor \hat{h} te faktor α . Faktor α je mjera koja ovisi o faktoru gruboće, a uobičajeno je jednaka njegovom kvadratu.

$$D_{GGXTR}(\mathbf{n}, \hat{h}, \alpha) = \frac{\alpha^2}{\pi((\mathbf{n} \cdot \hat{h})^2(\alpha^2 - 1) + 1)^2}$$

Slika 5.5: Funkcija distribucije normala GGX Trowbridge-Reitz [4]

Postoje mnoge funkcije koje aproksimiraju funkciju distribucije normala. Jedna od njih je funkcija distribucije normala GGX Trowbridge-Reitz korištena u implementaciji, prikazana na slici 5.5. Bitno je uočiti da je rezultat funkcija distribucije normala skalar.



Slika 5.6: Funkcija distribucije normala na kugli po različitim faktorima gruboće

Na slici 5.6 možemo uočiti vrijednosti funkcije distribucije normala u raznim točkama kugle za različite vrijednosti faktora gruboće. Za glatke plohe je velik omjer poravnatih mikrozrcala koncentriran na maloj površini dok je za grube plohe raširen po većem dijelu površine u manjim omjerima.

5.4. Fresnelov koeficijent











Fresnelov koeficijent F (engl. Fresnel coefficient) opisuje omjer svjetlosti koja se reflektira naspram svjetlosti koja se refraktira. On ovisi o polovičnom vektoru \hat{h} i kutu kojim se gleda na površinu \hat{v} . Na slici 5.7 je prikazana Fresnel-Schlickova funkcija, jedna od funkcija kojom se može aproksimirati Fresnelov koeficijent.

$$F_{Schlick}(\hat{h}, \hat{v}, F_0) = F_0 + (1 - F_0)(1 - (\hat{h} \cdot \hat{v}))^5$$

Slika 5.7: Fresnel-Schlickova funkcija [4]

Bazna reflektivnost površine je označena sa F_0 . To je boja koja za metalne površine poprima visoke vrijednosti dok za dielektrike poprima niske vrijednosti. Prikazane u tablici 5.1 su vrijednosti bazne reflektivnosti za neke površine.

Tablica 5.1: Vrijednosti bazne reflektivnosti za neke površine [3]

Materijal	F_0 (Linearna)	F_0 (sRGB)	Boja
Voda	(0.02, 0.02, 0.02)	(0.15, 0.15, 0.15)	
Plastika / staklo (niska)	(0.03, 0.03, 0.03)	(0.21, 0.21, 0.21)	
Plastika (visoka)	(0.05, 0.05, 0.05)	(0.24, 0.24, 0.24)	
Staklo (visoka) / rubin	(0.08, 0.08, 0.08)	(0.31, 0.31, 0.31)	
Dijamant	(0.17, 0.17, 0.17)	(0.45, 0.45, 0.45)	
Željezo	(0.56, 0.57, 0.58)	(0.77, 0.78, 0.78)	
Bakar	(0.95, 0.64, 0.54)	(0.98, 0.82, 0.76)	
Zlato	(1.00, 0.71, 0.29)	(1.00, 0.86, 0.57)	
Aluminij	(0.91, 0.92, 0.92)	(0.96, 0.96, 0.97)	
Srebro	(0.95, 0.93, 0.88)	(0.98, 0.97, 0.95)	

Također je moguće napisati i oblik funkcije koji ovisi o normali površine \mathbf{n} , vektoru pogleda \hat{v} , baznoj reflektivnosti F_0 te također i faktoru gruboće, ovdje označenog kao r na slici 5.8.

$$F_{Schlick-Roughness}(\mathbf{n}, \hat{v}, F_0, r) = F_0 + (\max((1 - r, 1 - r, 1 - r), F_0) - F_0)(1 - (\mathbf{n} \cdot \hat{v}))^5$$

Slika 5.8: Fresnel-Schlickova funkcija s faktorom gruboće [4]

Na slici 5.9 možemo vidjeti primjenu navedene funkcije na kugli za različite vrijednosti faktora gruboće.



Slika 5.9: Fresnel-Schlickova funkcija s faktorom gruboće po različitim faktorima gruboće

5.5. Geometrijska funkcija

Geometrijska funkcija G (engl. geometry function) je funkcija koja statistički aproksimira relativnu površinu gdje se njezini detalji na mikroskopskoj razini međusobno zasjenjuju uzrokujući upijanje zraka svjetlosti. Funkcija ovisi o normalu površine \mathbf{n} , vektoru smjera pogleda \hat{v} , vektoru smjera izvora svjetlosti \hat{l} te remapiranom koeficijentu gruboće k . Na slici 5.10 je prikazana Schlick-Beckmannova GGX aproksimacija geometrijske funkcije.

$$G_{SchlickGGX}(\mathbf{n}, \hat{v}, k) = \frac{\mathbf{n} \cdot \hat{v}}{(\mathbf{n} \cdot \hat{v})(1 - k) + k}$$

Slika 5.10: Geometrijska funkcija Schlick-Beckmann GGX [4]

Koeficijent k se može izračunati na različite načine. Na slici 5.11 su prikazana dva načina izračuna. Na lijevoj strani je prikazan izraz za izravan izračun, a na desnoj je prikazan izraz koji se koristi u kombinaciji s tehnikom osvjetljavanja temeljenog na slici.

$$k_{direct} = \frac{(\alpha + 1)^2}{8} \quad k_{IBL} = \frac{\alpha^2}{2}$$

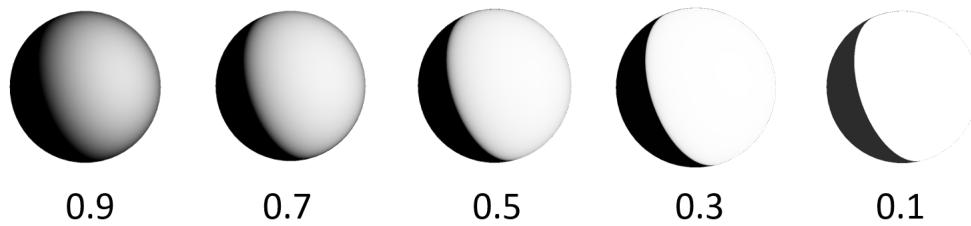
Slika 5.11: Konstante Schlick-Beckmann GGX geometrijske funkcije [4]

Za efektivniju aproksimaciju geometrijske funkcije, potrebno je još uključiti i vektor izvora svjetlosti \hat{l} . To možemo izvesti koristeći Smithovu metodu aproksimacije prikazanu na slici 5.12.

$$G_{Smith}(\mathbf{n}, \hat{v}, \hat{l}, k) = G_{SchlickGGX}(\mathbf{n}, \hat{v}, k)G_{SchlickGGX}(\mathbf{n}, \hat{l}, k)$$

Slika 5.12: Smithova metoda aproksimacije geometrijske funkcije [4]

Na slici 5.13 je prikazana usporedba Smithove aproksimacije geometrijske funkcije na kugli po različitim faktorima gruboće.



Slika 5.13: Smithova geometrijska funkcija po različitim faktorima gruboće

6. Osvjetljavanje temeljeno na slici

6.1. Uzorkovanje kubne teksture

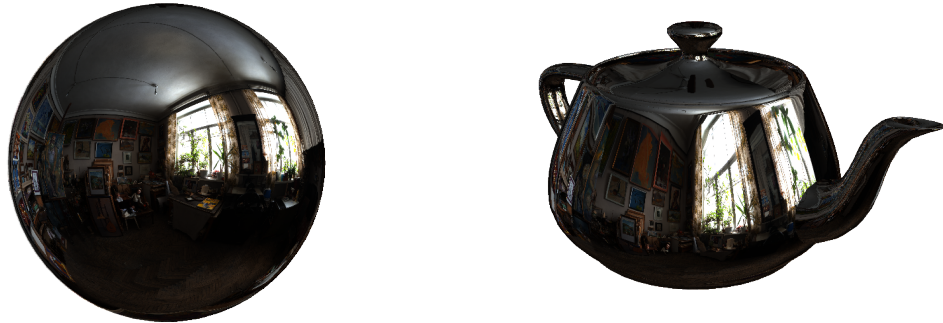
U programu sjenčara fragmenata često želimo realizirati zrcaljenje okoliša na objekt. Postoji nekoliko načina da to ostvarimo. Najjednostavniji način bi bio korištenje kubne teksture (engl. cubemap). Primjer kubne teksture je prikazan u nastavku na slici 6.1.



Slika 6.1: Primjer kubne teksture

Kubna tekstura se sastoji od 6 sastavnih tekstura, od kojih svaka odgovara jednoj od 6 ploha kocke. Za svaki fragment objekta na kojemu želimo ostvariti zrcaljenje ćemo koristiti njegovu normalu u toj točki za izračun boje kubne teksture koja bi bila zrcaljena upravo u tom fragmentu. Programski jezik GLSL ima ugrađenu metodu `texture` koja ima varijantu za parametar kubne teksture `samplerCube` i omogućuje nam upravo to bez potrebe za dodatnim kodom.

Na slici 6.2 možemo vidjeti rezultat zrcaljenja kubne teksture okoliša na objektu kugle i čajnika.



Slika 6.2: Primjer okoliša zrcaljenog na kugli i čajniku

6.2. Ravnopravna projekcija

Ravnopravna projekcija (engl. equirectangular projection) je jednostavni oblik projekcije kugle na ravnu plohu. U implementaciji, teksture u toj projekciji su korištene za zapis teksture okoliša iz razloga što su jednostavnije za prijenos. Za razliku od kubnih tekstura koje zahtjevaju 6 različitih tekstura za jedan okoliš, one zahtjevaju samo jednu. Primjer teksture okoliša u ravnopravnoj projekciji je prikazan na slici 6.3.



Slika 6.3: Primjer okoliša prikazanog ravnopravnom projekcijom

Teksture u ravnopravnoj projekciji bi željeli koristiti na isti način na koji koristimo i kubne teksture, a to je uzorkovanjem pomoću normale fragmenta. Nažalost, GLSL nam ne pruža ovu funkcionalnost izravno te ćemo morati napisati vlastitu funkciju koja pretvara koordinate smjera normale u 2D koordinate teksture. Navedena funkcija je prikazana na slici 6.4. Bitno za napomenuti jest da je ovaj pristup sporiji za razliku od izravnog korištenja kubne teksture, stoga je preporučeno da se statičke teksture u ravnopravnoj projekciji prije prvog korištenja pretvore u kubne teksture.

```
#define PI 3.14159265358979323846
vec2 SampleEquirectangular(vec3 v) {
    vec2 uv = vec2(atan(v.z, v.x), asin(v.y));
    uv *= vec2(0.5 / PI, 1.0 / PI);
    uv += 0.5;
    return uv;
}
```

Slika 6.4: Funkcija u GLSL-u za uzorkovanje teksture u ravnopravnoj projekciji [2]

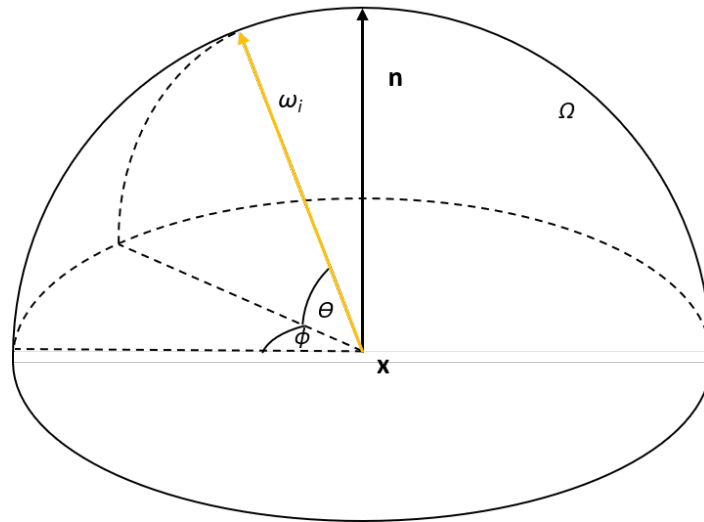
6.3. Difuzna obasjanost

Prisjetimo se jednadžbe reflektivnosti i Cook-Torranceove aproksimacije funkcije distribucije dvosmjerne refleksije. S obzirom da model definira funkciju kao zbroj međusobno neovisne difuzne i spekularne komponente pomnožene s vlastitim omjerom utjecaja, uvrštavanjem iste u jednadžbu reflektivnosti možemo rastaviti integralni izraz u dva podintegrala. Izraz za difuzni podintegral je prikazan u nastavku na slici 6.5.

$$L_d(\mathbf{x}, \omega_o, \lambda, t) = k_d \frac{c}{\pi} \int_{\Omega} L_i(\mathbf{x}, \omega_i, \lambda, t) (\omega_i \cdot \mathbf{n}) d\omega_i$$

Slika 6.5: Difuzni dio jednadžbe reflektivnosti za Cook-Torranceov model

Primjetimo da za svaki fragment integral difuzne komponente ovisi samo o ulaznom smjeru svjetlosti ω_i . To znači da možemo koristiti taj izraz da unaprijed izračunamo kubnu mapu integralnog dijela za neku sliku okoliša koju bismo koristili za računanje difuzne komponente. Prvo što moramo napraviti jest redefinirati integral pomoću stvarnih varijabli s kojima zapravo možemo računati.



Slika 6.6: Integriranje po svakoj ulaznoj zraki svjetlosti

Za obilazak čitavog prostora Ω , kao što je prikazano na slici 6.6, možemo se kretati po površini ravnine od kuta 0 do 2π , integrirajući za svaku vrijednost kuta θ ulazne zrake svjetlosti četvrtine kružnog isječka. Samim time skalarni umnožak jediničnih vektora ($\omega_i \cdot \mathbf{n}$) možemo izraziti pomoću $\cos(\frac{\pi}{2} - \theta)$, ili jednostavnije, $\sin \theta$. Također, za izračun ω_i pomoću kuteva možemo koristiti izraz $(\cos \phi \cos \theta, \sin \theta, -\sin \phi \cos \theta)$. Na slici 6.7 je prikazan konačan izraz.

$$L_d(\mathbf{x}, \omega_o, \lambda, t) = k_d \frac{c}{\pi} \int_{\phi=0}^{2\pi} \int_{\theta=0}^{\frac{\pi}{2}} L_i(\mathbf{x}, (\cos \phi \cos \theta, \sin \theta, -\sin \phi \cos \theta), \lambda, t) \sin \theta \, d\theta \, d\phi$$

Slika 6.7: Redefiniran difuzni dio jednadžbe reflektivnosti za Cook-Torranceov model

Pretvorbom integrala u Riemannove sume možemo aproksimirati kubnu mapu okoliša. Na primjeru kubne mape okoliša sa slike 6.8 ćemo dobiti kubnu mapu prikazanu na slici 6.9.



Slika 6.8: Kubna mapa okoliša



Slika 6.9: Kubna mapa difuzne obasjanosti

Vidimo da je kubna mapa difuzne obasjanosti dosta slična zamagljenoj ili zamućenoj slici. S obzirom da ljudsko oko ne zamjećuje sitne detalje, mogli bismo koristiti alternativne metode manje složenosti za izračun kubne mape koju ćemo koristiti kao mapu difuzne obasjanosti. Na slici 6.10 je prikazana kubna mapa okoliša na kojoj je provedeno Gaussovo zamućenje (engl. Gaussian blur). Slika izgleda dosta slična onoj koja je izračunata aproksimativnim postupkom. Alternativno tome, moguće je koristiti bilinearno uzorkovanje MIP mape niže rezolucije pozivom funkcije `textureLod`, kao što je prikazano na slici 6.11.



Slika 6.10: Kubna mapa dobivena Gausovim zamučivanjem



Slika 6.11: Kubna mapa dobivena bilinearnim uzorkovanjem

6.4. Spekularno osvjetljavanje

Preostalo je još razriješiti spekularni dio jednadžbe reflektivnosti. Nažalost, za razliku od difuznog dijela, ne možemo izdvojiti spekularnu komponentu izvan integrala jer njeni članovi ovise o više parametara osim samog ulaznog smjera svjetlosti ω_i . Na slici 6.12 je prikazan spekularni dio jednadžbe reflektivnosti.

$$L_s(\mathbf{x}, \omega_o, \lambda, t) = k_s \int_{\Omega} \frac{DFG}{4(\omega_o \cdot \mathbf{n})} L_i(\mathbf{x}, \omega_i, \lambda, t) d\omega_i$$

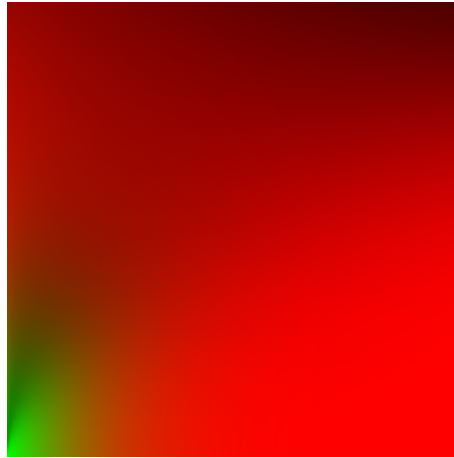
Slika 6.12: Spekularni dio jednadžbe reflektivnosti za Cook-Torranceov model

Pogoni kao što su Unreal Engine to razrješavaju tako da rastavljaju umnožak unutar integrala u konvoluciju dva integrala. Na slici 6.13 vidimo razdvojeni izraz. Integralni dio sa ulaznom svjetlosti L_i se rješava izračunom kubne mape slične onoj za difuznu obasjanost, no ovaj put trebamo uključiti i faktor gruboće. Za veći faktor gruboće vektori smjera su raspršeniji rezultirajući zamućenijim rezultatom. Koristeći funkciju distribucije normala s pretpostavkom da je vektor pogleda \hat{v} jednak vektoru smjera izlazne svjetlosti ω_o trebamo generirati nekoliko razina kubnih mapa, te one sa većom gruboćom možemo spremiti, umjesto u zasebnu teksturu, u razinu MIP mape s manjom rezolucijom. Na taj način, uzorkovanje trilinearne teksture možemo obavljati `textureLod` naredbom parametrizirajući razinu detalja (engl. level of detail) sa skaliranim faktorom gruboće. Naravno, moguće je aproksimirati isto i na jednostavniji način koristeći automatski generirane razine MIP mape kubne teksture okoliša, što će rezultirati manje kvalitetnim rezultatom.

$$L_s(\mathbf{x}, \omega_o, \lambda, t) = k_s \int_{\Omega} \frac{DFG}{4(\omega_o \cdot \mathbf{n})} d\omega_i * \int_{\Omega} L_i(\mathbf{x}, \omega_i, \lambda, t) d\omega_i$$

Slika 6.13: Razdvojeni spekularni dio jednadžbe reflektivnosti za Cook-Torranceov model

Dio integrala koji sadrži funkcija distribucije dvosmjerne refleksije se aproksimira principom razdvojenih suma (engl. split-sum approximation). Princip uključuje razdvajanje integrala uvrštavanjem Fresnelovog koeficijenta na sumu dva integrala. U sljedećem koraku je potrebno izračunati teksturu pregledne tablice (eng. look-up table, LUT) kojoj je na osi u postavljen faktor gruboće, a na osi v postavljen kosinus kuta između normale površine \mathbf{n} i vektora pogleda \hat{v} . Postupak je sličan prethodnom, a rezultatna tekstura bi trebala izgledati kao u nastavku na slici 6.14.



Slika 6.14: Tekstura pregledne tablice

7. Kvaliteta slike

7.1. Slike visokog dinamičkog raspona

Slike visokog dinamičkog raspona (engl. high-dynamic-range) su slike koje sadrže više informacija po pikselu nego uobičajenih 24 bita. Obično sadrže 48 bitova (16 bita po kanalu), ali moguće je i više. Razlog tomu jest što se onda slike dinamički mogu prilagoditi razini izlaganja zračenju (engl. exposure) za razliku od običnih slika koje pamte samo 8 bitova po kanalu piksela. U nastavku na slici 7.1 je prikazana ista slika s različito postavljenim razinama izlaganja zračenju.



Slika 7.1: Slika visokog dinamičkog raspona po različitim razinama izlaganja zračenju

7.2. Mapiranje tona slike

Mapiranje tona (engl. tone mapping) je postupak transformiranja boje numeričke vrijednosti s pomičnim zarezom iz visokog dinamičkog raspona u niski dinamički raspon $[0, 1]$ bez gubitka previše detalja. Postoji mnogo različitih metoda za postizanje mapiranja tona, a u nastavku na slici 7.2 je navedena Reinhardova metoda [2] dok na slici 7.3 je navedena metoda temeljena na razini izlaganja zračenju [2].

```
vec3 hdr_color = texture(hdr_texture , uv_coords).rgb;  
vec3 mapped = hdr_color / (hdr_color + vec3(1.0));
```

Slika 7.2: Reinhardova metoda mapiranja tona u jeziku GLSL

```
float exposure = 1.0; // varirajuća varijabla  
vec3 hdr_color = texture(hdr_texture , uv_coords).rgb;  
vec3 mapped = vec3(1.0) - exp(-hdr_color * exposure);
```

Slika 7.3: Metoda mapiranja tona temeljena na razini izlaganja zračenju u jeziku GLSL

7.3. Gama ispravljanje

Gama ispravljanje (engl. gamma correction) je postupak u kojemu se boja iz linearnog prostora (poznatiji kao RGB prostor) transformira u prostor boje monitora (poznatiji kao sRGB prostor) [2]. Na slici 7.4 je prikazan kod u GLSL-u za provođenje gama ispravljanja nad bojom.

```
const float gamma = 2.2;  
vec3 linear_color = texture(regular_texture , uv_coords).rgb;  
vec3 gamma_corrected = pow(linear_color , vec3(1.0 / gamma));
```

Slika 7.4: Gamma ispravljanje u jeziku GLSL

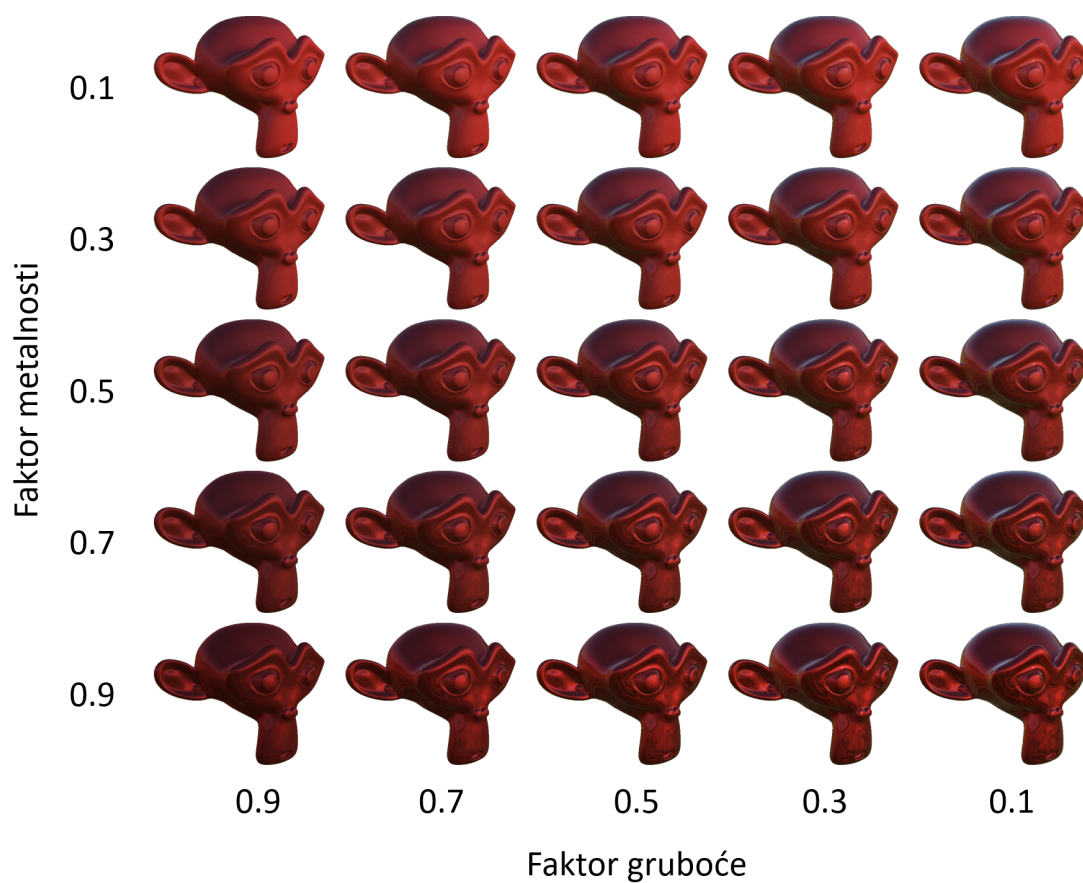
8. Programsko rješenje



Slika 8.1: Prikaz prozora programskog rješenja

Na slici 8.1 je prikazan isječak iz prozora programskog rješenja¹. Lijevi okvir omogućuje odabir modela te kontrolu albedo materijala, faktora ili teksture metalnosti, faktora ili teksture gruboće i faktora ili teksture okluzije ambijenta. Postavke za neke od materijala su unaprijed definirane, od kojih je trenutno odabrano zlato. Navigacija kamere je omogućena pomoću kombinacija tipki WASD i miša. Desni okvir omogućuje promjenu scene (okoliša), postavki izvora svjetla, razine izlaganja zračenju i konstante gama ispravljanja. Također se mogu samostalno prikazati različite komponente koje doprinose konačnom prikazu. Na slici 8.2 je prikazana usporedba modela majmuna za različite kombinacije faktora gruboće i metalnosti.

¹Programsko rješenje je dostupno na https://gitlab.com/__fastcall/pbr-explore repozitoriju



Slika 8.2: Usporedba modela majmuna pri različitim faktorima gruboće i metalnosti

9. Zaključak

Jedan od najdugoročnijih ciljeva računalne grafike je ostvarivanje fotorealističnog prikaza u stvarnom vremenu. Kroz godine se osmislilo mnogo modela i principa ostvarivanja prikaza koja su bila kompromis između kompleksnosti izračuna i kvalitete rezultata. U jedno od trenutno najmodernijih modela prikaza spada i fizikalno temeljeno ostvarivanje prikaza. Ideja tog modela jest da implementira principe iz stvarnog svijeta koji slijede pravila fizike, ili konkretnije, optike.

U ovom završnom radu je uveden fizikalno temeljen model ostvarivanja prikaza temeljen na Cook-Torranceovom modelu. Objašnjen je model mikrozrcala kojim se može opisati površina bilo kojega materijala. Objašnjena je uloga funkcije distribucije dvosmjerne refleksije te je predstavljena Cook-Torranceova aproksimacija te funkcije. Komponente koje sastavljaju funkciju su opisane, uključujući funkciju distribucije normala, Fresnelov koeficijent i geometrijsku funkciju. Načini zrcaljenja okoliša na površini objekta su uvedeni, uključujući i metodu osvjetljavanja temeljenog na slici.

LITERATURA

- Marco Alamia. *Coding Labs - Physically Based Rendering - Cook-Torrance*.
URL http://www.codinglabs.net/article_physically_based_rendering_cook_torrance.aspx. [1].
- Joey de Vries. *LearnOpenGL - PBR Tutorials*. URL <https://learnopengl.com/PBR/Theory>. [2].
- Naty Hoffman. *Physically-based shading models in film and game production*.
Technical report, ACM SIGGRAPH, 2010. URL https://renderwonk.com/publications/s2010-shading-course/hoffman/s2010_physically_based_shading_hoffman_a_notes.pdf. [3].
- Brian Karis. *Graphic Rants - Specular BRDF Reference*.
URL <http://graphicrants.blogspot.com/2013/08/specular-brdf-reference.html>. [4].
- OpenGL documentation*. Khronos Group. URL <https://www.opengl.org/documentation>. [5].

Fizikalno temeljen model ostvarivanja prikaza

Sažetak

U ovom završnom radu prikazane su osnove fizikalno temeljenog modela ostvarivanja prikaza pomoću Cook-Torranceovog modela. Napravljen je kratki uvod u OpenGL grafičko programsko sučelje zajedno s načinom rada sa sjenčarima. Uveden je model mikrozrcala kao temeljni model koji opisuje površinu bilo kojeg materijala, te je objašnjena općenita jednadžba ostvarivanja prikaza. Opisana je uloga funkcije distribucije dvosmjerne refleksije te je uveden Cook-Torranceov model kao najčešći model njene aproksimacije. Objasnjena je ideja funkcije distribucije normala, Fresnelovog koeficijenta te geometrijske funkcije. Prikazani su načini zrcaljenja okoliša i metode osvjetljavanja temeljenog na slici.

Ključne riječi: fizikalno temeljen model ostvarivanja prikaza, Cook-Torranceov model, model mikrozrcala, funkcija distribucije dvosmjerne refleksije, funkcija distribucije normala, Fresnelov koeficijent, geometrijska funkcija, zrcaljenje okoliša, osvjetljavanje temeljeno na slici, OpenGL, GLSL

Physically based rendering model

Abstract

In this thesis physically based rendering method based on the Cook-Torrance equation is explained. A short introduction to OpenGL graphical API is shown alongside with the shader workflow. The microfacet theory is introduced as a basis for describing any material surface and the general rendering equation is explained. The role of bidirectional reflectance distribution function is clarified and the Cook-Torrance equation is introduced as its most common form. The idea of normal distribution function, Fresnel equation and geometry function is explained. Some methods of environment reflection mapping and image based lighting are shown.

Keywords: physically based rendering model, Cook-Torrance equation, microfacet model, bidirectional reflectance distribution function, normal distribution function, Fresnel equation, geometry function, environment reflection mapping, image based lighting, OpenGL, GLSL