Osnove virtualnih okruženja

Laboratorijske vježbe

Vježba 2

Principi iscrtavanja: Praćenje zrake

FER – ZTE – Igor S. Pandžić
Suradnja na pripremi vježbe:
Milenka Gadže
Miran Mošmondor
Vjekoslav Levačić
Marko Pipal

1. Uvod

Praćenje zrake (engl. *ray tracing*) je klasična i izuzetno često upotrebljavana metoda iscrtavanja 3D scena. Metoda je zasnovana na zakonima optike (refleksija, refrakcija) uz korištenje pojednostavljenog modela osvjetljenja za izračunavanje intenziteta svjetlosti u određenoj točki.

Metodom praćenja zrake dobivaju se vrlo impresivne slike 3D scena koje uključuju efekte zrcaljenja, prozirnosti i sjena. Pritom je osnovna metoda vrlo jednostavna za implementaciju jer se sastoji od rekurzivnog ponavljanja nekoliko osnovnih operacija.

Cilj ove vježbe je implementirati osnovnu metodu praćenja zrake, te kroz to steći osnovne ideje o iscrtavanju slike 3D scene, modelu osvjetljenja i efektima sjene, zrcaljenja i prozirnosti.

2. Alati potrebni za izvođenje vježbe

Za izvođenje vježbe potrebna vam je Java ili C#, te alat za pisanje Java programa (Borland JBuilder, Eclipse ili običan tekst editor), odnosno alat za pisanje C# programa (Visual Studio ili običan uređivač teksta)

3. Teorijska podloga

Praćenje zrake je obrađeno na predavanju o iscrtavanju. Postupak praćenja zrake je jednostavan: za svaku točku ekrana, prati se zraka koja kroz tu točku ulazi u scenu. Za zraku se traži presjek sa predmetima u sceni. Ako se presjek nađe, računa se osvjetljenje u točki presjeka, te se računaju zrcaljena zraka i refraktirana zraka. Za ove dvije zrake postupak se rekurzivno ponavlja, te se zbrajaju doprinosi osvjetljenja svih nađenih točaka presjeka.

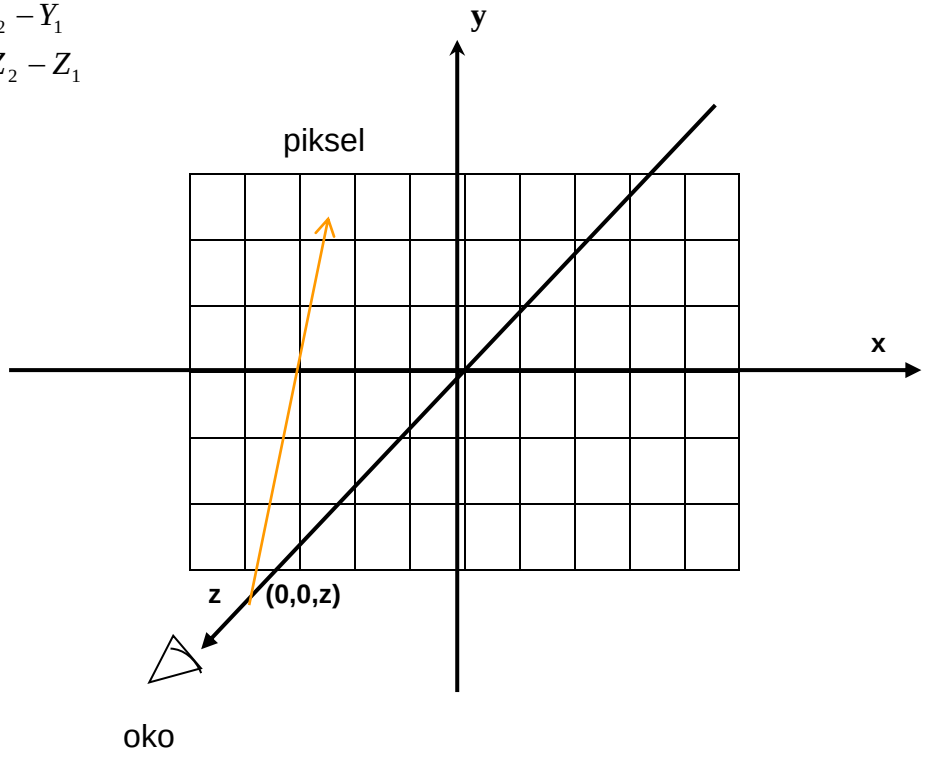
Pseudo-kod za postupak izgleda ovako:

```
za svaki x,y u prozoru iscrtavanja
     izračunaj zraku R kroz x,y
     boja C = TraceRay(R,0)
     obojaj točku x,y bojom C
kraj

funkcija TraceRay(R,dubina)
     ako je dubina>max. dubine, prekini funkciju i vrati crnu boju
     nađi najbliži presjek zrake R sa scenom
     ako nema presjeka, prekini funkciju i vrati kao rezultat boju pozadine
     izračunaj boju lokalnog osvjetljenja Clocal u točki presjeka
     izračunaj odbijenu zraku Rrefl
     boja Crefl = TraceRay(Rrefl, dubina+1)
     izračunaj refraktiranu zraku Rrefr
     boja Crefr = TraceRay(Rrefr, dubina+1)
     boja C = kombinacija(Clocal, Crefl,Crefr)
kraj: vrati boju C
```

3.1 Računanje zrake kroz točku x,y u prozoru iscrtavanja

Zraku promatramo kao pravac kroz dvije točke u trodimenzionalnom xyz-sustavu. Najbolje ju je predstaviti hvatištem i smjerom. Ekran za iscrtavanje postavljamo u xy-ravninu tako da točka (0,0,0) bude u središtu ekrana. Početna točka koja definira zraku je oko promatrača koje se nalazi na unaprijed odredjenoj i fiksiranoj poziciji (0,0,z). Druga točka koja definira zraku je piksel na ekranu za iscrtavanje. Položaj oka se uzima za hvatište zrake, a smjer zrake će biti normalizirani vektor od oka promatrača do piksela na ekranu i računa se kao smjer pravca kroz dvije točke (oko – točka 1, piksel – točka 2).

$$X_{SMJER} = X_2 - X_1$$
$$Y_{SMJER} = Y_2 - Y_1$$
$$Z_{SMJER} = Z_2 - Z_1$$



3.2 Testiranje dubine rekurzije

Dubinu rekurzije kod zbrajanja intenziteta možemo zadati kao konstantu (tj. da se funkcija *TraceRay* rekurzivno zove točno određen broj puta) ili je ograničiti tako da zadamo minimalni intenzitet koji ima smisla uračunati u ukupno lokalno osvjetljenje. Ako je izračunati intenzitet u rekurziji manji od zadanog onda prekidamo postupak rekurzije jer je utjecaj tog intenziteta zanemariv. U ovoj vježbi implementirati ćemo prvi, jednostavniji način, dakle konstantnu dubinu rekurzije.

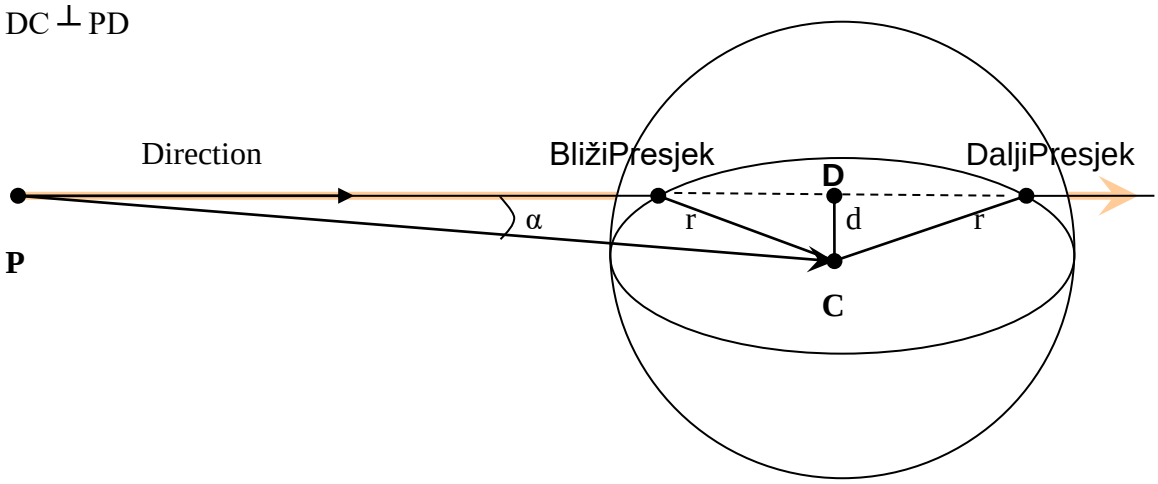
3.3 Nalaženje najbližeg presjeka zrake sa scenom

Jedini način da se odredi najbliži presjek zrake sa scenom je da se ispita i odredi presjek zrake sa svakim od objekata u sceni i uzme onaj čija je udaljenost od početka zrake najmanja.

U ovoj vježbi radit ćemo sa kuglama. Za zraku su poznati hvatište P i vektor smjera *Direction*, a za svaku kuglu su poznate koordinate središta C i radijus r.

Algoritam za nalaženje presjeka s kuglom:

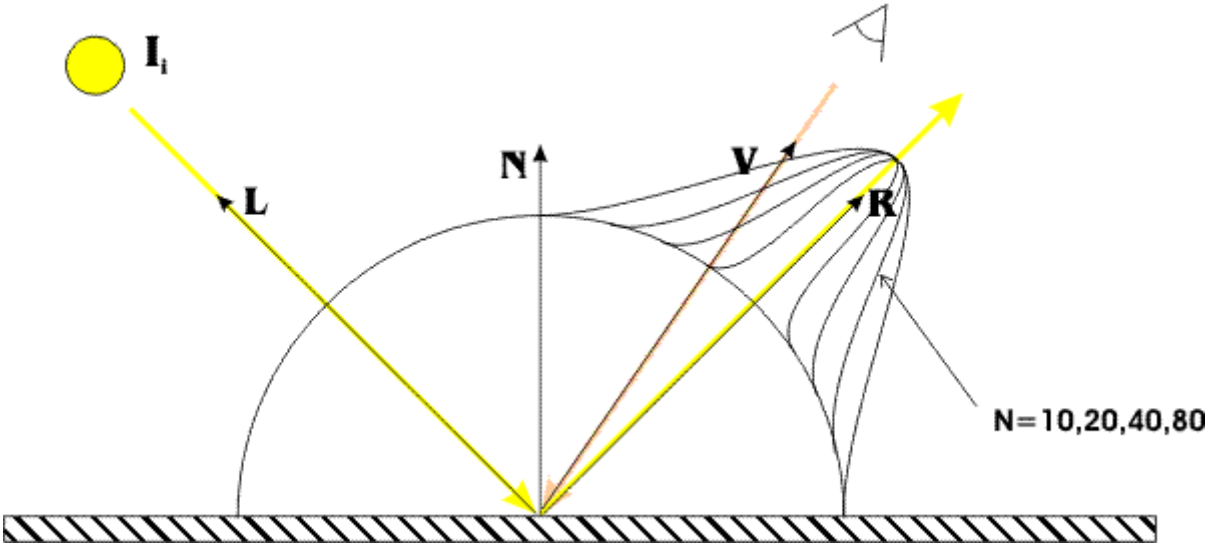
```
funkcija presjekPostoji(zraka)
     izračunaj vektor PC
     izračunaj kut α između vektora smjera zrake Direction i vektora PC
     ako je kut α veći od 90 stupnjeva zraka ne pokazuje u smjeru objekta
          pa prekini funkciju i vrati false
     nađi udaljenost d zrake od središta kugle ( d = |PC| * sin(α) )
     ako je d veće od radijusa kugle r prekini funkciju i vrati false
     odredi udaljenost hvatišta zrake P od točke D
          |PD| = √(|PC|² − d²)
     odredi udaljenost hvatišta zrake P od bližeg presjeka
          |PBližiPresjek| = |PD| − (√(r² − d²))

     ako je |PBližiPresjek| <= 0 + Epsilon[1] tada
          hvatiste se nalazi unutar kugle i vrijedi
          |PBližiPresjek|= |PDaljiPresjek|= |PD| + (√(r² − d²))

kraj: vrati true

funkcija vratiPresjek()
kraj: vrati BližiPresjek
```



3.4 Računanje lokalnog osvjetljenja u točki presjeka

Kad pronađemo presjek zrake i predmeta u sceni potrebno je dotičnom pikselu pripisati intenzitet koji se računa se prema Phongovom modelu odbijanja svjetlosti.

[1] Zbog ograničenosti računala u prikazu realnih brojeva u nekim nam slučajevima dolazi do toga da je desna strana izraza veća od nule unatoč tome što se hvatište nalazi unutar kugle. Stoga koristimo korekcijski faktor Epsilon koji mora biti dovoljno malen da izraz zadrži smisao, a obuhvati takve slučajeve.



Potrebno je razlikovati pojam zrake kao **zraka svjetlosti** (usmjerena suprotno vektoru L) i zraku koja **označava smjer našeg pogleda** (usmjerena suprotno vektoru V).

$$I = I_a k_a + I_i k_d (L \bullet N) + I_i k_s (R \bullet V)^n$$

Svi intenziteti I i koeficijenti k su zapravo vektori boje koji se sastoje od tri komponente: R, G, B – dakle crvena, zelena i plava. **Sve komponente poprimaju vrijednosti od 0 do 1**. Ukoliko neke od R, G, B vrijednosti premašuju granice zaokružite ih na maksimalnu, odnosno minimalnu vrijednost.

3.4.1 Ambijentna komponenta ($I_a k_a$)

Ambijentno svjetlo grubo aproksimira globalno osvjetljanje I daje minimum osvjetljenja kojim se izbjegava da dijelovi scene na koje svjetlo ne pada direktno budu potpuno crni, no koristi se u vrlo malim količinama jer daje efekt jednolične boje. Ambijentna svjetlost karakterizirana je intenzitetom Ia koji je konstanta za cijelu scenu, te ambijentnim koeficijentom materijala ka, koji određuje u kojoj mjeri materijal reagira na ambijentnu svjetlost.

3.4.2 Difuzna komponenta ($I_i k_d (L \bullet N)$)

Difuzna komponenta vjerno opisuje Lambertov zakon koji opisuje difuzno raspršivanje svjetla na predmetu. Difuzno odbijanje (raspršivanje) ide u svim smjerovima jednakim intenzitetom, a taj intenzitet ovisi o ulaznom kutu, te je najveći kada zraka upada okomito na površinu. Ova zakonitost je opisana skalarnim produktom vektora smjera upadne zrake L i normale na površinu N. Difuzna komponenta je uz to proporcionalna intenzitetu izvora svjetlosti Ii i difuznom koeficijentu materijala kd.

3.4.3 Spekularna komponenta ($I_i k_s (R \bullet V)^n$)

Spekularna komponenta aproksimira spekularni odsjaj na predmetu. Spekularni odsjaj je karakteriziran specifičnom krivuljom prikazanom na slici, koja ima oštar maksimum u smjeru

zrcaljenja R (R je simetričan L u odnosu na N). Što je materijal sjajniji to je ovaj maksimum više izražen, tj. krivulja je strmija. U prirodi se to manifestira oštrinom spekularnog odsjaja – što je materijal sjajniji, to je na njemu odsjaj svjetlosti oštriji. Ova karakteristika aproksimirana je prilično vjerno skalarnim produktom smjera gledanja V i smjera zrcaljenja R, te potenciranjem tog produkta faktorom n koji opisuje sjajnost materijala. Što je n veći to je materijal sjajniji. Spekularna komponenta je proporcionalna intenzitetu izvora svjetlosti Ii i spekularnom koeficijentu materijala ks.

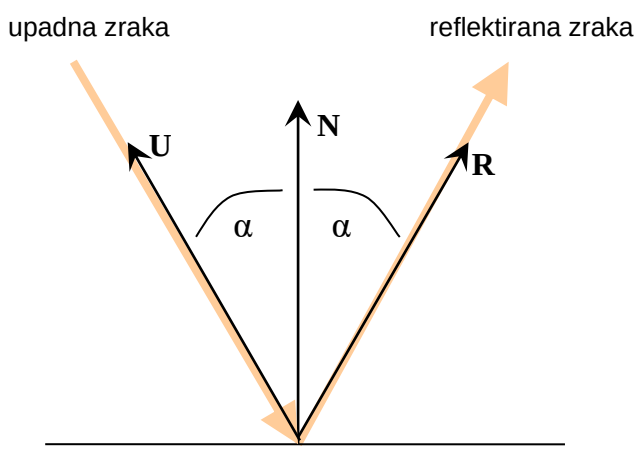
3.4.4 Utjecaj sjene

U svakoj točki presjeka zrake i predmeta u sceni računa se doprinos osvjetljenja pomoću modela lokalnog osvjetljenja, npr. Phongovim modelom. Model lokalnog osvjetljenja ne računa efekt sjene. Dakle, upotrijebimo li samo Phong model osvjetljenja, potpuno ćemo zanemariti činjenicu da se npr. između izvora svjetlosti i promatrane točke može naći još jedan predmet tako da ovaj izvor svjetlosti uopće ne baca svjetlo na našu promatranu točku pa su difuzne i spekularne komponente lokalnog osvjetljenja jednake nuli!

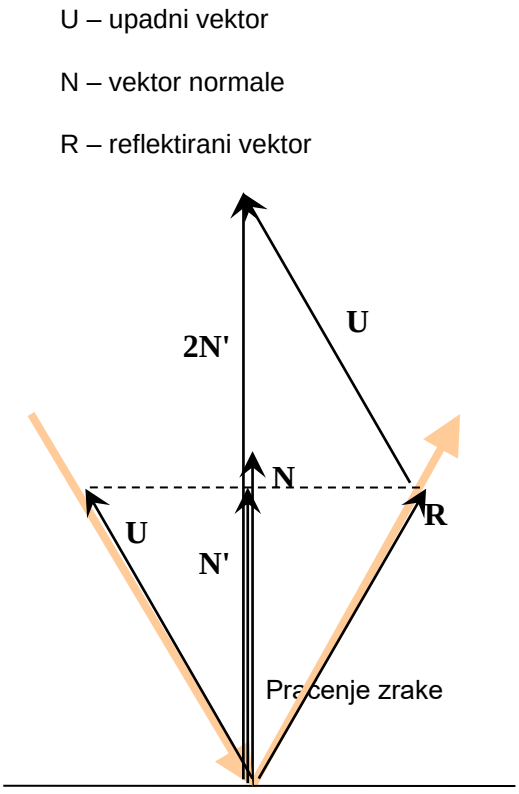
Ovaj problem se rješava zrakama za ispitivanje sjene (engl shadow ray, shadow feeler). Te zrake idu od mjesta presjeka do pojedinog izvora svjetlosti i provjeravaju nalazi li im se išta na putu tj. postoji li presjek sa scenom. U slučaju da postoji presjek zrake za ispitivanje sjene sa nekim objektom na njenom putu utjecaj **lokalnog osvjetljenja ne uračunava difuznu i spekularnu komponentu**. Naravno pri takvom postupku zanemarujemo činjenicu da objekti koji stoje na putu između promatrane točke i izvora svjetlosti mogu biti prozirni. Iako u ovoj vježbi to nije potrebno implementirati, ovaj se problem riješava na način da se promatranoj točki pripisuje jakost izvora svjetlosti I_i pomnožena sa faktorima prozirnosti tijela koja se nalaze na putu zrake.

Ovaj postupak je praktičan jer se za traženje presjeka sa zrakom za ispitivanje sjene koriste potpuno iste metode kao i za presjek sa ostalim zrakama u postupku, dakle uvođenjem sjena nismo bitno povećali složenost implementacije (naravno, složenost u smislu vremena zvođenja metode se povećava).

3.5 Računanje odbijene zrake



Kod računanja odbijene zrake potrebno je najprije odrediti vektor normale *N* na površinu kugle u točki



presjeka zrake s kuglom te upadni vektor *U* koji je suprotan vektoru smjera upadne zrake (slika).

Sada se može odrediti reflektirani vektor. Iz slike se vidi da je:

$$R = 2N' - U,$$

gdje je

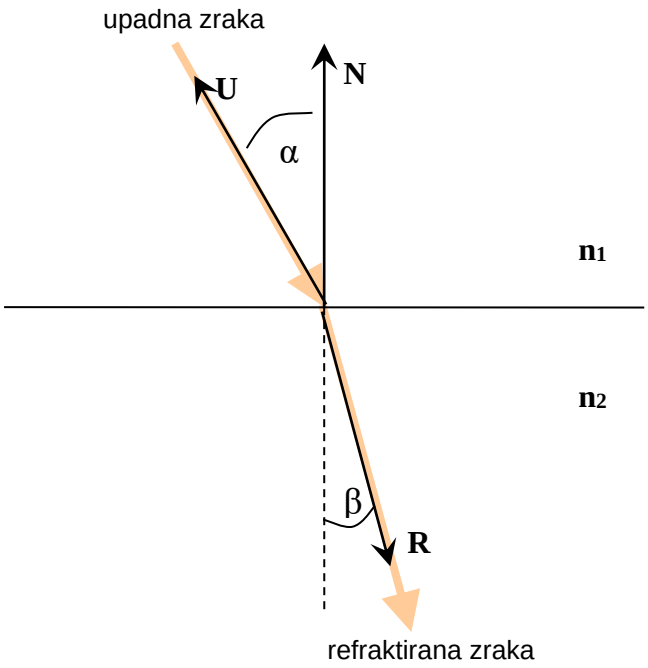
$$N' = (U \bullet N)N,$$

a (U•N) skalarni produkt upadnog vektora i vektora normale.

Dakle, reflektirani vektor oko normale N na površinu kugle u točki presjeka zrake s kuglom može se izračunati pomoću formule :

$$R = (2U \bullet N)N - U.$$

3.6 Računanje refraktirane zrake



U – upadni vektor

N – vektor normale

R – refraktirani vektor

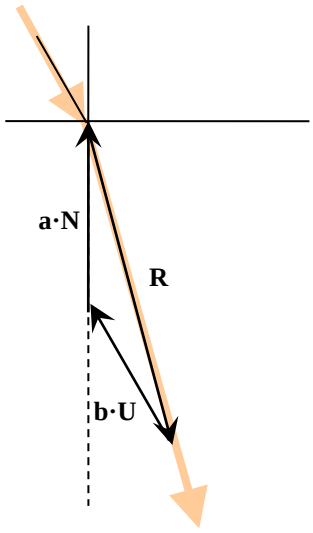
Kut reftraktirane zrake β se može jednostavno odrediti pomoću Snellovog zakona :

$$n_1 \cdot \sin(\alpha) = n_2 \cdot \sin(\beta)$$

gdje su n_1 i n_2 indeksi refrakcije sredstva iz kojeg dolazi zraka, odnosno sredstva nad kojim se lomi zraka.

Kod određivanja vektora smjera refraktirane zrake u prostoru polazi se od činjenice da je refraktirani vektor *R* kombinacija upadnog vektora *U* i vektora normale *N* (slika) :

$$R = - a \cdot N - b \cdot U$$



Važno je da su upadni vektor i vektor normale normalizirani. Sada je potrebno odrediti samo još koeficijente *a* i *b*.

Vrijedi da je :

$$b = \frac{n_1}{n_2} = ni \text{ (indeks loma)},$$

$$a = (-2 * b * \cos\alpha + \sqrt{D})/2,$$

gdje je

$$D = 4 * (b^2 * (\cos\alpha)^2 - b^2 + 1).$$

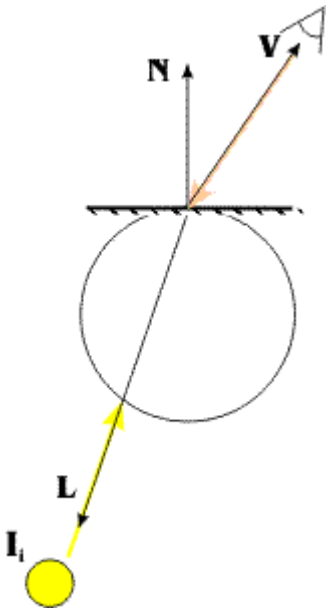
3.7 Kombiniranje lokalnog osvjetljenja i doprinosa odbijene i refraktirane zrake

Nakon što se odredi lokalno osvjetljenje, te doprinosi odbijene i refraktirane zrake potrebno ih je zbrojiti da se dobije konačno osvjetljenje u točki presjeka zrake sa scenom. Međutim, različiti objekti različito kombiniraju te doprinose. Na primjer, kod neprozirnog objekta udio doprinosa refraktirane zrake je jednak 0. Zbog toga se lokalna osvjetljenje (I_{illum}) i doprinos odbijene (I_{refl}) i refraktirane zrake (I_{refr}) kombiniraju na slijedeći način :

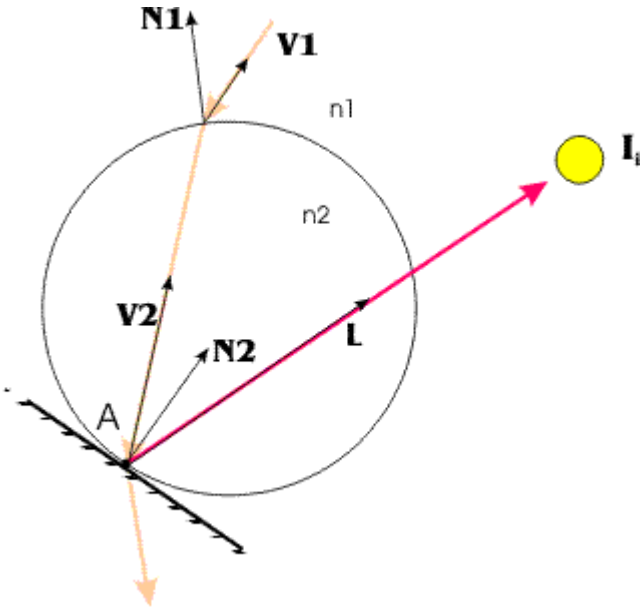
$$I = I_{illum} + k_1 I_{refl} + k_2 I_{refr}$$

gdje su k_1 i k_2 koeficijenti od 0 do 1 i predstavljaju udio odbijene, odnosno refraktirane zrake i karakteristike su materijala.

3.8 Dodatne napomene



Slika 1.



Slika 2.

Pogledajmo zanimljiv slučaj na slici 1. kada izvor svjetlosti ne baca svjetlo na promatranu točku. Očito je da tada vrijedi da je skalarni umnožak L i N manji od nule. U ovom slučaju ne računamo utjecaj difuzne i spekularne komponente.

Za bilo koju plohu, pa tako i za plašt kugle vrijedi da je vektor normale određen okomicom na plohu. Odabran smjer normale u općem je slučaju proizvoljan, tj. može pokazivati na jednu ili drugu stranu plohe, ali u našoj primjeni je važan. Potrebno je uvijek odabrati takav smjer normale da normala pokazuje na onu stranu s koje dolazi zraka. Funkcija *getNormal* uvijek vraća normalu koja pokazuje u smjeru od središta prema sjecištu. U slučaju da je skalarni umnožak V i N manji od nule potrebno je normali okrenuti smjer množenjem sa -1. Na slici 2. vidi se normala N2 koja je nastala okretanjem smjera. Obratite pozornost da se mijenja omjer n1 i n2 pri izlasku zrake iz kugle.

4. Opis zadatka

Potrebno je napraviti metodu praćenje zrake u programskim jezicima Java ili C#. Položaj oka te ekrana za iscrtavanje potrebno je fiksno odabrati. Pretpostaviti da se oko nalazi na poziciji (0,0,2) u koordinatnom sustavu, a ekran za iscrtavanje u xy-ravnini. U scenu je potrebno postaviti kugle te definirati sve parametre svake kugle (koeficijenti difuzne, ambijentne i spekularne komponente, indeks loma, udio odbijene i rekraktirane zrake), utjecaj sjena, zakona refleksije, refrakcije, disperzije te parametre izvora svjetlosti (ukupni intenzitet - RGB). Potrebno je rasporediti kugle tako da čine početno slovo vašeg prezimena i iscrtati sliku.

5. Upute za rad

Na stranicama predmeta pronaći ćete dvije zip datoteke, koje sadrže kosture koda jednog mogućeg rješenja napisane u Javi i C#. Također, rješenje napisano u Javi sadrži *Javadoc* projekta, te projekt možete otvoriti JBuilderom, a rješenje napisano u C# sadrži *VS solution*. U rješenju su opisane sve potrebne klase i pripadajuće metode. Ukoliko radite u Javi, project možete pokrenuti i u Eclipse-u tako da cijeli *src* folder kopirate u novi projekt, a *input.txt* kopirate u direktorij projekta. Ukoliko radite u C#, potrebno je datoteku *input*.txt kopirati u direktorij *Debug* koji se nalazi u direktoriju *bin*. Od vas se traži da implementirate metode objašnjene u ovim uputama, a koje u ponuđenom kosturu projekta nedostaju.

Svi parametri potrebni za crtanje slike (veličina i rezolucija ekrana, položaj svijetla, broj kugli u sceni te parametre svake kugle) se postavljaju čitanjem iz tekstualne datoteke, što je opisano u sljedećem poglavlju. Ovaj dio je već implementiran u rješenju.

5.1 Unos parametara

Najjednostavniji način unošenja svih potrebnih parametara je čitanjem iz tekstualne datoteke. Za to se može iskoristiti gotova klasa *FileReader*. Ona iz zadane tekstualne datoteke čita sve podatke koji su razdvojeni zarezom ili dvotočkom. Sve što se nalazi unutar zagrada <> se ignorira. Dakle, važan je samo redoslijed podataka u datoteci pa na to treba paziti.

Input.txt je primjer tekstualne datoteke iz koje se čitaju parametri. U njoj je zadana veličina prozora te rezolucija. Na primjer parametri <velicina prozora => 2: i <rezolucija prozora => 400: označavaju da je rezolucija ekrana 400*400 piksela, a da je prozor velik 2 mjerne jedinice u kooridnatnom sustavu.

FileReader koristi pomoćnu klasu *SphereParameters* koja nam služi kao struktura podataka u kojoj se nalaze svi parametri kugle.

5.2 Glavne klase i metode

Glavna petlja metode iscrtavanja praćenjem zrake, koja odgovara pseudo kodu na početku poglavlja 3, nalazi se u u klasi *Picture* u metodi *Paint*. Ovdje se za svaki pixel slike pomoću metode praćenja zrake odredi njegovu boju i tako se slika iscratva pixel po pixel.

Metoda *getPoint* koja za zadani pixel slike vraća 3D koordinate točke na ekranu kroz koju gledamo treba biti implementirana u klasi *Screen* (npr. za pixel -200,-200 treba vratiti

koordinate (-1,-1,0)). Zatim se konstruktorom klase Ray kreira zraka kroz tu točku. Ovaj dio postupka je opisan u poglavlju 3.1.

Za svaku zraku se zatim poziva glavna funkcija praćenja zrake *traceRay* u klasi *Scene*. Ova funkcija odgovara pseudokodu na početku poglavlja 3 (za iscrtavanje siluete potrebno je implementirati prva tri koraka pseudokoda iz 3. Poglavlja funkcije *traceRay*).

Potrebno je pronaći presjek zrake sa scenom (poglavlje 3.3). Metodu za presjek implementirajte u klasi *Sphere* koja nasljeđuje apstraktnu klasu *Object* i objektu daje karakteristiku kugle. Prije same izrade implementirajte računanje normale kugle u Sphere.java metoda *getNormal*. Također, unutar *ColorVector* klase u metodi *correct* implementirajte provjeru i ispravljanje R, G, B granica boje. U klasi *Object* nalaze se sve metode i svojstva karakteristična za sve objekte u sceni. Nakon što ste implementirali metodu presjeka, već možete iscrtati prvu sliku. U njoj će se vidjeti siluete scene. Sliku iscrtajte tako da pixele čije zrake sijeku scenu obojate, a ostalo ostavite crno. Ovako iscrtanu sliku siluete kugli potrebno je priložiti u izvještaju projekta.

Nakon ovoga izračunajte u metodi *traceRay* u klasi *Scene* komponente lokalnog osvjetljenja (intenziteti su tipa *ColorVector*, a koeficijenti tipa *PropertyVector*) prema priloženim formulama u poglavlju 3.4.1, 3.4.2, 3.4.3. Prvo implementirajte ambijentnu, zatim difuznu i na kraju spekularnu komponentu. Nakon dodavanja svake komponente iscrtajte sliku i priložite izvještaju projekta.

Sljedeći korak je utjecaj sjene. Na mjestu sjene difuzna i spekularna komponenta će biti jednake 0, a implementaciju metode *Shadow* izvedite u klasi *Scene* prema uputama u poglavlju 3.4.4. Ponovo iscrtajte sliku i priložite izvještaju projekta.

Sljedeći korak su reflektirane zrake. Njih implementirajte u klasi *Vector* uz pomoć formula iz poglavlja 3.5, te ih uvrstite rekurzivno u metodu *traceRay* kao što je opisano u pseudo-kodu (pripaziti na dubinu rekurzije, tj. njeno prekidanje nakon određene dubine). Zbrajanje doprinosa reflektirane zrake i lokalnog osvjetljenja objašnjeno je u poglavlju 3.7. Ponovo iscrtajte sliku i priložite izvještaju projekta.

Slijedeći korak su refraktirane zrake. Njih implementirajte u klasi *Vector* uz pomoć formula iz poglavlja 3.6, te ih uvrstite rekurzivno u metodu *traceRay* kao što je opisano u pseudo-kodu. Zbrajanje doprinosa refraktirane zrake i lokalnog osvjetljenja objašnjeno je u poglavlju 3.7.

Iscrtajte konačnu sliku sa svim komponentama i priložite izvještaju projekta.

Na kraju, potrebno je u datoteci *input.txt* posložiti veći broj kugli na takav način da tvore prvo slovo vašeg prezimena, te iscrtati sliku slova i uvrstiti je u izvještaj.

6. Predavanje rezultata vježbe

Rezultati vježbe se predaju zapakirani u arhivu OVO-V2- Rezultati-<ImePrezime>.zip koja treba sadržavati:

- Izvještaj o izvođenju vježbe koji treba sadržavati:
    - Kratak opis postupka iscrtavanja (max. jedna stranica)

- Iscrtane slike svih faza iscrtavanja prema uputama za izradu vježbe: (1) slika siluete scene (samo presjek), (2) slika s ambijentnom komponentom, (3) slika s ambijentnom i difuznom komponentom, (4) slika s ambijentnom, difuznom i spekularnom komponentom (puno lokalno osvjetljenje), (5) slika s lokalnim osvjetljenjem i sjenama, (6) slika s lokalnim osvjetljenjem, sjenama i reflektiranim zrakama, te (7) potpuna slika: lokalno osvjetljenje, refleksija, refrakcija i sjene. Svaku sliku treba komentirati, tj. objasniti razlike u odnosu na prethodnu sliku i zašto one nastaju.

- Na kraju, potrebno je u datoteci *input.txt* posložiti veći broj kugli na takav način da tvore prvo slovo vašeg prezimena, te iscrtati (8) sliku slova i uvrstiti je u izvještaj.

- Dobro komentirani izvorni kod zadatka (potrebno je dobro objasniti što se radi u svakom dijelu dokumenta koji ste sami dodavali), te izvršnu datoteku.

Navedena arhiva treba biti predana korištenjem aplikacije Moodle dostupne preko web stranica predmeta.

Napomena: Rezultati se šalju isključivo preko gore navedene aplikacije. U slučaju problema, javiti se email-om na adresu vo@fer.hr. Sačuvajte kopiju poslanih rezultata.

7. Napredni zadaci (ovaj dio nije obavezan)

7.1 Proširenje algoritma na druge geometrijske oblike

Napraviti algoritam praćenja zrake za bilo koje drugo geometrijsko tijelo osim kugle. Recimo najopćenitije bi bilo napraviti za trokut jer se onda od trokuta može sastaviti velik broj geometrijskih tijela. Ostali primjeri su elipsoid, kocka, torus, stožac itd.

7.2 Tekstura

Implementirati teksturiranje slike na geometriju.

7.3 Testiranje dubine rekurzije po doprinosu

Umjesto fiksne dubine rekurzije, kakva je implementirana u pseudo kodu u uputama za vježbu, implementirati postupak s adaptativnom dubinom rekurzije, pri kojem se rekurzija zaustavlja kada se doprinos reflektirane/refraktirane zrake spusti ispod zadanog praga.

7.4 Napredniji algoritam zrake za ispitivanje sjena

Implementirati takav algoritam zrake za ispitivanje sjena koji će u obzir uzeti sva tijela kroz koja je prošao i pomnožiti njihove udjele refraktirane zrake. Takav umnožak u programu iskoristiti kao koeficijent kojim množimo iznos spekularne i difuzne komponente u točki promatranja.