

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 1820

Reljefne tehnike teksturiranja

Anamarija Medić

Zagreb, listopad 2009.

Sadržaj:

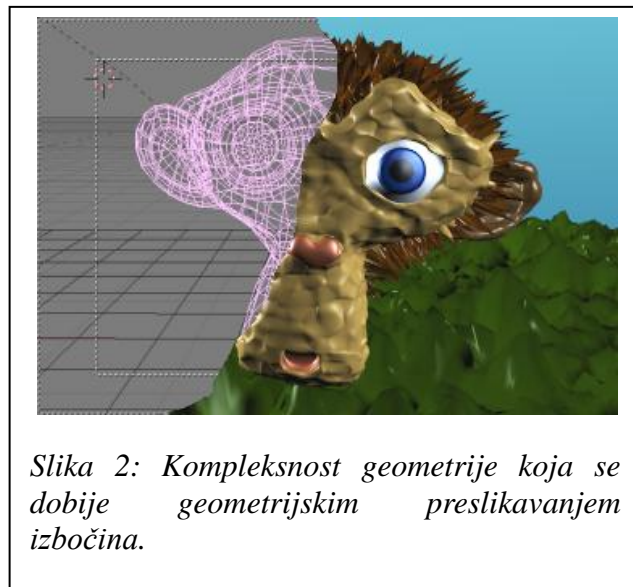
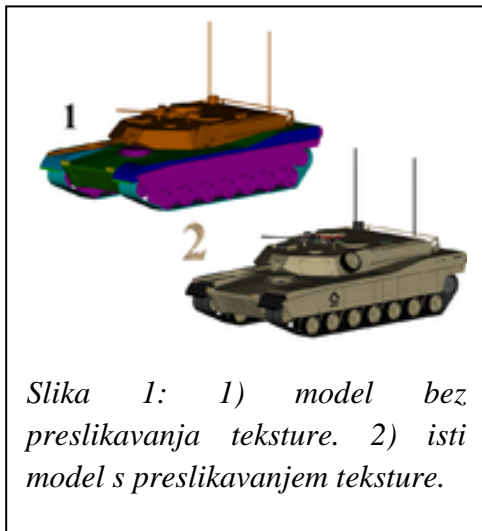
| | |
|--|----|
| 1. Uvod | 3 |
| 2. Kratak pregled tehnika za prikaz detalja trodimenzionalnih površina | 4 |
| 2.1. Preslikavanje izbočina | 5 |
| 2.2. Geometrijsko preslikavanje izbočina..... | 6 |
| 2.3. Preslikavanje uz efekt paralakse | 7 |
| 2.4. Preslikavanje uz efekt paralakse praćenjem zrake pogleda i reljefne tehnike preslikavanja . | 8 |
| 3. Pojava paralakse..... | 9 |
| 3.1. Paralaksa dobivena gibanjem..... | 9 |
| 3.2. Paralaksa dobivena promjenom kuta između vektora pogleda i površine | 10 |
| 4. Prikaz hrapavih površina preslikavanjem izbočina..... | 12 |
| 4.1. Dubinska karakteristika površine i pojam mape normala | 12 |
| 4.2. Tehnika preslikavanja izbočina..... | 14 |
| 4.2.1. Teorijska podloga preslikavanja izbočina | 15 |
| 4.3. Sustav tangente | 16 |
| 5. Prikaz reljefnih struktura preslikavanjem uz efekt paralakse | 17 |
| 5.1. Ulazne vrijednosti..... | 17 |
| 5.2. Tehnika preslikavanja uz efekt paralakse..... | 17 |
| 5.3. Ograničavanje odmaka..... | 20 |
| 6. Prikaz reljefnih struktura uz efekt paralakse praćenjem zrake pogleda | 21 |
| 6.1. Reljefno preslikavanje teksture..... | 21 |
| 6.1.1. Teorijska podloga reljefnog preslikavanja | 21 |
| 6.1.2. Implementacija reljefnog preslikavanja u stvarnom vremenu..... | 32 |
| 6.1.3. Samostalno bacanje sjene | 35 |
| 6.1.4. Ostale mogućnosti reljefnog preslikavanja | 37 |
| 6.2. Druge tehnike preslikavanja uz efekt paralakse praćenjem zrake pogleda | 39 |
| 6.2.1. Preslikavanje prekrivanja uslijed paralakse..... | 39 |
| 6.3. Preslikavanje strminom uz efekt paralakse..... | 40 |
| 6.4. Optimizacija metode pronalaženja sjecišta..... | 41 |
| 6.4.1. Optimizacija pronalaženja sjecišta uporabom mape stožaca | 41 |
| 7. Programska podrška..... | 43 |

| | | |
|----------------|---|----|
| 7.1. | Programski jezik za sjenčanje GLSL | 43 |
| 7.1.1. | Procesiranje vrhova | 44 |
| 7.1.2. | Procesiranje fragmenata | 45 |
| 7.2. | Jednostavan primjer programa za sjenčanje u GLSL-u | 46 |
| 8. | Rezultati..... | 47 |
| 8.1. | Rezultati dobiveni reljefnim preslikavanjem i greške dobivene krivim parametrima..... | 48 |
| 9. | Zaključak..... | 50 |
| Zahvala | | 50 |
| 10. | Literatura..... | 51 |
| Sažetak..... | | 52 |
| Abstract | | 53 |

1. Uvod

Dodavanje površinskih detalja objektima pridonosi realnosti tih objekata i cijele scene. Potrebne su nam površinske teksture za ispravno modeliranje objekata kao što su drveni ili kameni zidovi, ljudska koža i sl. Preslikavanje teksture je tehnika kojom se mogu prikazati složeni detalji površina i trodimenzionalnih modela. Tom tehnikom se preslikava uzorak nad geometrijskom karakteristikom objekta. Osnovna primjena jest preslikavanje boje na površinu. Osim toga omogućava sintezu slike, mnogobrojne varijacije u kvaliteti i efektima prikaza, promjenu izgleda površina dodavanjem zrcalnih efekata, preslikavanje i manipulaciju bojama, dodavanje površinskih odnaka točaka, promjenu vektora normale i sl.

Većina objekata nema glatke i jednolike površine. Dodavanjem tih dvodimenzionalnih detalja dobivamo ispravan prikaz slike, tapete na zidu ili omotnice na kutiji, no tradicionalnim preslikavanjem teksture ne možemo riješiti problem dodavanja trodimenzionalnih detalja. Razlog te ograničenosti leži u nemogućnosti prikaza paralakse dobivene pogledom i kretanjem.



Ovim radom ćemo dati pregled tehnika koja su dala rješenje za problem prikaza trodimenzionalnih detalja površina. Poseban naglasak će biti na reljefnim tehnikama preslikavanja tekstura.

2. Kratak pregled tehnika za prikaz detalja trodimenzionalnih površina

U ovom ćemo poglavlju dati kratak pregled najpopularnijih tehnika za prikaz detalja trodimenzionalnih površina. Detaljnija obrada slijedi u narednim poglavljima.

Trenutno postoje dva glavna pristupa za iscrtavanje trodimenzionalnih detalja objekata:

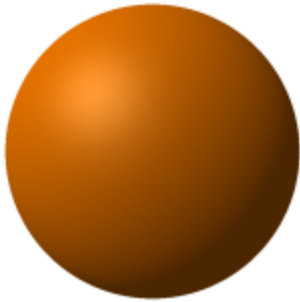
1. Pristup koji se temelji na promjeni geometrije (engl. *geometry-based rendering*)
2. Pristup koji se temelji na promjeni uzorka teksture (engl. *image-based rendering*).

Preslikavanje izbočina (engl. *bump mapping*), preslikavanje uz efekt paralakse (engl. *parallax mapping*) i preslikavanja uz efekt paralakse praćenjem zrake pogleda (engl. *relief mapping, parallax occlusion mapping, steep parallax mapping*) su najvažnije tehnike koje spadaju u pristup koji se temelji na promjeni uzorka teksture.

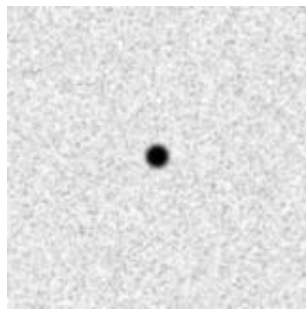
Geometrijsko preslikavanje izbočina (engl. *displacement mapping*) spada u pristup koji se temelji na promjeni geometrije.

2.1. Preslikavanje izbočina

Preslikavanje izbočina omogućuje detaljniji prikaz objekata na način da ispravno aproksimira hrapavost površine. Tehnika se temelji na obavljanju modulacija na normalama površine dobivajući modulirane vektore. Te modulirane vektore primjenjujemo u proračunu svijetla. Tehnika povećava realnost modela bez povećanja kompleksnosti geometrije (slika 5). No pretpostavka je uvijek da je visina neravnine jako mala s obzirom na veličinu same površine i zahtjeva osvjjetljenje po slikovnom elementu.



Slika 3: Sfera bez primjene efekta preslikavanja izbočina.



Slika 4: Tekstura koja se primjenjuje na model za preslikavanje izbočina.



Slika 5: Sfera je geometrijski ostala ista no na nju je primijenjena mapa koja opisuje preslikavanje izbočina. To mijenja način na koji površina sfere reagira na sjenčanje te joj daje izgled neravne površine.

2.2. Geometrijsko preslikavanje izbočina

Geometrijsko preslikavanje izbočina je tehnika u kontrastu s preslikavanjem izbočina i preslikavanjem uz efekt paralakse (engl. *parallax mapping*). Uz upotrebu teksture s dubinskim karakteristikama uzrokuje se odmak prave geometrijske pozicije točaka koji se nalaze na površini koja je preslikana teksturom (slika 7). Taj odmak se vrši duž normale površine, prema vrijednosti koja se dobije iz funkcije teksture. Ova tehnika daje površini osjećaj detaljnosti i dubine. Omogućava zaklanjanje samog sebe(engl. *self-occlusion*), bacanje sjene (engl. *self-shadowing*) i realniji izgled siluete. Loša strana je dodavanje velike količine dodatne geometrije. Ta geometrija se često iscrtava kao mreža mikropoligona.



Slika 6: Razlika između preslikavanja teksture, te plošnosti koju uzrokuje (desno) i geometrijskog preslikavanja izbočina te detaljnost koju uzrokuje (lijevo).



Slika 7: Dubinska karakteristika govori kako se koji element teksture odmiče od prvotne vrijednosti.

2.3. Preslikavanje uz efekt paralakse

Preslikavanje uz efekt paralakse (još se naziva preslikavanje uz virtualni pomak) je tehnika koja unapređuje tehniku preslikavanja izbočina i dodaje efekt paralakse. Kao rezultat krajnji korisnik vidi izražajnu dubinsku detaljnost površine (slika 8). Dubinsku karakteristiku predstavljaju vrijednosti čiji iznos odgovara dubini koja bi trebala biti u toj točki da je površina trodimenzionalna. Dubinsku detaljnost se postiže tako da se rade odmaci koordinata teksture nad slikovnim elementom koji se iscrtava. Ti pomaci su funkcija vektora pogleda i vrijednosti dubinske karakteristike. Ova tehnika daje jako dobre rezultate i krajnje je jednostavna, ali ima jako mnogo problema. Najočitiiji je da ne postoji zaklanjanje (engl. *occlusion*). Još jedan jako veliki nedostatak tehnike je da pri malim kutovima gledanja dolazi do deformacije i do isticanja plošnosti.



Slika 8: Kameni zid nad kojim je primijenjeno preslikavanje uz efekt paralakse.



Slika 9: Model slike 8 nad kojom je primijenjeno preslikavanje izbočina.



Slika 10: Model slike 8 nad kojom je primijenjeno preslikavanje teksture.

2.4. Preslikavanje uz efekt paralakse praćenjem zrake pogleda i reljefne tehnike preslikavanja

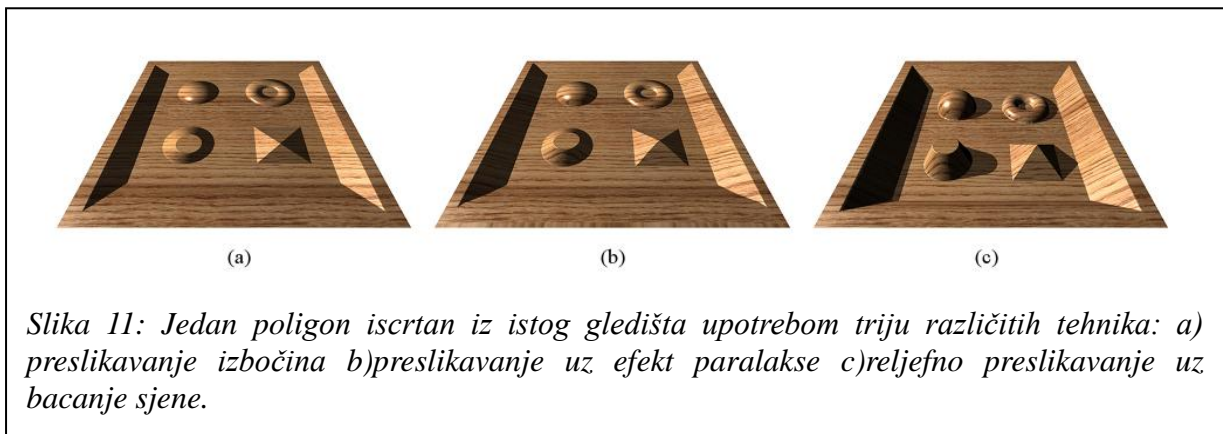
Preslikavanje uz efekt paralakse praćenjem zrake pogleda je tehnika koja poboljšava preslikavanje uz efekt paralakse na način da se preciznije određuje sjecište zrake pogleda i dubinske karakteristike površine. Također se rješava problem samozatamnjenja. Na osnovu načina određivanja sjecišta razlikujemo nekoliko tehnika. U vidu ovog rada posebna se pažnja posvećuje reljefnim tehnikama preslikavanja.

Reljefno preslikavanje predstavlja proširenje preslikavanje teksture tako da omogućava prikaz trodimenzionalnih detalja površina i ispravan prikaz paralakse uzrokovane pogledom i kretanjem.

Tehniku možemo gledati uz pomoću dva pristupa. Prvi pristup se temelji na faktorizaciji jednadžbe za deformaciju trodimenzionalne slike po McMillan-u i Bishop-u u "pred-deformaciju" i potom primjenu preslikavanja teksture. Drugi pristup se temelji na praćenju zrake pogleda i određivanje sjecišta. Drugi pristup je puno pogodniji i nastao je kao optimizacija prvog pristupa za prikaz u realnom vremenu.

Rezultati dobiveni ovom tehnikom rade ispravno za statičke promatrače i promatrače u pokretu koji mogu stajati daleko od površine ili blizu.

Reljefno preslikavanje značajno povećava efikasnost i funkcionalnost preslikavanja teksture te omogućava vjerodostojan prikaz površina i modela uz drastično smanjenje potrebne količine geometrije za taj prikaz.

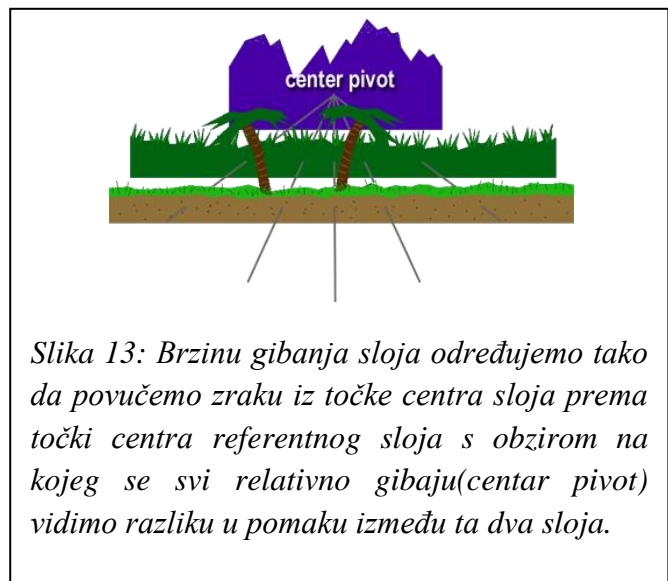
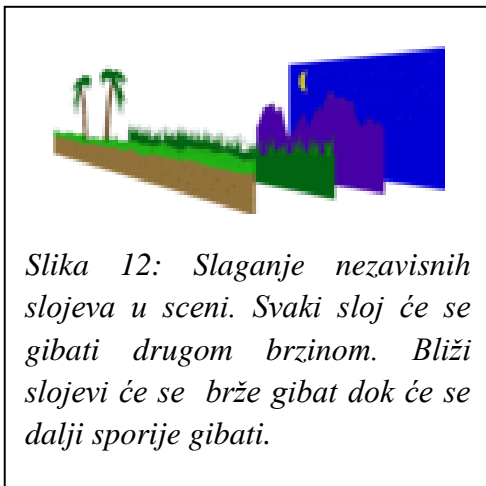


3. Pojava paralakse

Paralaksa je prividni odmak ili razlika u orijentaciji dvaju vektora pogleda koji gledaju u isti objekt iz različitih gledišta. Paralaksa se mjeri kutom kojeg zatvaraju ta dva pogleda. Dolazi od grčke riječi *parallaxis* koja znači "promjena". Objekti koji su blizu gledišta imaju veću paralaksu od objekata koji su na većoj udaljenosti. S obzirom na definiciju paralakse razlikovat ćemo dva pogleda na pojam. Paralaksu dobivenu gibanjem koja se koristi u dvodimenzionalnim igrama za iluziju dubine i paralaksa dobivena zbog promjene kuta između vektora pogleda i površine.

3.1. Paralaksa dobivena gibanjem

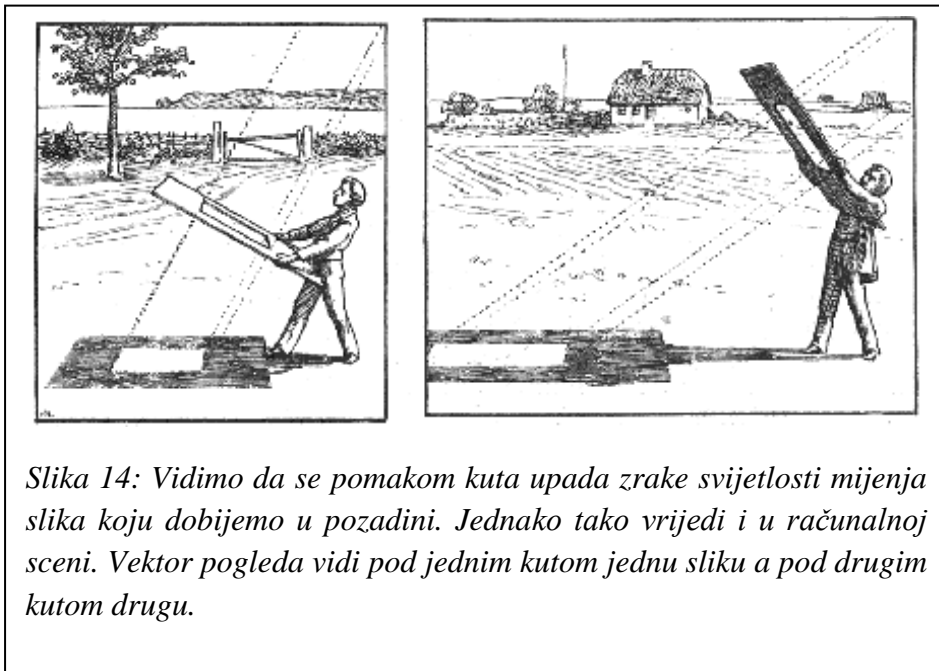
Paralaksa dobivena pokretom (engl. *parallax scrolling*) je iluzija dubine koja se dobiva kao rezultat gibanja promatrača u odnosu na objekt. Ova tehnika se koristi činjenicom da se objekti koji su bliži promatraču gibaju dulje vidnim poljem promatrača. U mnogim ranim aplikacijama scena bi se podijelila u nezavisne slojeve koji bi se pomicali različitim brzinama i time postigli osjećaj dubine prostora (slika 12).



3.2. Paralaksa dobivena promjenom kuta između vektora pogleda i površine

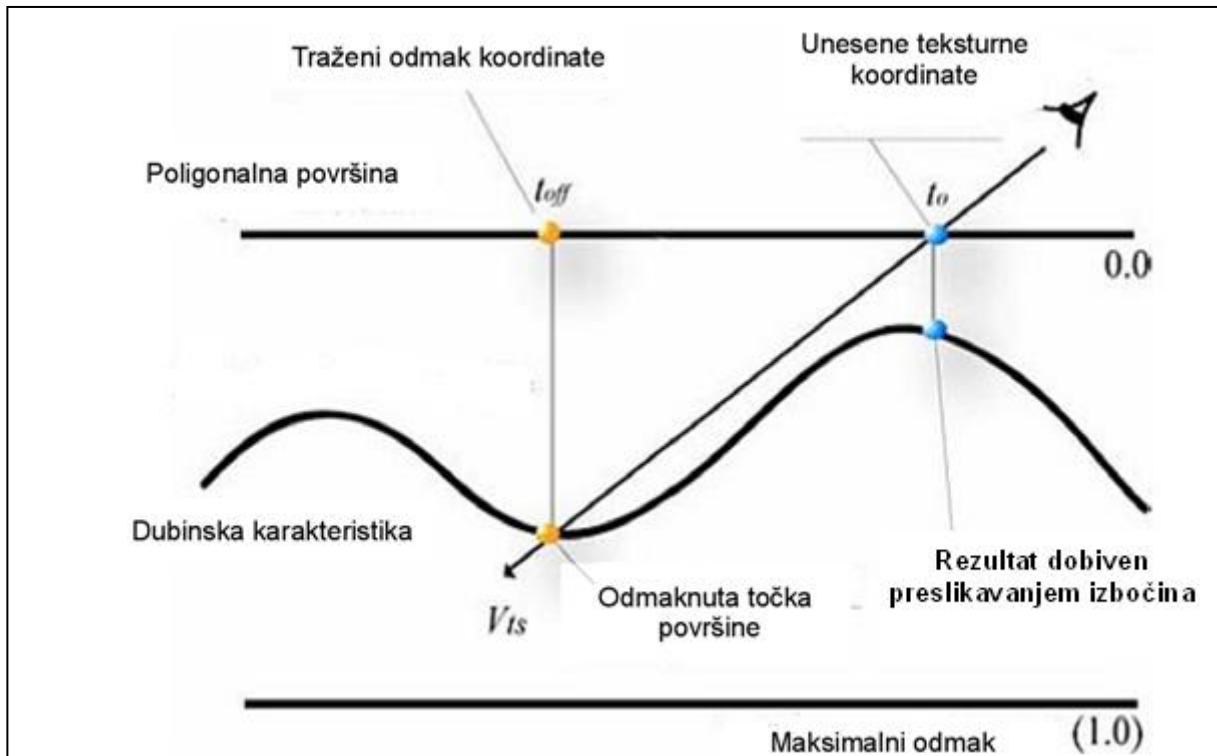
Kada nam se čini da je objekt promijenio položaj jer smo mi promijenili kut gledanja tada tu pojavu zovemo paralaksom. Kada neki predmet gledamo iz dva različita kuta tada slika koju vidimo nije ista. Razlog tome jest što kada gledamo iz jednog gledišta sjecišta našeg vektora gledanja i promatranog objekta nisu ista. Ovisno o kutu gledanja određena područja su više razdužena a drugo su sužena.

Astronomi vide efekt paralakse kada gledaju u zvijezde sa različitih mjesta na Zemlji.



Slika 14: Vidimo da se pomakom kuta upada zrake svjetlosti mijenja slika koju dobijemo u pozadini. Jednako tako vrijedi i u računalnoj sceni. Vektor pogleda vidi pod jednim kutom jednu sliku a pod drugim kutom drugu.

U računalnoj sceni efekt paralakse želimo postići nad površinskim objektima. Ti objekti će imati sebi pridružene plošne teksture. Ideja svih tehnika preslikavanja izbočina se temelji na tome da se vrši modulacija koordinata teksture nad slikovnim elementom koji se trenutno obrađuje i to na način da se realno imitira efekt paralakse koji bi se trebao pojaviti da je model uistinu trodimenzionalan. Svaka od tih tehnika daje određene rezultate koji su u određenim situacijama bolji ili lošiji. Najrealnija i zasigurno najbolja tehnika koja rješava ovaj problem je tehnika praćenje zrake pogleda. Tu tehniku čini zapravo skup tehnika koje se razlikuju u načinu rješavanja sjecišta vektora pogleda i dubinske karakteristike.



Slika 15: Promatrač gleda svojim vektorom pogleda na površinu. Kada bi se ovdje radilo o preslikavanju teksture promatrač bi video t_0 . Tada ne bi vidio paralaksu koja bi trebala postojati i površina je u potpunosti plosnata. Tehnike preslikavanja uz efekt paralakse uzimaju u obzir dubinsku karakteristiku koja bi postojala da je površina objekta trodimenzionalna.

Vidimo da presjekom vektora pogleda promatrača i te dubinske karakteristike dobijemo sjecište koje kada preslikamo na teksturu daje ispravnu vrijednost koordinate teksture koja bi se trebala prikazati u tom slikovnom elementu.

4. Prikaz hrapavih površina preslikavanjem izbočina

Unutar ovog poglavlja slijedi opis tehnike preslikavanje izbočina. Prvo ćemo opisat i definirat pojam dubinske karakteristike površine te pojam mape normala. Zatim slijedi opis tehnike preslikavanja izbočina i teorijska podloga nad kojom se temelji. Potom slijedi opis prostora kojeg nazivamo sustav tangente.

4.1. Dubinska karakteristika površine i pojam mape normala

Dubinska karakteristike površine čini skup vrijednosti koje odgovaraju iznosu dubine pojedine točke. Unutar našeg primjera dubinska karakteristika se smješta u teksturu. Komponenta α te teksture odgovara dubinskoj vrijednosti elementa koja se nalazi na tim istim (u,v) koordinatama u teksturi boje.



Slika 16: Lijeva slika predstavlja teksturu koja čini dubinsku karakteristiku desne teksture boje. Za svaku par (u,v) lijeva tekstura daje dubinsku vrijednost tog istog elementa na teksturi boje.

Kada imamo dubinsku karakteristiku sljedeći korak je izračun mape normala. Mapa normala je tekstura čija svaka vrijednost odgovara normalni koju bi taj slikovni element trebao imati kada bi bio izbočen kako to dubinska karakteristika opisuje. Te normale se računaju iz dubinske karakteristike tako da

$$n(i, j) = (|h(i, j)h(i + 1, j)|) \times (|h(i, j)h(i, j + 1)|) \quad [4.1.1.]$$

Iz ove jednadžbe vidimo da je normala na poziciji (i, j) jednaka vektorskom produktu dvaju vektora. Prvi vektor je između točaka visinske vrijednosti $h(i, j)$ prema susjednoj visinskoj vrijednosti $h(i + 1, j)$. Drugi vektor je između točaka visinske vrijednosti $h(i, j)$ prema susjednoj visinskoj vrijednosti $h(i, j + 1)$.

Sada kada smo dobili normalu u točki (i, j) potrebno je spremi vektor u teksturu. Najpogodnije je to napraviti tako da se svaka komponenta spremi u novi kanal. To ćemo napraviti tako da se \vec{i} komponenta spremi u R kanal, \vec{j} komponenta u G kanal i \vec{k} komponenta u B kanal. Jedini problem koji još imamo jest skaliranje vektora unutar granica [0, 1]. Razlog tog skaliranja jest taj što se vrijednosti RGB komponente nalaze u tom rasponu, a vektor normale se nalazi u rasponu [-1, 1]. Skaliranje postićemo :

$$R = \frac{(x+1.0)}{2}, G = \frac{(y+1.0)}{2}, B = \frac{(z+1.0)}{2} \text{ [4.1.2.]}$$

Sada kada smo dobili mapu normala čije vrijednosti odgovaraju normalama koje bi površina imala da je izbočena na način opisano u dubinskoj karakteristici možemo raditi preslikavanje izbočina.

4.2. Tehnika preslikavanja izbočina

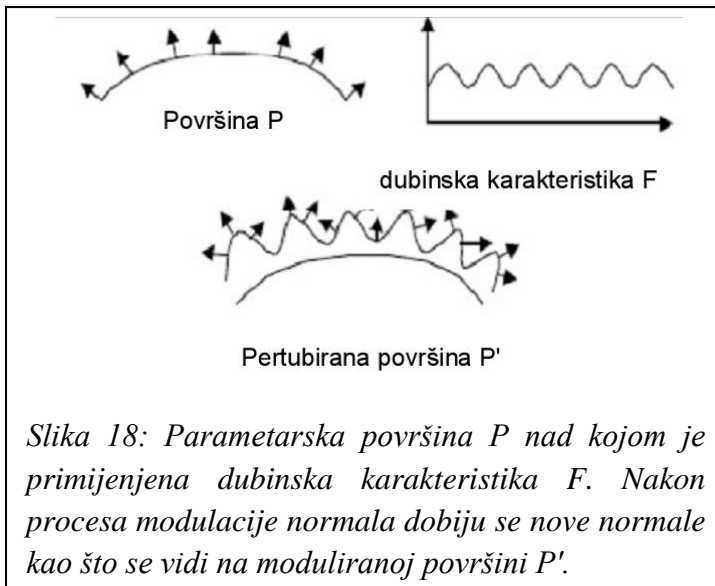
Preslikavanje izbočina se temelji na modulaciji normala površine koje će se koristiti u proračunu svijetla. Ova tehnika povećava realnost scene bez povećavanja dodatne geometrije. Tehnika u biti ne mijenja normale već samo koristi druge kako bi uspjela prevariti proračun svijetla. U sljedećem odjelu dat ćemo teorijsku podlogu kako se računaju nove normale iz dubinske karakteristike i parametarske površine.



Slika 17: Lijeva slika prikazuje model nad kojim je primijenjeno preslikavanje izbočina, a proračun svijetla vrši se na prethodno opisan način. Desna strana prikazuje model sa običnim preslikavanjem tekstone.

4.2.1. Teorijska podloga preslikavanja izbočina

Neka točka P čini točku na parametarskoj površini. Neka su površina i dubinska karakteristika opisane slikom 18. Idemo definirati normalu u točki P nad površinom. Ona je dana izrazom:



Slika 18: Parametarska površina P nad kojom je primijenjena dubinska karakteristika F. Nakon procesa modulacije normala dobiju se nove normale kao što se vidi na moduliranoj površini P'.

$$N(u, v) = \frac{\partial P(u, v)}{\partial u} \times \frac{\partial P(u, v)}{\partial v} \quad [4.2.1.1.]$$

Modulirani vektor pozicije nad površinom P' dobivamo izrazom:

$$P'(u, v) = P(u, v) + F(u, v) \frac{N(u, v)}{|N(u, v)|} \quad [4.2.1.2.]$$

Normala točke definirane u P' iznosi:

$$N'(u, v) = \frac{\partial P'(u, v)}{\partial u} \times \frac{\partial P'(u, v)}{\partial v} \quad [4.2.1.3.]$$

Kada nad time uvrstimo izraz [4.2.2.] dobivamo:

$$N' = N + \frac{\frac{\partial F}{\partial u} \left(N \times \frac{\partial P}{\partial v} \right) - \frac{\partial F}{\partial v} \left(N \times \frac{\partial P}{\partial u} \right)}{|N|} \quad [4.2.1.4.]$$

Sada kada smo dobili modulirane normale koje ćemo koristiti umjesto pravih ostatak posla je lagan. Pri proračunu svijetla koristit ćemo Phongov model osvjetljavanja:

$$I = k_a i_a + \sum_{\text{svijetla}} (k_d (L \cdot N) i_d) + k_s (R \cdot V)^\alpha i_s \quad [4.2.1.5.]$$

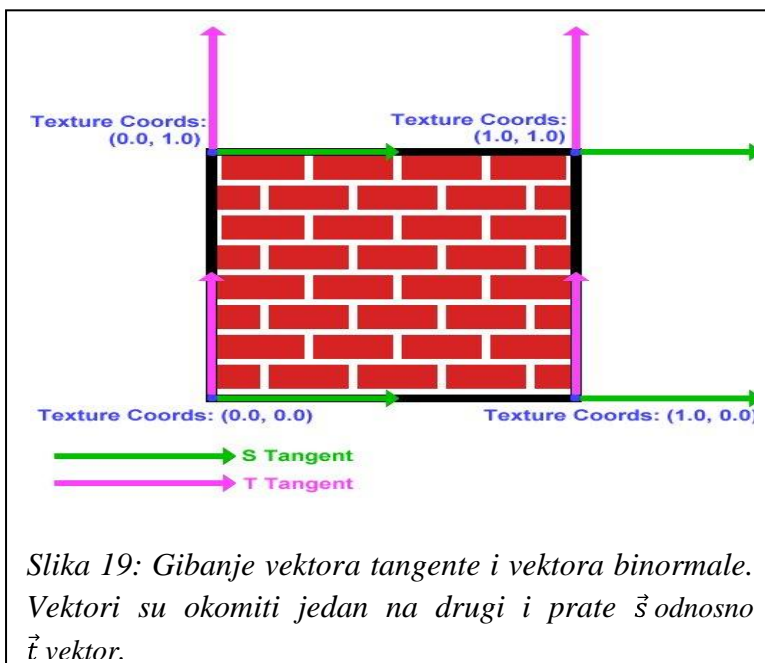
Na tom proračunu ćemo umjesto pravih vektora normale N koristiti N'. Na taj ćemo način dobiti ispravno osvjetljenje. I dalje ostaje problem paralakse, tj. ova tehnika ne stvara efekt paralakse. U sljedećim tehnikama koje ćemo koristiti primijetit će se da se kao tehnika osvjetljavanja koristi upravo opisana tehnika s moduliranim normalama. Svaka od sljedećih tehnika će biti samo proširenje preslikavanja izbočina.

U daljnjim razmatranjima će se za izračun moduliranih normala koristiti tehnika računanja mape normala opisane u jednom od prethodnih odjela. Ta tehnika je jako pogodna za realnu izvedbu jer se omogućava prenošenje normala preko tekstura programu za sjenčanje.

4.3. Sustav tangente

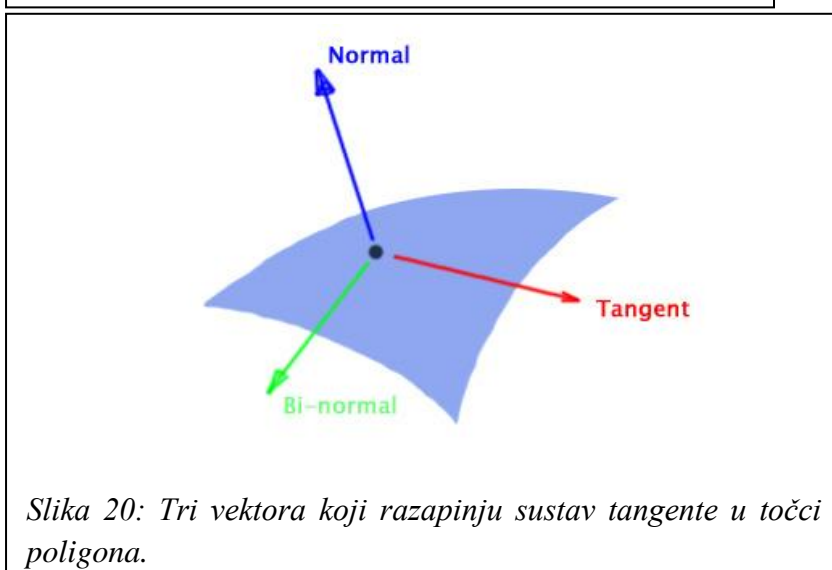
Da bi rezultat proračuna svjetla imao smisla, svi vektori koji se koriste pri tom proračunu moraju biti definirani unutar istog koordinatnog sustava. Kada bi se za taj sustav uzeo sustav koordinata oka tada bi izmijenjene normale koje se koriste unutar preslikavanja izbočina morale biti prebačene u taj sustav i još dodatno dodane pravoj normali i potom normalizirane. Takve operacije su jako skupe unutar programa za sjenčanja. Zbog toga se uvodi sustav tangente koji će varirati nad svakim vrhom poligona.

Koordinatni sustav tangente razapinju tri okomita vektora. Prvi vektor je vektor tangente $T = (T_x, T_y, T_z)$. To je vektor koji je okomit na normalu i prati vektor \vec{s} . Drugi vektor je vektor normale $N = (N_x, N_y, N_z)$. Treći vektor je vektor binormale $B = (B_x, B_y, B_z)$ i on se dobije kao vektorski produkt prethodna dva. Zajedno ova tri vektora čine ortogonalnu bazu ovog sustava (slika 20).



Da bi prebacili vektor iz lokalnog prostora objekta koristit ćemo izraz:

$$\begin{bmatrix} S_x \\ S_y \\ S_z \end{bmatrix} = \begin{bmatrix} T_x & T_y & T_z \\ B_x & B_y & B_z \\ N_x & N_y & N_z \end{bmatrix} \begin{bmatrix} O_x \\ O_y \\ O_z \end{bmatrix} \quad [4.3.1.]$$

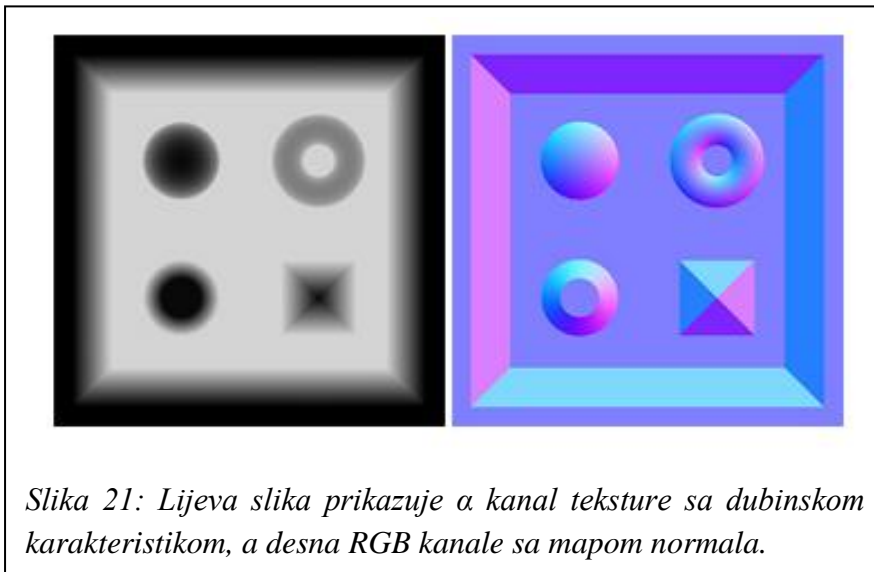


5. Prikaz reljefnih struktura preslikavanjem uz efekt paralakse

Preslikavanje uz efekt paralakse je tehnika koja u stvarnom vremenu izvodi aproksimaciju izbočenja objekta. Svaka koordinata teksture slikovnog elementa se primjereno oblikuje kako bi za vrijeme iscrtavanja dala iluziju dubine koje nema. Efekt se prilagođava vektoru pogleda promatrača. Glavna prednost ove tehnike je efikasnost. To je najjednostavnija tehnika za ostvarivanje dubine uz efekt paralakse u realnom vremenu. Glavni nedostatak je kvaliteta. Preslikavanje uz efekt paralakse ne daje tako dobre rezultate kao što to daju zahtjevnije tehnike preslikavanja uz efekt paralakse praćenjem zrake pogleda.

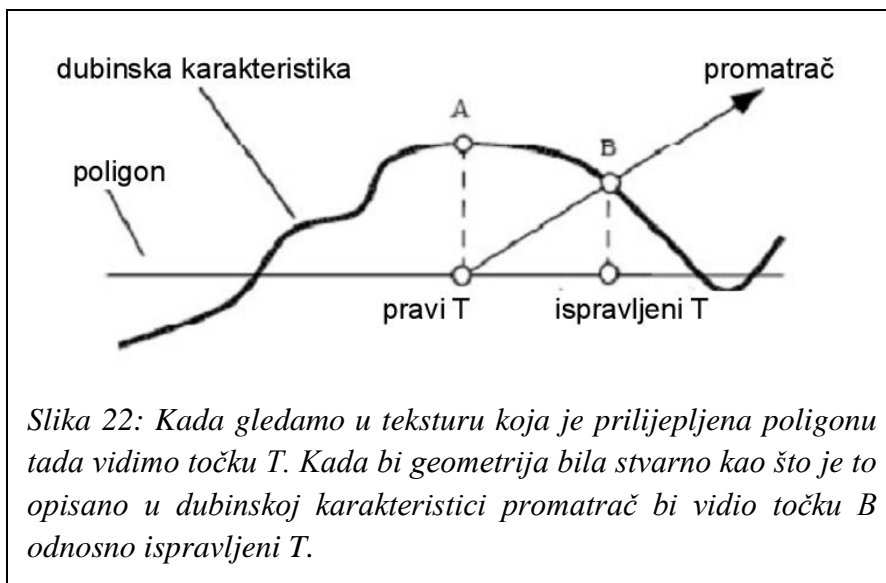
5.1. Ulazne vrijednosti

Preslikavanje uz efekt paralakse je tehnika koja proširuje preslikavanje izbočina time što rješava problem paralakse. Kao što smo vidjeli preslikavanje izbočina nema efekt paralakse uvjetovane pogledom i kretnjom. Ova tehnika koristi istu mapu normala koju koristi preslikavanje izbočina za proračun svijetla. Međutim koristi i dubinsku karakteristiku u svojim proračunima oblikovanja koordinata teksture. Mapa normala i dubinska karakteristika se zajedno spremaju u jednu teksturu na način da se normale spremaju u RGB kanale, a dubina za pojedinu točku u α kanal.

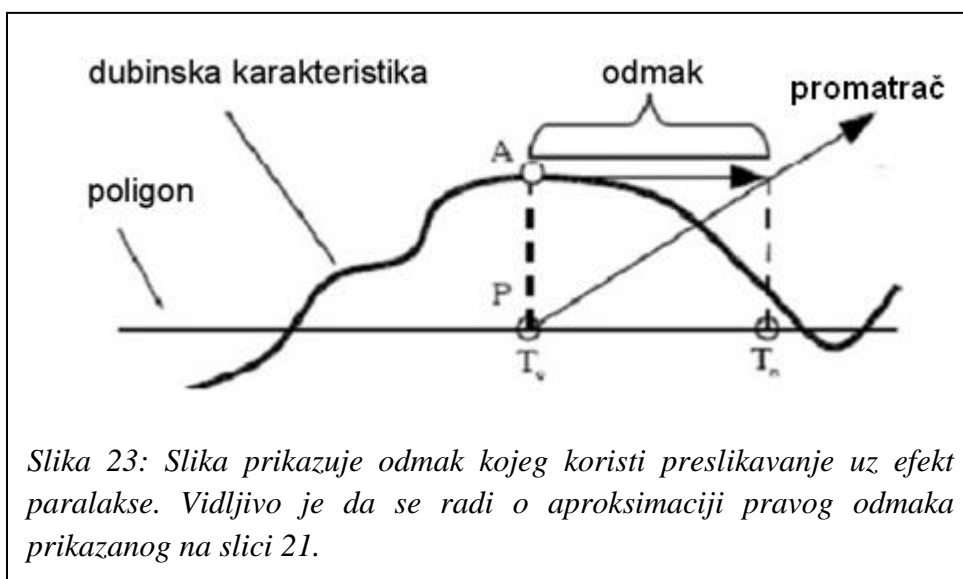


5.2. Tehnika preslikavanja uz efekt paralakse

Algoritam tehnike se temelji na računanje odmaka koordinata teksture na danom slikovnom elementu. Čitav proračun se izvodi u sustavu tangente kao što je to bio slučaj u preslikavanju izbočina. Pošto se koristi sustav tangente z komponenta vektora govori o okomitoj udaljenosti od površine, a x i y komponenta govore o pomacima unutar površine.



Kao što vidimo na slici 22 promatrač kada gleda u površinu vidi točku A (odnosno točku T na teksturi). Da je površina geometrijski jednaka dubinskoj karakteristici promatrač bi vidio točku B (na teksturi ispravljeni T). Preslikavanje uz efekt paralakse aproksimira izračun odmaka koji je potreban da promatrač vidi ispravnu točku. Slika 23 opisuje aproksimaciju koju preslikavanje uz efekt paralakse koristi.



Jednadžba koja opisuje računanje točke T_n i time opisuje modeliranje preslikavanja uz efekt paralakse je:

$$T_n = T_v + h' \frac{V_{\{x,y\}}}{V_{\{z\}}} \quad [5.2.1.]$$

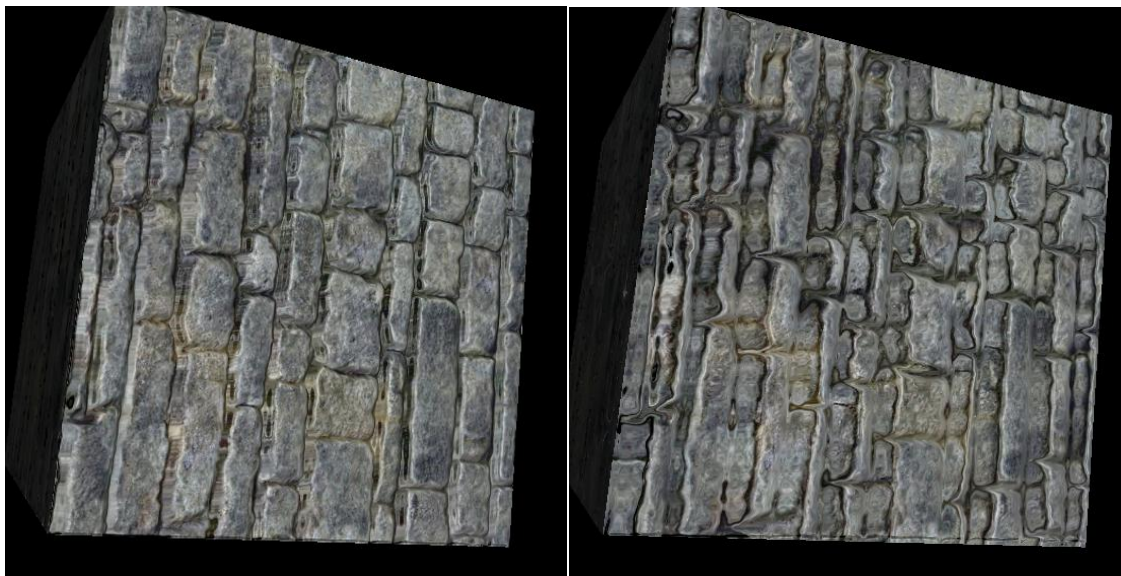
T_n i T_v su promijenjene koordinate teksteure odnosno početne koordinate teksteure. V je vektor pogleda koji ide od promatrača prema slikovnom elementu i definiran je u sustavu tangente.

h' je skalirana verzija vrijednosti dubinske karakteristike.

Geometrijski je jednadžba paralakse krajnje jednostavna [5.2.1.]. Dijeljenje x i y komponente vektora pogleda sa z komponentom daje nagib. Dubinska vrijednost h' predstavlja rast normalom. Umnožak nagiba i dubinske vrijednosti h' daje proporcionalno tangentno kretanje nad površinom poligona u smjeru vektora pogleda. Taj tangentni odmak se dodaje početnoj koordinati teksture.

Dubinska vrijednost h' se skalira. Razlog tome jest što kada mi izvadimo iz dubinske karakteristike vrijednost za taj slikovni element vrijednost je skalirana u granicama $[0, 1]$. Kada bi ostala u tim granicama i kada bi se ta vrijednost koristila u jednadžbi paralakse promijenjene koordinate teksture bi se u velikim mjerama preklapale i ispravljale međusobno. Rezultat toga bi dao slijevanje slikovnih elemenata.

$$h' = h * s + b \text{ [5.2.2.]}$$



Slika 24: Lijeva slika prikazuje model nad kojem je primijenjeno skaliranje dubinske vrijednosti s parametrima $s = 0.06$ i $b = -0.02$. Primjećujemo da se ne vide toliko greške aproksimacije. Desna slika ima za parametre skaliranja $s = 0.116$ i $b = 0.000$. Primjećujemo slijevanje koordinata teksture.

Preslikavanje uz efekt paralakse je samo gruba aproksimacija i za sobom nosi mnogo nedostataka.

Kada promatrač gleda poligon iz profila primjećuje da je poligon potpuno ravan. Tehnika ne vodi računa o tome kada jedan dio teksture zaklanja drugi dio. Kada gledamo poligon pri malim kutovima dolazi do pojave slijevanja koordinata tekstura zbog strmog nagiba u dubinskoj karakteristici. Dolazi do preklapanja i diskontinuiranosti što je rezultat aproksimacije odmaka. Kao što vidimo na slici 22 promijenjena koordinata teksture nije jednaka koordinati teksture koju bi promatrač trebao vidjeti iz tog položaja. Jednadžba paralakse uzima u obzir samo dubinsku vrijednost točke koja se trenutno obrađuje te zanemaruje oblik površine. Ide od pretpostavke da cijela površina ima konstantnu dubinsku vrijednost i to vrijednost dubine točke koju trenutno obrađuje.

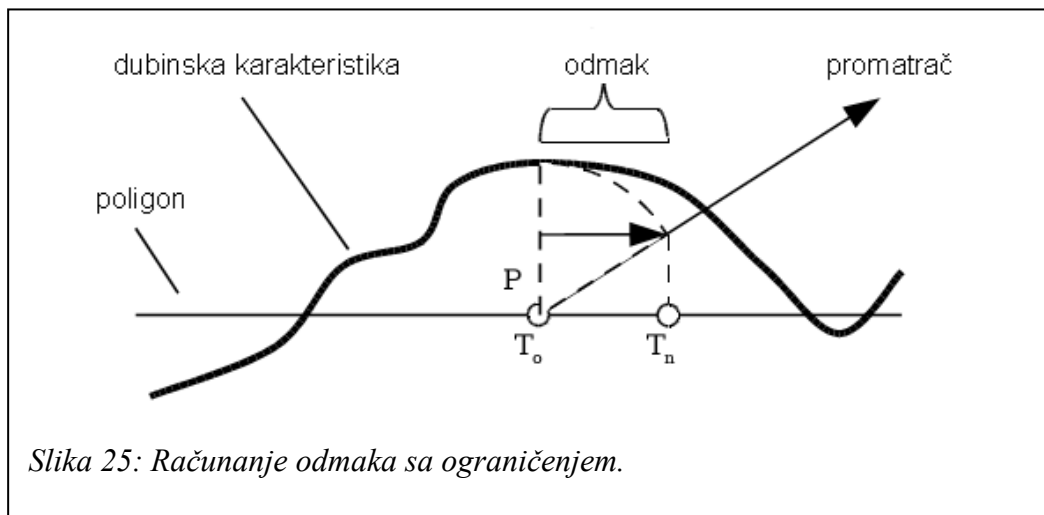
5.3. Ograničavanje odmaka

Kako bi se smanjila pojava slijevanja koordinata teksture odstranjuje se $1/z$ izraz iz jednadžbe paralakse. Ovu modificiranu verziju preslikavanja uz efekt paralakse nazivamo preslikavanje uz efekt paralakse s ograničenjem odmaka (engl. parallax mapping with offset limiting). Modificirana jednadžba glasi:

$$T_n = T_v + h' * V_{\{x,y\}} \quad [5.3.1.]$$

Odstranjivanjem z podjele ograničuje pomak na maksimum od h' u oba smjera teksture čime se uvelike smanjuju greške dobivene prevelikim preklapanjem. Mora se samo napomenut da u ovom slučaju x i y koordinate predstavljaju mjeru skaliranja i time vektor pogleda mora biti normaliziran.

Slika 21 nam pokazuje kako dubina više nije konstantno jednaka dubini u točki T_o .



6. Prikaz reljefnih struktura uz efekt paralakse praćenjem zrake pogleda

Tehnika preslikavanja uz efekt paralakse ima mnogo nedostataka kao što su nemogućnost samozatamnjenja i pojava slijevanja koordinata teksture. Razlog tome jest što je tehnika samo gruba aproksimacija i time računanje odmaka nije nikad točno. Poboljšanje toga bi bilo da se točnije računa odmak odnosno da se točnije računa sjecište vektora pogleda i dubinske karakteristike. Tehnika preslikavanja uz efekt paralakse praćenjem zrake pogleda jest upravo to. Temelji se na iterativnom traženju sjecišta. Unutar toga razlikujemo više pristupa. Jedan od pristupa jest reljefno preslikavanje. Unutar ovog poglavlja dan je opis tehnike i njene implementacije u stvarnom vremenu praćenjem zrake pogleda. Druge tehnike kao što su preslikavanje prekrivanja uslijed paralakse (engl. *parallax occlusion mapping*), ili preslikavanje strminom uz efekt paralakse (engl. *steep parallax mapping*) imaju sličan pristup kao reljefno preslikavanje samo što se razlikuju u načinu određivanja sjecišta. Na kraju samog poglavlja je dan jedan od algoritama optimizacije reljefnog preslikavanja.

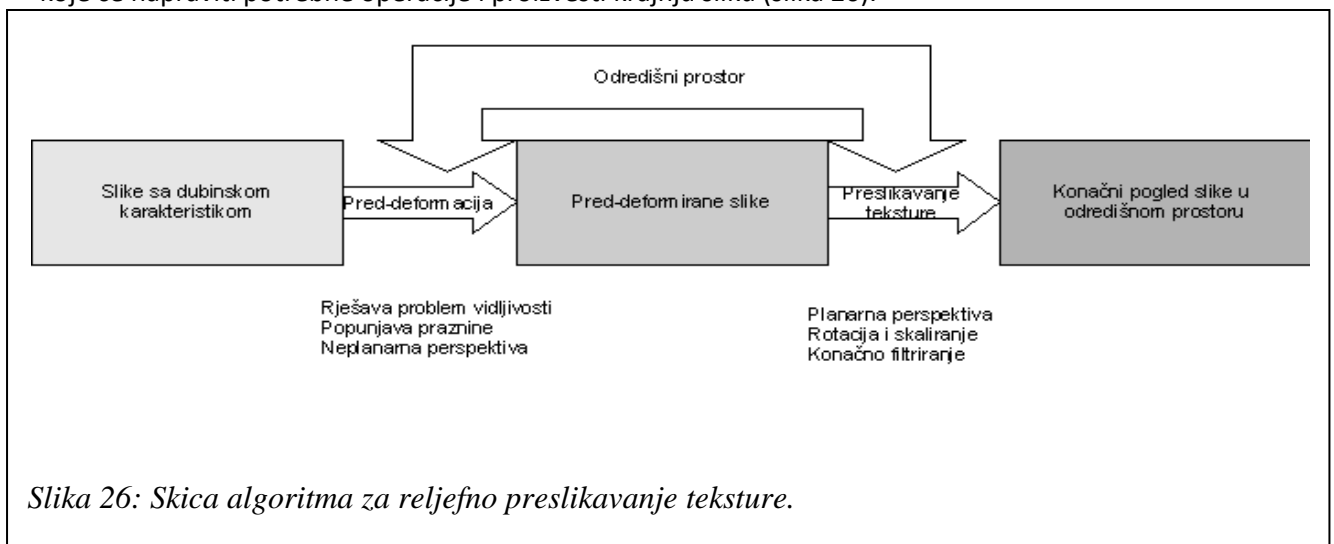
6.1. Reljefno preslikavanje teksture

Slijedi pregled teorijske podloge tehnike reljefnog preslikavanja, njena implementacija u stvarnom vremenu praćenjem zrake pogleda pomoću programa za sjenčanje te druge mogućnosti. To su na primjer tehnike koje uključuju iscrtavanje modela u više slojeva što kao rezultat daje strukture koje ispravno prikazuju siluete.

6.1.1. Teorijska podloga reljefnog preslikavanja

6.1.1.1. Pregled algoritma

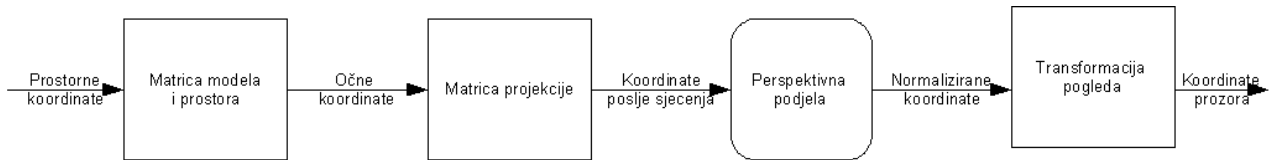
Reljefno preslikavanje teksture daje efekt paralakse uzrokovan pogledom i kretanjem na način da deformira (engl. *pre-warp*) teksturu prije samog preslikavanja na površinu. Kao ulaz u algoritam imamo dubinsku karakteristiku slike i vektor pogleda. Pred-deformacija je odgovorna za rješavanje problema vidljivosti i popunjavanje praznina. Izlazna slika se koristi kao ulaz u preslikavanje teksture koje će napraviti potrebne operacije i proizvesti krajnju sliku (slika 26).



Slika 26: Skica algoritma za reljefno preslikavanje teksture.

6.1.1.2. Trodimenzionalna deformacija slike

Prvo ćemo pokazati put točke kroz grafički protočni sustav koji se iz lokalnih koordinata objekta projicira u koordinate prozora.



Objekt je dan u koordinatnom sustavu objekta:

$$P_{object} = [x_o, y_o, z_o, 1]^T$$

i matricom pogleda i modela:

$$M_{modelview}$$

Zatim transformiramo koordinate točke u sustav oka $P_{eye} = [x_{eye}, y_{eye}, z_{eye}, w_{eye}]^T$. Ova će se transformacija dobiti iz izraza:

$$P_{eye} = M_{modelview} P_{object}$$

Na sličan način, ako je $M_{projection}$ projekcijska matrica onda su $P_{clip} = [x_{clip}, y_{clip}, z_{clip}, w_{clip}]^T$ točke dobivene nakon odsijecanja. Te točke se dobiju iz izraza:

$$P_{clip} = M_{projection} P_{eye}$$

Normalizirane koordinate točaka se dobiju iz izraza :

$$P_{NDC} \stackrel{\div w}{=} P_{clip}$$

Simbol $\stackrel{\div w}{=}$ označava jednakost poslije operacije dijeljenja homogenim faktorom w s desne strane. X, Y i Z komponenta od P_{NDC} su skalirani u granicama $[-1, 1]$.

Pretpostavimo da je prozor gledanja širine W i dužine H te da je ishodište u centru gledanja.

Neka je dubinski opseg $[n, f]$. P_{NDC} je projiciran u točku unutar prozora gledanja preko transformacijske matrice $M_{viewport}$. Ta točka iznosi: $P_{win} = [x_{win}, y_{win}, z_{win}, 1]^T$ i dobiva se preko izraza:

$$P_{win} = M_{viewport} P_{NDC}$$

$$M_{\text{viewport}} = \begin{bmatrix} W/2 & 0 & 0 & W/2 \\ 0 & H/2 & 0 & H/2 \\ 0 & 0 & (f-n)/2 & (f+n)/2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

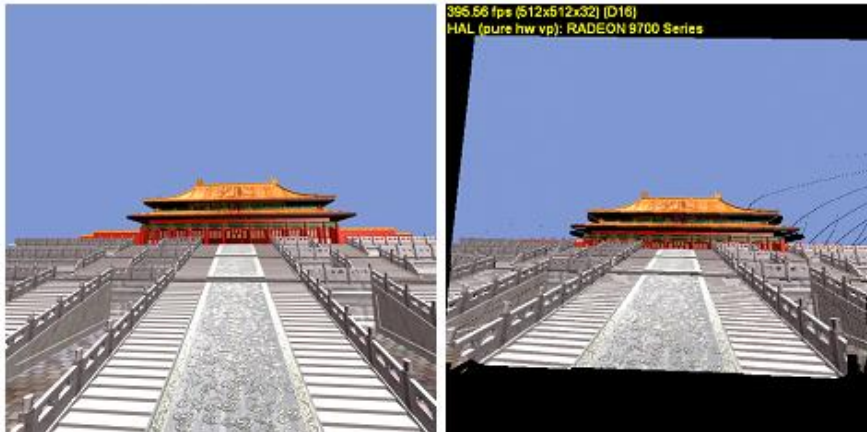
Sve se matrice mogu spojiti u jednu krajnju matricu M_n :

$$M_n = M_{\text{viewport}} M_{\text{projection}} M_{\text{modelview}}$$

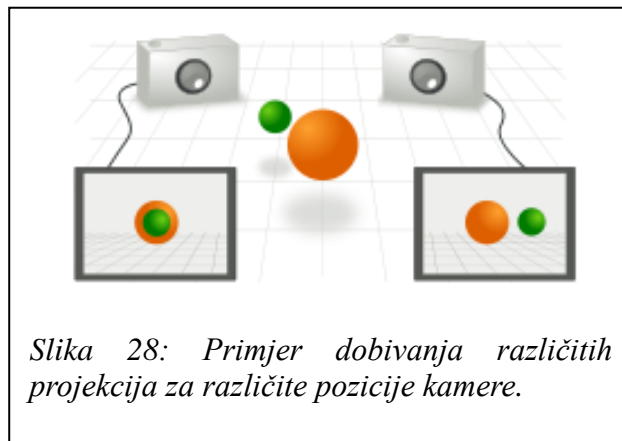
$$P_{\text{win}} \stackrel{w}{=} M_n P_{\text{object}}$$

Gornji izraz generira referentnu sliku. Transformacija slikovnog elementa unutar trodimenzionalne deformacije slike je transformacija koja se sastoji od dva koraka. Prvi korak je vratiti točke natrag u lokalne koordinate objekta (inverzno preslikavanje). Drugi korak je preslikati dobivene točke u drugo odredišno gledište kroz isti protočni sustav.

$$M_{\text{warping}} = M_{d\pi} M_{s\pi}^{-1}$$

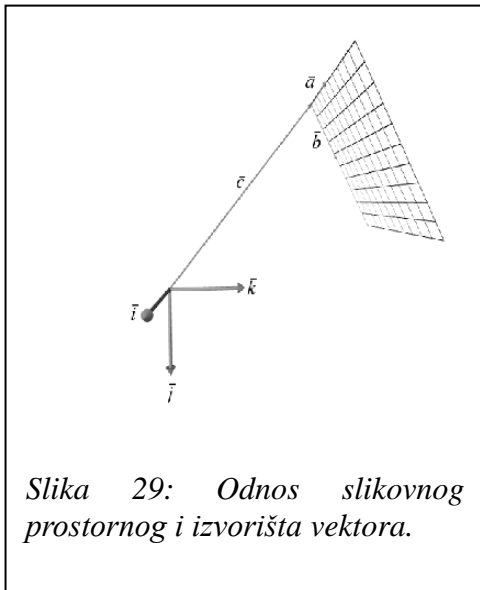


Slika 27: Lijeva slika predstavlja originalnu referentnu sliku, desna deformiranu sliku.



Slika 28: Primjer dobivanja različitih projekcija za različite pozicije kamere.

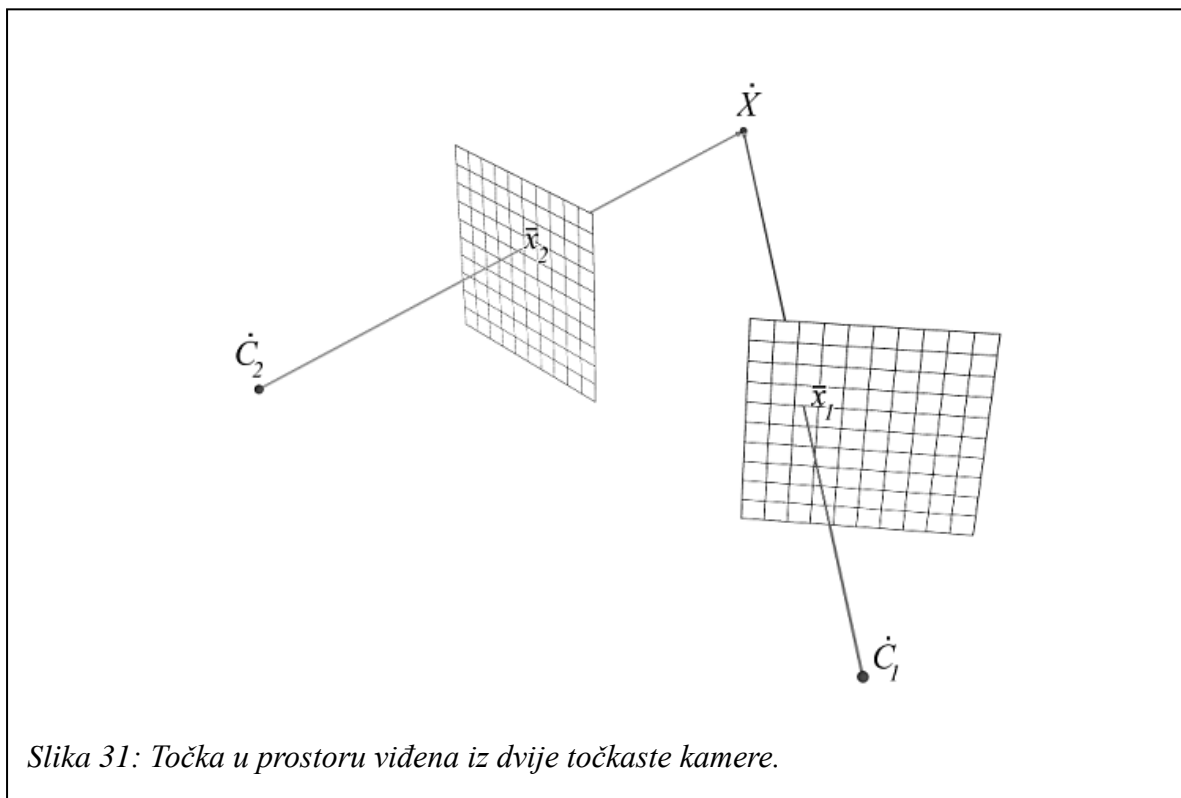
6.1.1.3. Formalni opis trodimenzionalne deformacije slike



$$\bar{d} = \begin{bmatrix} d_i \\ d_j \\ d_k \end{bmatrix} = \begin{bmatrix} a_i & b_i & c_i \\ a_j & b_j & c_j \\ a_k & b_k & c_k \end{bmatrix} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \mathbf{P} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$$

Slika 30: Funkcija planarnog projiciranja točke u prostor slike opisan ravninom u trodimenzionalnom prostoru. Rezultirajući vektor d je vektor koji polazi od izvorišta koordinatnog sustava i i prolazi kroz točku (u,v) unutar prostora slike. Vrijednosti u i v u matrici P čine vektori a , b i c . Vektori a i b razapinju dvodimenzionalni prostor slike. Vektor c je vektor iz izvorišta sustava u početnu točku ravnine slikovnog prostora.

Trodimenzionalna deformacija slike je geometrijska transformacija koja preslikava referentnu izvornu sliku i_s u određenu sliku i_d (slika 31).



Geometrijski model scene se opisuje kombinacijom dubinske karakteristike i modelom kamere asociran s izvornom slikom. Neka je x točka u euklidskom prostoru čija projekcija na slikovnu ravninu i_s ima koordinate (u_s, v_s) . Projekcija x u proizvoljnu odredišnu ravninu i_d se izražava sa:

$$\vec{x}_t \triangleq P_t^{-1} P_s \vec{x}_s + P_t^{-1} (C_s - C_t) \delta_s(u_s, v_s) \quad [6.1.1.3.1.]$$

\triangleq označava jednakost čijem rezultatu nedostaje jedan skalarni produkt parametara.

i_t - predstavlja odredišni prostor (*target space*).

i_s - predstavlja referentni prostor (*source space*).

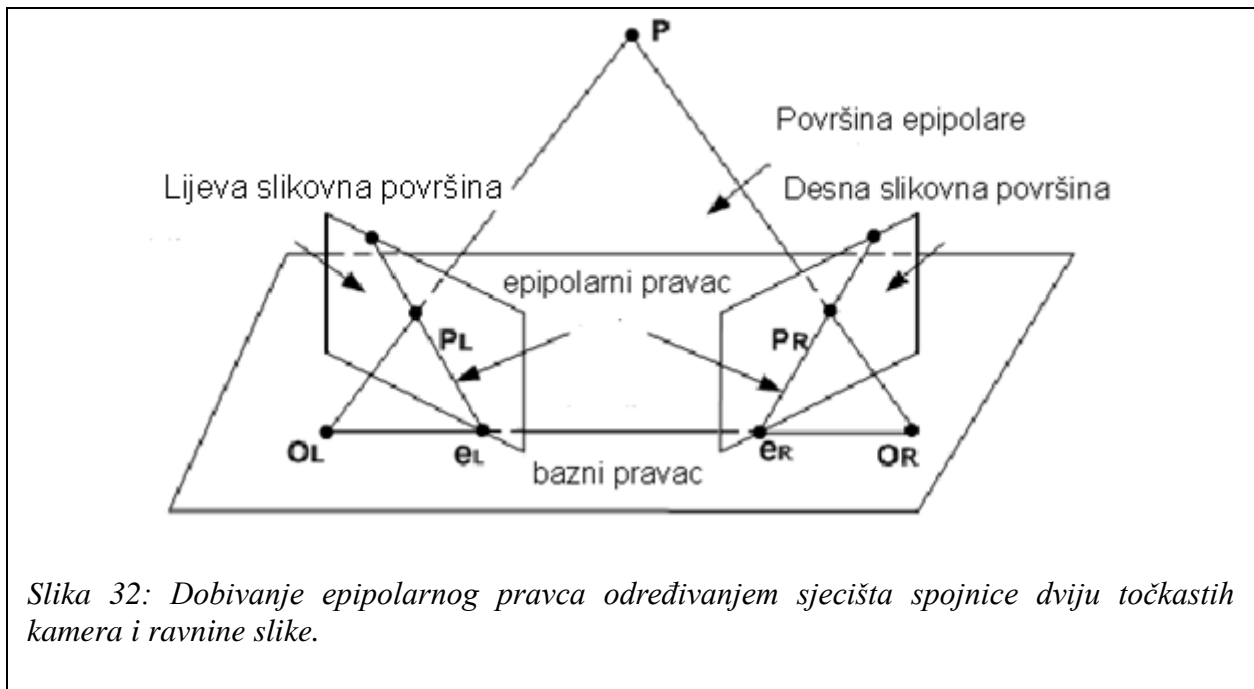
Matrice P_t, P_s - predstavljaju 3x3 matrice kamere asocirane sa prostorom i_t odnosno i_s .

$x_s = [u_s, v_s, 1]^T$ i $x_t = [u_t, v_t, 1]^T$ predstavljaju točke (u, v) u izvorišnoj slici i_s , odnosno odredišnoj i_t .

C_s i C_t su centri projekcije točkastih kamera asocirane sa prostorom i_s odnosno i_t .

$\delta_s(u_s, v_s)$ je generalno nepodudaranje izvornog slikovnog elementa (u_s, v_s) .

Relacija pokazuje da se odredišna slika može dobiti primjenom planarno-perspektivne transformacije nad izvornom slikom nakon koje slijedi proporcionalni pomak $\delta_s(u_s, v_s)$ točaka po slikovnom elementu u smjeru epipolarnog pravca. Epipolarni pravac je spojnica sjecišta linije koja spaja pozicije točkaste kamere sa slikovnom prostornom ravninom i točke x koja daje sjecište odgovarajuće kamere i svoje prostorne slikovne ravnine i_k (slika 32).



Ovakva se faktorizacija naziva površina-plus-paralaksa (engl. *plane-plus-parallax*). Preslikavanje teksture je specijalni slučaj trodimenzionalne slikovne deformacije gdje sve točke dijele jednako nepodudaranje $\delta_s(u_s, v_s)$.

Reljefno preslikavanje teksture uvodi faktorizaciju jednadžbe [6.1.3.1.] kako bi se poslije procesa deformacije ispravno primijenilo preslikavanje teksture.

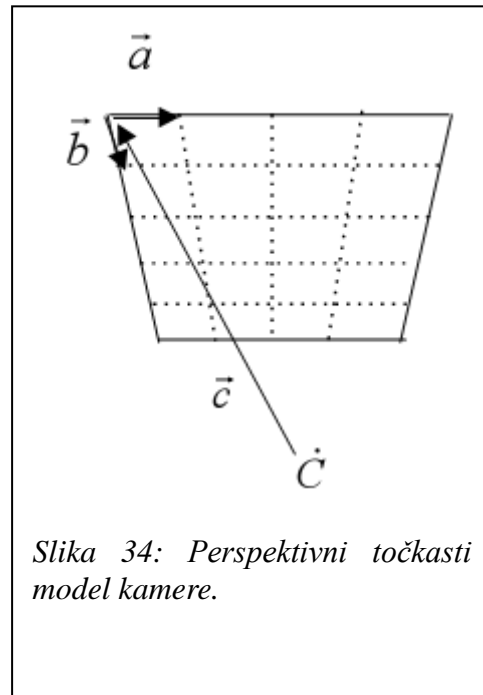
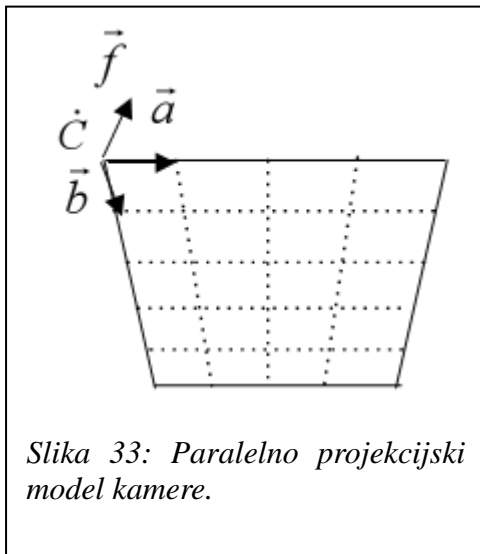
Prva prednost ovog sustava jest upotreba grafičkog sklopovlja za preslikavanje teksture koji će izvesti potrebne transformacije i filtriranje.

Druga prednost je mogućnost da se deformacija prikaže kao jednodimenzionalna slikovna operacija koja će se vršiti po stupcima i recima zahtijevajući samo dva susjedna elementa teksture.

Treća prednost je što se tehnika prirodno integrira sa grafičkim sučeljem.

6.1.1.4. Relacija za opis "Pred-deformacije"

Kako bi se dodala idealna faktorizacija potrebno je prethodno pronaći "pred-deformaciju" p . Kompozicija te deformacije p i preslikavanja teksture m je ekvivalentno trodimenzionalnoj slikovnoj deformaciji w prethodno definiranoj, tj. $w = m \cdot p$. Neka su $(u_i, v_i) = (u_s + \Delta u, v_s + \Delta v)$ među koordinate dobivene nakon pomaka izvornih koordinata (u_s, v_s) za pomak $(\Delta u, \Delta v)$. Tokom deformacije se elementi teksture pomiču okomito i vodoravno u prostoru teksture. Pomiču se za količinu koja se dobije ovisno o visinskoj karakteristici i karakteristici kamere. U reljefnom preslikavanju teksture umjesto perspektivne točkaste kamere koristimo model paralelne projekcijske kamere (slika 33).



Vektori a i b predstavljaju bazu za slikovni prostori i . Vektor f u tom paralelnom projekcijskom modelu kamere predstavlja vektor okomit na slikovni prostor i . Uzimajući u obzir paralelno projekcijski model točka \dot{x} se dobiva iz izraza:

$$\dot{x} = \dot{C}_s + \begin{bmatrix} a_{st} & b_{st} & f_{st} \\ a_{sj} & b_{sj} & f_{sj} \\ a_{sk} & b_{sk} & f_{sk} \end{bmatrix} \begin{bmatrix} u_s \\ v_s \\ displ(u_s, v_s) \end{bmatrix} = \dot{C}_s + P'_s \vec{x}'_s \quad [6.1.1.4.1.]$$

Vektori a , b i f su prethodno definirani. \dot{C}_s predstavlja ishodište slikovne prostorne ravnine. $Displ(u_s, v_s)$ je ortogonalni pomak definiran dubinskom karakteristikom točke (u_s, v_s) .

Projekcija koordinata iz izvorišnog sustava i_s u odredišnu perspektivnu projekcijsku kameru je dana izrazom:

$$u_t = \frac{Au_s + Bv_s + D + C' \text{displ}(u_s, v_s)}{Iu_s + Jv_s + L + K' \text{displ}(u_s, v_s)}$$

$$v_t = \frac{Eu_s + Fv_s + H + G' \text{displ}(u_s, v_s)}{Iu_s + Jv_s + L + K' \text{displ}(u_s, v_s)}$$

Gdje:

$$A = \vec{a}_s \cdot (\vec{b}_t \times \vec{c}_t), B = \vec{b}_s \cdot (\vec{b}_t \times \vec{c}_t), C = \vec{f}_s \cdot (\vec{b}_t \times \vec{c}_t),$$

$$D = (\dot{C}_s - \dot{C}_t) \cdot (\vec{b}_t \times \vec{c}_t), E = \vec{a}_s \cdot (\vec{c}_t \times \vec{a}_t), F = \vec{b}_s \cdot (\vec{c}_t \times \vec{a}_t),$$

$$G' = \vec{f}_s \cdot (\vec{c}_t \times \vec{a}_t), H = (\dot{C}_s - \dot{C}_t) \cdot (\vec{c}_t \times \vec{a}_t), I = \vec{a}_s \cdot (\vec{a}_t \times \vec{b}_t),$$

$$J = \vec{b}_s \cdot (\vec{a}_t \times \vec{b}_t), K' = \vec{f}_s \cdot (\vec{a}_t \times \vec{b}_t), L = (\dot{C}_s - \dot{C}_t) \cdot (\vec{a}_t \times \vec{b}_t)$$

Odgovarajući će se izrazi za preslikavanje teksture dobiti iz prethodnih izraza tako da se $\text{displ}(u_s, v_s)$ izjednači s 0 za sve slikovne elemente izvorne slike.

$$\frac{Au_i + Bv_i + D}{Iu_i + Jv_i + L} = \frac{Au_s + Bv_s + D + C' \text{displ}(u_s, v_s)}{Iu_s + Jv_s + L + K' \text{displ}(u_s, v_s)}$$

$$\frac{Eu_i + Fv_i + H}{Iu_i + Jv_i + L} = \frac{Eu_s + Fv_s + H + G' \text{displ}(u_s, v_s)}{Iu_s + Jv_s + L + K' \text{displ}(u_s, v_s)}$$

Rješavanjem gornjih izraza dolazimo do vrijednosti (u_i, v_i) :

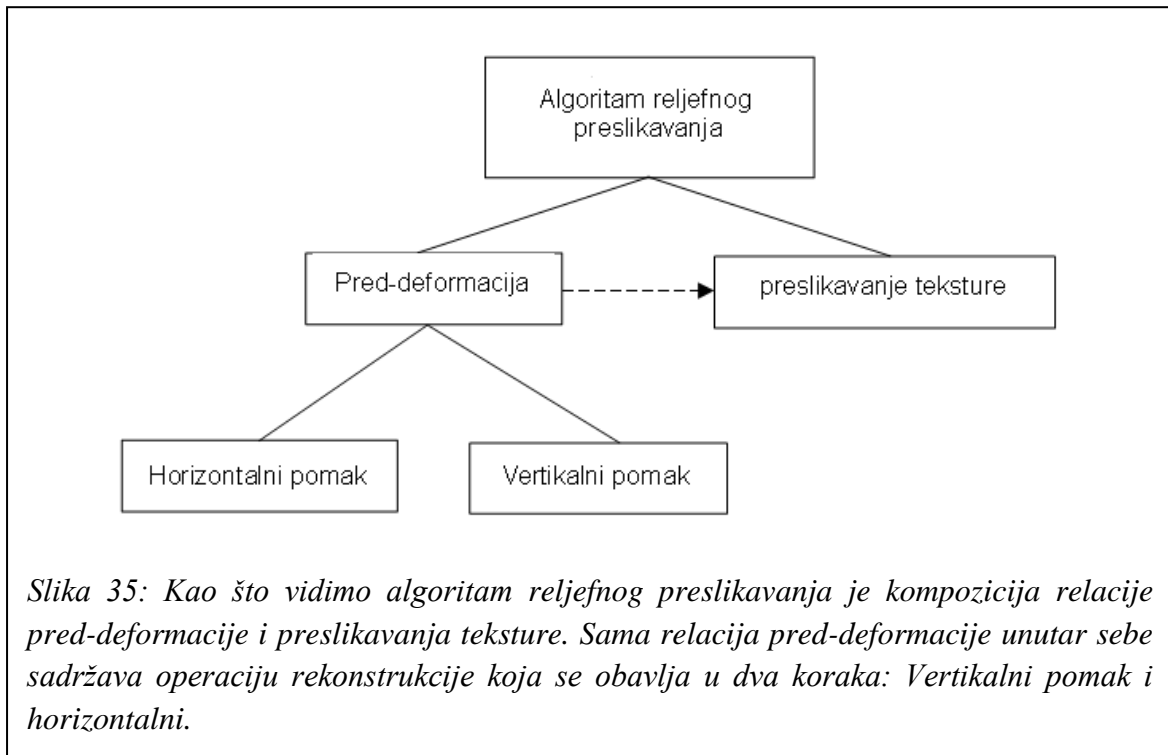
$$u_i = \frac{u_s + k_1 \text{displ}(u_s, v_s)}{1 + k_3 \text{displ}(u_s, v_s)}$$

$$v_i = \frac{v_s + k_2 \text{displ}(u_s, v_s)}{1 + k_3 \text{displ}(u_s, v_s)}$$

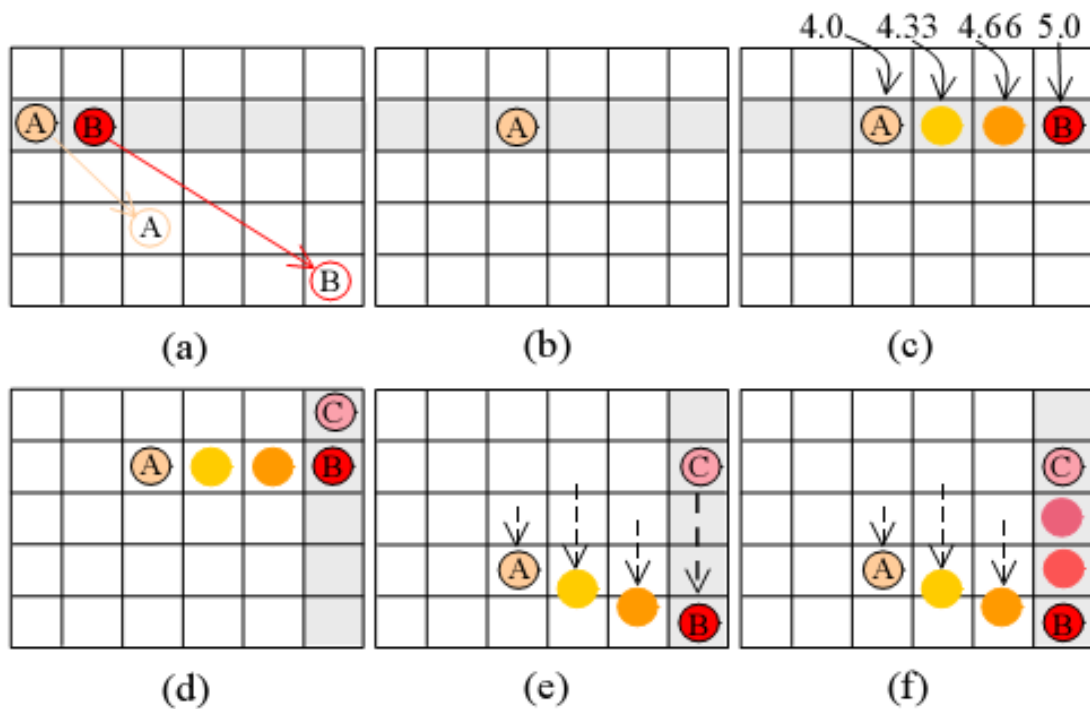
k_1, k_2 i k_3 predstavljaju konstante koje opisuju karakteristike odredišne i izvorne kamere, te zajedno sa $\text{displ}(u_s, v_s)$ čine pomak $(\Delta u, \Delta v)$ u izvornim elementima teksture.

6.1.1.5. Rekonstrukcija pred-deformirane teksture

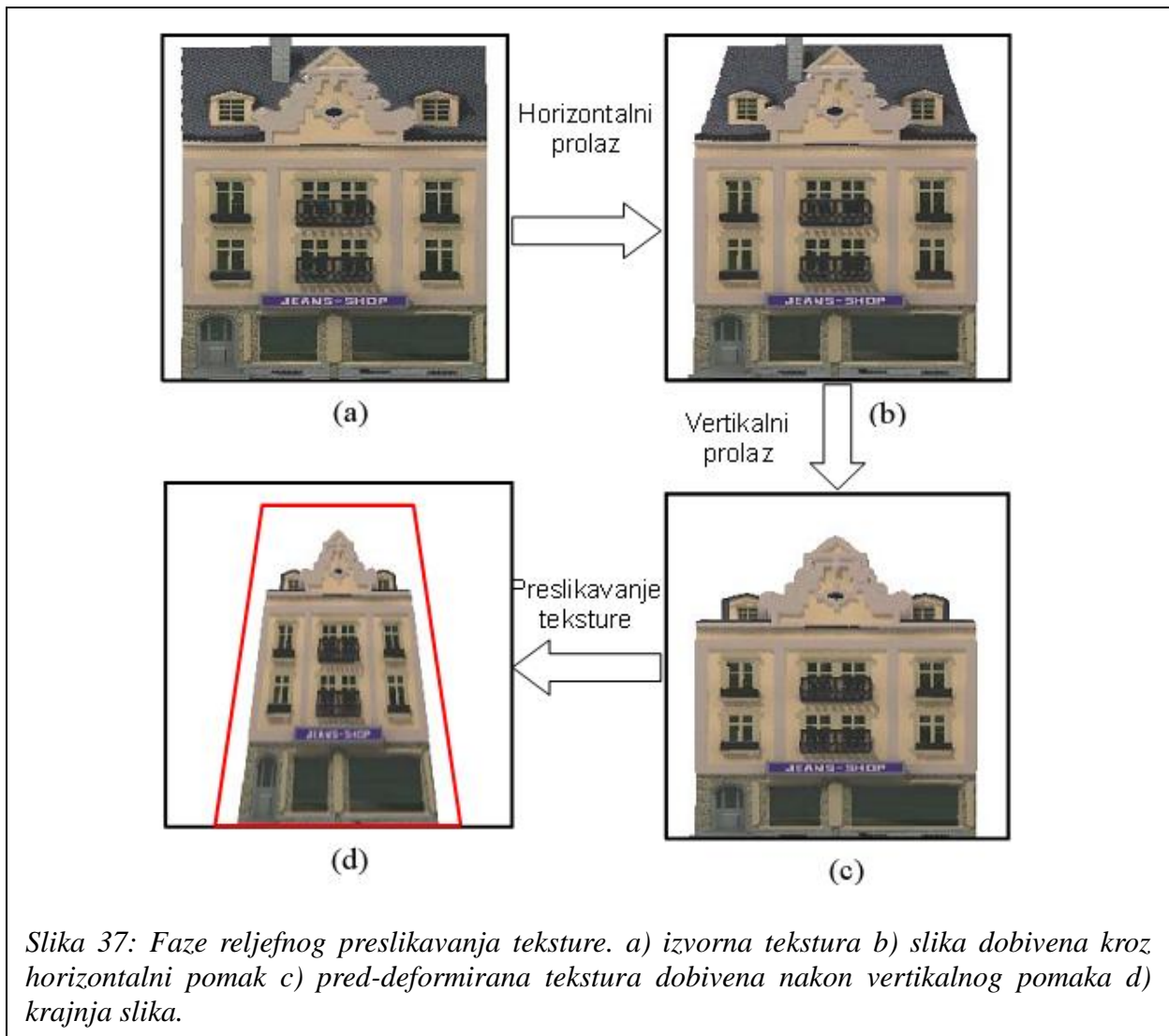
Računanje među točaka je samo jedna faza "pred-deformacije". Druga mnogo skuplja faza jest rekonstrukcija podataka u tablicu slikovnih elemenata među-slike (slika 35). Rekonstrukcija rješava problem neželjenih učinaka uslijed diskretizacije (engl. *anti aliasing*). Specijalna struktura "pred-deformacijske" relacije omogućuje implementaciju rekonstrukcije kao dvofazni proces. Proces se sastoji od horizontalnog prolaza i vertikalnog prolaza.



Slika 35: Kao što vidimo algoritam reljefnog preslikavanja je kompozicija relacije pred-deformacije i preslikavanja teksture. Sama relacija pred-deformacije unutar sebe sadržava operaciju rekonstrukcije koja se obavlja u dva koraka: Vertikalni pomak i horizontalni.



Slika 36: Deformacija jednog elementa teksture. a) elementi teksture u izvornoj teksturi i njihovi ekvivalenti nakon pred-deformacije. b) prvi element teksture u trenutnom retku se prebacuje u svoj krajnji stupac c) drugi element teksture se prebacuje u svoj krajnji stupac i vrši se interpolacija boje tijekom rasterizacije. d) nakon što su svi reci deformirani, element teksture C se prilagođava prema elementu teksture B e) duž svih stupaca elementi teksture se pomiču prema krajnjim recima. f) Boja se interpolira tokom rasterizacije.



Pseudokod algoritma pred-deformacije (C-boja, D-odmak, (U,V) – koordinate teksture):

dohvati U_{in} , V_{in} , C_{in} , D_{in}

$U_{next} = \text{relacija_za_izracun_ui}(U_{in}, D_{in})$

$V_{next} = \text{relacija_za_izracun_vi}(V_{in}, D_{in})$

za ($U_{out} = \text{integer}(U_{prev} + 1)$; $U_{out} \leq U_{next}$; $U_{out}++$) {

 linearno interpoliraj C_{out} između C_{prev} i C_{in}

 linearno interpoliraj V_{out} između V_{prev} i V_{in}

 postavi C_{out} i V_{out} kod U_{out}

$U_{prev} = U_{next}$; $V_{prev} = V_{next}$; $C_{prev} = C_{next}$;

}

6.1.2. Implementacija reljefnog preslikavanja u stvarnom vremenu

U ovom pristupu se preslikavanje teksture obavlja u vektorskom prostoru tangente. Zbog toga se tehnika može primijeniti na zakrivljenim površinama proizvodeći ispravno samo zatamnjenje, sjene i osvjetljenja obavljenog nad slikovnim elementom. Tehnika se zasniva na efikasnom algoritmu za određivanje sjecišta dubinske karakteristike i vektora pogleda koji se temelji na inverznom postupku (gledamo iz slikovnog elementa) praćenja zrake.

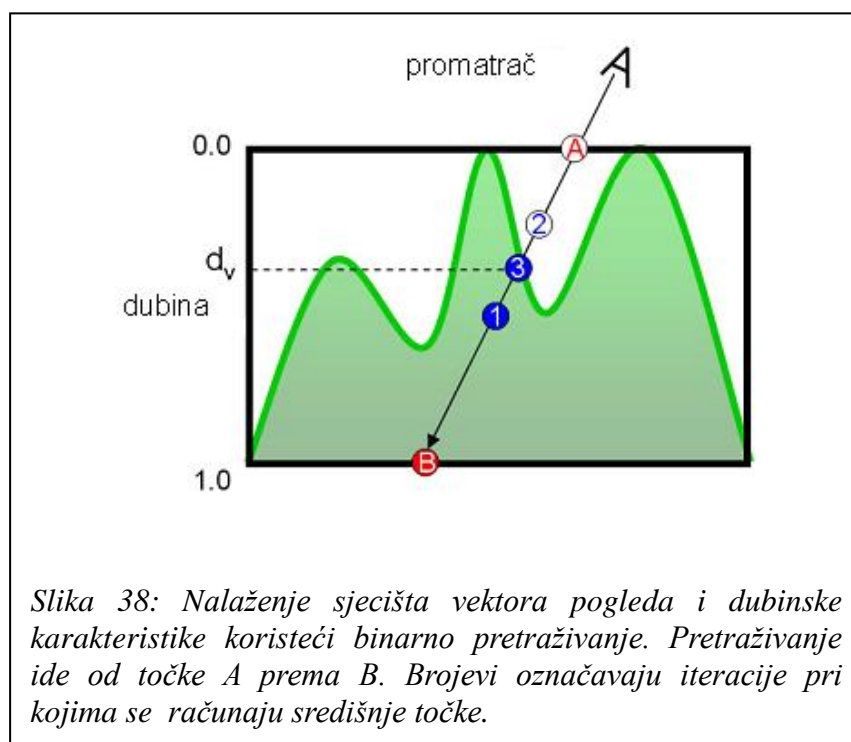
Jako dobro svojstvo ove implementacije je što podržava ispravno gledanje dodanih trodimenzionalnih svojstava površina pri jakom malim udaljenostima i kutovima.

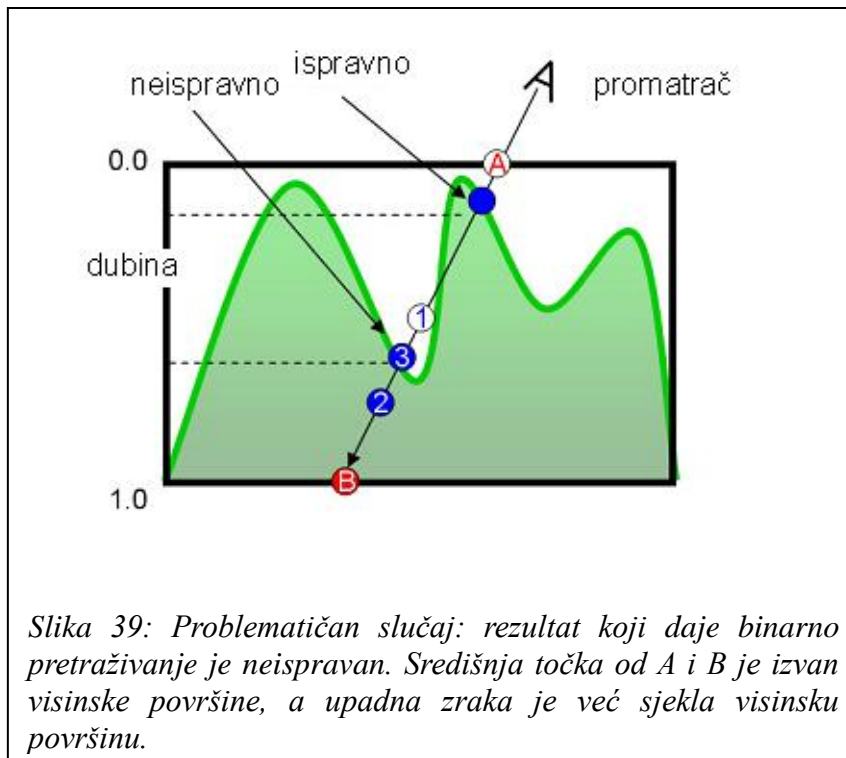
Ulazne vrijednosti u algoritam predstavljaju tekstura s dubinskom karakteristikom i mapom normala i tekstura s bojom.

Proces preslikavanja reljefnih podataka na proizvoljnu površinu temelji se na sljedećim koracima:

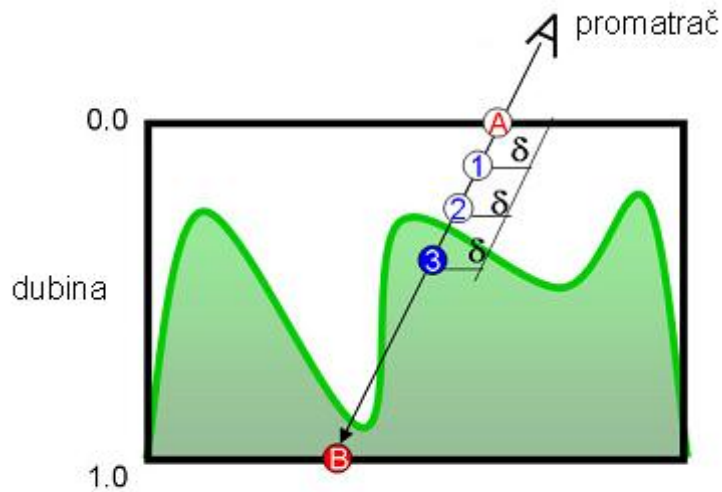
Za svaki fragment koji će biti iscrtan:

- izračunaj vektor pogleda (engl. *VD-viewing direction*) kao vektor od gledišta do točke na proizvoljnoj poligonalnoj površini.
- prebaci vektor pogleda VD u vektorski prostor tangente koji je asociran s trenutnim fragmentom koji se obrađuje
- koristi VD' (transformirani VD) i $A, (s,t)$ koordinatu teksture fragmenta koji se trenutno obrađuje, kako bi izračunao $B, (u,v)$ koordinatu teksture gdje zraka dohvaća dubinsku vrijednost 1.0 .
- izračunaj sjecište između VD' i dubinske karakteristike koristeći algoritam za binarno pretraživanje počevši od A do B (slika 38).
- obavi sjenčanje fragmenta koristeći attribute (normala, boja, visina) asocirane s koordinatom teksture koja se dobije iz prethodno izračunatog sjecišta.





Točka A ima dodijeljenu dubinsku vrijednost 0.0, a točka B 1.0. U svakom se koraku računa središnja točka za dani interval, te joj se dodjeli aritmetička vrijednost dubina i koordinata teksture krajnjih točaka. Krug označen sa 1 predstavlja prvu središnju točku. Središnja vrijednost koordinata teksture služi za dobivanje dubinske vrijednosti u teksturi s dubinskim karakteristikama. Ako je spremljena vrijednost manja od izračunate, točka duž zrake je unutar visinske površine kao u slučaju točke 1. Binarno pretraživanje nastavlja svoje izvođenje dok god je jedna krajnja vrijednost unutar, a druga izvan visinske površine. Broj unutar kruga označava redoslijed unutar kojeg su dohvaćene vrijednosti. U praksi se često događa da je oko 8 iteracija dovoljno za dobivanje prihvatljivih rezultata. Obavljanje 8 iteracija označava podjelu dubinskog opsega na 256 jednakih dijelova. Binarno pretraživanje, prethodno opisano, može dovesti do grešaka ako upadna zraka siječe dubinsku karakteristiku u više točaka (slika 39). Kako bi se izbjegli ovakvi pogrešni rezultati započinjemo s linearnim pretraživanjem (slika 40). Pretraživanje počinje od točke A i vršimo pomake od δ puta dužine AB tražeći prvu točku presjeka. Linearno se pretraživanje za pristup vrijednostima može izvoditi paralelno, pošto ne postoji uvjetni pristup koordinatama teksture.



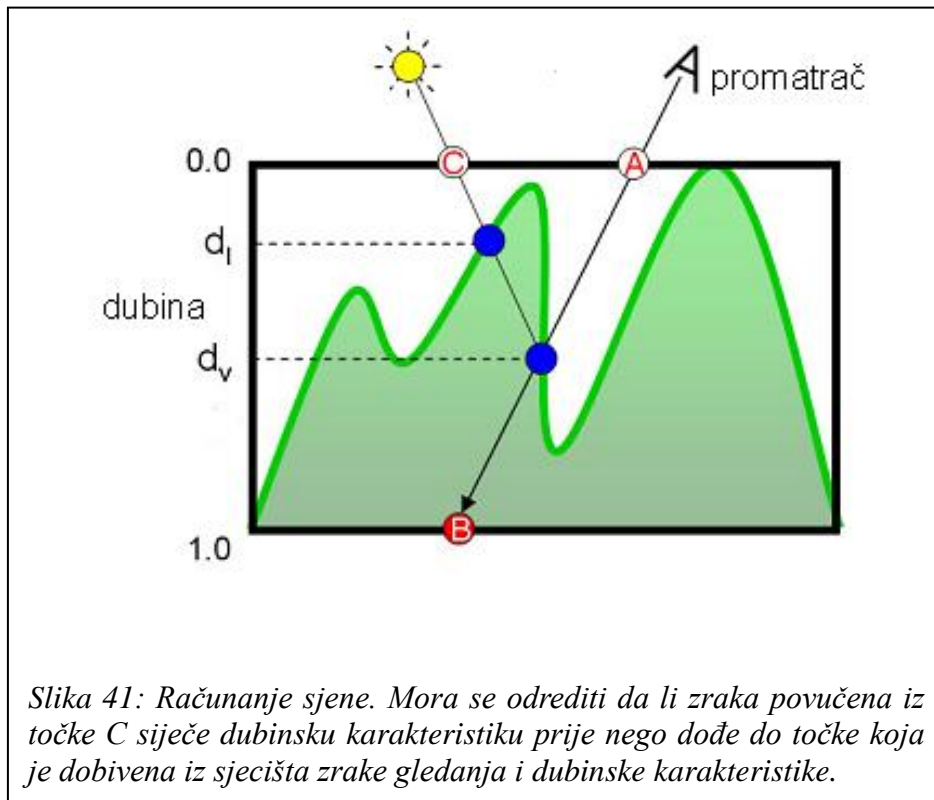
Slika 40: Linearno pretraživanje duž pravca AB za početnu točku A.

Jednom kada je pronađeno prvo sjecište, binarno pretraživanje koristi posljednju točku izvan površine i trenutnu točku.

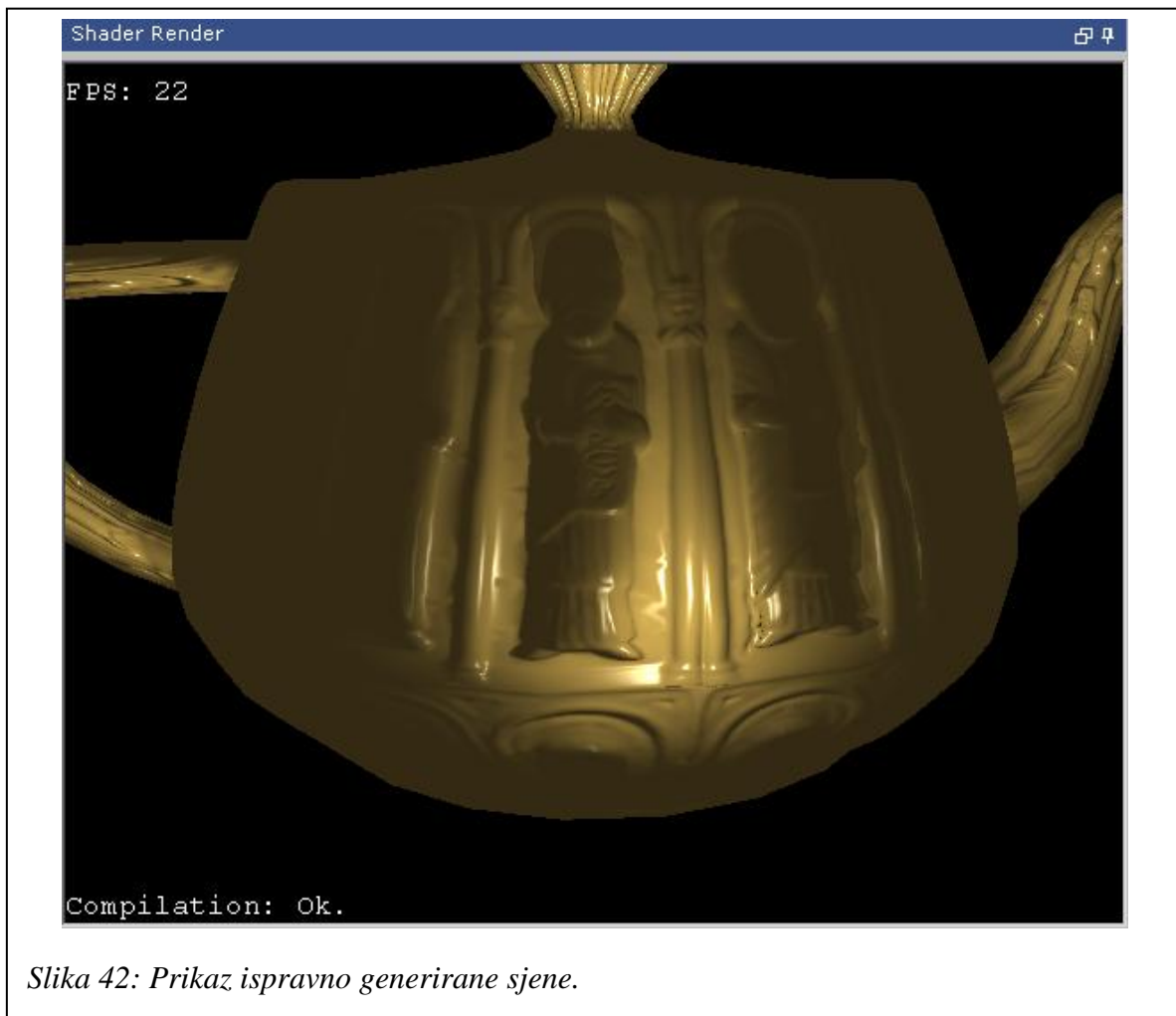
6.1.3. Samostalno bacanje sjene

Postoji jednostavan algoritam za određivanje da li je neki fragment u sjeni ili ne. Unutar ovog odjela ćemo dati kratak opis algoritma.

U ovom slučaju koristimo zraku svjetlosti i tražimo da li zraka siječe visinsku površinu duž puta prema točki koja se trenutno obrađuje (slika 41). Ako siječe, točka koja se obrađuje je u sjeni, inače nije.



Primjer dobiven s bacanjem sjene:



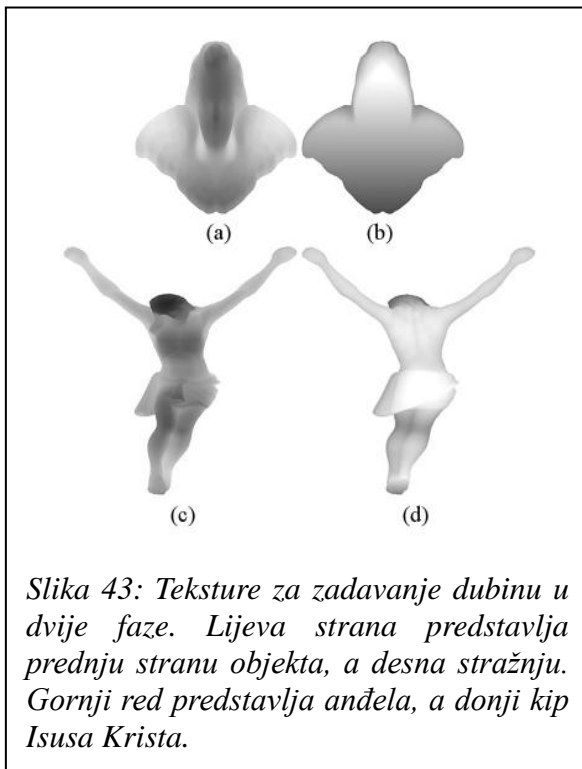
Slika 42: Prikaz ispravno generirane sjene.

6.1.4. Ostale mogućnosti reljefnog preslikavanja

6.1.4.1. Reljefno preslikavanje sa dubinskom karakteristikom definiranom u dva sloja za ispravan prikaz silueta

Jedna jako dobra tehnika za obradu modela pomoću reljefnog preslikavanja teksture je reljefno preslikavanje teksture gdje se dubinska karakteristika zadaje u dvije faze. U tom slučaju imamo zadane dvije dubinske karakteristike. Kada imamo samo jednu reljefnu teksturu nemamo dovoljno informacija kako bi mogli u potpunosti modelirati cijeli objekt. Tada ne postoji informacija "što leži iza". Kako bi mogli opisati u potpunosti objekt definiramo dodatnu dubinsku karakteristiku koja će sadržavati dubine stražnje strane objekta. Točka je unutar objekta ako:

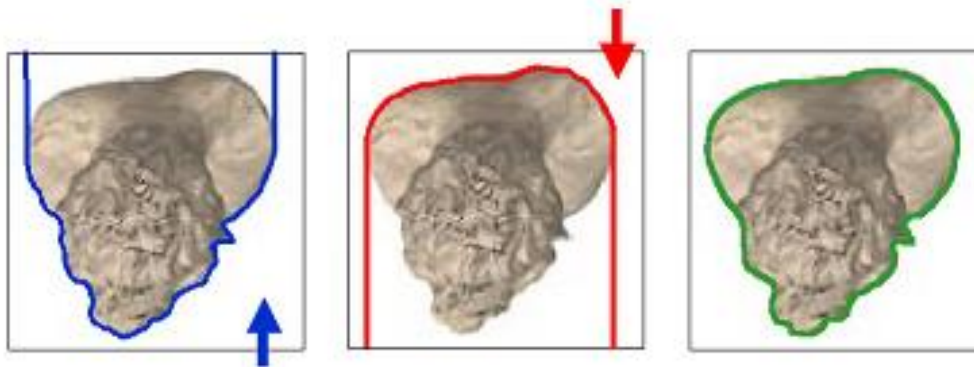
dubina prednje strane \leq dubina točke koja se obrađuje \leq dubina stražnje strane



Slika 43: Teksture za zadavanje dubinu u dvije faze. Lijeva strana predstavlja prednju stranu objekta, a desna stražnju. Gornji red predstavlja anđela, a donji kip Isusa Krista.



Slika 44: Lijeva strana predstavlja objekt nad kojem je primijenjeno obično reljefno preslikavanje dok desna strana predstavlja preslikavanje dubine u dvije faze.

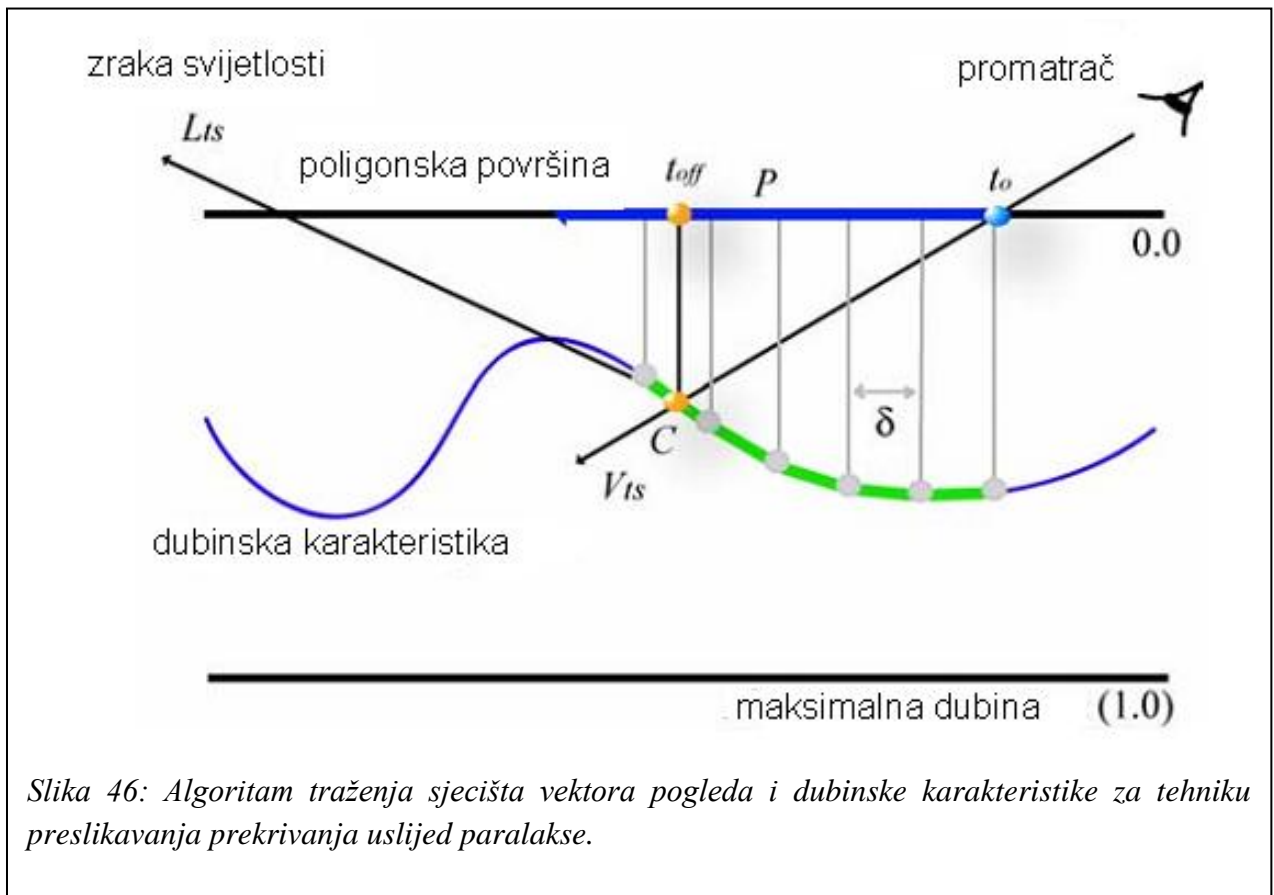


Slika 45: Reljefno preslikavanje s dvofaznom dubinskom karakteristikom. Kombinirano se koristi prednji i stražnji dubinski sloj.

6.2. Druge tehnike preslikavanja uz efekt paralakse praćenjem zrake pogleda

6.2.1. Preslikavanje prekrivanja uslijed paralakse

Preslikavanje prekrivanja uslijed paralakse kao što smo već rekli spada u isti skup tehnika kao reljefno preslikavanje. Tehnika kao reljefno preslikavanje koristi jednake ulaze te vrši praćenje zrake od minimalne dubine (nad površinom) do sjecišta. Jedina razlika kao što vidimo na slici 46 jest što ne koristi binarno pretraživanje već koristi samo linearno pretraživanje koje se kreće duž unaprijed određenog broja uzoraka.



Slika 46: Algoritam traženja sjecišta vektora pogleda i dubinske karakteristike za tehniku preslikavanja prekrivanja uslijed paralakse.

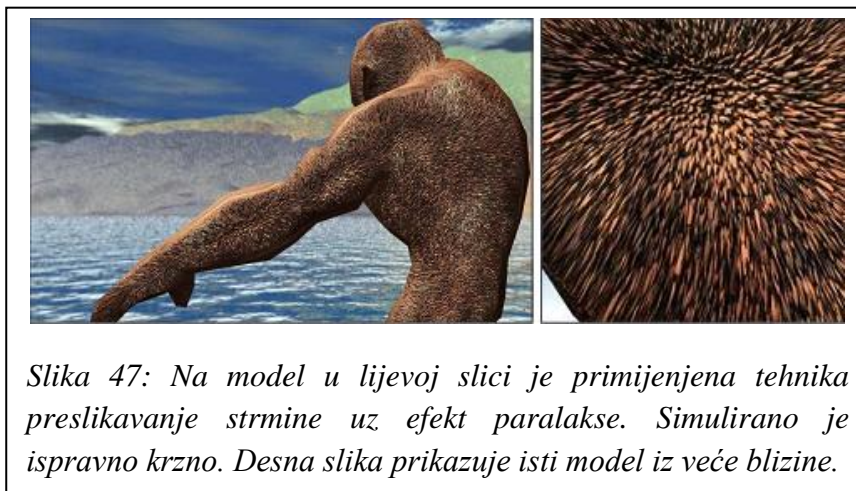
Algoritam traženja sjecišta počinje u točki t_0 zatim se kreće duž vektora pogleda V_{ts} u intervalima δ radeći time linearno pretraživanje. Svakim intervalom dobivamo nove dvije točke koji su potencijalni kandidati za sjecište. Za dvije točke ispitujemo da li je unutar njih sjecište na način da izvadimo njihove dubinske vrijednosti iz dubinske karakteristike te vidimo da li su poviše trenutne dubine ili ne. Jednom kad nađemo takav par točaka računamo sjecište vektora pogleda i tog linearnog segmenta i tako dolazimo do točke C . Točka C je točka koju bi promatrač trebao vidjeti.

6.3. Preslikavanje strminom uz efekt paralakse

Ova tehnika predstavlja još jednu tehniku unutar skupa tehnika preslikavanja praćenjem zrake pogleda uz efekt paralakse. Tehnika se kao i prethodna temelji na linearnom pretraživanju. Jedinu razliku koju uvodi jest da vodi računa od stupnja detaljnosti (level of detail). Kada dolazi do dohvaćanje elementa iz teksture, bilo dubina ili boja, izvodi se pomoću funkcije koja uzima u obzir detaljnost. Pseudokod koji opisuje tehniku:

```
float step = 1.0 / n
vec2 dt = E.xy * bumpScale / (n * E.z)
float height = 1.0;
vec2 t = texCoord.xy;
vec4 nb = texture2DLod(NB, t, LOD);
while (nb.a < height) {
    height -= step; t += dt;
    nb = texture2DLod(NB, t, LOD);
}
// ... Shade using N = nb.rgb
```

Vektor E predstavlja vektor pogleda. Algoritam kao i u drugim dosad opisanim kreće od površine prema promatraču u smjeru vektora pogleda. Kreće se u koracima od $1/n$ i pomiče se svakom iteracijom za dt u smjeru vektora pogleda, dok ne dođe do sjecišta.

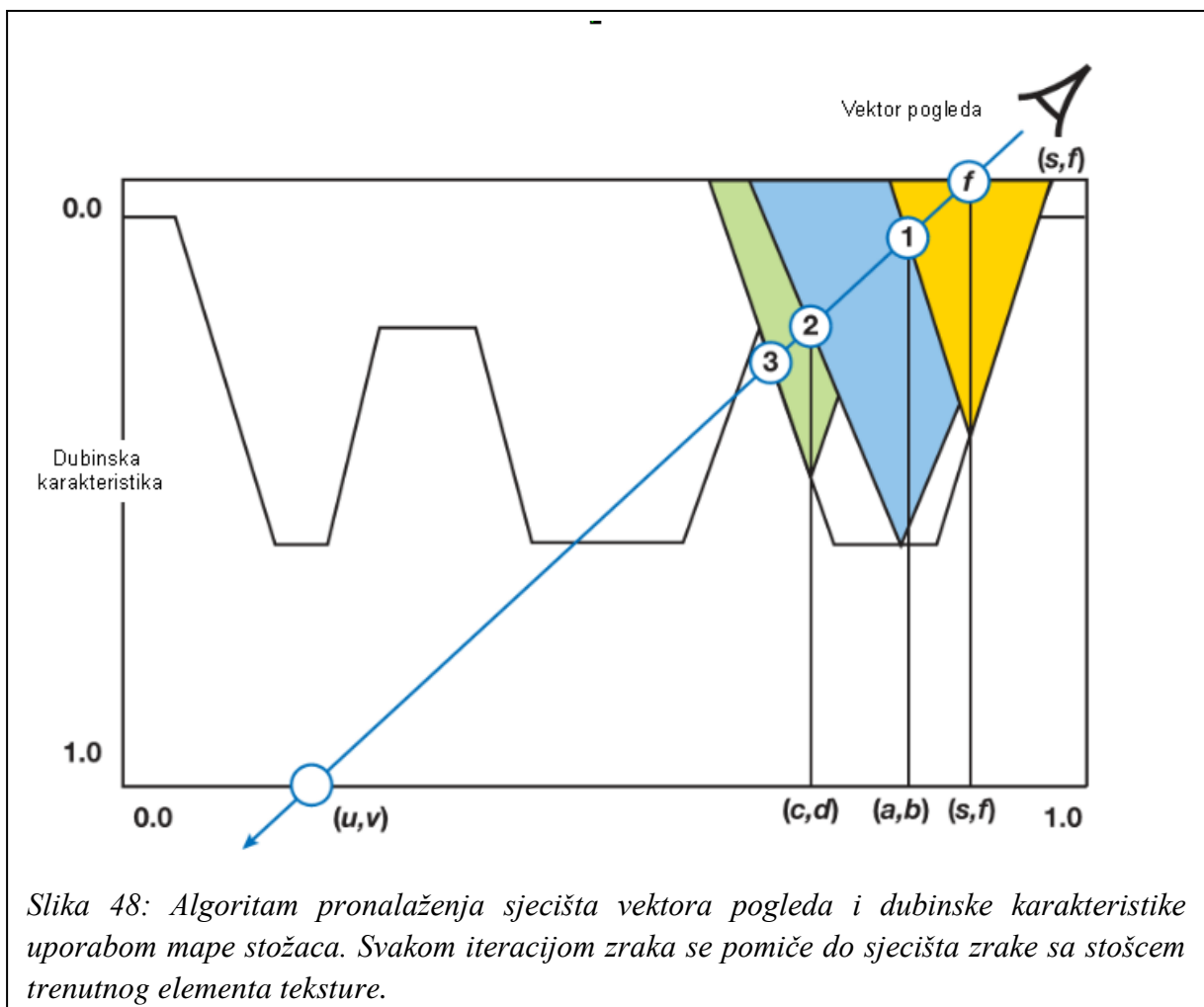


6.4. Optimizacija metode pronalaženja sjecišta

6.4.1. Optimizacija pronalaženja sjecišta uporabom mape stožaca

Reljefno preslikavanje u svim svojim varijacijama upotrebe (prikazivanje proizvoljnih poligonskih površina, ispravno prikazivanje siluete ili prikaz površinskih detalja koji nisu opisane dubinskom karakteristikom) uvijek koristi isti način nalaženja sjecišta. Algoritam pronalaženja sjecišta vektora pogleda i dubinske karakteristike temelji se kao što smo vidjeli na binarnom i linearnom pretraživanju. Linearno pretraživanje je jako pogodno stvaranju neželjenih učinaka uslijed diskretizacije (engl. *aliasing*). Od tada su smišljene mnogi algoritmi koji pokušavaju riješiti problem efikasnijeg i točnijeg pronalaženja sjecišta. Jedan takav algoritam je algoritam pronalaženja sjecišta uporabom mape stožaca (engl. *Cone step mapping*). Algoritam se temelji na tome da se koristi unaprijed izračunata mapa stožaca (engl. *cone map*).

Mapa stožaca asocira svakom elementu unutar teksture koja sadržava dubinsku karakteristiku jedan kružni stožac. Kut stošca je maksimalni kut koji stožac ima a da ne siječe dubinsku karakteristiku površine (slika 48).



Algoritam osigurava da će uvijek doći do sjecišta, no može imati previše koraka. Zbog toga se uvodi maksimalni broj koraka. To uvođenje maksimalnog broja koraka ima za posljedicu da se nekad neće sjecište točno izračunati što dovodi do grešaka. Taj problem se može riješiti uvođenjem dodatnog binarnog pretraživanja posljednje među-točke i točke nad površinom.

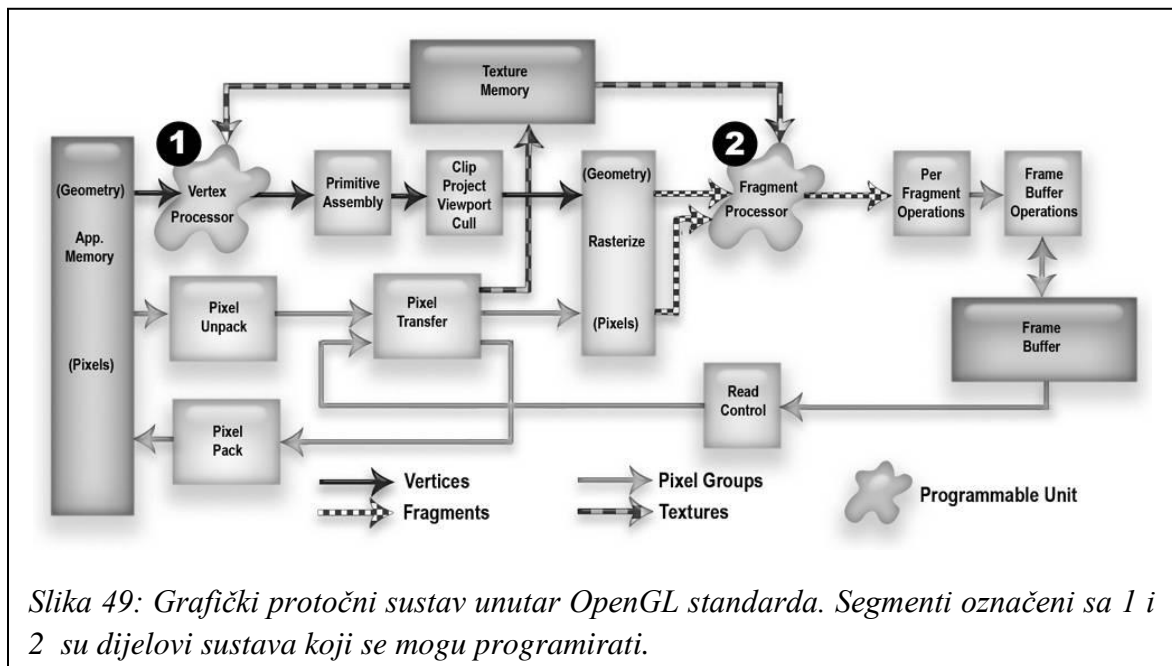
7. Programska podrška

Slijedi kratak opis jezika za sjenčanje dan unutar OpenGL grafičkog standarda i kratki primjeri ispravnih programa napisanih u tom jeziku.

7.1. Programski jezik za sjenčanje GLSL

GLSL (engl. *OpenGL Shading Language*) je unutar OpenGL standarda od verzije 2.0. Omogućava programiranje dviju ključnih faza unutar grafičkog protočnog sustava. Fiksni protok za procesiranje vrhova (engl. *vertex processing*) i fragmenata (engl. *fragment processing*) se zamjenjuje programirljivom jedinicom koja može obavljati sve funkcije tog fiksnog protoka i još mnogo više. Kod generiran jezikom GLSL nazivamo programom za sjenčanje (engl. *shader*). Pošto imamo dvije faze fiksnog protočnog sustava kojih zamjenjujemo, razlikujemo program za sjenčanje vrhova (engl. *vertex shader*) i program za sjenčanje fragmenata (engl. *fragment shader*). Unutar ta dva programa za sjenčanje koristimo jednak jezik uz mali skup promjena.

Kao što vidimo na slici 49 dijelovi protočnog sustava koji se mogu programirati su procesiranje vrhova i procesiranje fragmenata. Podaci putuju od aplikacije prema procesoru vrhova, nadalje prema procesoru fragmenata do slikovne prikazne memorije (engl. *frame buffer*). Jezik za sjenčanje GLSL je dizajniran na način da omogućuje paralelno izvođenje procesiranja vrhova i fragmenata.



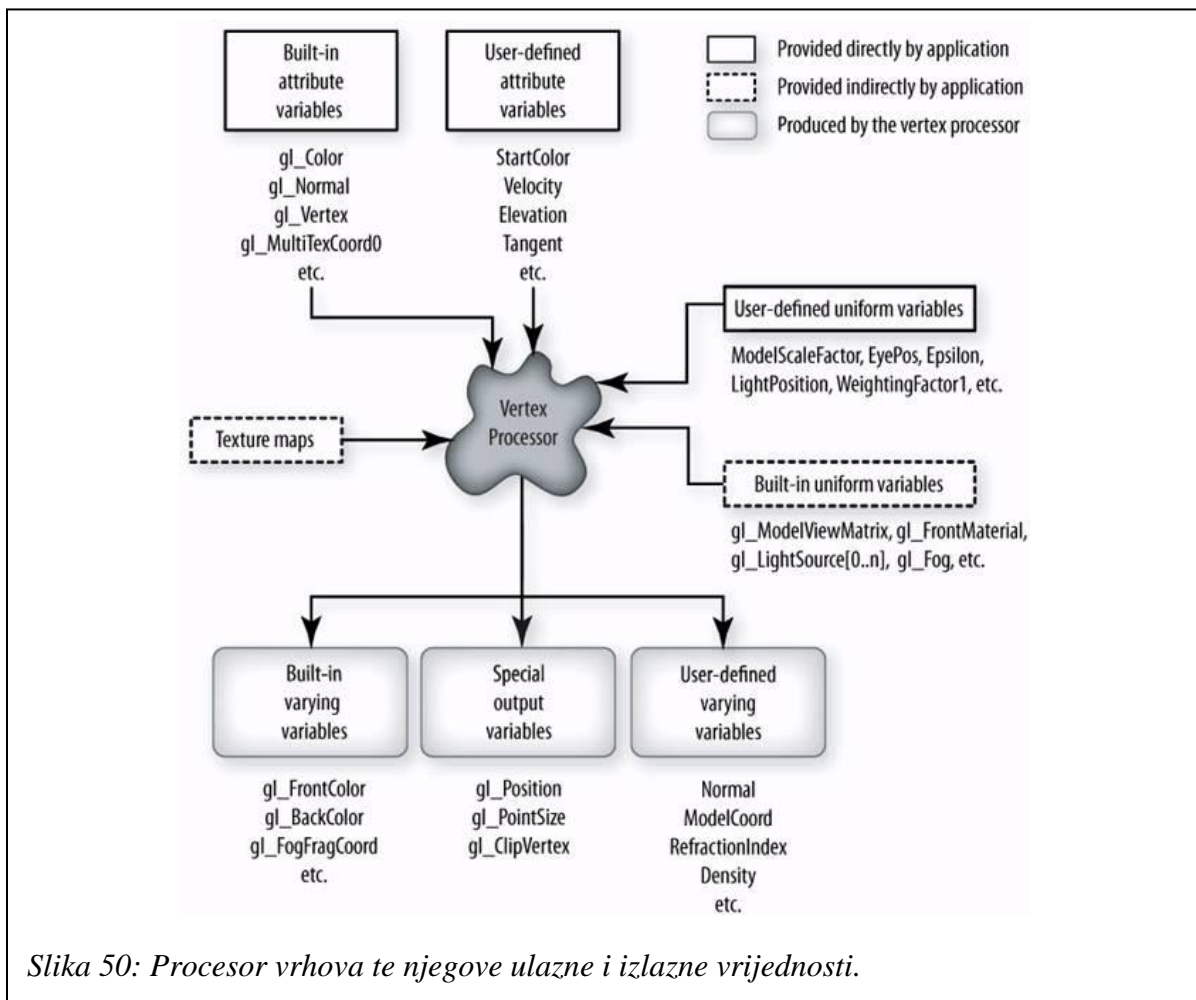
Slika 49: Grafički protočni sustav unutar OpenGL standarda. Segmenti označeni sa 1 i 2 su dijelovi sustava koji se mogu programirati.

7.1.1. Procesiranje vrhova

Procesor vrhova je programibilna jedinica koja operira na ulaznim vrhovima i podacima pridruženi tim vrhovima. On obavlja klasične grafičke operacije kao što su:

- Transformacija vrhova
- Transformacija normala i normalizacija normala
- Generiranje koordinata teksture
- Transformacija koordinata teksture
- Osvjetljenje
- Primjena materijala

Uvođenjem programiranja, fazi se procesiranja vrhova omogućuje izvođenje još raznih drugih proračuna.



Slika 50: Procesor vrhova te njegove ulazne i izlazne vrijednosti.

Kao što vidimo na slici 50 razlikujemo mnoge ulazne i izlazne vrijednosti procesora vrhova. Razlikujemo korisnički postavljene i ugrađene vrijednosti.

- Atributne (engl. *attribute*) varijable su varijable koje se zadaju na razini vrha. Ova vrsta varijabli se odnosi isključivo samo na procesor vrhova.

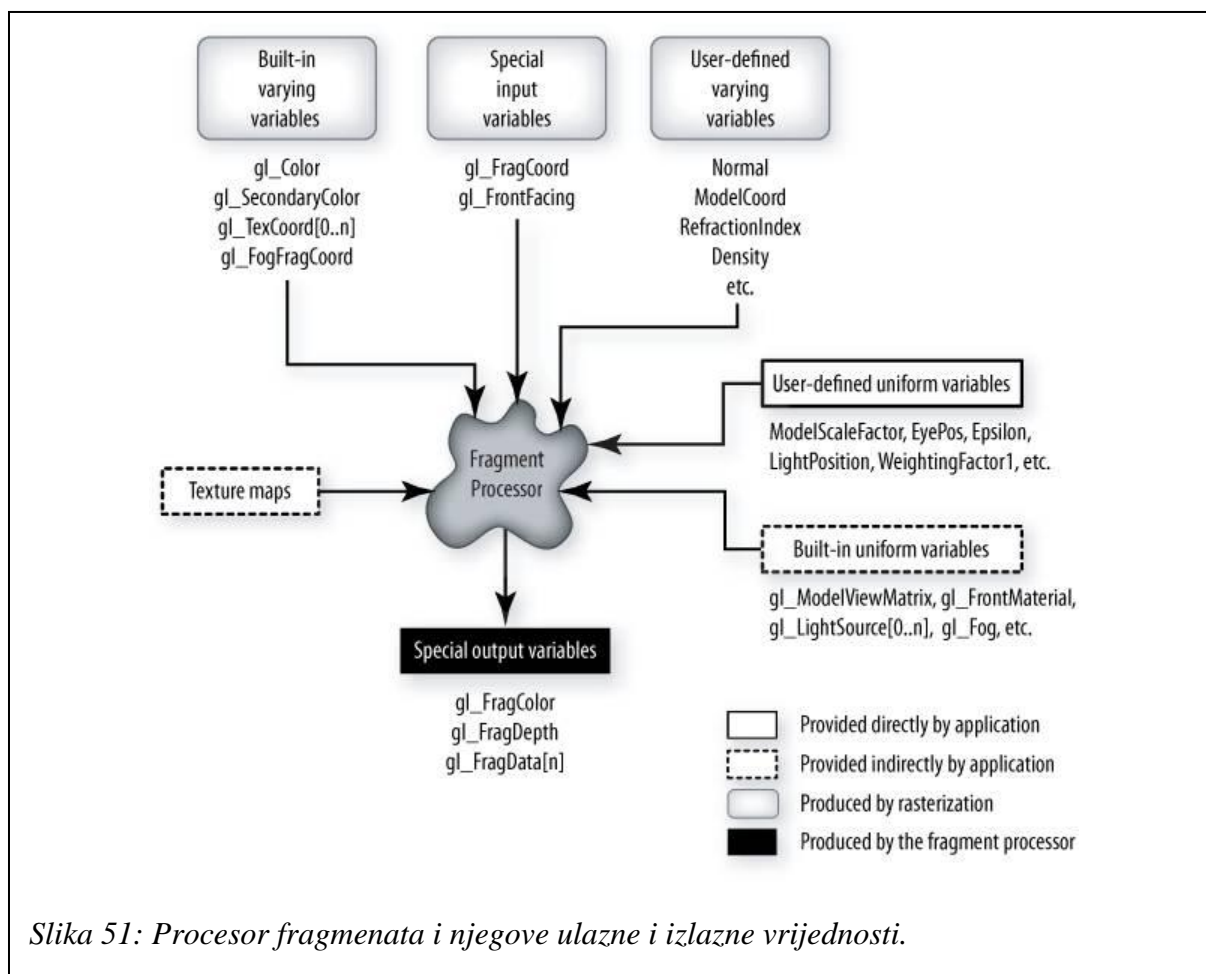
- Uniformne (engl. *uniform*) varijable su varijable zadane na razini primitive. Ove se varijable prenose ili procesoru vrhova ili procesoru fragmenata.
- Promjenjive (engl. *varying*) varijable su varijable koje procesor vrhova prenosi procesoru fragmenata.

7.1.2. Procesiranje fragmenata

Procesor fragmenata je programibilna jedinica koja operira na fragmentima i podacima asocirane tim fragmentima. Procesor fragmenata obavlja klasične operacije kao što su:

- Operacije na interpoliranim vrijednostima
- Dohvat tekstura
- Primjena tekstura
- Magla
- Proračun boje

Kao i kod procesora vrhova uvođenjem programibilnosti mogući su još razni proračuni unutar procesora fragmenata.



Slika 51: Procesor fragmenata i njegove ulazne i izlazne vrijednosti.

Kao što vidimo na slici 51 primarne ulazne vrijednosti su promjenjive vrijednosti. Kao i u procesoru vrhova imamo uniformne ulazne vrijednosti. Izlazne vrijednosti su zadane ugrađenim specijalnim vrijednostima. Za svaki fragment procesor može ili izračunati boju, dubinu i proizvoljne vrijednosti (pisanje u izlazne varijable *gl_FragColor*, *gl_FragDepth* i *gl_FragData*) ili u potpunosti odbaciti fragment.

7.2. Jednostavan primjer programa za sjenčanje u GLSL-u

Program za sjenčanje vrhova:

```
uniform float CoolestTemp;
uniform float TempRange;
uniform float VertexTemp;

varying float temp;

void main(){

    temp = (VertexTemp - CoolestTemp);
    temp /= TempRange;
    gl_Position = glTransform();

}
```

Program za sjenčanje fragmenata:

```
uniform vec3 HottestColor;
uniform vec3 CoolestColor;

varying float temp;

void main(){

    vec3 color = mix(CoolestColor, HottestColor, temp);
    gl_FragColor = vec4(color, 1.0);

}
```

8. Rezultati

Sva testiranja obavljena su na prijenosnom računalu sa sljedećim karakteristikama:

TOSHIBA A200-23P

- Intel Core 2 Duo Mobile Processor T7500 (Cpu brzina:2,2 GHz, Brzina sabirnice: 800MHz, Veličina Cache: 4MB) procesor.
- ATI Mobile Radeon HD 2600 256 MB (PCI Express) grafička kartica
- 2 GB radne memorije
- Windows XP

Svi programi za sjenčanje su obrađeni i iscrtani unutar AMD RenderMonkey 1.82. RenderMonkey je razvojno okruženje za razvijanje programa za sjenčanje i prikaz rezultata u stvarnom vremenu. Omogućava dinamičko učitavanje modela, tekstura i raznih drugih parametara. Parametri se mogu mijenjati tokom iscrtavanja scene i rezultati su odmah vidljivi na modelu.

Unutar ovog poglavlja slijedi usporedba tehnika po pouzdanosti, kvaliteti i zahtjevnosti.

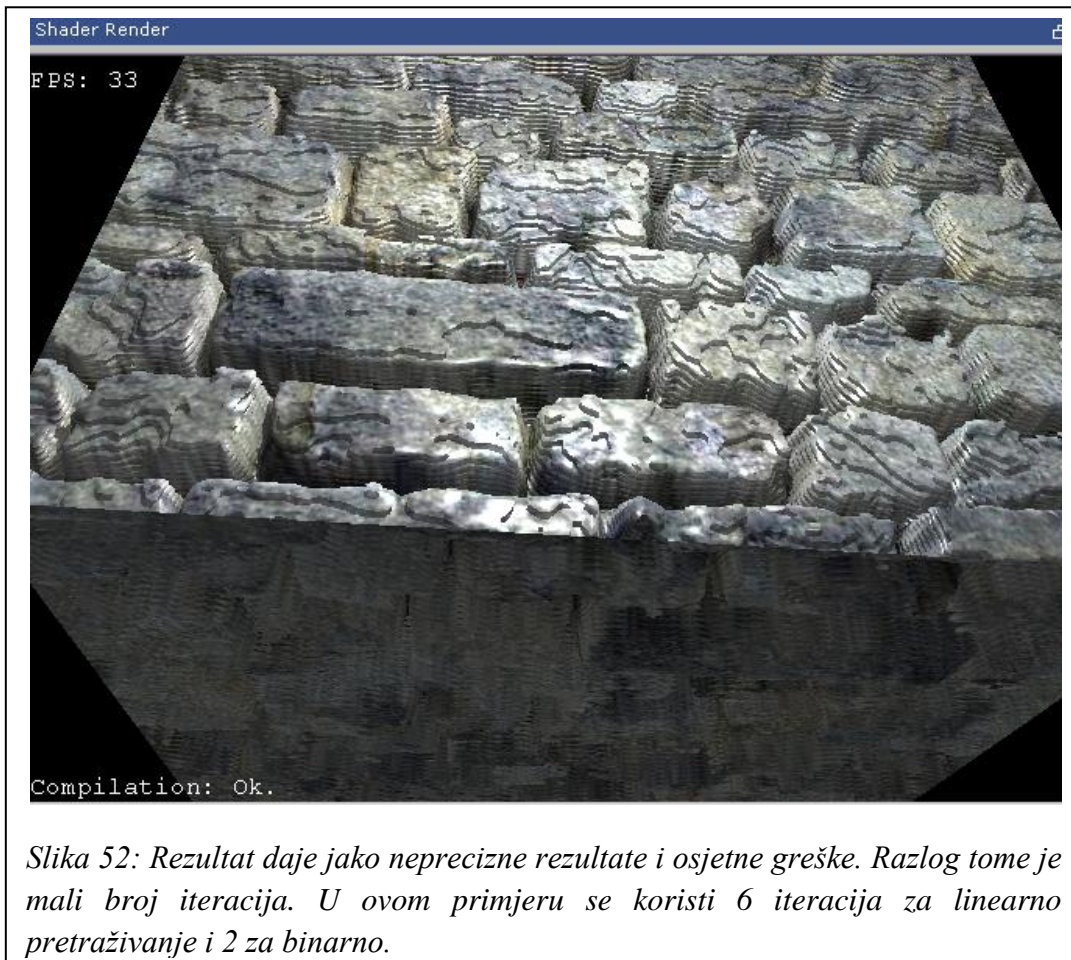
| Naziv tehnike | Osvjetljenje po fragmentu | Brzina izvođenja (1 - 5) | Izvediv u jednom prolazu | Ocjena kvalitete (1 - 5) | Daje ispravnu paralaksu |
|---|---------------------------|--------------------------|--------------------------|--------------------------|-------------------------|
| Preslikavanje teksture | Ne | 5 | Da | 1 | Ne |
| Preslikavanje izbočina | Da | 4 | Da | 2 | Ne |
| Preslikavanje uz efekt paralakse | Da | 3 | Da | 3 | Da |
| Reljefno preslikavanje | Da | 2 | Ne | 4 | Da |
| Reljefno preslikavanje uz upotrebu mape stožaca | Da | 1 | Ne | 5 | Da |

Tablica 1: Usporedba tehnika prema raznim kriterijima.

8.1. Rezultati dobiveni reljefnim preslikavanjem i greške dobivene krivim parametrima

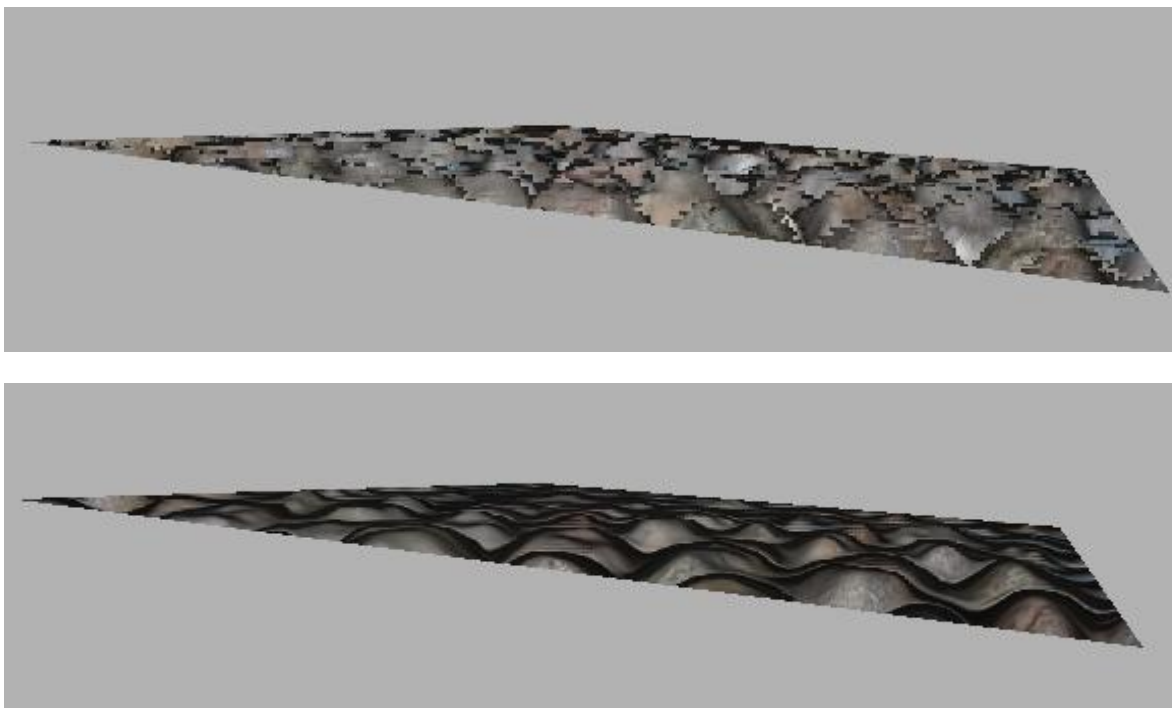
Rezultati dobiveni reljefnim preslikavanjem jako ovise o parametrima programa. Ti parametri su broj koraka linearnog pretraživanja i broj koraka binarnog pretraživanja.

Kada imamo premalo koraka tada dolazi do pojave stepenica na modelu kao što vidimo na slici 52.



Da bi reljefno preslikavanje dalo dobre rezultate potrebno je oko 5 iteracija za binarno pretraživanje i 11 za linearno.

Na gornjoj slici unutar slike 53 vidimo greške dobivene pri malim kutovima kod reljefnog preslikavanja. One se javljaju zbog toga što se krivo računaju sjecišta. Donja slika unutar slike 53 je ista scena samo sa optimizacijom reljefnog preslikavanja pomoću mape stožaca. Kao što vidimo ta tehnika daje puno točnije rezultate.



Slika 53: Gornja slika prikazuje model na kojim je primijenjeno reljefno preslikavanje sa 11 iteracija za linearno pretraživanje i 5 za binarno. Donja slika prikazuje isti model nad kojim je primijenjeno reljefno preslikavanje sa optimizacijom pomoću mape stožaca.

9. Zaključak

Kroz ovaj rad smo vidjeli da postoji mnogo tehnika za simuliranje trodimenzionalnih detalja površina manipulacijom uzorkom teksture. Svaka od tih tehnika ima svoje prednosti i nedostatke. Reljefno preslikavanje daje jako dobre rezultate. Omogućava ispravan prikaz modela uz ne dodavanje dodatne geometrije. Implementacija algoritma se na današnjim grafičkim procesorima može obraditi u stvarnom vremenu sa zadovoljavajućom kvalitetom. Tehnika je s obzirom na jednostavnije kao što su preslikavanje izbočina i preslikavanje uz efekt paralakse mnogo zahtjevnija za računalo ali zato daje mnogo bolje rezultate zauzvrat.

Danas se jako mnogo vremena troši na nalaženje optimizacije reljefnog preslikavanja kako bi se dobili još točniji rezultati. Jedan način optimizacije, a i ujedno danas najrašireniji, je reljefno preslikavanje optimizirano mapom stožaca.

Sve tehnike koje su se spomenule unutar ovog rada se danas aktivno koriste u industriji igara. Omogućavaju realniji prikaz scene i time bolji doživljaj za igrača.

Zahvala

Zahvaljujem svojoj mentorici prof.dr.sc. Željki Mihajlović na strpljenju, pomoći i vodstvu pri izradi ovog diplomskog rada.

10. Literatura

1. Manuel M. Oliveira, Gary Bishop, David McAllister „Relief Texture Mapping“. Proceedings of SIGGRAPH 2000 (New Orleans, La) , July 23-28, 2000.
2. Oliveira, Manuel M. „Relief Texture Mapping“. Ph.D. Dissertation. UNC Computer Science Technical Report TR00-009, University of North Carolina, March 3, 2000.
3. Fabio Policarpo, Manuel M. Oliveira Comba, Joao „Real-Time Relief Mapping on Arbitrary Polygonal Surfaces“. April 3-6, 2005.
4. McMillan, L. „An Image-Based Approach to Three-Dimensional Computer Graphics“. Ph.D. Dissertation. UNC Computer Science Technical Report, TR97-013, University of North Carolina, April 1997.
5. BLINN, J. F. „Simulation of Wrinkled Surfaces“. In Proceedings of the 5 annual conference on Computer graphics and interactive techniques, 1978.
6. Dummer, Jonathan, „Cone Step Mapping: An Iterative Ray-Heightfield Intersection Algorithm.“ Dostupno na <http://www.lonesock.net/files/ConeStepMapping.pdf>, 2006.
7. Policarpo, Fabio, i Oliveira, Manuel M. „Relief Mapping of Non-Height-Field Surface Details.“ In Proceedings of the 2006 Symposium on Interactive 3DGraphics and Games, pp. 55–62, 2006.
8. Eric Risser, Musawir Shah i Sumanta Pattanaik „Interval Mapping.“ University of Central Florida Technical Report, 2005.
Dostupno na <http://graphics.cs.ucf.edu/IntervalMapping/images/IntervalMapping.pdf>.
9. Policarpo, Fabio i Oliveira, Manuel M. „Relaxed Cone Step Mapping for Relief Mapping“. Dostupno na:
http://developer.download.nvidia.com/books/gpu_gems_3/samples/gems3_ch18.pdf .
GPU GEMS 3, 2007.
10. McGuire, Morgan i McGuire Max „Steep Parallax Mapping“. 2005.
11. Astel, Dave „More Open GL Game Programming“. Thompson Course Technology, 2006.
12. Killgard, Mark J. „A Practical and Robust Bump-mapping Technique for Today's GPUs“, NVIDIA, July 2000.
13. Welsh , T. „Parallax Mapping with Offset Limiting: A Per-Pixel Aproximation of Uneven Surfaces“, Infiscape Corporation, January 2004.
14. Rost, Randi J. „The OpenGL Shading Language“, Second Edition, Addison-Wesley, December 2006.
15. <http://en.wikipedia.org/wiki/Parallax> .
16. <http://graphics.cs.brown.edu/games/SteepParallax/index.html> .
17. http://en.wikipedia.org/wiki/Displacement_mapping .
18. Tatarchuck, N. „Practical Dynamic Parallax Occlusion Mapping“, SIGGRAPH, 2005.
19. Tatarchuck, N. „Dynamic Parallax Occlusion Mapping with Soft Shadows“, 2006.

Sažetak

(Reljefne tehnike teksturiranja)

U radu je dan pregled svih danas najkorištenijih tehnika za prikaz trodimenzionalnih detalja površina. Dana je teorijska i matematička podloga nad kojom se temelji reljefno preslikavanje. Opisana je implementacije algoritma reljefnog preslikavanja u stvarnom vremenu. Napravljen je program za sjenčanje pomoću jezika za sjenčanje GLSL unutar AMD RenderMonkey razvojnog okruženja koji implementira algoritam reljefnog preslikavanja u stvarnom vremenu. Također su napravljeni programi za sjenčanje drugih jednostavnijih tehnika za usporedbu i analizu. Dan je pregled nedostataka tehnike i greške do kojih se može doći. Također je navedena jedna od tehnika koja se danas koristi za optimizaciju reljefnog preslikavanja.

Ključne riječi: reljefno preslikavanje teksture, geometrijsko preslikavanje izbočina, preslikavanje izbočina, preslikavanje uz efekt paralakse, preslikavanje prekrivanja uz efekt paralakse, preslikavanje strminom uz efekt paralakse, optimizacija reljefnog preslikavanja, reljefno preslikavanje mapom stožaca, iscrtavanje pomoću promjene uzorka teksture, iscrtavanje u stvarnom vremenu.

Abstract

(Relief mapping techniques)

This paper presents an overview of the most used techniques today in presenting three-dimensional surface details. A theoretical and mathematical explanation is given for understanding relief mapping. A description is presented that describes the implementation of relief mapping in real time. A shader is given which implements the algorithm of relief mapping in real time. The shader is written in the GLSL shading language using AMD RenderMonkey as the development environment. There is also a set of shaders which implement the algorithms of the simpler techniques for comparison and analysis. The flaws of relief mapping are explained and shown. A technique is described for optimizing relief mapping.

Keywords: relief mapping, displacement mapping, bump mapping, parallax mapping, parallax occlusion mapping, steep parallax mapping, optimizing relief mapping, cone step mapping, image-based rendering, real time rendering.