SVEUČILIŠTE U ZAGREBU

FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 1819

Nefotorealistični prikaz objekta

Ante Radman

Zagreb, studeni 2009.

Sadržaj

Popis slika

Popis tablica

1 Uvod

Od početka svog razvoja, računalna grafika se najviše bavila ciljem da predmete prikaže što vjernijim stvarnosti, na način kao što je to moguće na fotografijama. No, fotografije i općenito fotorealistične tehnike nisu uvijek najbolji način da se prenesu vizualne informacije. Njihova složenost obično nosi više informacija nego je promatraču potrebno za razumijevanje prikaza, a razumijevanje može otežati i nedovoljna raspoznatljivost predmeta uslijed sporednih čimbenika kao što su neprikladna rasvjeta, prevelika međusobna sličnost predmeta u njihovoj boji, nemogućnost promatrača da uopće razlikuje boje i slično.

Nefotorealistični prikaz je relativno nova grana računalne grafike, koja ima zadatak da vizualnu informaciju prenese brže i učinkovitije nego što je moguće fotorealističnim tehnikama. U tu svrhu koristi se naglašavanje oblika ili pojedinih bitnih karakteristika površine nauštrb manje bitnih informacija, te se iskorištava informacijska učinkovitost crta i drugih jednostavnih elemenata iz već postojećih načina prikaza, s kojima smo odrasli i koje najbolje razumijemo. S druge strane, mogućnost nefotorealističnog prikaza je i ciljano oponašanje estetskog dojma iz tradicionalnih tehnika, čega ranije nije bilo u računalnim tehnikama prikaza. Tako je moguće iz "hladnog" računalnog prikaza dobiti i toplinu crtića, stripova ili umjetničkih poteza kistom.

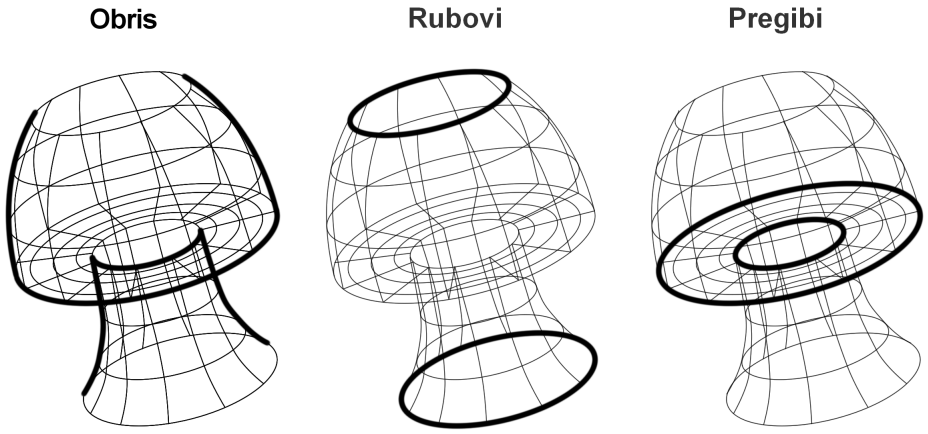
U ovom radu odabrana je kao tema za nefotorealistični prikaz starinska estetika crtanja potezima, rezbarenja, sjenčanja točkama i slično. Kao temelj za ostvarenje odabrana je tehnika tonskih mapa koju su osmislili Praun i suradnici [1], a koja se može učinkovito izvesti korištenjem postojeće računalne podrške za rad s teksturama. Modeli tijela građeni su uobičajenim trodimenzionalnim poligonalnim mrežama.

U sljedećem poglavlju govori se o prikazu oblika tijela najosnovnijim karakterističnim crtama, kao prvi korak u ostvarenju nefotorealističnog prikaza. Zatim se u poglavlju 3 izlažu precizne zamisli o optimalnom korištenju tekstura za oponašanje likovnih

zahvata. U poglavlju 4 dan je matematički model za prilagodbu tekstura obliku tijela. Poglavlje koje zatim slijedi opisuje programsko ostvarenje ove tehnike. Rezultati su izloženi u poglavlju 6. Zaključuje se sumiranjem teme i rezultata koji su ostvareni, s razmišljanjima o nedostatcima ove tehnike i mogućnostima za buduće poboljšanje programskog rješenja.

2 Prikaz oblika osnovnim crtama

Karakteristične crte koje najviše govore o obliku tijela su njegov obris i oštri bridovi – rubovi tijela i pregibi – i zato su one nezaobilazan korak u nefotorealističnom prikazu. Spomenute tri vrste takvih crta prikazane su na slici 2.1. Tehnike koje imaju za cilj estetski dojam koriste različite likovne izvedbe; za ovaj rad su najprikladnije jednostavne, pune crte.



Slika 2.1. Karakteristične crte tijela

Njihov pronalazak traži određenu analizu tijela prije samog iscrtavanja. U toj, početnoj obradi modela, potrebno je prije svega utvrditi strukturu poligonalne mreže i relacije susjedstva među poligonima. To je postupak koji je potrebno provesti samo jednom; kasnije se koristi dobiveni skup informacija za prikaz svih triju vrsta osnovnih crta tijela.

2.1 Obris tijela

Obris, kontura ili silueta je skup crta povučenih oko tijela, koje ga jasno odvajaju od okoline. Kod složenih tijela obris može rezultirati i s više krivulja. U 3D-modelima koji

su građeni od poligonalne mreže jednostavno ga je ostvariti biranjem bridova poligona koji su na granici između vidljivih i skrivenih područja. Brid poligona smatra se dijelom obrisa ako je poligon vidljiv i na tom bridu graniči s poligonom koji je skriven.

Vidljivost pojedinog poligona se po toj definiciji može pojednostavnjeno utvrditi skalarnim umnoškom normale i vektora pogleda prema očištu, pri čemu se poligon smatra vidljivim, odnosno okrenutim prema promatraču ako je umnožak veći od nule.

Uvjet za takav pojednostavnjeni izračun je usmjerenost svih poligona modela u smjeru kazalje na satu. Ovdje ostvareno programsko rješenje pretpostavlja takvu usmjerenost. U slučaju suprotne orijentacije, predznak umnoška bi imao suprotno značenje. Druga pretpostavka za tako jednostavnu izvedbu algoritma je korištenje Z-spremnika, koji će i u samom obrisu efektivno razvrstati bridove u dijelove koji su vidljivi i one koji su skriveni.

Ako tijelo s vremenom mijenja svoj smjer ili položaj, potrebno je iznova potražiti bridove obrisa. U priloženom programskom rješenju početni proračun se izvodi pri učitavanju modela i kasnije pri svakoj rotaciji i pomicanju iznova.

2.2 Rubovi

Slično kao što su rubovi u stvarnom svijetu završetci ploha, u poligonalnim 3D-mrežama treba rubovima modela smatrati one bridove poligona koji ga ne povezuju ni s jednim drugim poligonom.

Za utvrđivanje rubova se dakle koristi već dobivena informacija o susjedstvima među poligonima, a dodatne analize nije potrebno provoditi.

2.3 Pregibi

Pregibi su crte oko kojih se normala površine relativno naglo mijenja. Također su vrlo bitni za razumijevanje oblika tijela, te ih treba iscrtati jednako naglašeno kao druge

dvije spomenute vrste crta. Pregibe pronalazimo izračunom kutova između svaka dva susjedna poligona, pri čemu jednostavan prelazak određenog graničnog iznosa oštrog kuta po dogovoru znači postojanje pregiba na bridu.

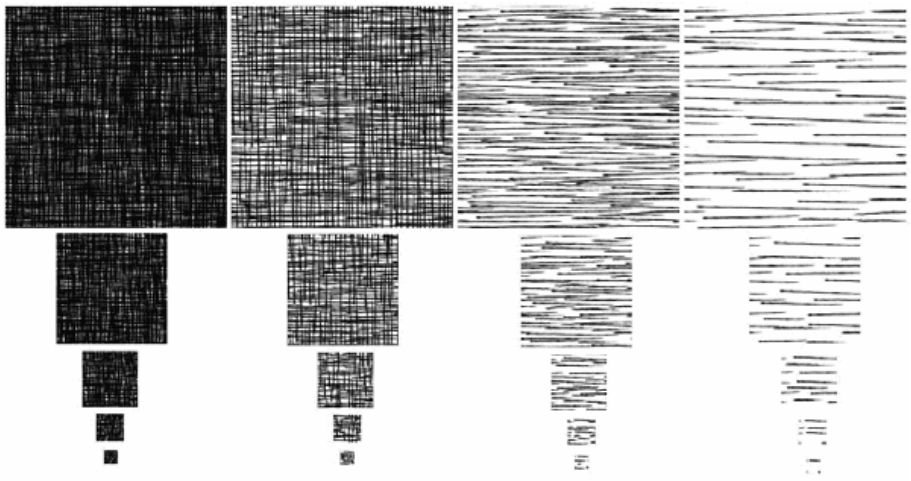
Programsko rješenje dopušta korisniku proizvoljno zadavanje oštrog kuta, odn. granice kada tupi kut prelazi u oštar. Time je omogućeno da se prikaz prilagodi konkretnom modelu.

3 Prikaz oblika tonskim mapama

Koncept tonskih mapa osmislili su E. Praun i suradnici [1] za višenamjensko ostvarenje nefotorealističnog prikaza gdje se mogu oponašati estetika rezbarenja, crtanja olovkom, ugljenom, oslikavanje točkicama i slično. Zamisao je bila iskoristiti mogućnosti brzog računalnog teksturiranja umjesto izračunavanja i iscrtavanja svakog osnovnog elementa prikaza zasebno, uz riješena neka pitanja koja su mučila ranija rješenja. Tako je dobiveno jedno zaokruženo i dosljedno rješenje prikaza koje omogućuje i animirano izvođenje u stvarnom vremenu.

3.1 Priprema tekstura

Potezi kistom ili drugi željeni likovni materijal pripremi se i iscrta unaprijed na nizu sličica, za svaku nijansu osvjetljenja posebno i za svaku razinu detalja (veličinu teksture) posebno. Ukupno gledano, gradi se dvodimenzionalna mapa kao na slici 3.1; kasnije će se površina tijela teksturirati tim sličicama slijedeći posebna pravila.



Slika 3.1. tonska mapa; svaki stupac odgovara po jednoj nijansi osvjetljenja, a redak razini detalja za mip-mapping

Najveće sličice su dimenzija 256×256 točaka, sljedeća je razina 128×128 i tako dalje, slijedeći potencije broja 2 do veličine 1×1. Svaka sličica pritom mora biti toroidalna, tj.

omogućiti popločavanje svojim vlastitim ponavljanjem bez šavova u smjerovima gore, dolje, lijevo i desno.

Imati što više razina detalja za automatski odabir najprikladnije razine (mip-mapping) je od ključne važnosti u ovoj tehnici, jer pregrubo iscrtani elementi ne ostavljaju dojam tradicionalne tehnike koju se želi oponašati.

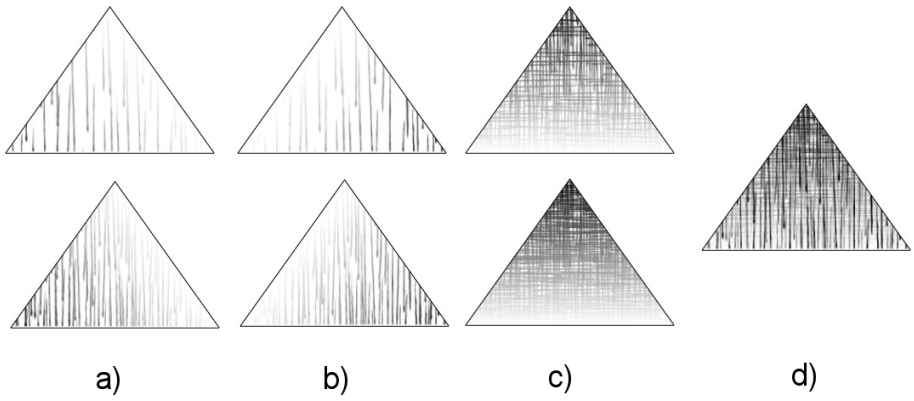
3.2 Stapanje

Od velike važnosti je i ostvarenje prostorne i vremenske koherentnosti prikaza. Zahtjev za prostornom koherentnosti, odnosno ispravnog predočavanja oblika, postavlja dva uvjeta: prijelaz s poligona tamnije nijanse na svjetliji poligon mora biti kontinuiran, a prijelaz s više razine detalja na nižu razinu, primjerice prilikom prelaska s područja krupnijih poligona na finije područje s manjim poligonima, ne smije biti previše uočljiv. Vremenska koherentnost, nužna za kvalitetno ostvarenje animacije modela ili osvjetljenja, uvjetuje da prijelaz sa svjetlijeg vrha na tamniji vrh unutar svakog poligona također mora biti gladak.

Ti zahtjevi rješavaju se s jedne strane izvedbom sličica na način da teksture tamnijih nijansi sadrže teksture svjetlijih: sa svakom novom nijansom dodaju se novi potezi kista na već postojeće. Slično je i u vertikalnom smjeru: veće teksture se izvode kao nadskup manjih, osim u slučaju najmanjih tekstura, koje teško mogu imati uopće raspoznatljivu sliku.

Drugi dio rješenja je stapanje više tekstura na svakom poligonu, s ciljem da se u samom sjenčanju ostvari kontinuirani prijelaz s tamnijeg područja na svjetlije te da se izgladi kvantiziranost osvjetljenja, s obzirom da sličice prikazuju samo nekoliko nijansi. Tako je potrebno koristiti zasebne teksture za svaki od triju vrhova poligona, a zatim ostvariti kontinuirani prijelaz među tim teksturama od vrha do vrha, tj. treba ih baricentrično zbrojiti, slično kao u Gouraudovom sjenčanju. Još je potrebno zbog izbjegavanja skokovitih promjena osvjetljenja na svakom od vrhova stopiti dvije teksture – najbližu svjet-

liju i najbližu tamniju nijansu – u omjeru koji odgovara pravom, nekvantiziranom osvjetljenju vrha. Ukupno tako izvodimo linearno stapanje po šest tekstura na svakom poligonu, što je prikazano na slici 3.2.



Slika 3.2. Stapanje šest slika iz tonske mape na poligonu; svaki stupac od a) do c) odgovara po jednom vrhu i pokazuje dvije najbliže nijanse, koje se stapaju u konačnu nijansu vrha; konačni izgled čitavog poligona je ukupna kombinacija, prikazana sasvim desno u stupcu d)

Uzevši u obzir i mip-mapping, koji se izvodi strojno za svaku teksturu, grafička kartica ima zadatak na svakom poligonu stopiti ukupno 12 tekstura. Iako je to puno posla, u konačnici će poligon ovakvim postupkom sigurno imati traženu estetiku i koherentnost.

3.3 Izračun osvjetljenja

Teksture se iscrtavaju bez OpenGL-ovog ugrađenog osvjetljenja. Premda bi ono ostvarilo bolje performanse, provodi se vlastiti izračun i na temelju dobivenih vrijednosti odabiru se pojedine teksture za neosvijetljeni prikaz.

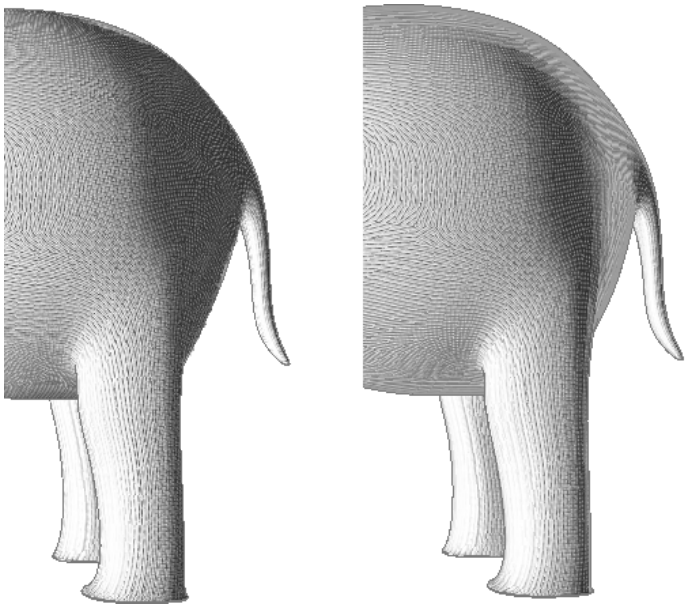
U fotorealističnom svijetu osvjetljenje se obično računa pomoću difuzne, ambijentne i zrcalne komponente. U ovom slučaju, ambijentna svjetlost je već određena teksturom najtamnijeg tona, dok zrcalna nije nikad ni zaživjela u razmatranim tehnikama crtanja. Stoga računamo samo difuznu komponentu, kao skalarni umnožak normale pojedinog vrha i vektora pogleda prema očištu.

Položaj izvora svjetla se ne preporučuje uzeti kao čvrsto određen jednom točkom u sceni. Umjesto toga, izvor se treba kretati zajedno s očištem, pomaknut za otprilike 45 stupnjeva ulijevo i prema gore. Takvo osvjetljenje je uobičajeno u tradicionalnim, nefotorealističnim ilustracijama, i vjerojatno je najbolji izbor u smislu razumljivog predočavanja oblika tijela.

S obzirom da sličice dijele među sobom iste poteze, konačni efekt je da će približavanjem ili pomicanjem sa svjetla u sjenu na površini tijela neki potezi polagano blijediti ili se pojavljivati novi, dok će većina poteza biti ista. To savršeno odgovara tradicionalnim tehnikama.

3.4 Dodatni zahtjevi

Korištenje anizotropnog filtriranja se pokazuje kao nezaobilazan prateći uvjet za izvedbu ove tehnike, jer dojam prirodnog crtanja rukom može biti potpuno uništen deformacijama koje se javljaju na teksturama na rubnim područjima pod oštrim kutom gledanja. Jedan primjer problematične konfiguracije prikazan je na slici .



Slika 3.3. Prikaz s korištenjem anizotropnog filitriranja i bez njega

4 Prilagodba teksture zakrivljenosti površine

Oblik tijela se, osim crtama obrisa, rubova i pregiba, izvrsno predočava crtama koje prate najveću lokalnu zakrivljenost površine. [2] Na gotovo svakom mjestu na površini tijela moguće je utvrditi smjer najveće zakrivljenosti površine kao tzv. prvi glavni smjer, poznat iz diferencijalne geometrije. Izuzetak su dijelovi kugle, koji su jednako zakrivljeni u svim smjerovima te dijelovi ravnine, koji nisu nimalo zakrivljeni, a tu su i neki posebni slučajevi u kojima je zakrivljenost u nekim (tzv. asimptotskim) smjerovima također nula. Te iznimke treba posebno zbrinuti.

Teksture je dakle potrebno na svakom pojedinom poligonu usmjeriti s njegovim lokalnim prvim glavnim smjerom, čime će potezi iscrtani na njima slijediti slične putove onima koje bi odabrala ljudska ruka u tradicionalnim tehnikama, i tako najbolje ispuniti zadatak prikazivanja oblika tijela.

Osim određivanja smjera u svakom vrhu, bit će zatim potrebno i distribuirati dobivene smjerove na one vrhove pri kojima nije bilo moguće odrediti smjer zbog pogreške. Naposlijetku će se izvesti završno zaglađivanje smjerova.

4.1 Izračunavanje glavnih smjerova općenito

Postoje analitički postupci koji izračunavaju smjerove najveće i najmanje zakrivljenosti u pojedinoj točki na mreži poligona. Oni su svi ovisni prvenstveno o kvaliteti same mreže, kojom se aproksimira željeni oblik, a koji treba "izvući" iz aproksimacije i prikazati linijama. K tome su svi ti postupci u odr. mjeri podložni manjim greškama.

U diferencijalnoj geometriji [3], normalna zakrivljenost K_t plohe u odr. točki T i u odr. smjeru **t** se definira kao recipročna vrijednost polumjera oskulacijske kružnice koja leži u ravnini s točkom T, normalom plohe i tangentom **t** i najbolje aproksimira presjek plohe u tom smjeru. Ekstremne vrijednosti zakrivljenosti K_1 i K_2 nazivamo glavnim zakrivljenostima plohe u točki T, a vektore tangenata pridruženih im krivulja nazivamo

glavnim smjerovima u T (dalje označeni kao $\mathbf{p_1}$ i $\mathbf{p_2}$). Glavni su smjerovi uvijek međusobno okomiti. Tzv. prvi glavni smjer pokazuje smjer najveće zakrivljenosti u točki, dok drugi glavni smjerpokazuje smjer najmanje zakrivljenosti.

Spomenute posebne slučajeve kada je normalna zakrivljenost u točki T jednaka nuli nazivamo asimptotskim smjerovima plohe u točki T.

Na glatkim plohama vrijedi za bilo koji jedinični vektor na nekoj njenoj tangencijalnoj ravnini sljedeća jednakost:

$$K_t = \begin{bmatrix} x & y \end{bmatrix} W \begin{bmatrix} x \\ y \end{bmatrix}$$

gdje su komponente x i y vektora izražene u lokalnom ortonormiranom koordinatnom sustavu, a W je tzv. Weingartenova matrica, odn. druga temeljna forma. Za nju je karakteristično da se rotacijom spomenutog lokalnog koordinatnog sistema može dijagonalizirati i dati jednakost:

$$K_t = \begin{bmatrix} x' & y' \end{bmatrix} \begin{bmatrix} K_1 & 0 \\ 0 & K_2 \end{bmatrix} \begin{bmatrix} x' \\ y' \end{bmatrix} = K_1 x'^2 + K_2 y'^2$$

gdje su K_1 i K_2 glavni smjerovi, a x' i y' su sad izraženi pomoću glavnih smjerova kao komponenti.

Ako je X(u, v) lokalna parametrizacija plohe u okolini točke T, možemo koristeći $X_u(T)$, $X_v(T)$ i normalu N_T kao lokalni koordinatni sistem izračunati glavne zakrivljenosti i glavne smjerove u točki na sljedeći općenit način. Weingartenova matrica ima oblik:

$$W = \begin{bmatrix} \dfrac{eG - fF}{EG - F^2} & \dfrac{fE - eF}{EG - F^2} \\ \dfrac{fG - gF}{EG - F^2} & \dfrac{gE - fF}{EG - F^2} \end{bmatrix}$$

gdje su:

$$e = N_T \cdot X_{uu}(T) \qquad E = X_u(T) \cdot X_u(T)$$
$$f = N_T \cdot X_{uv}(T) \qquad F = X_u(T) \cdot X_v(T)$$
$$g = N_T \cdot X_{vv}(T) \qquad G = X_v(T) \cdot X_v(T)$$

i, u slučaju kad su X_u i X_v ortogonalni, dobivamo simetričnu matricu:

$$W = \begin{bmatrix} e & f \\ f & g \end{bmatrix}$$

Ako su λ_1 i λ_2 vlastite vrijednosti matrice W, slijedi da one odgovaraju glavnim zakrivljenostima K_1 i K_2, a vlastiti vektori $\mathbf{t_1} = [t_{11}\ t_{12}]^T$ i $\mathbf{t_2} = [t_{21}\ t_{22}]^T$ pridruženi tim vrijednostima su glavni smjerovi plohe u točki T, no izraženi u lokalnim koordinatama. Glavni smjer povezan s većom vlastitom vrijednosti matrice je tzv. prvi glavni smjer i odgovara najvećoj zakrivljenosti, dok drugi glavni smjer pokazuje smjer najmanje zakrivljenosti.

Ti vektori se korištenjem izabrane parametrizacije preslikavaju natrag u euklidski prostor R^3 u vektore:

$$t_1 = t_{11}X_u + t_{12}X_v$$
$$t_2 = t_{21}X_u + t_{22}X_v$$

Time je teorijski određen zadatak izračunavanja glavnih zakrivljenosti, odn. glavnih smjerova na plohi. U konkretnom praktičnom zadatku, prvi korak u izračunavanju glavnih smjerova je odrediti normalu u svakoj točki, a to znači izračunati je u svakom vrhu poligonalne mreže modela. Zatim treba odabrati vektore koji će činiti novi, ortogonalni koordinatni sustav. Zatim se nekim od poznatih postupaka aproksimira matrica W i izračunavaju njeni vlastiti vektori.

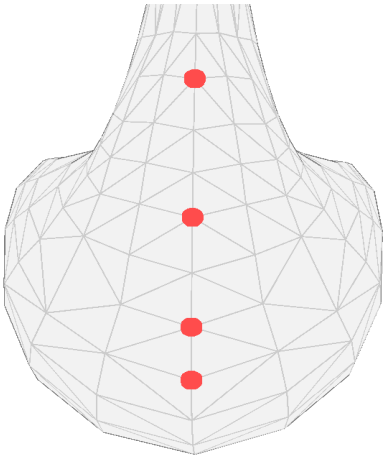
Uobičajeni postupci kojima se te vrijednosti izračunavaju mogu se podijeliti u tri

skupine, ovisno o tome da li se zasnivaju na aproksimaciji područja analitičkom plohom, računanjem srednje vrijednosti normalnih zakrivljenosti (nakon aproksimacije zakrivljenosti kružnicom u što više smjerova) ili računanjem srednje vrijednosti posebno definiranog tenzora zakrivljenosti. Ovdje je odabran postupak aproksimacije kvadratičnom plohom, s dodatnom posebnom analizom namijenjenom objektima s oštrim bridovima i dijelovima ravnina, kao što su to vrlo česta pravokutna tijela.

4.2 Aproksimacija površine tijela geometrijskom plohom

Zamisao je geometrijsku plohu, odn. polinom, umetnuti u okolinu promatrane točke T i prilagoditi najbližim okolnim točkama, a to su u ovom slučaju susjedni vrhovi na poligonalnoj mreži. (Susjedstvo je određeno kao povezanost točaka nekim od bridova poligona.) Zatim se iz takve plohe mogu izračunati tražene zakrivljenosti, s pretpostavkom da će ona, provučena kroz ostale točke na najbolji način, biti dovoljno slična pravoj plohi da dâ zadovoljavajuće rezultate. Pokazano je da u općem slučaju glatke plohe rezultati konvergiraju k točnim vrijednostima, ako se ne radi o posebnim degeneriranim slučajevima. [4]

Nestabilnost postupka se očituje kad su položaji susjednih vrhova (vrhova uzetih u izračun) smješteni na dva pravca koji se sijeku. Tada se radi o spomenutim asimptotskim smjerovima. Jedan od primjera je dan na slici 4.1.



Slika 4.1. Nestabilne točke na primjeru modela patke

Za svaki vrh T na modelu pokušat će se što je bolje moguće prilagoditi kvadratičnu plohu susjednim vrhovima S*i*. Svaki od njih se preslika u lokalne koordinate (x_i, y_i, z_i), a T postaje (0, 0, 0). Normala N_T leži duž pozitivne osi Z, a koristi se, u ovom primjeru, kvadratična ploha određena izrazom:

$$z = f(x,y) = \frac{A}{2}x^2 + Bxy + \frac{C}{2}y^2$$

Neka je W nepoznata Weingartenova matrica izražena u lokalnom koordinatnom sustavu oko vrha T. Za ovakvu plohu matrica W je jednaka:

$$W = \begin{bmatrix} A & B \\ B & C \end{bmatrix}$$

Susjedne točke uvode se u matrični sustav jednadžbi:

$$\begin{bmatrix} \frac{1}{2}x_i^2 & x_i y_i & \frac{1}{2}y_i^2 \end{bmatrix} \mathbf{x} = z_i$$

gdje je vektor **x** sastavljen od traženih vrijednosti $[A\ B\ C]^T$.

Sustav nema egzaktno rješenje, već mora zadovoljiti polinom u svim uključenim točkama što je bolje moguće. Zato jedino preostaje pronaći najbolje rješenje prema metodi najmanjih kvadrata.

Izračunatim koeficijentima A, B i C određena je dakle kvadratična ploha koja najbolje aproksimira površinu tijela u promatranoj točki. Pripadna Weingartenova matrica, popunjena prema ranije navedenom izrazu, u svojim vlastitim vektorima sadrži traženu aproksimaciju glavnih smjerova u lokalnom koordinatnom sustavu.

Konačne vrijednosti vektora prvog glavnog smjera dobivaju se njegovim preslikavanjem natrag u izvorni koordinatni sustav, a to konkretno znači još samo rotaciju kojom se od smjera vektora normale N_T vraćamo na smjer pozitivne osi Z.

4.3 Zaglađivanje smjerova

Pronađene smjerove u vrhovima poželjno je još donekle ujednačiti, da razlika među jednako usmjerenim područjima koja je nastala isključivo zbog nesavršenosti mreže poligona bude što manje očita.

Izračun se izvodi zbrajanjem smjerova među susjednim vrhovima i davanjem prednosti sličnim smjerovima. Pri tom zbrajanju uvažava se posebna aritmetika "dvosmjernih" smjerova, koja suprotne smjerove zbraja kao iste, jer rezultiraju istom orijentacijom tekstura na poligonima.

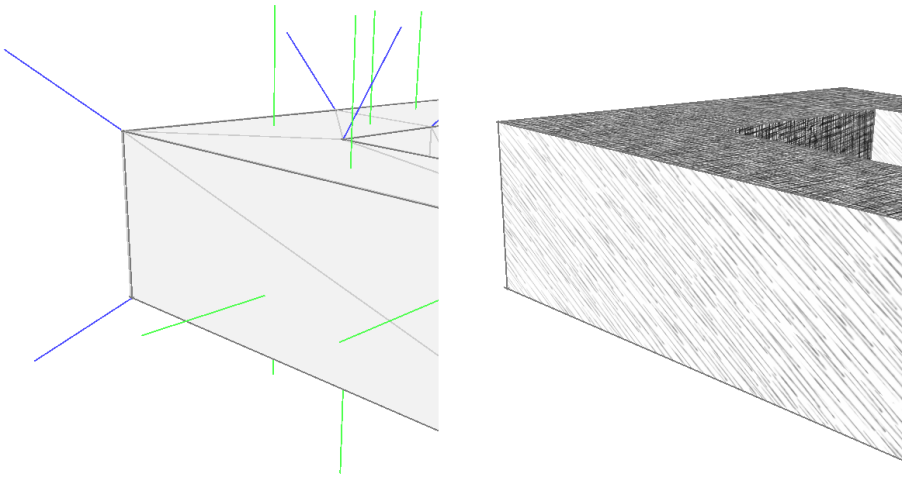
4.4 Posebnosti kod pravokutnih tijela

Smjer teksture na poligonu koji ima oštar brid (rub ili pregib) ne određuje se pomoću smjerova zakrivljenosti u vrhovima, jer su oni izračunati s pretpostavkom glatke plohe i pretpostavkom postojanja tangencijalnih ravnina u vrhovima. Na oštrim rubovima to zapravo nije ispunjeno te izračunati smjer nema smisla, pa treba prenijeti smjer distribucijom sa susjednih poligona ili, ako to nije moguće, odrediti neovisnim odabirom.

Jedno rješenje koje se u tom slučaju pokazuje estetski prihvatljivim je uzeti smjer okomit na koordinatnu os koja je paralelna s ravninom poligona, odnosno onu os koja je tome najbliža. To je posljedica vrlo čestog običaja da su pravokutni modeli građeni s ravninama paralelnim s koordinatnim osima. Tada se uzima vektorski umnožak normale i dotične osi kao najbolje rješenje za smjer.

Pri iscrtavanju poligona koji imaju oštre bridove, te bridove će se istaknuti uočljivom tamnom crtom kao i obris tijela, zbog sugeriranja oblika tijela na način tipičan za nefotorealistični prikaz. Osim toga, oštrina bridova naglasit će se i pri "sjenčanju", i to posebnim računanjem osvjetljenja. Osvjetljenje se inače računa pojedinačno za svaki vrh poligona; sada ako je utvrđeno da se vrh nalazi na nekom oštrom bridu, osvjetljenje će se računati pomoću normale trokuta, a ne normale koja je bila izvorno izračunata na vrhu s pretpostavkom da se radi o glatkoj plohi.

Na slici 4.2 prikazan je primjer pravokutnog tijela s prikazanim normalama koje su izračunate u vrhovima (plava boja). Umjesto tih normala, za osvjetljenje se koriste se normale poligona (zelena boja). Rezultat je jasan i ispravan dojam oštrog brida na tijelu (desno).



Slika 4.2. Normale trokuta (zeleno) su bolji odabir od normala vrhova (plavo) za osvjetljavanje pravokutnih tijela

5 Programsko ostvarenje

Uz rad je priložen program koji se razvijao tijekom rada na temi, s pripadajućim izvornim kodom. Korišten je jezik C++ uz OpenGL kao grafičko sučelje. Teksture su nacrtane koristeći poseban grafički program.

Program učitava 3D-model iz datoteke i generira prikaz uz mogućnost korisnikovog mijenjanja pogleda i promjene parametara prikaza.

Odabran je Wavefrontov OBJ-format datoteke modela kao glavni format. S obzirom da obrada složenih modela može potrajati i do pola minute, izvedena je mogućnost snimanja već obrađenih modela u vlastiti format s nastavkom NPR. Kasnije se takva ulazna datoteka može otvoriti umjesto izvorne OBJ-datoteke bez čekanja na obradu.

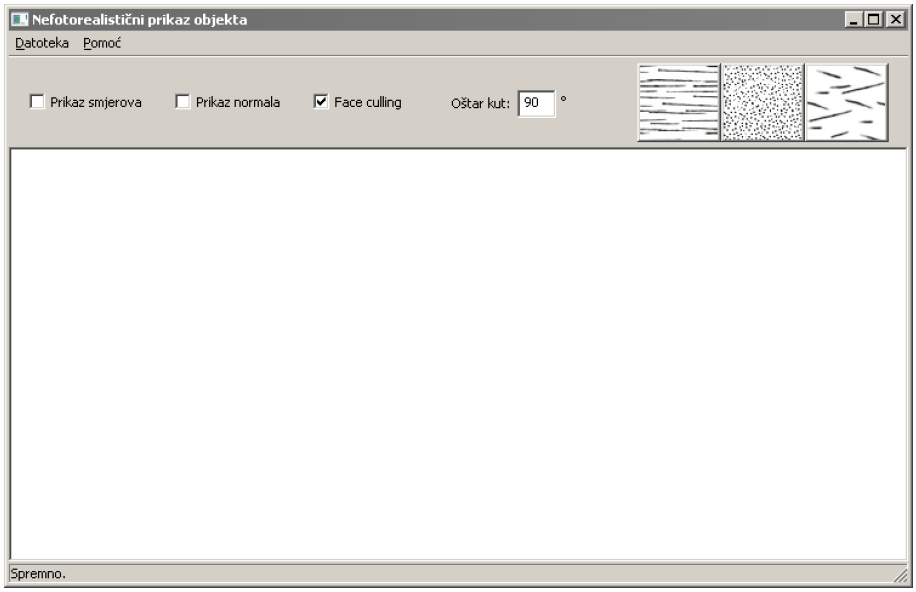
Uz program je priloženo nekoliko primjera modela u OBJ-formatu, koji se mogu slobodno koristiti u akademske svrhe.

Rješenje je namijenjeno Microsoft Windows platformi s uobičajenom podrškom za OpenGL u vidu biblioteka opengl32.dll i glu32.dll, koje su priložene uz operacijski sustav, ili njihovih zamjena. Poželjno je da računalo ima suvremenu grafičku karticu, zbog naglaska na radu s teksturama.

5.1 Korištenje programa

Pri pokretanju programa moguće je navesti ime ulazne datoteke kao parametar. Ona se može i kasnije odabrati putem izbornika *Datoteka → Otvori* ili pak dovući mišem iz drugih programa i ispustiti na radnu površinu. Obrađena datoteka može se snimiti za kasnije brzo učitavanje putem izbornika *Datoteka → Snimi.*

Izgled radnog prozora prikazan je na slici 5.1.



Slika 5.1. Glavni prozor programa

Na alatnoj traci na raspolaganju su korisničke opcije:

- prikaz smjerova zakrivljenosti izračunatih u vrhovima; koristi se crvena boja za vektor smjera u pojedinom vrhu
- prikaz normala trokuta i normala vrhova; koristi se plava boja za normale vrhova i zelena boja za normale trokuta
- uključenje *face cullinga*, za rasterećenje prikaza kod vrlo složenih modela
- unos graničnog iznosa oštrog kuta, koji će se utvrđivati postojanje pregibâ na modelu; preporuka je kut od 90 stupnjeva ili veći
- različiti stilovi crtačkih tehnika za oponašanje

Rotiranje učitanog modela se izvodi prevlačenjem lijevom tipkom miša preko radne površine, dok se kotačićem na mišu prikaz zumira u željenoj mjeri. Prevlačenje desnom tipkom miša pomiče model, odnosno pogled u stranu.

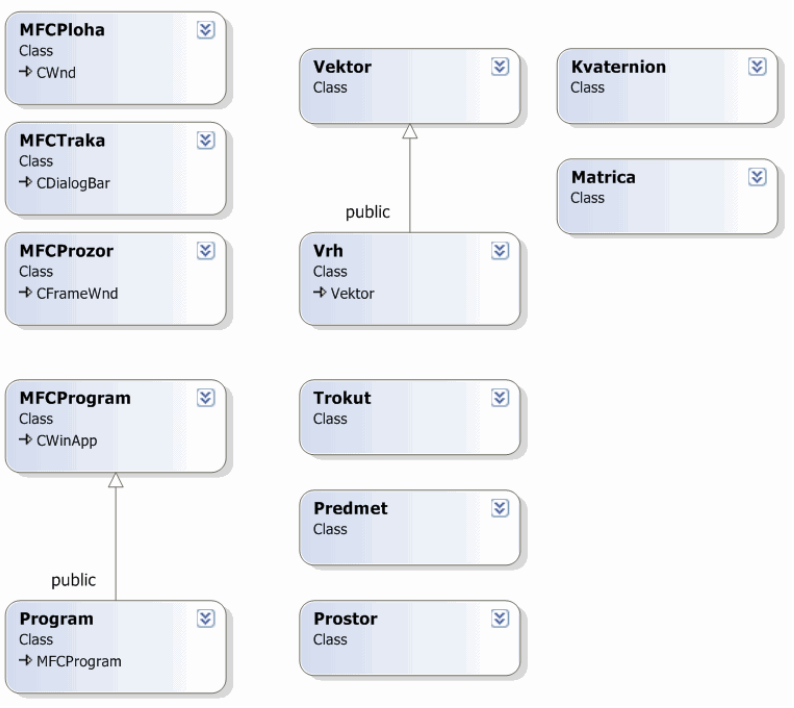
Korisniku su na raspolaganju i tipke gore/dolje/lijevo/desno za rotiranje modela preko tipkovnice, kao i tipka *Escape* za brzi izlazak iz programa.

Ako tijelo nije ispravno iscrtano, pogotovo nakon što se uključi *face culling*, preporuka je uključiti prikaz normala i provjeriti da li su sve normale usmjerene van tijela. Ako nisu, pripadni poligoni su orijentirani u smjeru suprotnom od kazaljke na satu. Program ne podržava takvu orijentaciju poligona.

Drugi problemi s pogrešnim iscrtavanjem modela koji imaju uzroke u samim modelima mogu biti kada se više točaka preklapa na istom mjestu, odnosno kad se više poligona preklapa u istoj ravnini. Takvi slučajevi mogu se pak brzo ispraviti u nekom 3D editoru.

5.2 Organizacija izvornog koda

Dijagram razvijenih C++ razreda prikazan je na slici 5.2.



Slika 5.2. Dijagram razreda

Svaki razred pohranjen je u vlastitoj .cpp datoteci, s pripadnom .h datotekom. U tablici 5.1 dan je kratak opis svake od njih.

Tablica 5.1. Organizacija izvornog koda po datotekama

Datoteke	Opis funkcionalnosti
vektor.cpp vektor.h	Razred Vektor, koji implementira vektor kao geometrijski element
kvaternion.cpp kvaternion.h	Razred Kvaternion, koji služi za rotacije vektora
matrica.cpp matrica.h	Razred Matrica, potreban za matrične izračune oko normalnih zakrivljenosti
vrh.h	Razred Vrh je jednostavna nadgradnja razreda Vektor, predstavlja element poligona
trokut.cpp trokut.h	Razred Trokut predstavlja poligon 3D-mreže
predmet.cpp predmet.h	Razred Predmet predstavlja jedan objekt u sceni
prostor.cpp prostor.h	Razred Prostor predstavlja scenu, i sadrži cjelokupan opis prostora
program.cpp program.h datoteka.cpp	Razred Program je glavni program, rasterećen svih nebitnih poslova; dio posvećen čitanju datoteka modela je izdvojen u zasebnu datoteku
mfcprogram.cpp mfcprogram.h	Razred MFCProgram predstavlja tipični program za MS Windowse; iz njega je deriviran razred Program
mfcprozor.cpp mfcprozor.h	Razred MFCProzor ostvaruje glavni prozor aplikacije u MS Windowsima
mfctraka.cpp mfctraka.h	Razred MFCTraka je alatna traka na glavnom prozoru
mfcploha.cpp mfcploha.h	Razred MFCPloha predstavlja radnu površinu u sredini prozora

Slike tekstura (mipmape) pohranjene su u svojim zasebnim mapama.

5.3 Programi jedinica za sjenčanje

Zbog optimizacije prikaza, stapanje tekstura je vlastoručno programirano u jedinicama za sjenčanje, u jeziku GLSL. Koristi se mogućnost procesora vrhova i fragmenata da sami utvrde koje točno teksture treba kombinirati na bilo kojem mjestu na poligonu, bez da moraju izvoditi bilo kakve nepotrebne poslove. Ranije je izračun bio moguć u samo tri krajnja vrha na pojedinom poligonu, pa je nužno bilo stapati šest tekstura na svakom poligonu kako bi oni u konačnici imali ispravne prijelaze između krajnjih nijansi.

U ostvarenom mehanizmu glavni program daje na raspolaganje procesoru vrhova sve teksture i položaj izvora svjetla putem uniformnih varijabli, te standardnim putem normalu interpoliranu između tri vrhova poligona za određenu točku na poligonu. Procesor vrhova prosljeđuje procesoru fragmenata prostorne koordinate dotične točke, danu normalu i koordinate tekstura. Procesor fragmenata dobivenim podacima izračuna intenzitet osvjetljenja i odredi potrebne teksture koje treba stopiti, te izračuna konačnu boju kao zbroj najviše dviju tekstura.

Program za procesor vrhova u GLSL-u je sljedeći:

```
varying vec3 vrh, normala;
void main () {
  gl_TexCoord[0] = gl_TextureMatrix[0] * gl_MultiTexCoord0;
  gl_TexCoord[1] = gl_TextureMatrix[1] * gl_MultiTexCoord1;
  gl_TexCoord[2] = gl_TextureMatrix[2] * gl_MultiTexCoord2;
  gl_TexCoord[3] = gl_TextureMatrix[3] * gl_MultiTexCoord3;
  gl_TexCoord[4] = gl_TextureMatrix[4] * gl_MultiTexCoord4;
  normala = gl_Normal;
  vrh = gl_Vertex;
  gl_Position = ftransform();
}
```

Program za procesor fragmenata:

```
uniform sampler2D tex1, tex2, tex3, tex4, tex5;
uniform vec3 svjetlo;
varying vec3 vrh, normala;
void main () {
  vec4 boja1 = texture2D (tex1, gl_TexCoord[0].st);
  vec4 boja2 = texture2D (tex2, gl_TexCoord[1].st);
  vec4 boja3 = texture2D (tex3, gl_TexCoord[2].st);
  vec4 boja4 = texture2D (tex4, gl_TexCoord[3].st);
  vec4 boja5 = texture2D (tex5, gl_TexCoord[4].st);
  float intenzitet = max (0.0, dot (
    normalize (svjetlo - vrh), normalize (normala))) * 4;
  float razina = floor (intenzitet);
  float ostatak = intenzitet – razina;
  if (razina <= 0)
    gl_FragColor = boja2 * ostatak + boja1 * (1.0 – ostatak);
  if (razina == 1)
    gl_FragColor = boja3 * ostatak + boja2 * (1.0 – ostatak);
  if (razina == 2)
    gl_FragColor = boja4 * ostatak + boja3 * (1.0 – ostatak);
  if (razina == 3)
    gl_FragColor = boja5 * ostatak + boja4 * (1.0 – ostatak);
  if (razina >= 4)
    gl_FragColor = boja5;
}
```

5.4 Napomene pri prevođenju

Rješenje je razvijeno u Microsoft Visual Studiju 2008, sa statički povezanim MFC-bibliotekama i uključenom punom podrškom za Unicode. Pri povezivanju se koriste uobičajene biblioteke opengl32.lib i glu32.lib za OpenGL funkcije te glee.lib za pomoć pri programiranju jedinica za sjenčanje.

Svi resursi aplikacije, uključujući bitmape za teksture, uključeni su u sam projekt i moraju biti povezani u konačnu izvršnu datoteku.

6 Rezultati

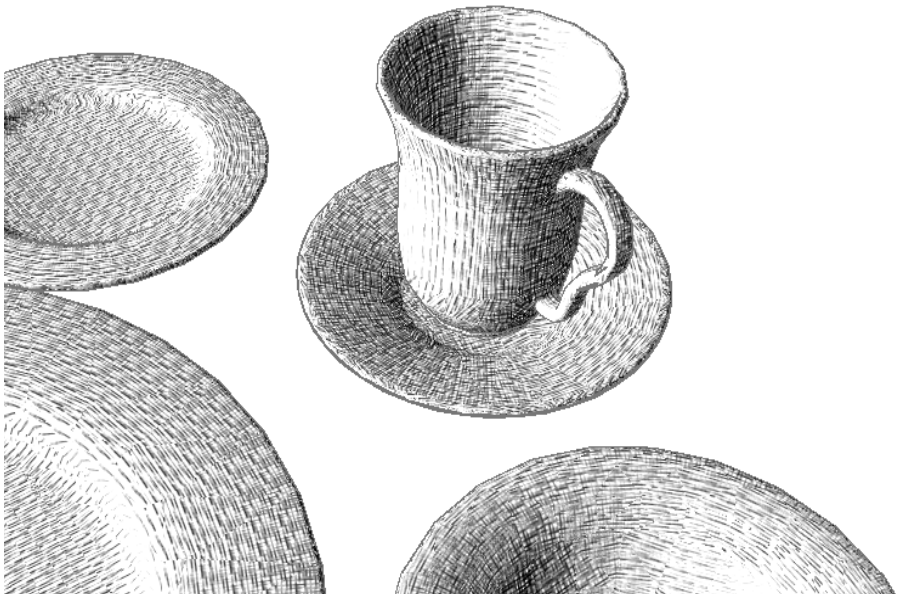
Na slikama 6.1–6.4 dan je konačni prikaz četiriju modela koristeći različite stilove tonskih mapa. Broj poligona u modelima varira od 800 do 16.000.

Modeli su pokazali koherentnost prilikom statičnog prikaza iz različitih kutova gledanja te prilikom animacije rotiranjem i postupnim uvećavanjem i smanjivanjem (pomicanjem očišta).

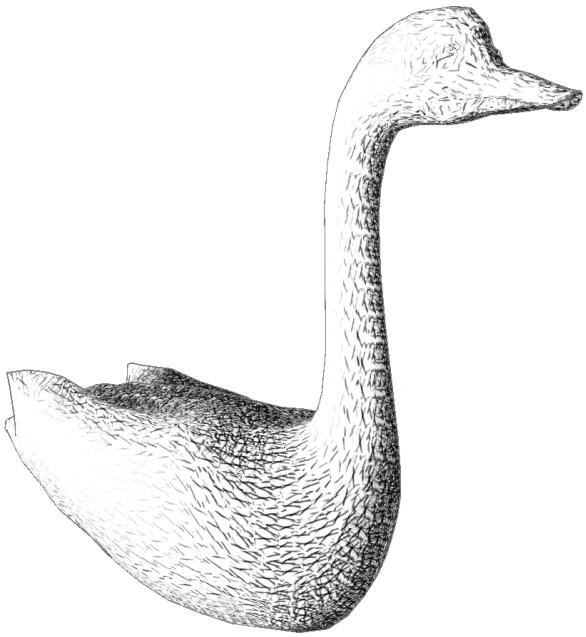
Na rezultate u kvalitativnom smislu jako utječe rezolucija zaslona. Modeli su relativno fino modelirani velikim brojem poligona, što ima za posljedicu da je poligon prosječne veličine relativno mali kad se projicira na zaslon, i utoliko traži da rezolucija bude što veća kako bi mogao pokazati svoje detalje. Drugi čimbenik koji traži najveće moguće povećanje rezolucije je sama priroda tradicionalnih tehnika, koje su crtane na papiru i nisu pikselizirane.

Zato, s druge strane, na sasvim jednostavnim, grubim modelima detalji koji su iscrtani na teksturama dolaze do punog izražaja i na najnižim rezolucijama.

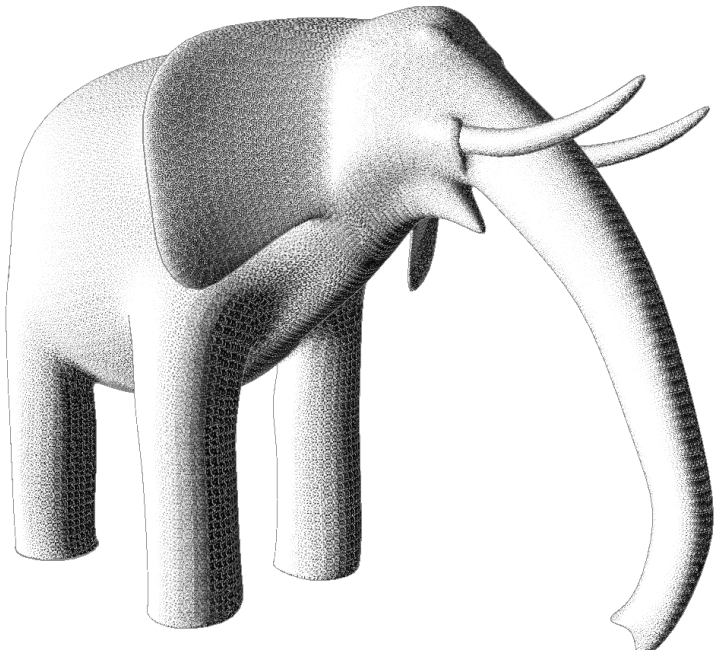
Najzahvalnijim tradicionalnim tehnikama s obzirom na problem s pikselizacijom pokazale su se one zasnovane na točkastim elementima. One su i pri najvećem smanjenju tekstura uslijed *mip-mappinga* pokazale sposobnost da zadrže svoj svojstven izgled.



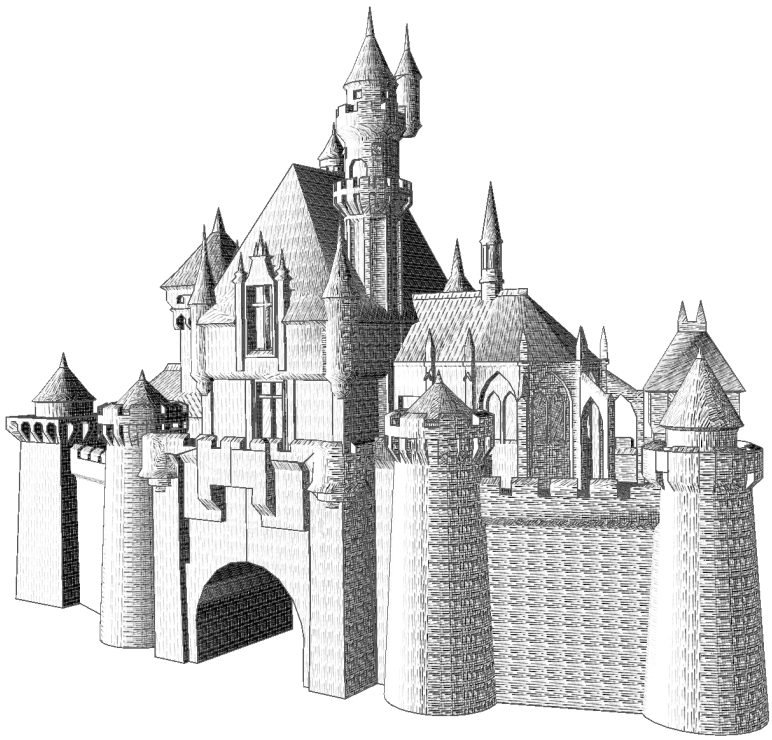
Slika 6.1: Tehnika dugih crta ostavlja dojam crtanja olovkom



Slika 6.2. Tehnika raspršenih crtica pogodna je za prikaz krzna ili perja



Slika 6.3. Oslikavanje finim točkicama ostavlja dojam specifične slikarske tehnike



Slika 6.4. Objekti s oštrim bridovima zadržavaju njihovu naglašenost

Na ispitnom računalu, koje možemo po snazi svrstati u nižu klasu, s procesorom Intel Core 2 Duo s dvije jezgre na 1,86 GHz i karticom GeForce 7600, programsko rješenje u animaciji postiže vrlo ugodnu brzinu osvježavanja prikaza, više nego dovoljnu za prikaz u stvarnom vremenu za sve ispitane modele, pa i one s više od 60.000 poligona. Mjerenje brzine je izvedeno pri jednolikoj rotaciji, pri čemu su tijela rotirana jednakim brojem punih rotacija oko osi, te je izračunat prosjek brzina tijekom proteklog vremena. Dobiveni rezultati navedeni su u tablici 6.1.

Tablica 6.1. Broj slika u sekundi (FPS) izmjeren u animaciji modela, s programiranim jedinicama za sjenčanje (shaderima) i bez njih

Ime modela	**Broj vrhova**	**Broj poligona**	**FPS bez *shadera***	**FPS sa *shaderima***
elephant.obj	9.402	18.792	13	86
dinnerware.obj	3.409	6.798	28	87
crank25k.obj	12.489	24.999	10	70
swan.obj	842	1.678	85	87
castle.obj	7.687	13.103	16	86
panda.obj	2.400	4.757	41	86
treefrog.obj	9.413	18.792	12	86
griffon.obj	18.656	37.039	6	46
f16.obj	2.344	4.592	42	87
columns.obj	2.110	4.210	21	87
gargoyle.obj	21.379	40.348	5	45
cafehippo.obj	16.002	61.374	4	42

Na rezultate je najviše utjecaja imala optimizacija iscrtavanja tekstura vlastitim programiranjem jedinica za sjenčanje, koja je toliko ubrzala prikaz da je usko grlo jednostavno premješteno s grafike na ostale dijelove sustava. Nakon tog zahvata brzina praktički više ne ovisi o složenosti objekta.

Gledano na sustav prije optimizacije, od velikog utjecaja je bilo korištenje tehnike *face*

cullinga, kojom se stražnje poligone izuzimalo iz razmatranja, čime se otprilike prepolovio i broj potrebnih operacija s teksturama. Ukupno gledano, na operacije s teksturama trošilo se praktički čitavo vrijeme obrade.

Izračun obrisa tijela je pak u odnosu na rad s teksturama potpuno zanemariv, iako se izvodi svaki put iznova pri svakoj promjeni tijekom animacije.

U poslovima početne obrade modela, što uglavnom zauzima analitičko izračunavanje glavnih smjerova u vrhovima, potroši se na korištenom računalu od nekoliko sekundi kod srednje složenih do dvadesetak sekundi kod najsloženijih od spomenutih modela.

7 Zaključak

Zamisao prikaza tradicionalnih poteza kistom i drugih likovnih tehnika pomoću tonskih mapa je vrlo zahvalna za izvedbu široke palete tradicionalnih tehnika. Pojava svojstvena tom rješenju (stapanju šest tekstura zajedno) je da potezi postepeno blijede ili se pojavljuju novi kako se površina tijela kreće od svjetla prema tami ili obratno. No, to nije prikladno za neke tehnike koje ovdje nisu razmstrsne, primjerice crtanje poteza tintom, gdje potezi trebaju postati deblji, a ne jače vidljivi. Također ni tehnike koje crtaju valovito, oponašajući drhtanje ljudske ruke, ne bi se mogle izvesti teksturama i rasporediti po poligonima.

Drugi problem vezan je za samu tehnologiju prikaza tekstura, koja otvara problem s područjima gledanim pod oštrim kutom i traži rješenje u vidu anizotropnog filtriranja.

Ugodnim iznenađenjem pokazao se izračun obrisa tijela, koji i u jednostavnom, neoptimiziranom obliku nije bio nimalo zahtjevan ni kod najsloženijih modela. U tom pogledu zaključak je da visoko optimizirani postupci kakvi su osmišljeni ranije, a koji su svjesno prihvaćali i određenu grešku kao žrtvu performansama, na današnjim računalima više nisu potrebni.

Još ugodnije iznenađenje bili su rezultati programiranja jedinica za sjenčanje, koji su jednostavno otklonili sve brige oko performansa izvođenja.

Naposlijetku, ostvareno programsko rješenje u svom algoritmu za analizu tijela traži prilično velike količine memorije, što doseže i na stotine megabajta kod modela građenih od više desetaka tisuća poligona, pa to za sada ostavlja najveći prostor za poboljšanje.

8 Literatura

1. E. Praun, H. Hoppe, M. Webb, A. Finkelstein: *Real-time hatching*. Proceedings of SIGGRAPH 2001, str. 581.

2. A. Girshick, V. Interrante, S. Haker, T. Lemoine: *Line Direction Matters: An Argument for the Use of Principal Directions in 3D Line Drawings*. Proceeding of NPAR June 2000, str. 43–52.

3. *Uvod u diferencijalnu geometriju,* http://www.grad.hr/itproject_math/

4. F. Cazals, M. Pouget: *Estimating Differential Quantities Using Polynomial Fitting of Osculating Jets*. Computer Aided Geometric Design 22, 2005. str. 121–146.

5. Mihajlović, Ž: *Računalna grafika*, predavanja, FER, ZEMRIS, 2006.

9 Sažetak

Rad ostvaruje prikaz proizvoljnog trodimenzionalnog poligonalnog modela nefotorealističnom tehnikom zasnovanom na korištenju tekstura organiziranih u tonske mape. Izvedeno je oponašanje više tradicionalnih tehnika crtanja potezima i oslikavanjem površine točkama. Ostvaren je jedan od uobičajenih matematičkih modela ispitivanja zakrivljenosti površine te model prilagodbe tekstura za što sugestivnije predstavljanje oblika te prostornu i vremensku koherentnost prikaza, uz dodatnu analizu za prikaz tijela pravokutnih oblika. Konačno rješenje optimizirano je vlastitim programiranjem jedinica za sjenčanje.

Ključne riječi: nefotorealistični prikaz, teksture, crtanje, rezbarenje, trodimenzionalni model

10 Abstract

This paper describes an implementation of a non-photorealistic rendering scheme using a collection of images called a tonal art map, intended for use on arbitrary three-dimensional meshes. Some traditional techniques of hatching and stippling have been shown. One of the common curvature analysis models has been implemented, as well as a model for adaptation of textures to the shape in a suggestive manner, with special attention on rectangular forms analysis. Resulting solution has been further optimized by custom shader programming.

Keywords: non-photorealistic rendering, textures, hatching, stippling, three-dimensional mesh