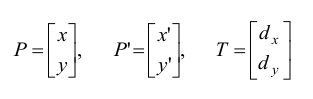
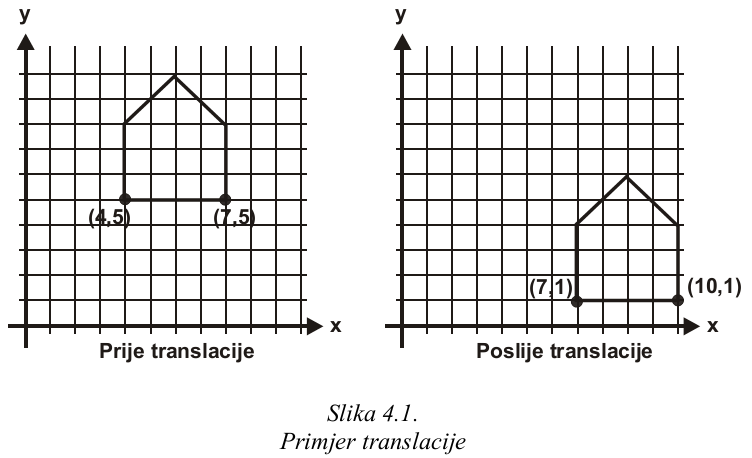
**GEOMETRIJSKE TRANSFORMACIJE**

**2D TRANSFORMACIJE**

*Translacija*

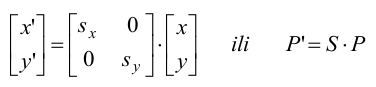
Translatiramo sve točke neke dužine translatiranjem samo njenih rubnih točaka i iscrtavanjem ravne crte među njima. Isto vrijedi i za skaliranje i rotaciju.



*Skaliranje*

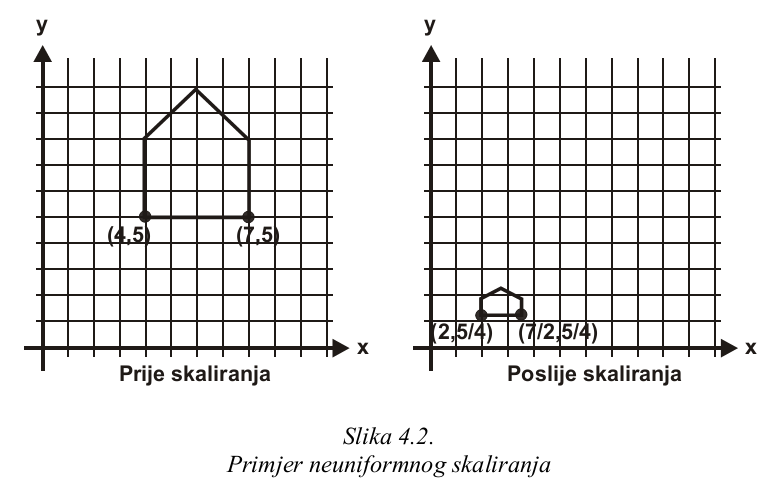
Točke mogu biti skalirane za vrijednost sx po x osi i sy po y osi, a nove koordinate točaka računaju se iz umnoška.

S – matrica skaliranja

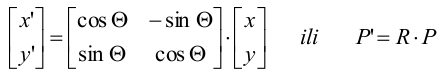
Skaliranje može biti:

* Diferencijalno skaliranje (sxsy)
* Uniformno skaliranje (sx = sy)

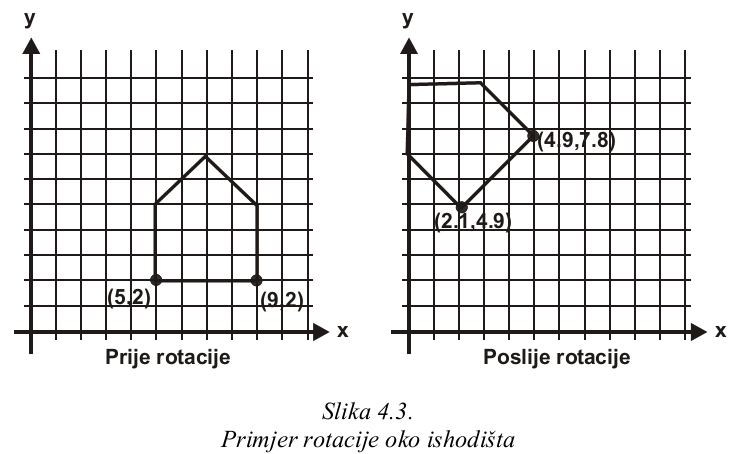
**

*Rotacija*

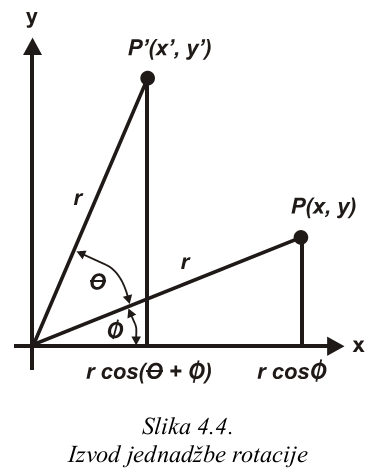
Rotacija točaka oko ishodišta za kut može se matematički definirati kao



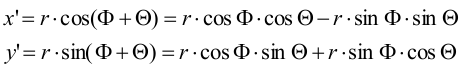
R – matrica rotacije



Pozitivna orijentacija kuta rotacije je obrnuta od smjera rotacije kazaljki na satu. Dokaz:









*Homogene koordinate i matrični zapis 2D transformacija*

Matrični zapis translacije, skaliranja i rotacije su respektivno:

translacija - zbrajanje matrica

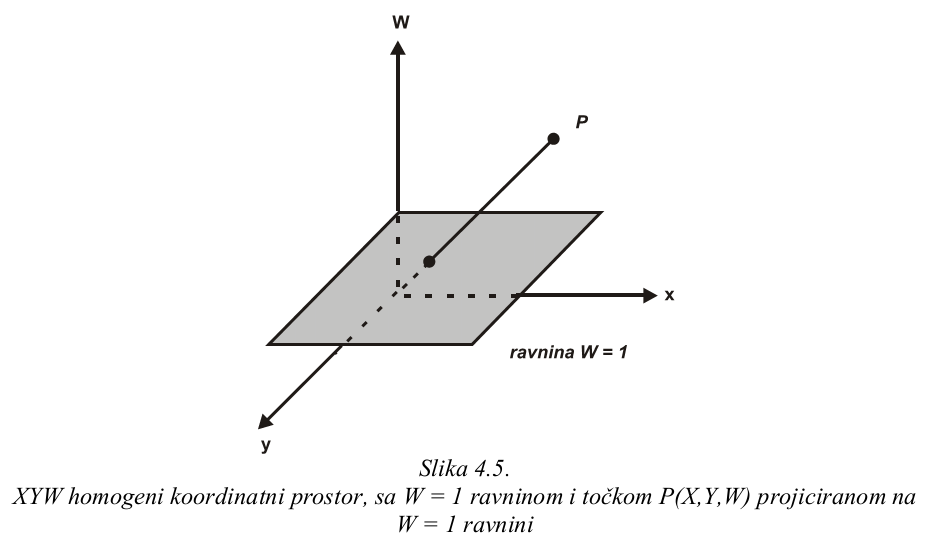
rotacija i skaliranje - množenje matrica

Problem je pri kombiniranju različitih transformacija. Predstavljanje točaka homogenim koordinatama omogućava predstavljanje sve tri transformacije kao množenje. Umjesto predstavljanja para brojeva (x,y), svaku točku predstavljamo s uređenom trojkom (x,y,W). Dva skupa homogenih koordinata (x,y,W) i (x',y',W') predstavljaju istu točku ako i samo ako je x' = tx; y' = ty i W' = tW gdje je t konstanta proporcionalnosti.

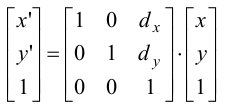
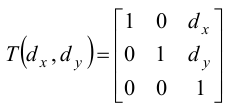
Na primjer:

(2,3,6) i (4,6,12) predstavljaju istu točku predstavljenu različitim uređenim trojkama.

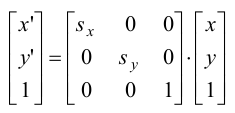
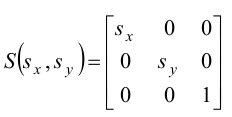
Bar jedna od homogenih koordinata mora biti različita od nule: (0,0,0) nije dozvoljeno. (x,y,W) predstavlja istu točku kao i (x/W,y/W,1). x/W i y/W nazivamo Kartezijeve koordinate homogenih točaka.



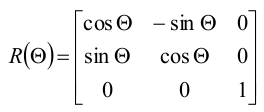
Translacijska matrica 3x3 za homogene koordinate ima oblik

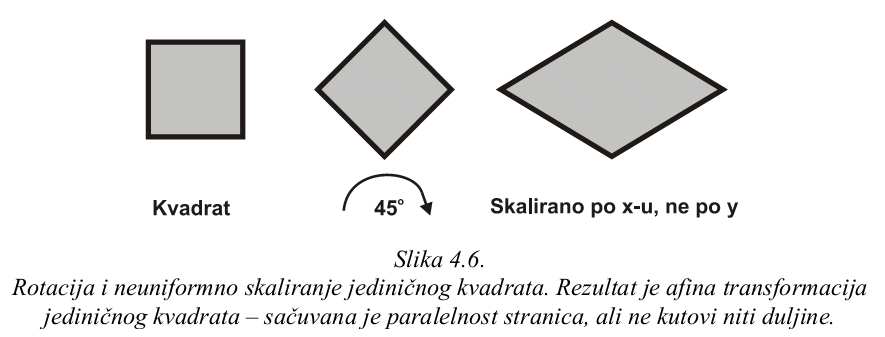
Jednadžbe skaliranja dane izrazom mogu se napisati u matričnoj formi na sljedeći način

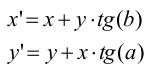
Rotacijska jednadžba može se napisati kao

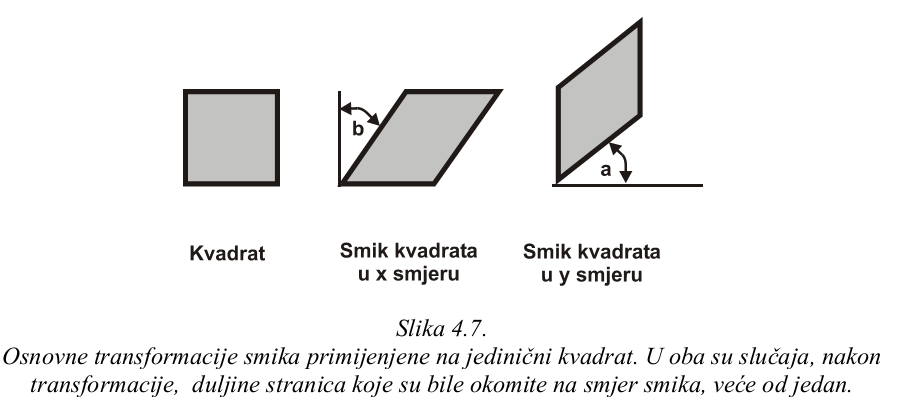
  

Navedene transformacije su afine transformacije tj. proizvoljna sekvenca takvih transformacija (rotacije, translacije i skaliranja) sačuvat će paralelnost linija ali ne duljinu i kutove.

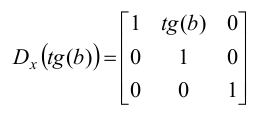
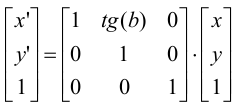


Spomenimo još jednu vrstu osnovnih transformacija koja je također afina. To je transformacija smika. Postoje dvije vrste transformacija smika: smik po x osi i smik po y osi (slika 4.7). Transformacija smika za kut a u odnosu na koordinatnu os x i kut b u odnosu na koordinatnu os y definirana je sljedećim jednadžbama:

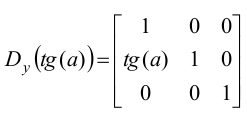
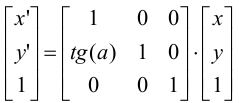




Definirajmo matricu smika po x-u Dx kao

Smik po y osi:

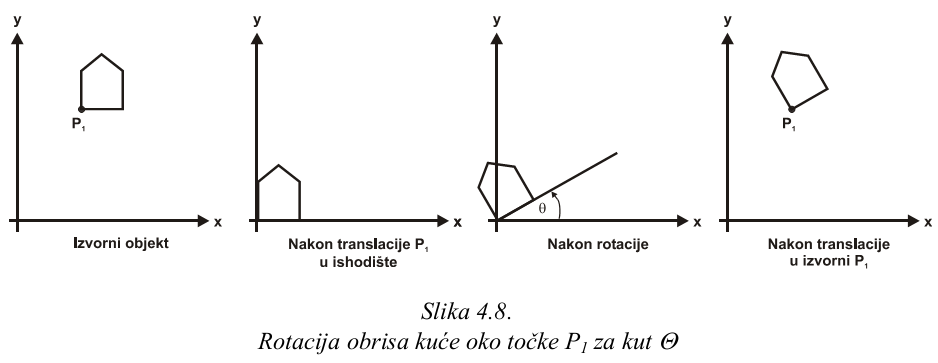
Smik po x i y osi:

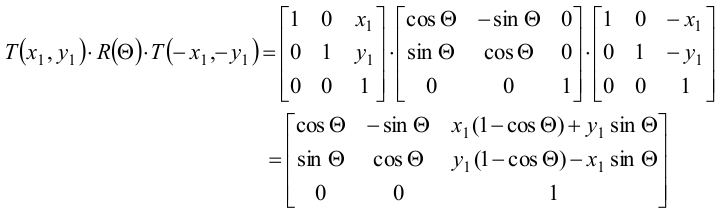
*Kompozicija 2D transformacija*

Primjer uzmimo rotaciju nekog objekta oko proizvoljne točke P1. Matrica rotacije vrši rotaciju s obzirom na ishodište, pa stoga problem rješavamo u tri koraka:

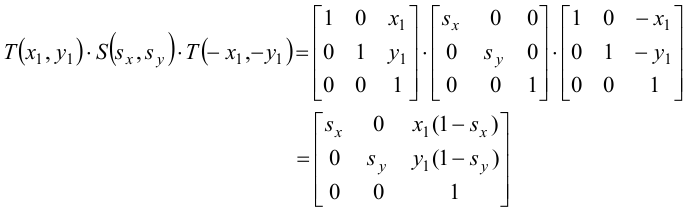
1. Translatiramo objekt tako da se točka P1 preslika u ishodište
2. Rotiramo objekt
3. Translatiramo objekt tako da se točka ishodišta preslika u točku P1



Rezultat je različit od rezultata koji bi dobili da smo primijenili samo rotaciju.



Sličan pristup rješavanju složene transformacije koristimo kada želimo skalirati neki objekt s obzirom na proizvoljnu točku P1. U prvom koraku vršimo translaciju tako da se točka P1 preslika u ishodište, zatim skaliramo, a potom translatiramo natrag u P1.



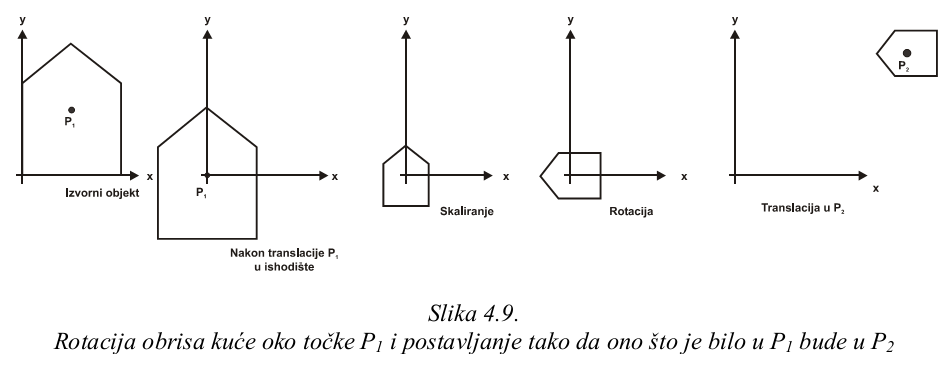
ukupna transformacija

Primjer:

Skaliranje, rotacija, pozicioniraje

1. translacija P1 u ishodište
2. skaliranje i rotacija
3. translacija u novu točku P2





*Transformacija prozor – otvor*

U računalnoj grafici sliku možemo definirati kao skup projekcija objekata iz prostora nekog svijeta koji može biti stvaran ili virtualan. Dimenzije i položaj slike se mogu slobodno odabrati. Nužno je obaviti transformaciju scene iz svijeta (odnosno koordinatnog sustava svijeta) u koordinatni sustav prikaznog uređaja. Taj proces može se odvijati u više koraka i ti koraci se na različit način definiraju u okviru različitih grafičkih standarda.

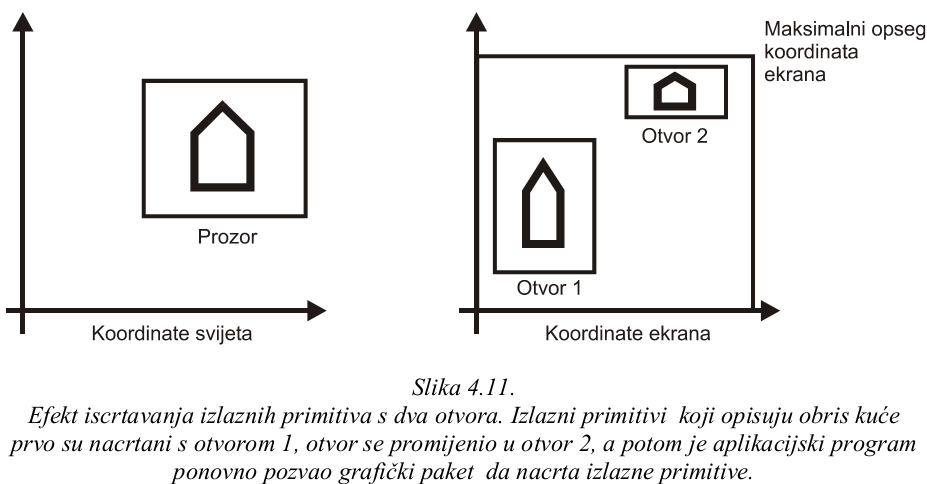
Izvorna scena smještena u koordinatnom sustavu svijeta, a konačna slika u koordinatnom sustavu prikaznog uređaja.

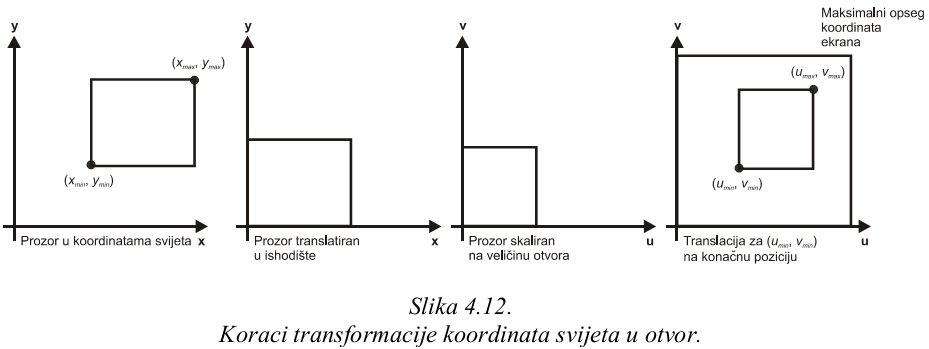
**Prozor** (*window*): Izdvojeni dio koordinatnog sustava svijeta koji sadrži scenu koju se želi prikazati.

**Otvor** (*viewport*): Izdvojeni dio koordinatnog sustava prikaznog uređaja koji sadrži sliku odabrane scene.

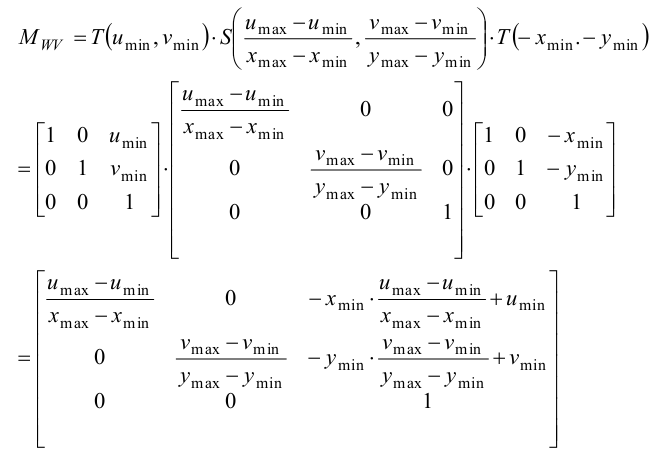
U okviru prikazne transformacije aplikacijski program izvodi transformaciju prozora u otvor uz moguću nejednoliku promjenu faktora proporcionalnosti. Postupak transformacije koordinata sadrži sljedeće elementarne transformacije:

* translacija prozora u ishodište (T1),
* promjena faktora proporcionalnosti (S),
* translacija otvora u željeni položaj (T2).

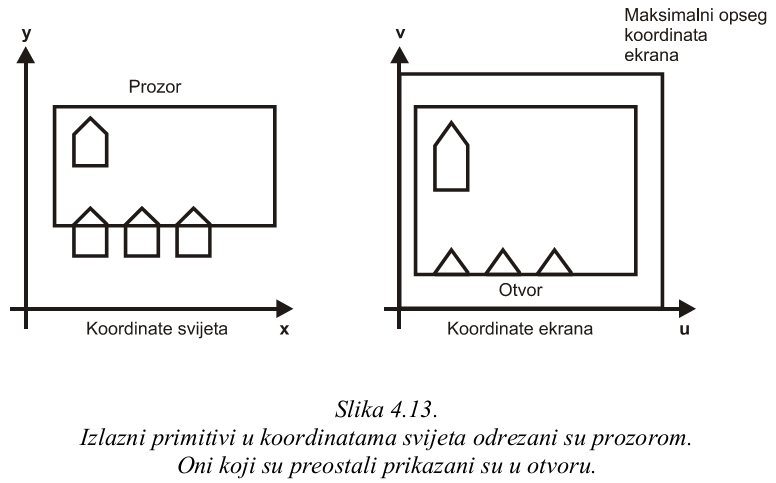




Transformaciju prozor – otvor obavljamo matricom složene transformacije koja se dobije kao umnožak elementarnih transformacijskih matrica:



Mnogi grafički paketi kombiniraju transformaciju prozor – otvor s odrezivanjem izlaznih primitiva prozorom.



**3D TRANSFORMACIJE**

*Matrični prikaz 3D transformacija*

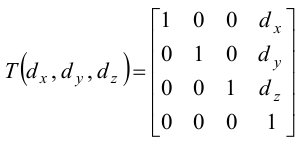
Umjesto prikazivanja točke u prostoru s 3 koordinate (x,y,z), koristimo zapis (x,y,z,W) gdje dvije uređene četvorke predstavljaju istu točku ako je jedna sastavljena od višekratnika prve (različitih od nula). Uređena četvorka (0,0,0,0) nije dozvoljena. Uobičajeni zapis točke (x,y,z,W) uz je (x/W,y/W,z/W,1). Ovaj način zapisa se zove homogeni zapis.

3D koordinatni sustav koji ćemo koristiti je definiran pravilom desne ruke.

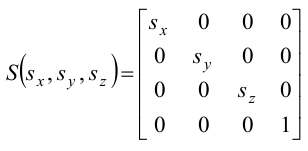


*Translacija*

Translacija u 3D je jednostavno proširenje one iz 2D:

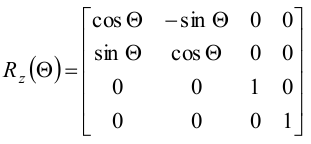
 

*Skaliranje*



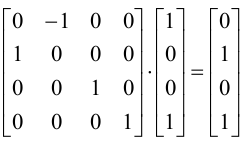
Provjera: 

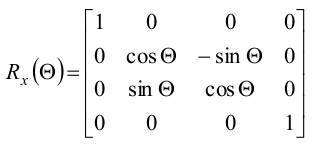
*Rotacija*

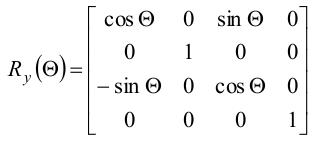
Izraz za 2D rotaciju je zapravo 3D rotacija oko osi z, dakle 

Provjerimo ovu tvrdnju:

jedinični vektor duž osi x zarotirajmo za 90° i rezultat bi trebao biti jedinični vektor duž osi y tj. :

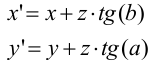


Matrica rotacije oko osi x: 

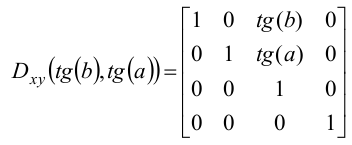
Matrica rotacije oko osi y: 

Stupci (i redci) lijevih gornjih 3x3 submatrica navedenih matrica rotacije su međusobno okomiti jedinični vektori. Determinante tih submatrica imaju determinantu = 1 što znači da su ortogonalne te čuvaju kutove i duljine.

3D transformacije smika po z osi možemo napisati na sljedeći način:



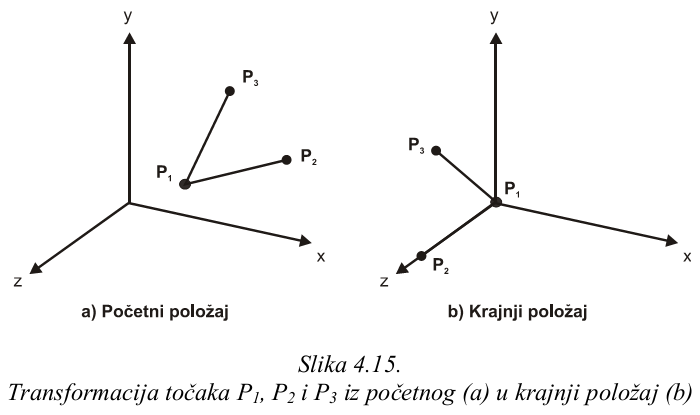
ili u matričnom obliku



Transformacije smika po x i y osi imaju sličan oblik.

**GEOMETRIJSKE TRANSFORMACIJE I OBJEKTI U 3D PROSTORU**

**KOMPOZICIJA 3D TRANSFORMACIJA**



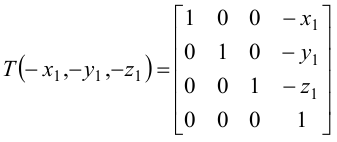
Potrebno je translatirati točku P1 u ishodište tako da segment P1P2 leži na pozitivnoj z osi, a segment P1P3 leži na pozitivnoj polovici y osi ravnine (y,z). Duljine segmenata ostaju nepromijenjene nakon transformacije.

Rješenje:

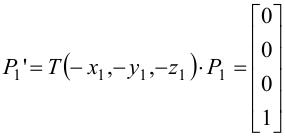
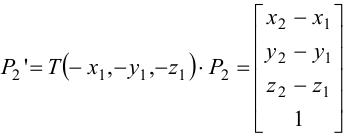
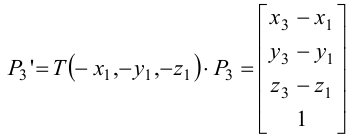
1. Translacija točke P1 u ishodište
2. Rotacija oko osi y tako da P1P2 leži u ravnini (y,z)
3. Rotacija oko osi x tako da P1P2 leži na osi z
4. Rotacija oko osi z tako da P1P3 leži u ravnini (y,z)

1. Translacija točke P1 u ishodište

Napišimo matricu translacije

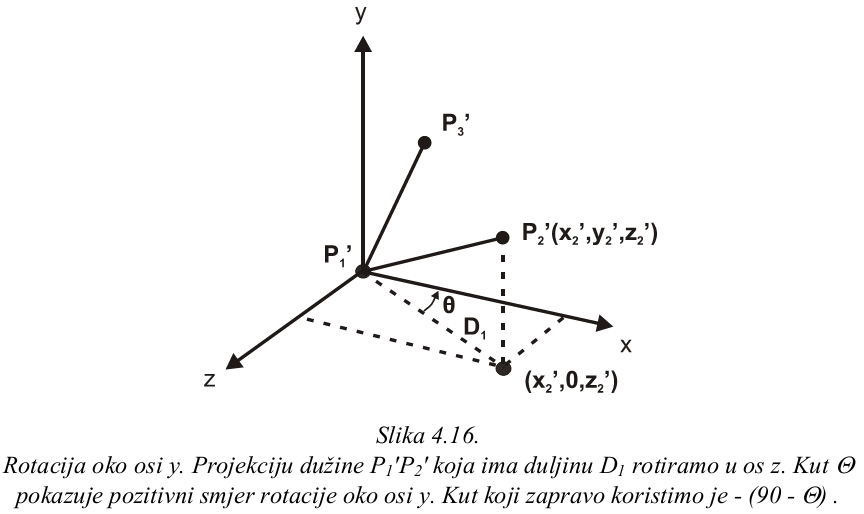


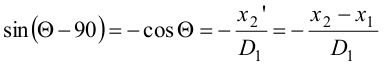
Primjenom matrice T na točke P1, P2 i P3 dobijemo

2. Rotacija oko osi y

Projekciju dužine P1'P2' koja ima duljinu D1 rotiramo u os z. Kut θ pokazuje pozitivni smjer rotacije oko osi y. Kut koji zapravo koristimo je: - (90 - θ) .

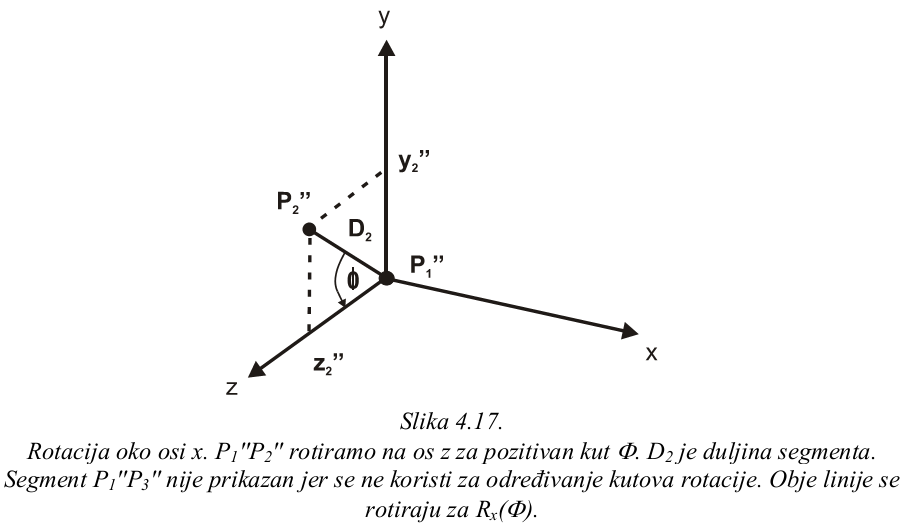


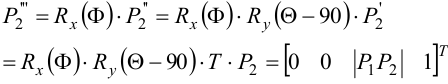
 

3. Rotacija oko osi x

P1''P2'' rotiramo na os z za pozitivan kut Φ. D2 je duljina segmenta. Segment P1''P3'' nije prikazan jer se ne koristi za određivanje kutova rotacije. Obje linije se rotiraju za Rx (Φ).

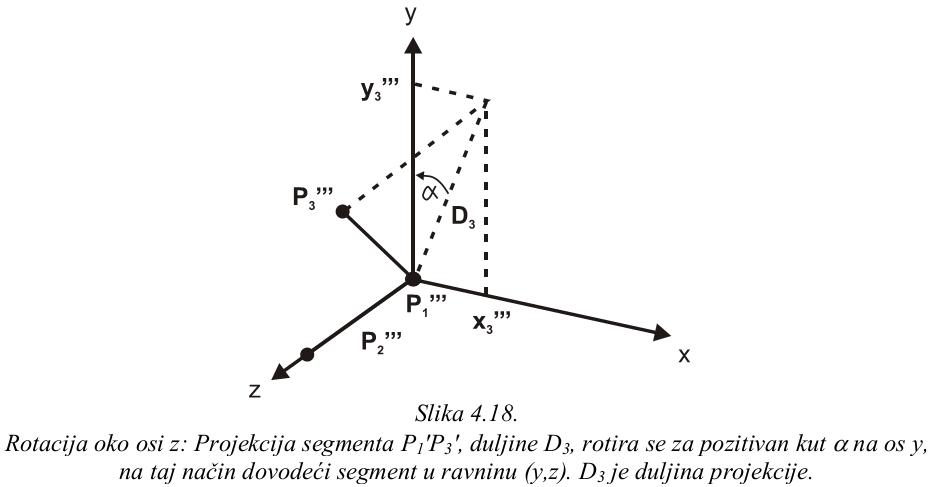




4. Rotacija oko osi z

Projekcija segmenta P1'P3', duljine D3, rotira se za pozitivan kut α na os y, na taj način dovodeći segment u ravninu (y,z). D3 je duljina projekcije.



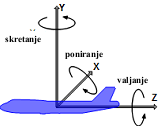




Ukupna matrica transformacije:



**ROTACIJA POMOĆU EULEROVIH KUTOVA**

****

Način prikaza općenite rotacije pomoću rotacija oko 3 osi. Rotacija se definira:

R=Rz(r) x Rx(p) x Ry(h)

gdje su h – skretanje (engl. head); p – poniranje (engl. pitch); r – valjanje (engl. roll)

Rotacije se vrše u lokalnom koordinatnom sustavu. Intuitivno je.

Problemi:

* Blokada kardana (engl. gimbal lock)
* Moguće nepredvidljivosti prilikom interpolacija.

**ROTACIJA POMOĆU QUATERNIONA**

Rješavaju probleme koji se pojavljuju kod rotacije Eulerovim kutovima. Quaternioni su 4-dimenzionalni vektori. Predstavljaju proširenje kompleksnih brojeva s 3 imaginarne komponente. Neintuitivni – teško je predočiti rotaciju na temelju quaterniona.

**OBJEKTI U 3D PROSTORU**

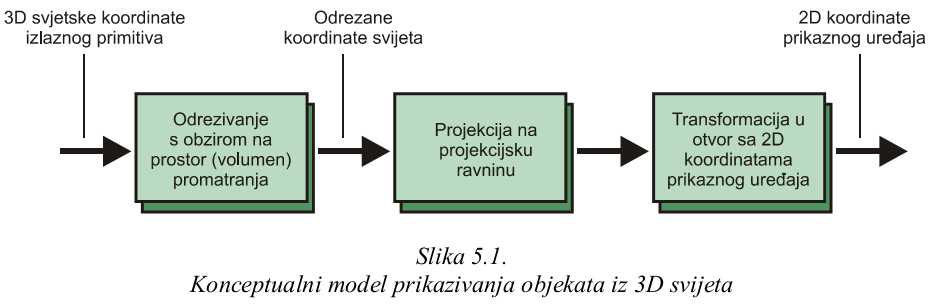
U 2D svijetu, postupak prikazivanja obuhvaća:

1. određivanje prozora iz tog svijeta
2. njegovo prikazivanje u otvoru (viewport) na nekoj 2D površini koju promatramo (npr. ekranu).

Konceptualno, objekti iz svijeta odrezuju se nekim prozorom i potom transformiraju u neki otvor za prikaz.

Prikaz 3D objekata je složeniji (nego prikaz 2D objekata) zbog:

* dodatne dimenzije
* ograničenosti prikaznih uređaja na dvije dimenzije.



**3D – PROJEKCIJE**

Projekcije:

* ravninske
* neravninske\*
* negeometrijske\*

\*Projekcija na zakrivljenu površinu odnosno uz uporabu zakrivljenih projekcijskih zraka (npr. mnoge kartografske projekcije).

*Projekcije*

Ravninske geometrijske projekcije se u načelu odvijaju na sljedeći način:

1. **projekcijske zrake** (ili projektori) izlaze iz projekcijskog središta (*COP – center of projection*)
2. projekcijske zrake prolaze kroz sve točke objekta i presijecaju **projekcijsku ravninu** (*projection plane*) tvoreći u njoj oblik projekcije.

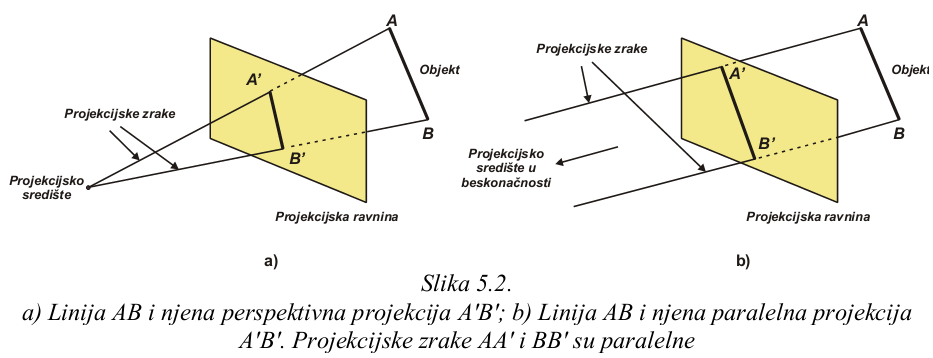
*Ravninske geometrijske projekcije*

Ravninske geometrijske projekcije se dijele na:

* perspektivne projekcije
* paralelne projekcije

Temeljna razlika je u odnosu projekcijskog središta i projekcijske ravnine.

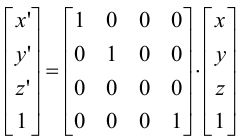
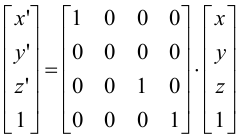
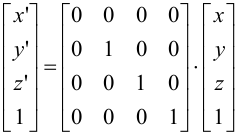
* **Paralelne** projekcije: udaljenost projekcijskog središta i projekcijske ravnine je beskonačna.
* **Perspektivne** projekcije: udaljenost projekcijskog središta i projekcijske ravnine je konačna.

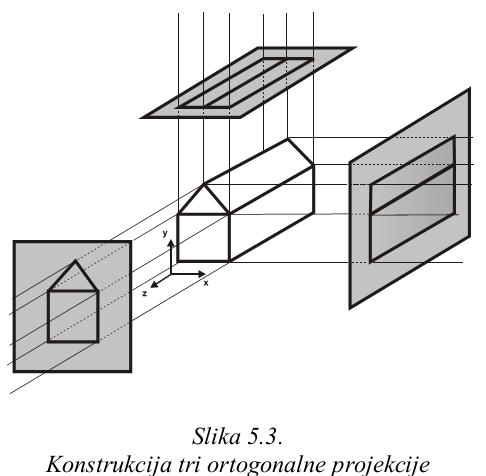


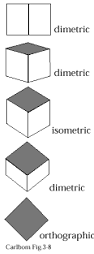
*Ortogonalna projekcija*

Ortogonalna projekcija je najjednostavnija paralelna projekcija. Ortogonalna projekcija preslikava sve točke objekta paralelnim zrakama na određenu ravninu. U slučaju ortogonalne projekcije na koordinatne ravnine projekcijske zrake su okomite na ravninu projekcije.

xy-ravnina xz-ravnina yz-ravnina



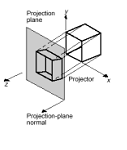


*Aksonometrijske projekcije*

Paralelne projekcije kod kojih projekcijska ravnina nije paralelna sa niti jednom od koordinatnih ravnina. Paralelne linije se skraćuju s istim faktorom.

* **Izometrijska**: Kutovi među sve tri glavne osi su jednaki (120º). Jednako skaliranje duž svake osi.
* **Dimetrijska**: Kutovi između dvije osnovne osi su jednaki; potrebna dva omjera skaliranja.
* **Trimetrijska**: Različiti kutovi između tri koordinatne osi; potrebna 3 faktora skaliranja.

*Kosa projekcija*



Kosa projekcija također pripada paralelnim projekcijama kao i ortogonalna tj. projekcijske zrake su međusobno paralelne. Razlika je u tom da projekcijske zrake nisu ortogonalne na projekcijsku ravninu nego na nju upadaju pod nekim kutom. Likovi koji su paralelni s ravninom projekcije projicirani su bez izobličenja.

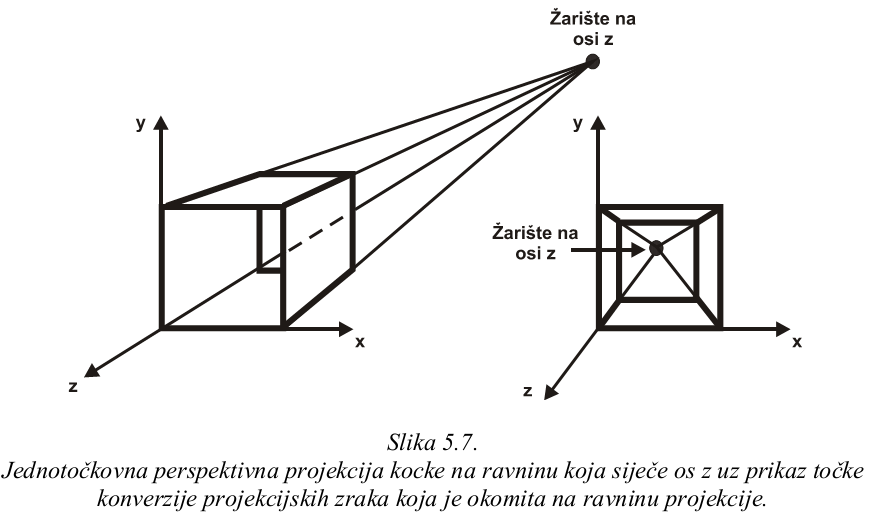
Neka je zadana točka P s homogenim koordinatama. Ako želimo odrediti kosu projekciju P'' točke P na xy-ravninu, potrebno je :

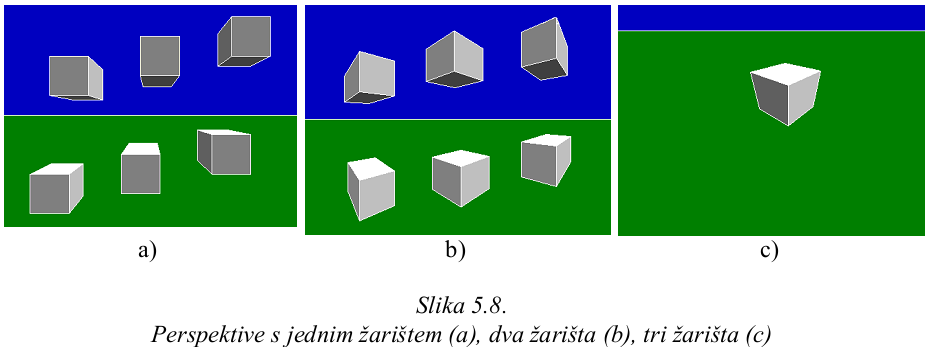
1. odrediti kut projekcije α za koji će se os z prikloniti prema osi x
2. odrediti i faktor prikrate n kojim ćemo množiti z vrijednosti točaka.

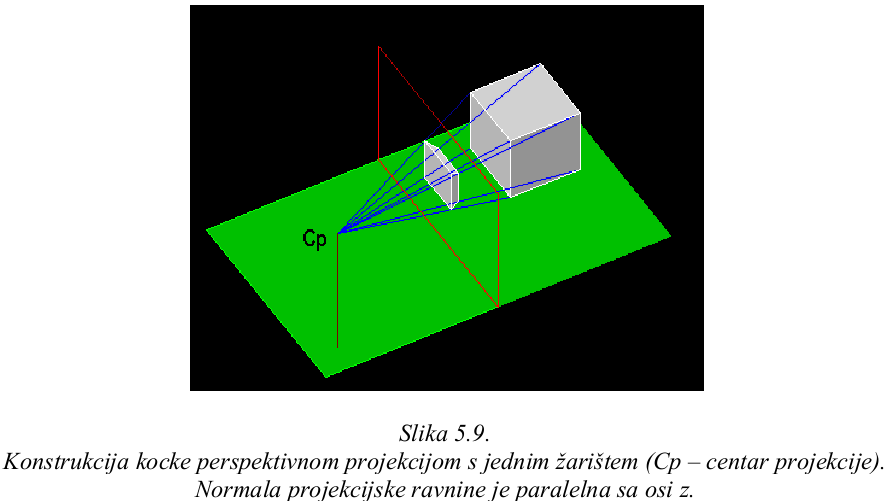
*Perspektivne projekcije*

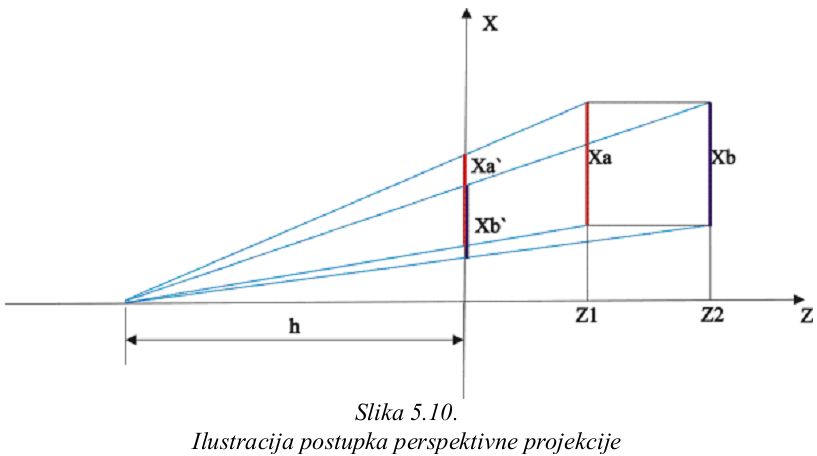
Perspektivne projekcije imaju neku konačnu udaljenost do projekcijskog središta u kojem konvergiraju paralelne linije koje nisu paralelne s projekcijskom ravninom. U 3D prostoru paralelne linije mogu se sječi samo u beskonačnosti -> projekcijsko središte možemo shvatiti i kao projekciju točke u beskonačnosti.

Ovakvu vrstu perspektive imamo kod fotografije i kod ljudskog vida. Stoga takvu vrstu projekcija nazivamo i fotografska a projekcijsko središte nazivamo i motrište.

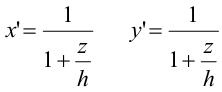
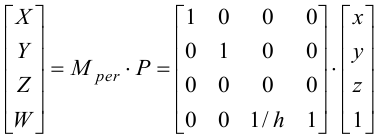




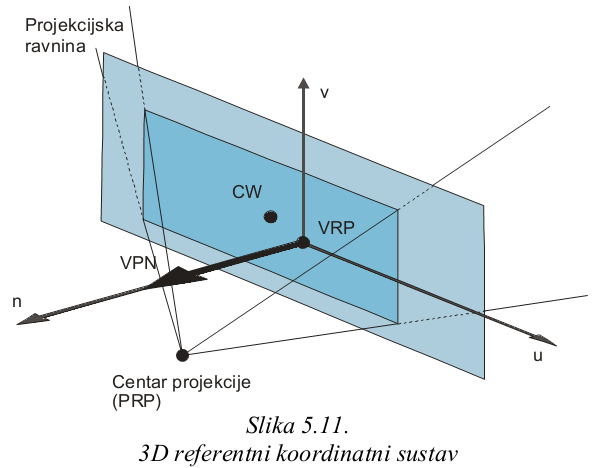




Promatrač se nalazi na udaljenosti h od projekcijske ravnine. Bridovi xa i xb preslikavaju se u dužine xa' i xb'.

**DEFINICIJA PROJEKCIJSKE RAVNINE OPSEGA POGLEDA**

****

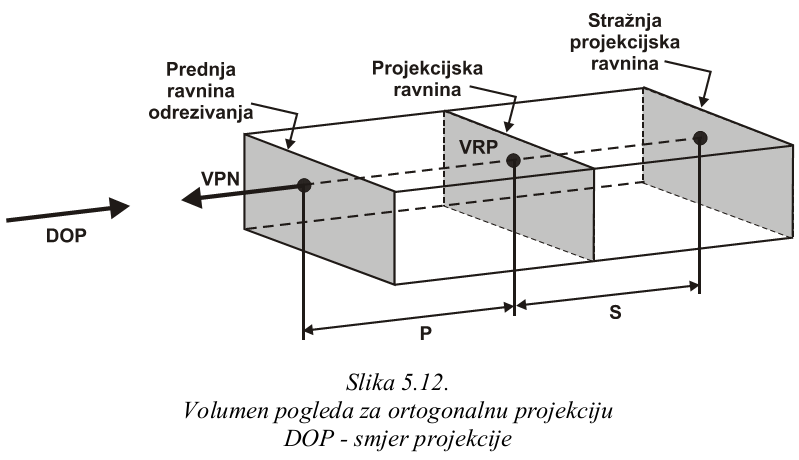
Pogled na 3D objekt u 2D koordinatnom sustavu zahtijeva projekcijsku ravninu i projekcijski volumen (view frustum, 3D view volume). Opseg pogleda definira koji dio prostora se projicira. Nakon što smo odredili o kojem se dijelu prostora (volumenu) radi -> potrebno je odrezati dijelove predmeta koji su izvan tog područja.

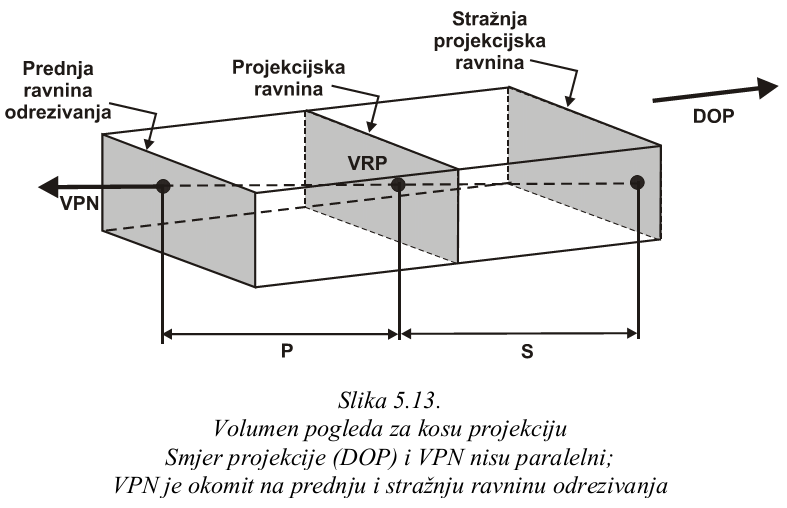
Ravnina pogleda može biti bilo gdje s obzirom na objekte u svijetu ili sceni koje želimo projicirati. Projekcijska ravnina ili ravnina pogleda (view plane) određena je točkom na ravnini koju nazivamo referentna točka pogleda (view reference point – VRP) i okomicom na ravninu pogleda (view plane normal – VPN). Uz zadanu ravninu pogleda, potreban je i prozor na toj ravnini. Za definiranje prozora u ravnini pogleda, potrebno je odrediti minimalne i maksimalne vrijednosti koordinata prozora duž dvije ortogonalne osi.

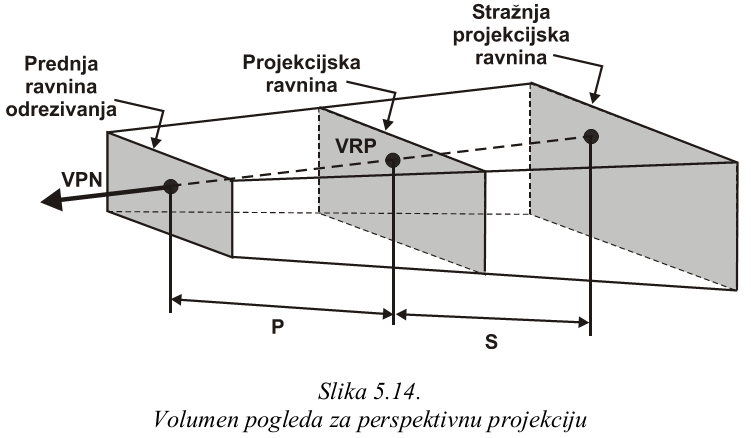
Ove osi su dio 3D referentnog koordinatnog sustava (View Reference Coordinate - VRC). Ishodište VRC sustava je točka VRP. Smjer osi u se određuje tako da jedinični vektori u, v i n formiraju koordinatni sustav po pravilu desne ruke.

**VOLUMEN POGLEDA (PROJEKCIJSKI VOLUMEN)**

Volumen pogleda ograničavamo kako bi se ograničili samo na neki konačni prostor promatranja. Prednja i stražnja ravnina odrezivanja su paralelne s projekcijskom ravninom i njihova normala je označena s VPN.







**PRIMJENA RAVNINSKIH GEOMETRIJSKIH PROJEKCIJA**

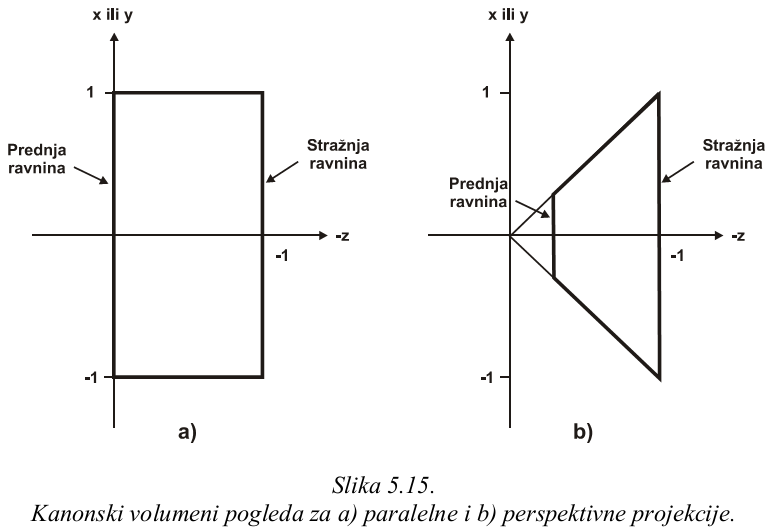
*Postupak prikazivanja 3D objekata koji se provodi*

Zadani su volumen pogleda i projekcija. Linije se (s obzirom na volumen pogleda) odrezuju izračunavanjem njihova sjecišta sa svakom od šest ravnina koje definiraju taj volumen. Linije koje ostanu nakon odrezivanja projiciraju se na ravninu promatranja rješavanjem sustava jednadžbi koje se odnose na presjek projekcijskih zraka i ravnine promatranja. Opisana procedura zahtijeva veliki broj kalkulacija koje se ponavljaju za svaku liniju koju želimo prikazati.

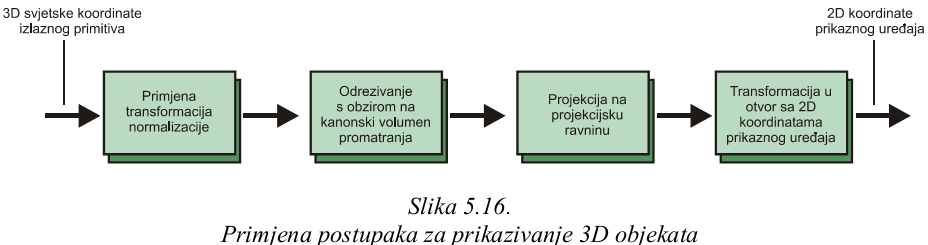
Učinkovitija procedura koja se temelji na podijeli cijelog postupka na više malih, jednostavnijih koraka. Neke volumene pogleda lakše je odrezivati od drugih. Na primjer, presjek linije sa svakom ravninom volumena pogleda paralelne projekcije jednostavno se računa ako je zadano sljedećih šest ravnina: 

Također, izračunavanje je jednostavno i ako je volumen pogleda perspektivne projekcije zadan sa ravninama:

To su tzv. kanonski volumeni pogleda.



Strategija rješavanja temelji se na pronalaženju tzv. transformacija normalizacije Npar i Nper koje transformiraju neki proizvoljni volumen pogleda (paralelne projekcije ili perspektivne projekcije) u paralelni ili perspektivni kanonski volumen pogleda, respektivno. Potom se provodi odrezivanje, projiciranje u 2D. Ova strategija transformira i one točke koje će se naknadno odrezati, ali je zato odrezivanje jednostavno.



Normalizacijska transformacija paralelne projekcije Npar sastoji se od sljedećih transformacija:

1. Translacija referentne točke pogleda (VRP) u ishodište.
2. Rotacija referentnog koordinatnog sustava tako da os n (VPN) postane z os, os u postane x os, a os v postane os y.
3. Smik kojim smjer projekcije postaje paralelan sa osi z.
4. Translacija i skaliranje u kanonski volumen pogleda paralelne projekcije

Normalizacijska transformacija perspektivne projekcije Nper sastoji se od sljedećih transformacija:

1. Translacija referentne točke pogleda (VRP) u ishodište.
2. Rotacija referentnog koordinatnog sustava tako da os n (VPN) postane z os, os u postane x os, a os v postane os y.
3. Translacija tako da središte projekcije (COP), zadano točkom PRP bude u ishodištu.
4. Smik kojim središnja linija volumena pogleda postaje os z.
5. Skaliranje u kanonski volumen pogleda perspektivne projekcije.

**KRIVULJE**

**MODELIRANJE**

Uobičajeni načini modeliranja:

* **POLIGONALNO MODELIRANJE** - zapisivanje točaka preko njegovih X, Y i Z koordinata, koje se kasnije linearno spajaju stvarajući obris objekta. Računalo zatim izračunava plohe omeđene tim linijama. Te plohe se nazivaju poligonima.
* **NURBS** (engl. non-uniform, rational B-spline) je modeliranje s linijama čija zakrivljenost ovisi o upravljačkim točkama. Smještaj točaka u prostoru regulira zakrivljenost, dužinu i poziciju linija. Tako grupa od šest točaka jednako razmaknutim jedna od druge tvore krug. Dodavanjem sedme točke u blizini jedne od postojećih, krug ce se izdužiti u smjeru tih dviju točaka.

Korak modeliranja sastoji se od oblikovanja zasebnih objekata koji se kasnije koriste u sceni. Postoje brojne tehnike modeliranja, npr.:

* **Modeliranje konstruktivnim gotovim oblicima** (engl. *constructive solid geometry*) - tehnika modeliranja u kojem se gotov model dobije sastavljanjem od dvaju ili više geometrijska objekta.
* **Modeliranje** **preko** **implicitnih** **površina** (engl. *implicit* *surfaces*) - odnosi se na modeliranje jednostavnih geometrijskih formi (poligona) koji su smješteni jedan do drugog, te u prostoru stvaraju plohu, odnosno volumen.
* **Modeliranje** **sa** **pod**-**podijeljenim** **površinama** (engl. *subdivision* *surfaces*) - pri modeliranju glatkih površina, ovom tehnikom se postojeći poligoni dijele, što stvara veću iluziju fine zakrivljenosti objekta.

**KRIVULJE: UVOD**

Na najnižem nivou grafičko sklopovlje prikazuje točke, linijske segmente i poligone (obično predstavljeni kao trokuti i četverokuti).

Glatke krivulje – aproksimacija velikog broja linijskih segmenata.

Matematički opis krivulja – mali broj parametara (kontrolnih točaka).

Posljedica: manji utrošak memorije, precizan opis, moguća bilo koja preciznost renderiranja, izračun vektora normala u bilo kojoj točci.

**KRIVULJE I POVRŠINE**

*Krivulje*

Najjednostavniji pristup modeliranju krivulje je linearna aproksimacija (prvog reda). Krivulja se aproksimira višestrukim crtama (niz povezanih ravnih crta) ili mnogokutima.

Točnost aproksimacije određena je brojem linearnih segmenata kojima se aproksimira pojedini dio krivulje. Za visoku razinu podudarnosti linearnog aproksimacijskog modela i željene krivulje potreban je velik broj linearnih segmenata.

Veća razina podudarnosti odnosno bolja aproksimacija uz manji broj pojedinačnih segmenata - primjena aproksimacija višeg reda Najčešće se koriste aproksimacije trećeg reda. Aproksimacije nižeg reda ne daju dovoljno fleksibilnosti za oblikovanje različitih krivulja. Aproksimacije višeg reda su računski zahtjevnije i složenije za primjenu.

Postoji više oblika matematičkog prikaza krivulja za aproksimacije višeg reda:

1. eksplicitni
2. implicitni
3. parametarski

Eksplicitni oblik koordinate y i z izražava kao eksplicitne funkcije koordinate x: y=y(x) , z=z(x). Ovaj oblik zapisa ima sljedeće nedostatke:

* nisu moguće višestruke vrijednosti x-a (kao npr. kod kružnica);
* nije sačuvana rotacijska invarijantnost (nije jednostavno rotirati krivulju),
* teškoće s vertikalnim tangentama (zbog beskonačnog iznosa nagiba).

Implicitni oblik jednadžbe krivulje zapisujemo na sljedeći način: f(x,y,z)=0.

Njegovi nedostatci u primjenama računalne grafike su sljedeći:

* problem s višestrukim rješenjima (potrebno je postavljati dodatne uvjete za izbor željenog rješenja),
* problem s kontinuitetom tangenti u dodirnim točkama različitih segmenata (podudarnost smjera).

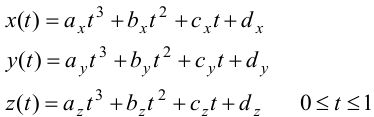
Parametarski oblik jednadžbe krivulje sve tri koordinate izražava kao funkcije parametra t: x=x(t), y=y(t), z=z(t). Ovaj oblik jednadžbe krivulje nema prethodne navedene nedostatke eksplicitnog i implicitnog oblika te je stoga najprikladniji za modeliranje krivulja u računalnoj grafici.

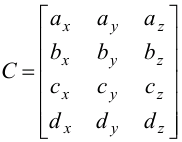
*Parametarske krivulje trećeg reda*

Najčešće se koriste za modeliranje krivulja u računalnoj grafici jer omogućavaju dovoljno fleksibilnosti za oblikovanje različitih krivulja uz prihvatljivu razinu složenosti. Polinomi nižeg reda općenito nemaju dovoljno fleksibilnosti u oblikovanju željene krivulje. Polinomi višeg reda složeniji i računski zahtjevniji.

Polinomi prvog i drugog reda ne ispunjavaju zahtjev da željena krivulja ima zadanu početnu i krajnju točku uz zadane vrijednosti derivacije u njima. Parametarske krivulje trećeg reda su krivulje najnižeg reda koje nisu ravninske u 3D.

Model krivulje se specificira po odsječcima polinomima trećeg reda. Svaki odsječak Q opisan je s tri funkcije (polinoma trećeg reda) x, y i z parametra t na sljedeći način:



vektor potencija parametra t: 

matrični oblik koeficijenata polinoma:

izraz za model odsječka krivulje : 

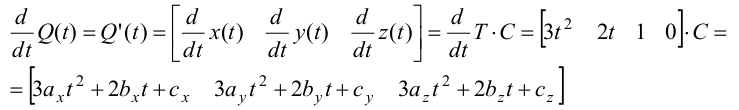
Matricu koeficijenata C, možemo napisati kao umnožak : C=M\*G

M - bazna matrica veličine 4 x 4,

G - geometrijski vektor (vektor s četiri elementa koji predstavljaju geometrijska ograničenja)

Geometrijska ograničenja su, na primjer, krajnje točke ili tangente koji određuju neku krivulju.

vektor smjera tangente:



Cjeloviti model željene krivulje tvori se sastavljanjem modela pojedinih odsječaka.

Razina glatkoće krivulje na spoju dvaju odsječaka izražava se u smislu dviju vrsta kontinuiteta:

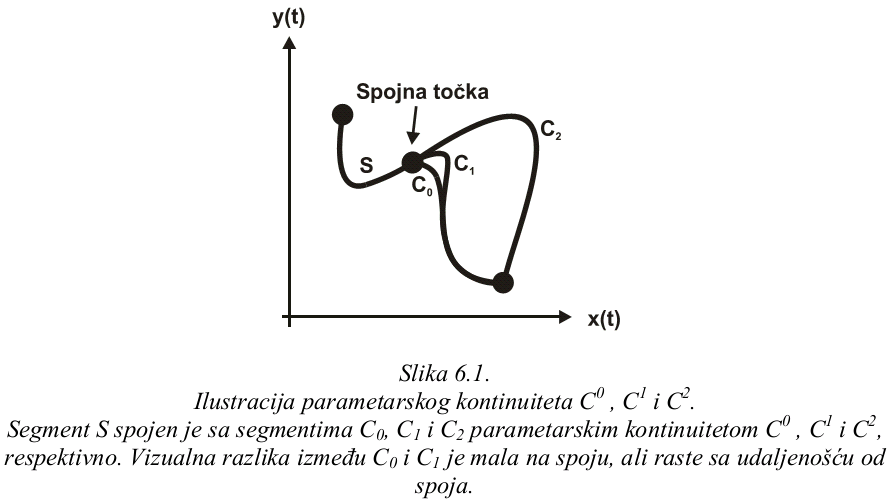
* geometrijskog kontinuiteta G
* parametarskog kontinuiteta C

Geometrijski kontinuitet definiran je na sljedeći način:

* geometrijski kontinuitet G0 - neprekinutost krivulje u točki dodira segmenata
* geometrijski kontinuitet G1 - jednakost smjera vektora tangente u točki dodira segmenata.

Parametarski kontinuitet definiran je na sljedeći način:

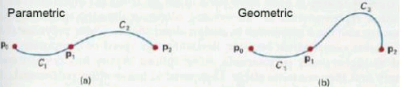
* parametarski kontinuitet C0 – segmenti su spojeni
* parametarski kontinuitet C1 – u točki dodira segmenata, prve derivacije po parametru t su jednake
* parametarski kontinuitet Cn - jednakost n-te derivacije Q(t) u točki dodira segmenata.





1. C0 : točke x, y, z u t2 prve krivulje jednake su točkama x, y, z u t1 druge
2. C1 : prva derivacija u točki spajanja je jednaka
3. C2 : i prve i druge parametarske derivacije oba segementa su jednake u točki spajanja

1. G0 : isti kao i parametarski C0
2. G1 : parametarske prve derivacije su proporcionalne na presjeku dva segmenta
3. G2 : i prva i druga parametarska derivacija dva segmenta su proporcionalne na krajnjim točkama



Općenito, parametarski kontinuitet C1 implicira geometrijski kontinuitet G1, ali obrat ne vrijedi. Kontinuitet G1 manje restriktivan od C1. Spojne točke s kontinuitetom G1 izgledati podjednako glatke kao i one s kontinuitetom C1.

Segment krivulje Q(t) se definira zadavanjem:

* krajnjih točaka
* vektora tangenti
* kontinuiteta između segmenata krivulje

Svaka parametarska krivulja trećeg reda ima četiri koeficijenta, pa su potrebna četiri uvjeta za postavljanje četiri jednadžbe sa četiri nepoznanice i njihovo rješavanje.

Tri glavne vrste krivulja koje će biti opisane su:

* **Hermiteove krivulje**: definirane sa dvije krajnje točke i dva vektora smjera u krajnjim točkama,
* **Bezierove krivulje**: definirane sa dvije krajnje točke i dvije dodatne točke koje određuju vektore smjera u krajnjim točkama,
* **B-krivulje** (B-splineovi) i **β-krivulje** (β-splineovi): definirane sa četiri kontrolne točke.

**HERMITEOVE KRIVULJE**

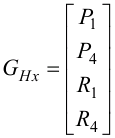
Hermiteove krivulje definiramo pomoću: - dvije krajnje točke (P1 i P4) i - dva vektora smjera u krajnjim točkama (R1 i R4). Prisjetimo se općeg zapisa segmenta:

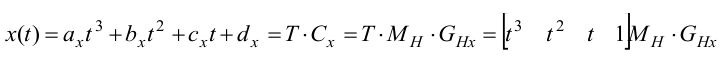
 

M - bazna matrica veličine 4 x 4,

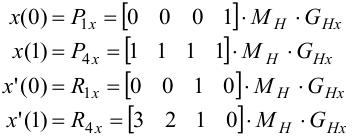
G - geometrijski vektor (vektor s četiri elementa koji predstavljaju geometrijska ograničenja)

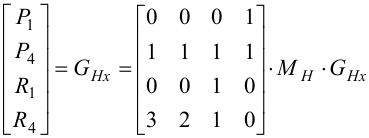
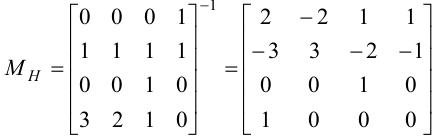
Hermiteovu baznu matricu MH, koja se množi s Hermiteovim geometrijskim vektorom GH, izračunavamo postavljanjem četiri jednadžbe (po jedna jednadžba za svako ograničenje) za četiri nepoznata koeficijenta polinoma.

Dakle, x komponenta Hermiteova geometrijske matrice ima oblik:

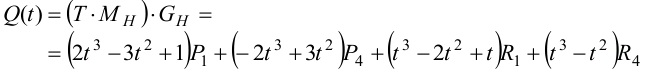


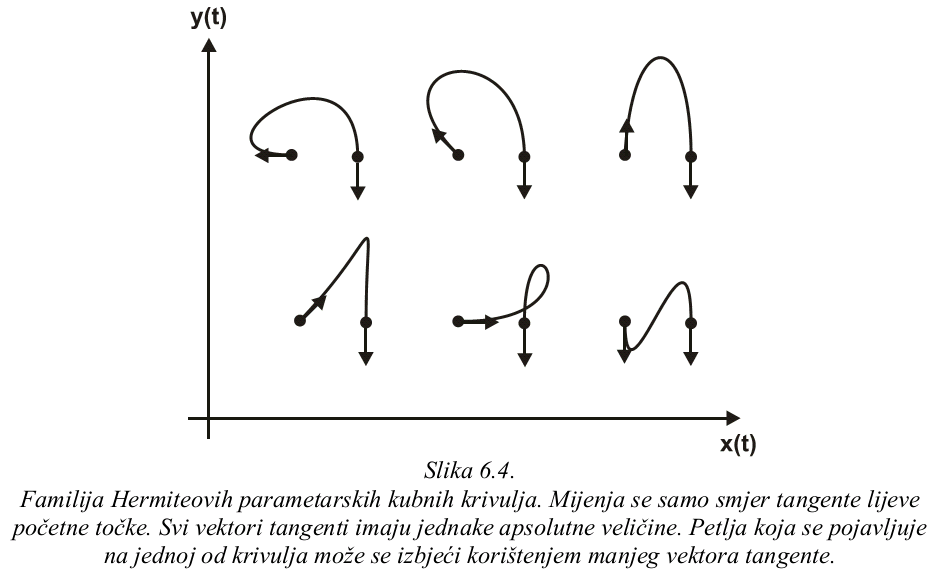
odakle možemo iskazati izraze za ograničenja P1, P4, R1 i R4:



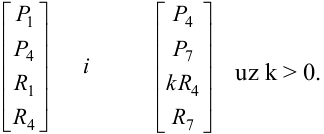
 

Izraz iz kojeg možemo izračunati utjecaj svakog elementa geometrijskog vektora za neku vrijednost t:





Za dvije Hermiteove parametarske kubne krivulje koje imaju zajedničku krajnju točku s geometrijskim kontinuitetom G1, geometrijski vektori moraju biti oblika

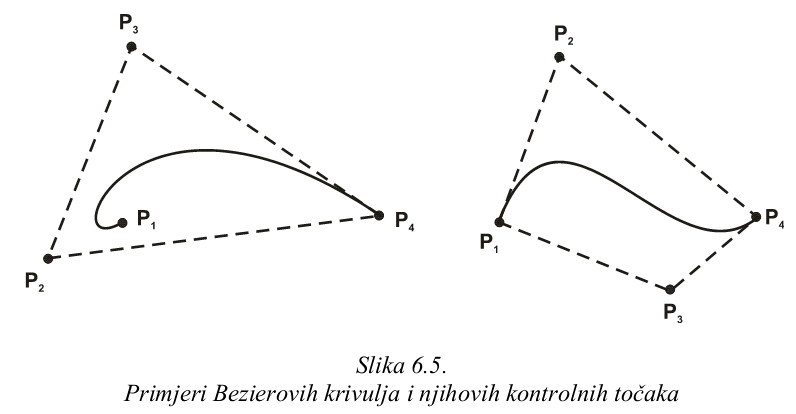


**BEZIEROVE KRIVULJE**

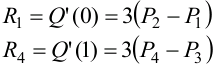
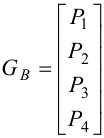
Najčešće jednostavne krivulje trećeg reda. Razvio Pierre Bézier u 70-tim za CAD/CAM. Za crtanje ga koristi Adobe PostScript, Adobe Illustrator, Macromedia Freehand, Fontographer,... Koristi se i za animaciju.

Bezierove krivulje su definiramo pomoću:

* dvije krajnje točke i
* dvije dodatne točke koje nisu na krivulji (određuju vektore smjera u krajnjim točkama)



Pomoću dviju kontrolnih točaka posredno su definirani vektori smjera tangenti R1 i R4 u dvjema krajnjim točkama. Vektor smjera tangente u početnoj točci odgovara derivaciji krivulje Q(t) za vrijednost parametra t = 0, a vektor smjera tangente u krajnjoj točci odgovara derivaciji krivulje Q(t) za vrijednost parametra t = 1:



Geometrijski vektor GB za Bezierove krivulje definiran je s četiri točke

Matrica MHB definira relaciju između Hermiteovog geometrijskog vektora GH i Bezierovog geometrijskog vektora GB: 

Jednadžba odsječka Bezierove krivulje: 

Uvjet za G1 kontinuitet jest da točke P3, P4 i P5 moraju biti različite i kolinearne:

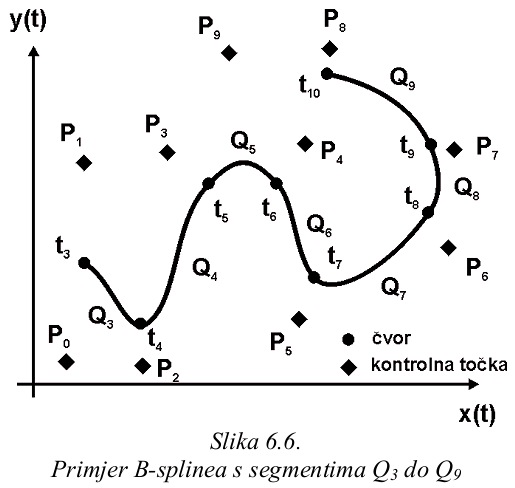
P3 - P4 = k (P4 - P5), k > 0

Ukoliko je zadovoljen i uvjet da je k = 1, tada je postignut i kontinuitet C1.

**B-SPLINE KRIVULJE**

Pojam:

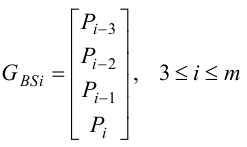
«Spline» (krivulje) potiče od naziva fleksibilnih metalnih traka koje su se koristile pri izradi površina aviona, auta i brodova. Utezi postavljeni na splineove korišteni su za oblikovanje krivulja. Općenito, metalni splineovi imaju kontinuitet drugog reda.

Matematički oblik ovih krivulja jest prirodni kubni spline. To je neprekidni polinom trećeg reda s kontinuitetom C0, C1 i C2 koji prolazi kontrolnim točkama. Splineovi su općenito glatkiji od Hermiteovih i Bezierovih krivulja.

B-krivulje se sastoje od segmenata krivulje čiji su koeficijenti polinoma ovisni o nekoliko kontrolnih točaka. Pomicanje kontrolne točke djeluje samo na mali dio krivulje. Za razliku od prirodnih splineova, B-krivulje (splineovi) ne prolaze kontrolnim točkama, ali imaju isti kontinuitet. Kubnu B-krivulju možemo opisati pomoću niza od m+1 kontrolnih točaka P0, P1,…, Pm, i sastoji se od m-2 segmenata (parametarskih polinoma trećeg reda) Q3, Q4,…, Qm.

Iako bi i ove segmente mogli definirati s parametrom t iz intervala [0,1] za svaki pojedini segment, parametar ćemo prilagoditi (supstitucija t = t + k) tako da njegova domena varira ovisno o rednom broju segmenta.

Svaki od m-1 segmenata krivulje definiran je s četiri od ukupno m+1 kontrolnih točaka. Na primjer, segment Qi definiran je s točkama Pi-3, Pi-2, Pi-1 i Pi pa geometrijski vektor GBs B-splinea za segment Qi ima oblik:



Kao što je svaki segment definiran s četiri kontrolne točke, svaka kontrolna točka (osim onih na početku i kraju sekvence P0, P1,…, Pm) djeluje na četiri segmenta krivulje. Pomicanje kontrolne točke u nekom smjeru pomiče četiri segmenta na koje djeluje u istom smjeru, dok na ostalim segmentima nema promjene.

**NAČINI CRTANJA PARAMETARSKIH KRIVULJA**

Postupak crtanja parametarskih krivulja uključuje:

* određivanje diskretnog skupa točaka na modelu krivulje i
* iscrtavanje ravnih crta koje ih povezuju.

Mogući pristupi crtanja parametarskih krivulja:

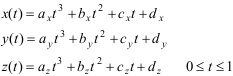
* *iterativni proračun* x(t), y(t) i z(t) za niz bliskih rastućih vrijednosti parametra t,
* *rekurzivna podjela* do zadovoljavajuće bliskosti kontrolnih točaka samoj krivulji,
* *hibridni pristup* kao kombinacija iterativnog i rekurzivnog načina.

Iterativni proračun odvija se na način da se vrijednosti koordinanata x(t), y(t), i z(t) pojedinih točaka izračunavaju za niz vrijednosti parametra t međusobno udaljenih za unaprijed određeni konstantan iznos d. Proračunate točke spajaju se ravnim crtama. *Problem*: unaprijed treba odrediti razmak točaka s obzirom na parametar t. Prevelik razmak -> slaba kvaliteta aproksimacije. Premalen razmak -> nepotrebno proračunsko opterećenje.

*Rekurzivna podjela* odvija se na način da se između dvije izračunate točke umeće treća točka. Rekurzivna podjela zaustavlja se adaptivno kada odsječak krivulje postane dovoljno ravan da se može aproksimirati ravnom crtom. Pojedinosti postupka različite su za pojedine vrste krivulja. Ovaj pristup je posebice prikladan za Bezierove krivulje.

*Hibridni pristup* kombinira najbolja svojstva iterativnog i rekurzivnog pristupa. U osnovi se može opisati kao iterativni proračun s adaptivnim korakom.

Primjer:



Prikaz parametarskih krivulja je jednostavan. Evaluacijom izraza u n uzastopnih vrijednosti od t (uz neki korak d) dobiju se točke zadane krivulje. Kod za prikaz parametarskih krivulja trećeg reda može se napisati u sljedećem obliku:

typedef double CoefficientArray[4]

void CrtajKrivulju (

CoefficientArray cx, /\* koeficijenti za x(t): Cx = MGx \*/

CoefficientArray cy, /\* koeficijenti za y(t): Cy = MGy \*/

CoefficientArray cz, /\* koeficijenti za z(t): Cz = MGz \*/

int n) /\* broj koraka \*/

{

int i;

double d = 1.0 / n;

double t = 0;

MoveAbs3 (cx[3], cy[3], cz[3]); /\* t = 0: pocetak u x(0), y(0), z(0) \*/

for (i = 0; i < n; i++) {

double t2, t3, x, y, z;

t += d;

t2 = t \* t;

t3 = t2 \* t;

x = cx[0] \* t3 + cx[1] \* t2 + cx[2] \* t + cx[3];

y = cy[0] \* t3 + cy[1] \* t2 + cy[2] \* t + cy[3];

z = cz[0] \* t3 + cz[1] \* t2 + cz[2] \* t + cz[3];

DrawAbs3 (x, y, z);

}

} /\* CrtajKrivulju \*/

**POVRŠINE**

Najčešće korišteni načini prikaza površina u 3D prostoru temelje se na primjeni:

* **Parametarskih površina**: generalizacija parametarskih krivulja na način da se umjesto jednog parametra koriste dva.
* **Mreža mnogokuta**: skup povezanih mnogokutnih ravnih površina kojima se aproksimira željena površina proizvoljnog oblika.
* **Površina drugog reda**: površine definirane općom jednadžbom drugog reda u implicitnom obliku.

**DVOPARAMETARSKE POVRŠINE TREĆEG REDA**

Dvoparametarske površine trećeg reda predstavljaju generalizaciju parametarskih krivulja trećeg reda. U jednadžbi parametarske krivulje parametar t zamijenimo parametrom s.



**BEZIEROVE POVRŠINE**

Bezierova geometrijska matrica se sastoji od 16 kontrolnih točaka.



Bezierove površine su prikladne za interaktivno crtanje iz istog razloga kao što su zanimljive i Bezierove krivulje (jer su vektori smjera tangenti u krajnjim točkama eksplicitno izraženi). Uvjet za C0 i G0 kontinuitet na spojevima dijelova površina je jednakost četiriju rubnih kontrolnih točaka. G1 kontinuitet nastupa kada su dva skupa od po četiri kontrolne točke sa svake strane ruba kolinearne s točkama ruba.

**B-SPLINE POVRŠINE**

Kontinuitet C2 na rubovima je kod B-sjplineova automatski - nisu potrebna nikakva posebna podešavanja kontrolnih točaka. Potrebno je izbjeći njihovo dupliranje.

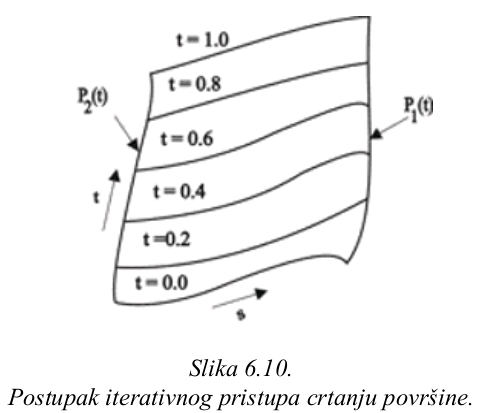
**NAČINI CRTANJA DVOPARAMETARSKIH POVRŠINA TREĆEG REDA**

Kao i u slučaju crtanja parametarskih krivulja, osnovni pristupi crtanju parametarskih površina uključuju:

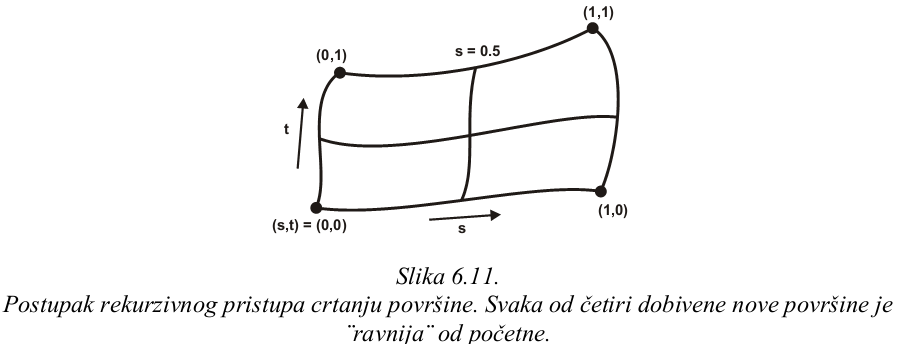
* iterativni postupak
* rekurzivni postupak

*Iterativni postupak* podrazumijeva proračun polinoma trećeg reda za niz bliskih rastućih vrijednosti parametara t i s. Mogući pristup:

* iscrtavanje niza krivulja kao funkcija parametra t (dok se vrijednost parametra s drži konstantnom za pojedinu krivulju, a mijenja se od krivulje do krivulje).
* iscrtava se niz krivulja kao funkcija parametra s (dok se vrijednost parametra t drži konstantnom za pojedinu krivulju, a mijenja se od krivulje do krivulje).



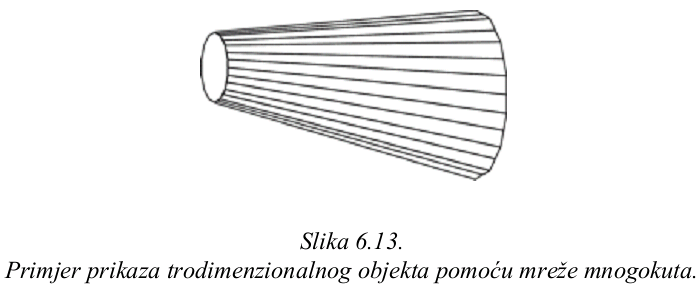
Postupak rekurzivne podjele podrazumijeva dijeljenje površine na četverokute (omeđene krivuljama) do zadovoljavajuće ravnoće rezultirajućih četverokuta. Podjela se obavlja dijeljenjem prvo po jednom, a zatim po drugom parametru. Dijeljenje se obavlja crtanjem krivulje po jednom parametru uz konstantnu vrijednost drugog parametra. Konstantna vrijednost drugog parametra određuje se kao aritmetička sredina između vrijednosti tog parametra koje su korištene u prethodnoj podjeli. Ovaj postupak ilustriran je na slici 6.11.



**MREŽE MNOGOKUTA**

Mreža mnogokuta: skup bridova, vrhova i mnogokuta povezanih tako da oblikuju površinu koja dovoljno dobro aproksimira željejnu površinu.

* Svaki brid zajednički je za najviše dva mnogokuta, a svaki vrh zajednički je za najmanje dva brida.
* Svaki brid povezuje dva vrha.
* Mnogokut je zatvoreni skup povezanih bridova (stranica).



Najčešće korišteni načini prikazivanja modela mreže mnogokuta su:

* eksplicitni način,
* struktura pokazivača na listu vrhova,
* struktura pokazivača na listu bridova.

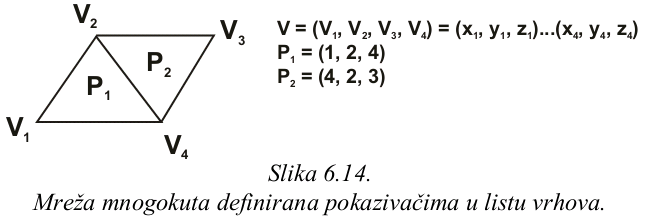
U eksplicitnom prikazu svaki je mnogokut predstavljen listom koordinata vrhova:

P = ((x1, y1, z1), (x2, y2, z2), ...,(xn, yn, zn)).

Nedostatak: vrhovi se pojavljuju više puta jer ne postoji eksplicitna predstava zajedničkih vrhova.

U prikazu temeljenom na strukturi pokazivača na listu vrhova svaki vrh se pohranjuje samo jednom u listu vrhova V = ((x1, y1, z1), (x2, y2, z2), ...,(xn, yn, zn)).

Mnogokut se definira listom pokazivača (indeksa) u listu vrhova.



Memorijski zahtjevi su znatno manji nego u eksplicitnom prikazu. Promjena koordinata jednog vrha je jednostavna jer nije potrebno uspoređivanje i pretraživanje. Međutim, u ovom prikazu relativno je teško odrediti mnogokute koji imaju zajednički vrh, a i zajednički bridovi se iscrtavaju dva puta.

Navedeni problemi otklonjeni su u prikazu temeljenom na strukturi pokazivača u listu bridova. U ovom prikazu svaki se brid unosi jednom P = (E1, E2, ..., E3) i to na sljedeći način: E = (V1, V2, P1, P2) gdje su prva dva elementa koordinate vrhova koji određuju brid, a druga dva elementa oznake mnogokuta kojima je taj brid zajednički.



**POVRŠINE DRUGOG REDA**

Površine drugog reda kao što su površina kugle, površina elipsoida ili cilindra mogu poslužiti kao elementarne površine za definiranje modela složenijih površina u trodimenzionalnom prostoru. Ova skupina površina može se definirati implicitnom jednadžbom slijedećeg oblika:

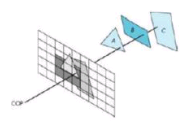
f(x,y,z) = ax2 + by2 + cz2 + 2dxy + 2eyz + 2fxz + 2gx + 2hy + 2jz + k = 0.

Vrsta površine određena je vrijednostima parametara a, b, c, d, e, f, g, h, i, j, k.

Prednosti ovog pristupa su:

* prikladnost za niz proračuna kao što su normale na površinu u pojedinoj točci;
* određivanje presjeka;
* određivanje pripadnosti točke površini.

Ove prednosti temelje se na jasnoj matematičkoj formi definicije površine.

**ODREĐIVANJE VIDLJIVIH POVRŠINA**

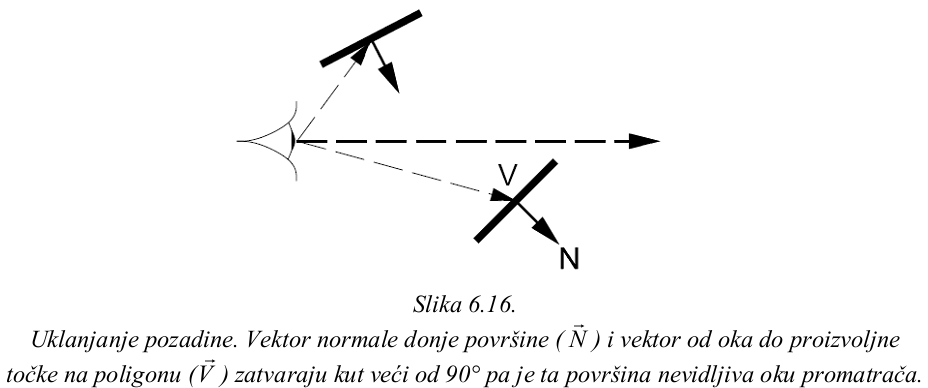
Ukoliko imamo zadanu grupu 3D objekata, potrebno je odrediti koji su rubovi odnosno površine vidljivi iz motrišta (za perspektivne projekcije) ili s obzirom na smjer projekcije (za paralelne projekcije).

Na taj način mogu se prikazati samo vidljive linije i površine. Ovaj postupak nazivamo određivanje vidljivih linija i površina odnosno eliminiranje skrivenih linija i površina.

**ODBACIVANJE POZADINE**

Postupak odbacivanja pozadine (engl. backface culling) tj. eliminiranja svih poligona koji su okrenuti suprotno od oka promatrača. Pozadinski poligon je onaj čija je normala usmjerena od promatrača.

Ovaj postupak zahtjeva određvanje normale poligona, vektora od motrišta do poligona, te kuta koji zatvaraju vektori i . Ukoliko su poligoni dvostrani tj. nisu zatvoreni u volumen, metoda nije primjenjiva. Odbacuje minimalno 50% poligona i može se lako sklopovski integrirati.



**Z-BUFFER ALGORITAM**

Z-Buffer algoritam (ili algoritam dubinskog međuspremnika): jedna od najčešće korištenih rutina. Za implementaciju Z-buffer algoritma potrebno je osim međuspremnika okvira (frame buffer) F u koji pohranjujemo informaciju o boji određenog piksela, imati i z-međuspremnik (z-buffer) Z u kome se pohranjuje z vrijednost svakog piksela.

Postupak:

1. Z-međuspremnik se inicijalizira u nulu (ili neku drugu vrijednost koja odgovara maksimalnoj dubini)
2. Međuspremnik okvira se postavlja u boju pozadine.
3. Najveća vrijednost koju možemo pohraniti u Z međuspremnik predstavlja z vrijednost prednje ravnine odsijecanja.
4. Poligoni se rasteriziraju u međuspremnik okvira po proizvoljnom redoslijedu.
5. Ako poligon kojeg rasteriziramo u točci (x,y) nije udaljeniji od točke čija je dubina i boja trenutno pohranjena u međuspremnicima, nova boja i dubina se upisuju preko starih vrijednosti.

Cijeli prikazani postupak svodi se na pretraživanje svih vezanih parova {Zi(x,y), Fi(x,y)} za određeni x i y kako bi se pronašla najveća vrijednost Zi. Međuspremnik okvira i z međuspremnik pohranjuju informaciju vezanu uz najveći z koji je pronađen za svaki par (x,y). Svaki poligon se rasterizira liniju po liniju i vrijednosti se pohranjuju u međuspremnike.

Z-buffer algoritam djeluje i za objekte koji nisu poligoni, odnosno može se primijeniti za bilo koji objekt kojem možemo odrediti osvjetljenje i z vrijednost u svakoj točki projekcije.

Prednosti:

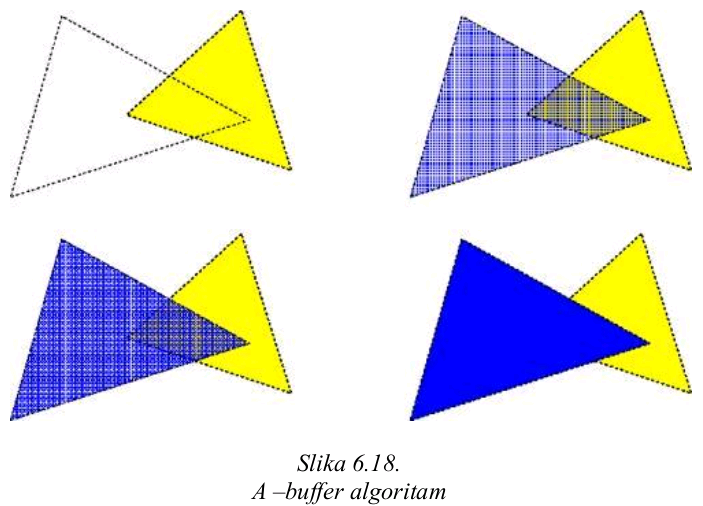
* jednostavan
* lak za primjenu u sklopovlju
* poligoni se mogu obrađivati u proizvoljnom poretku

Nedostatci:

* zahtijeva dodatnu memoriju
* zahtijeva brzu memoriju
* teško provođenje analitičkog antialiasinga
* dijeljeni rubovi se obrađuju nekonzistentno (ovisi o redoslijedu)
* teško simuliranje prozirnih poligona (odbacujemo poligone iza najbližeg)

**A-BUFFER ALGORITAM**

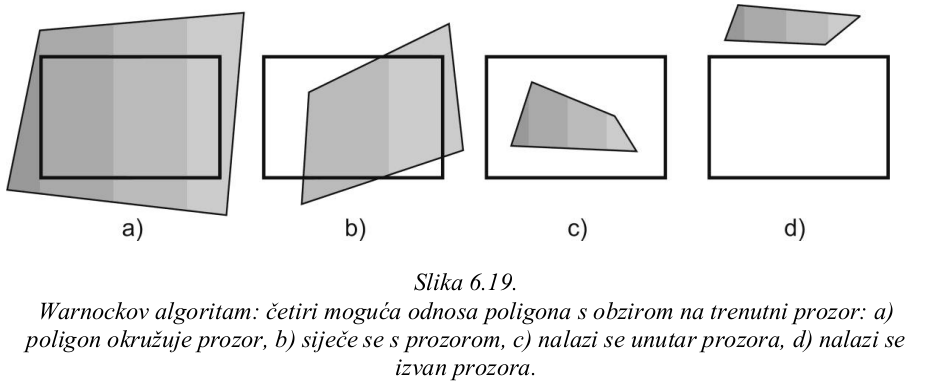
Služi za rješavanje problema aliasinga. Proširenje Z – buffer algoritma: A – buffer algoritam koji koristi diskretnu aproksimaciju netežinskog uzorkovanja područja (unweighted area sampling). Algoritam je razvio Lucasfilm. Z-Buffer algoritam radi samo za neprozirne poligone. Zbog toga akumuliramo vrijednosti prozirnih objekata, a svaki pojedinačno težinski doprinosi konačnoj slici.



Kad nacrtamo piksel, ukoliko crtamo proziran objekt, pohranjujemo z vrijednost, boju, i postotak prozirnosti t. Ako nakon toga iscrtamo neki neprozirni objekt ispred, stari objekt (piksel) je zaklonjen pa informaciju odbacujemo. Ako iscrtamo objekt u pozadini, pohranjujemo tu boju s postotkom 100-t, jer je toliki udio u boji piksela.

**WARNOCKOV ALGORITAM**

Ovaj algoritam za uklanjanje skrivenih površina pada u skupinu tzv. ¨podijeli-pa-vladaj¨ algoritama. Algoritam klasificira poligone s obzirom na trenutni prozor u trivijalne i ne-trivijalne slučajeve. Trivijalne slučajeve je lako obraditi dok se za ne-trivijalne slučajeve trenutni prozor promatranja rekurzivno dijeli u četiri jednaka podprozora. Svaki dobiveni podprozor se nadalje koristi za reklasifikaciju preostalih poligona. Ova rekurzivna procedura se ponavlja dokle se svi poligoni ne klasificiraju trivijalno ili dokle trenutni prozor ne dosegne rezoluciju slikovnih elemenata zaslona. U tom trenutku, algoritam se vraća na jednostavni z-buffer algoritam sortiranja poligona koji se presjecaju i boja pojedinog slikovnog elementa postaje jednaka boji poligona najbližeg promatraču.



Opisani algoritam čuva koherentnost područja tj. svi poligoni koji su klasificirani kao okružujući ili vanjski ostaju tako klasificirani i za sve podprozore dobivene rekurzijom. To svojstvo algoritma je i temelj njegove učinkovitosti. Uz samo deset rekurzivnih poziva algoritma, prozor veličine 1024 x 1024 slikovna elementa može se razbiti na nivo pojedinih piksela. Postoje i alternativne strategije podjele prozora koje ponešto kompliciraju algoritam, ali ga zato ubrzavaju.

**OSVJETLJENJE**

**ISCRTAVANJE (RENDERIRANJE)**

Postupak stvaranja realističnih slika iz zadanog opisa scene. Svjetlost tj. elektromagnetska energija u oko promatrača stiže nakon interakcije s fizičkom okolinom. U oku, dolazi do fizikalnih i kemijskih promjena na osnovu kojih dolazi do stvaranja električnih impulsa koje mozak interpretira.

**OKO**

Ljudsko oko je vrlo složen sustav. Aproksimacija: sfera koja ima promjer od oko 20 mm. Fleksibilna leća oka koristi se za fokusiranje primljeno svjetlo na područje u pozadini hemisfere oka – mrežnici. Mrežnica sadrži dvije vrste receptora: čunjiće i štapiće.

**Čunjići** (*Cones*):

* 6 do 7 milijuna čunjića su koncentrirani u središtu stražnje hemisfere oka i svaki je vezan sa po jednim živcem.
* osjetljivi samo na relativno visoke razine svjetlosti
* koriste se za rješavanje finih detalja.

**Štapići** (*rods*):

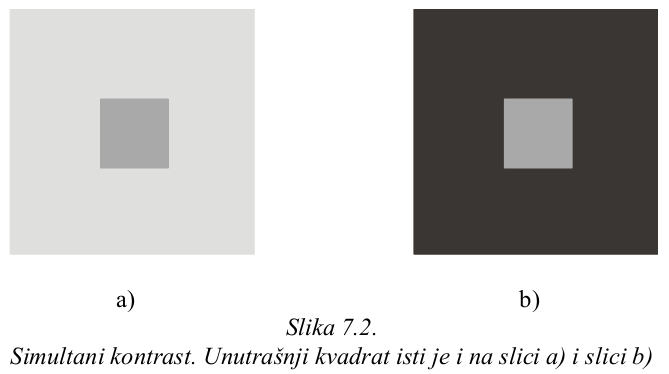
* Distribuirani su po retini i ima ih između 75 i 150 milijuna.
* Sa jednim živcem povezano je više štapića pa oni ne služe za percipiranje finih detalja.

Štapići su osjetljivi na vrlo niske razine svijetla. Samo čunjići služe za opažanje boja. Čunjići osjetljivi samo na veće razine osvjetljenja -> objekte koje gledamo pod slabim svijetlom, ne vidimo u boji (crvena ruža noću, u vrtu, izgleda crno ili tamno sivo).

Eksperimentalno je dokazano da je osjetljivost oka na svjetlost logaritamskog oblika. Oko ne može simultano reagirati na cijeli opseg svjetlosti već reagira u nekom relativnom opsegu svijetlosti koje je centrirano oko razine prilagodbe svijetlu. U tom opsegu svijetla kojem se oko prilagodi, ekstremne vrijednosti percipiramo kao bijelo ili crno.

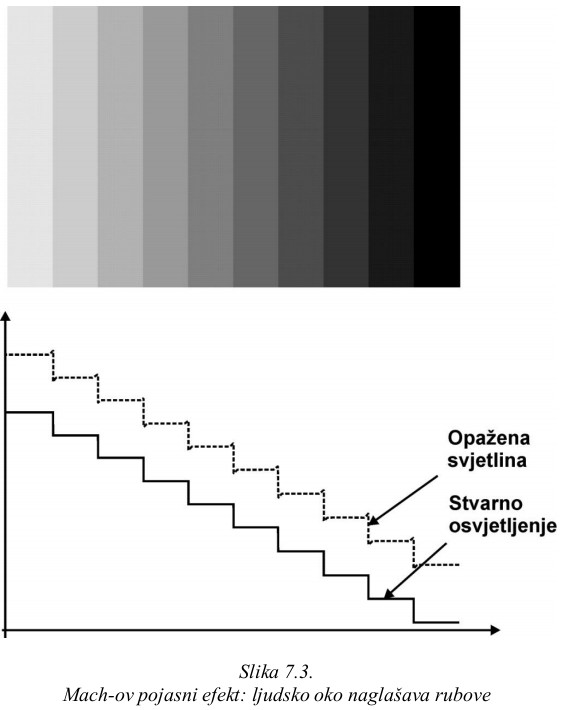
Simultani kontrast :

* Oko se prilagođava prosječnoj razini svijetlosti u sceni
* *Posljedica*: područje konstantne razine svijetlosti ili intenziteta koje je okruženo tamnim područjem čini nam se svjetlije nego to isto područje okruženo svijetlim područjem



*Mach-ov pojasni efekt*

Područja konstantnog intenziteta nam izgledaju kao područja promjenljivog intenziteta. Oko ima tendenciju prebačaja ili prelijevanja rubova područja konstantnog intenziteta. Mach-ov pojasni efekt nastupa u slučaju nagle promjene nagiba krivulje svjetlosnog intenziteta.



**MODELI OSVJETLJAVANJA**

Kada svjetlosna energija pada na neku površinu, ona može biti apsorbirana, reflektirana ili transmitirana. Neki objekt je vidljiv zbog postojanja reflektirane ili transmitirane svjetlosti. Ako je upadna zraka svjetlosti u cijelosti apsorbirana, objekt je nevidljiv i nazivamo ga crnim tijelom. Količina apsorbirane, reflektirane i transmitirane energije ovisi o valnoj duljini svjetlosti.

Ako se intenzitet upadne zrake svjetlosti reducira približno podjednako za sve valne duljine, objekt osvijetljen bijelom svjetlosti (sve valne duljine) je siv. Ako je gotovo sva svijetlost apsorbirana, objekt je crn. Ako je samo mali dio apsorbiran, objekt je bijel.Ako su neke valne duljine selektivno apsorbirane, reflektirano i/ili transmitirano svjetlo koje izlazi iz objekta ima drugačiju distribuciju energija, objekt je obojen.

Svojstva svjetlosti koja se reflektira od ili transmitira kroz površinu objekta ovisi o:

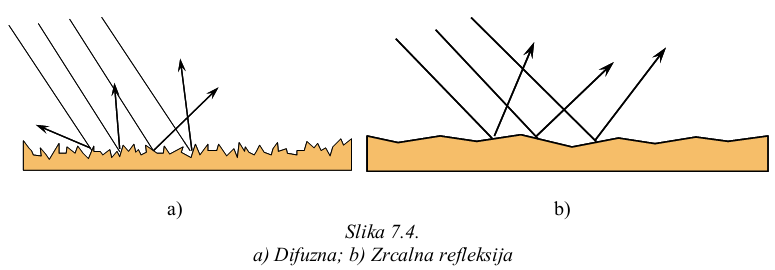
* sastavu,
* smjeru
* geometriji izvora svjetlosti,
* orijentaciji površine
* svojstvima površine objekta.

Reflektirana svjetlost može biti a) difuzna i b) zrcalna:

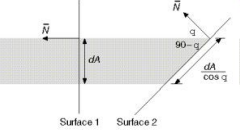
**Difuzno** reflektirana ili transmitirana svjetlost je ona svjetlost koja je, nakon što je ušla ispod površine nekog objekta, apsorbirana i potom reemitirana. Difuzno reflektirana zraka raspršena je u svim smjerovima podjednako pa je položaj promatrača nevažan.

**Zrcalno** reflektirana zraka reflektira se od vanjske površine objekta -> ne ulazi pod površinu -> nije

raspršena. Uska zraka svjetlosti koja pada na površinu i uska zraka svjetlosti se reflektira od površine. Svojstva reflektirane svjetlosti ostaju u osnovi nepromijenjena (tj. ako je dolazno svjetlo bijelo ili npr. crveno) reflektirano svjetlo je bijelo odnosno crveno, te je uglavnom neovisno o boji površine).



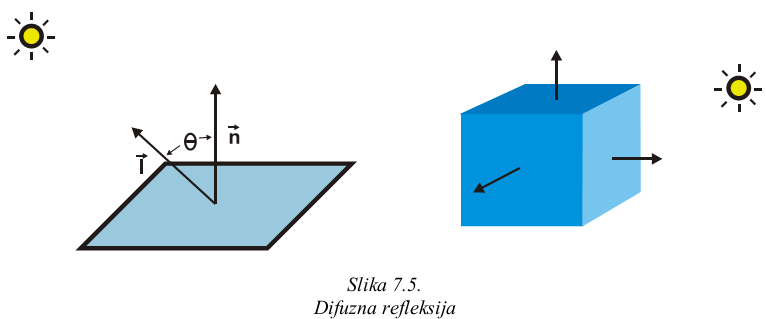
Materijali površina su općenito dielektrici (izolatori) i vodiči. Dielektrični materijali su prozirni, a vodljivi materijali su neprozirni. Prozirni (dielektrični) materijali imaju niski koeficijent refleksije. Koeficijent refleksije za neprozirne materijale je visok.

**JEDNOSTAVNI MODEL OSVJETLJAVANJA**

Po Lambertovom zakonu djelić neki površine dS rasvijetljen je proporcionalno kosinusu kuta pod kojim zraka svjetlosti upada na površinu. Po Lambertovom zakonu djelić neki površine dS rasvijetljen je proporcionalno kosinusu kuta pod kojim zraka svjetlosti upada na površinu:



reflektirani intenzitet = intenzitet upadne svjetlosti iz točkastog izvora \* koeficijent difuzne refleksije \* kosinus kuta između vektora prema izvoru svjetlosti i vektora normale na površinu



Objekti renderirani pomoću jednostavnog modela Lambertove difuzne refleksije imaju jednoličnu matiranu površinu. Objekti koji ne primaju svjetlost direktno od izvora izgledaju crni jer je pretpostavljen točkasti izvor svjetla koji je smješten u oku ili motrištu. U stvarnosti, objekti primaju raspršeno svjetlo iz okoline, na primjer zidova sobe. Ambijentno osvjetljenje - distribuirani izvor svjetla.

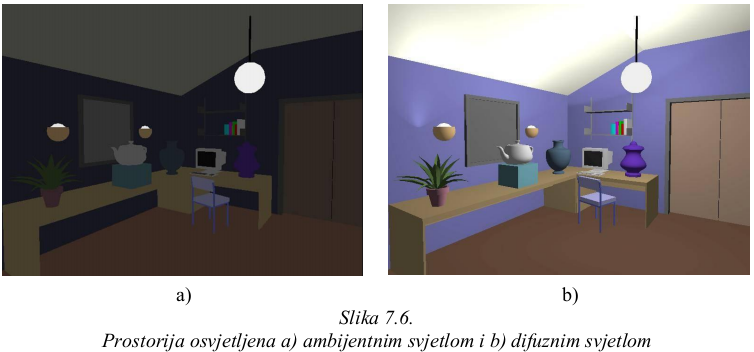
Zbog zahtjevnosti proračuna, služimo se aproksimacijom kojom pretpostavljamo da je ambijentno osvjetljenje konstantno u cijelom prostoru pa tu činjenicu koristimo linearnoj kombinaciji ambijentne i difuzne komponente. Jednostavni model osvjetljavanja u tom slučaju možemo opisati izrazom:



gdje je Ia(λ) intenzitet ambijentne svjetlosti, a ka(λ) koeficijent reflektirane ambijentne svjetlosti

(0 ≤ ka(λ) ≤1). Općenito ka i Ia se također u ovom jednostavnom modelu osvjetljenja smatraju konstantnim vrijednostima. Tada možemo izraz (7.2) napisati na sljedeći način:



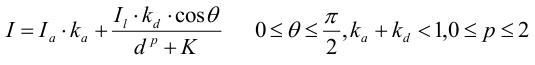


Određivanja intenziteta svjetlosti reflektirane od dva objekta iste orijentacije prema izvoru svjetla, a koja se nalaze na različitim udaljenostima od izvora Rezultat: jednaki intenzitet oba objekta. Ako se objekti preklapaju - nije ih moguće razlikovati.

Dobro je poznato da intenzitet svjetlosti opada recipročno kvadratu udaljenosti od izvora -> udaljeniji objekti izgledaju tamniji. Nažalost, ako se izvor svjetlosti nalazi u beskonačnosti -> udaljenost do objekta je također beskonačna -> nulti doprinos ovakvog osvjetljenja ukupnom intenzitetu.

Ako primijenimo perspektivnu transformaciju scene: udaljenost od perspektivnog žarišta do objekta, d, se koristi kao konstanta proporcionalnosti za difuzni izraz. Ako je perspektivno žarište blizu objektu, izraz jako varira pa se može dogoditi da objekti koji imaju gotovo istu udaljenost od žarišta (svjetla) imaju preveliku razliku u osvjetljenju.

Preveliku razliku u osvjetljenju korigiramo uvođenjem iskustvene konstante K i atenuacijom izraza za kvadrat udaljenosti korištenjem nekog drugog, manjeg eksponenta p:



Kada je žarište u beskonačnosti, udaljenost d se često određuje s obzirom na objekt koji mu je najbliži. Na ovaj način se objekt najbliži žarištu osvjetljava s punim intenzitetom izvora svjetlosti, a ostali, udaljeniji objekti s manjim intenzitetom. Ako je površina u boji, model osvjetljenja se primjenjuje na svaku od tri primarne boje pojedinačno.

**ZRCALNA REFLEKSIJA (JEDNOSTAVNI MODEL)**

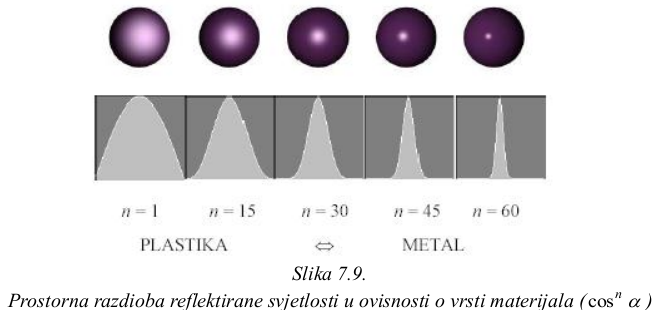
Intenzitet zrcalno reflektiranog svjetla ovisi o kutu upadne zrake, njenoj valnoj duljini i svojstvima materijala. Zrcalna refleksija svjetla je usmjerena. Savršeno reflektirajuća površina (zrcalo) ima kut

refleksije jednak kutu upada. Zbog toga, samo promatrač koji se nalazi na točno tom kutu (refleksije) može vidjeti reflektiranu zraku svjetlosti.

Ako se radi o nesavršenoj reflektirajućoj površini, količina svjetlosti koja dopire do promatrača ovisi o prostornoj razdiobi zrcalno reflektiranog svjetla. Za glatke površine, prostorna razdioba je uska ili fokusirana, dok je razdioba za hrapave površine široka. Blještavilo (engl. highlights) sjajnog objekta nastaje zbog zrcalne refleksije. Zbog toga što je zrcalno reflektirano svjetlo fokusirano duž vektora refleksije, blještavilo se pomiče s promatračem. Reflektirano svjetlo ima svojstva upadnog svjetla jer se svjetlost odbija od vanjskog dijela površine (osim za neke metale i boje). Jednadžba zrcalne komponente svjetlosti može se napisati na sljedeći način:



gdje je ω(i,λ), krivulja refleksije koja predstavlja omjer zrcalno reflektirane svjetlosti i upadne svjetlosti u funkciji upadnog kuta, i, i valne duljine λ. Potencija n aproksimira prostornu razdiobu zrcalno reflektirane slike.





Zrcalna refleksija je usmjerena tj. ovisi o kutu upada svjetlosti. Svjetlo koje pogađa površinu pod pravim kutom zrcalno reflektira samo određeni postotak svjetlosti, a ostatak se apsorbira ili difuzno reflektira. Postotci ovise o svojstvima materijala i valnim duljinama. Za neke nemetale refleksija pada i do 4% dok za neke metale raste do 80%.

**PHONG-OV MODEL (JEDNOSTAVNI MODEL)**

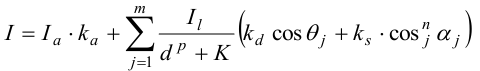
Kombiniranjem rezultata zrcalne refleksije i izraza za ambijentno i difuzno svjetlo dobije se tzv. *Phong-ov model osvjetljenja*:



Jednostavni model osvjetljenja koji zanemaruje promjenu valne duljine može se izraziti sljedećom jednadžbom:



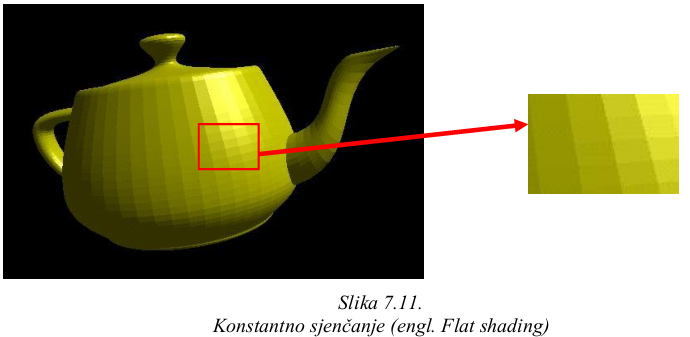
U slučaju više izvora svjetlosti, efekti se linearno zbrajaju. Tada jednostavni model osvjetljenja postaje:



gdje je m broj izvora svjetlosti a konstanta ks koeficijent zrcalne refleksije.

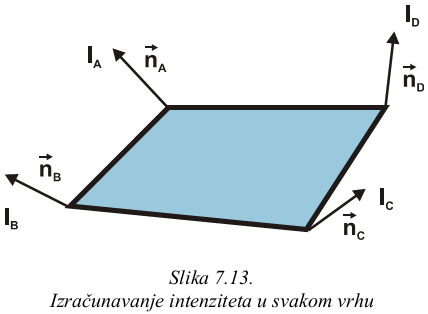
**GOURAUD SJENČANJE**

Ako na neku poligonalnu površinu primijenimo model osvjetljenja s po jednom jediničnom normalom za svaku površinu unutar jednog poligona, rezultat takvog sjenčanja će biti kao na slici

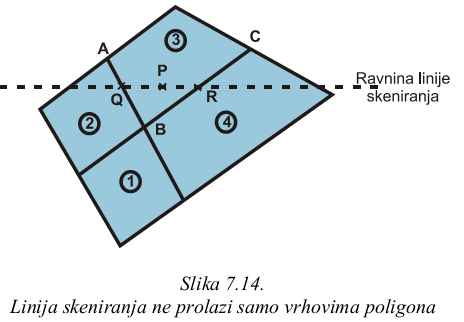


Bolji prelazi se mogu postići korištenjem tehnike koju je razvio Gouraud. Ako koristimo algoritam koji skenira liniju po liniju za renderiranje nekog objekta, vrijednost intenziteta u svakoj točci duž linije skeniranja mora se odrediti na osnovu modela osvjetljavanja. Normale na površinu aproksimiraju se na vrhovima poligona površine.





Nakon toga se pomoću bilinearne interpolacije određuje intenzitet za svaki piksel na liniji skeniranja:



Intenzitet u točci P određen je na sljedeći način: Prvo linearnom interpolacijom intenziteta na vrhovima poligona A i B dobijemo intenzitet u točci Q tj.



gdje je u = AQ/AB. Na sličan način intenzitet u točci R koja se nalazi na liniji skeniranja računamo iz jednadžbe



gdje je v = BR/ BC.

Na kraju, intenzitet točke P na liniji skeniranja dobije se linearnom interpolacijom duž linije skeniranja između točaka Q i R tj.



gdje je t = QP/QR.



Nedostatak Gouraud-ovog sjenčanja je što ne uklanja u potpunosti Mach-ov pojasni efekt. Ova pojava nastaje jer algoritam osigurava kontinuitet intenziteta duž rubova poligona ali ne i kontinuitet promjene intenziteta. Također, silueta objekta ostaje poligonalna jer je temeljna struktura podataka poligonalna.

**PHONG SJENČANJE**

Računski zahtjevnije od Gouraud sjenčanja ali Phong-ovo sjenčanje daje bolje rezultate.

**Gouraud**: interpolira vrijednosti intenziteta duž linije skaniranja.

**Phong**: interpolira vektore normala duž linije skaniranja.

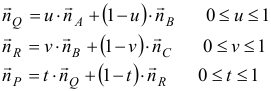
Model osvjetljenja se potom primjenjuje za svaki piksel, korištenjem interpoliranih normala za izračunavanje intenziteta.

Ova tehnika daje bolju lokalnu aproksimaciju zakrivljenosti površine, pa stoga i bolje renderiranje površine. Naročito je poboljšana realističnost zrcalnog sjaja.

Phong-ovo sjenčanje prvo aproksimira zakrivljenost površine u vrhovima poligona aproksimacijom

normala u vrhovima. Nakon toga primjenjujemo bilinearnu aproksimaciju za određivanje normale u svakom pikselu.

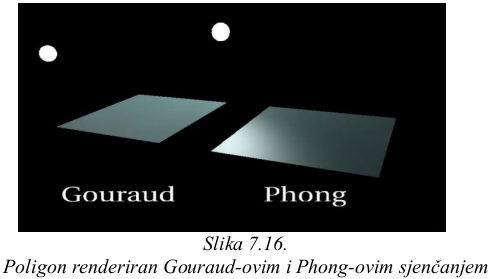
Promotrimo još jednom primjer sa slike 7.14. Normala u točci P određuje se linearnom interpolacijom između A i B da bi dobili Q, između B i C da dobijemo R i napokon između Q i R da bi dobili P. Dakle:



gdje su u = AQ/AB, v = BR/BC, t = QP/QR. I u ovom slučaju možemo upotrijebiti inkrementalnu metodu za proračun normale duž linije skaniranja:



gdje indeksi 1 i 2 označavaju uzastopne piksele duž linije skeniranja.



Iako Phong-ovo sjenčanje reducira najveći broj problema koje povezujemo s Gouraud-ovim sjenčanjem, i dalje primjenjuje linearnu interpolaciju. Zbog toga i dalje ostaje problem diskontinuiteta prve derivacije intenziteta koji dovodi do Mach-ovog pojasnog efekta. Ipak, općenito uzevši, ovi efekti su znatno manji nego kod Gouraud-ove tehnike.

**TEKSTURE**

Tekstura predstavlja fizičku osobinu nekog objekta i odnosi se prvenstveno na veličinu, oblik i raspored njegovih dijelova. Postupak kojim nekom objektu dodajemo razna svojstva (boja, prozirnost, hrapavost, osvjetljenje,...) naziva se teksturiranje.Teksture i koristimo s ciljem redukcije velikog broja poligona potrebnih za prikaz detalja nekog objekta.

Teksture su bitmape tj. slike koje mogu biti ručno crtane, skenirane ili računalno generirane. Proces dodavanja tekstura na neki objekt ili neki njegov dio naziva se teksturiranje (engl.Texture mapping). Osnovna ideja je jednom teksturom zamijeniti stotine ili pak tisuće poligona, jednostavnim lijepljenjem slike odnosno teksture na poligon.



U uvodu smo koristili izraz ''zaljepiti'', no upravo ta riječ jest osnovni problem kojeg je potrebno razmotriti, dakle treba odredit način na koji se neku teksturu može zalijepiti na odabrani objekt. Opisat ćemo tri različita pogleda na teksture i njihovo ''ljepljenje''. Krenimo redom:

* **Prvi** aspekt jest dodavanje odvojene teksture na glatku površinu, tzv. funkcija mapiranja. Nakon dodavanja površina je još glatka.
* **Drugi** aspekt jest dodavanje ''privida'' hrapavosti na površinu, tzv. funkcija preturbiranja.
* **Treći** aspekt govori o simulaciji okoline, izvođenju sjena i svjetla pomoću tekstura.

**MAPIRANJE**

Zajednički naziv za bilo kakvo ''ljepljenje'' tekstura na bilo kakve površine se naziva mapiranje. Problem mapiranja uzorka na glatku površinu je reduciran na problem transformacije iz jednog koordinatnog sustava u drugi. Razlikujemo dva različita transformiranja (parametriziranja) iz jednog u drugi koordinatni sustav:

* **parametrizacija površine** (engl. *surface* *parametrization*) jest transformacija iz prostora teksture u prostor objekta, a to bi značilo da sliku moramo toliko ''iskriviti'' da odgovara objektu na kojeg je želimo zalijepiti.
* **transformacija pogleda** (engl. *viewing* *transformation*) tj. postupak kojim objekt na kojem je mapirana tekstura transformiramo u nama potrebnu perspektivu (položaj)

Da bi uzorak teksture, definiran u koordinatnom sustavu teksture (u,v) i površinu objekta, definiranog u drugom koordinatnom sustavu (x,y,z), a predstavljenog u parametarskom prostoru (Θ, Φ), gdje je x(Θ, Φ), y(Θ, Φ), z(Θ, Φ), mapirali na tu površinu , moramo odrediti funkciju mapiranja između 2 prostora: prostor teksture -> parametarski prostor:



a inverzna funkcija tj. funkcija mapiranja koja transformira parametarski prostor u prostor teksture:



Funkcija mapiranja uglavnom je linearna:



gdje su vrijednosti A, B, C, D dobivene iz veza između poznatih točaka 2 koordinatna sustava. Nedostaci linearnog mapiranja su vidljivi već pri mapiranju neke teksture na sferu. U tom slučaju dešavaju se distorzije te trebamo primjeniti nelinearno mapiranje.

**ALGORITMI MAPIRANJA**

Kada je prostor slike tj. ''onoga što vidimo'' kao rezultat mapiranja neki rasterski prikazni uređaj (računalni monitor), tj. kada govorimo o mapiranju u računalnoj grafici, onda najčešće primjenjujemo dvije, u osnovi slične tehnike.

1. Catmullov algoritam oduzimanja zakrivljenih površina

Ovaj algoritam djeli površinu na poddjelove, sve dok dijeljeni dio ne pokriva centar jednog piksela. Nakon toga parametarske vrijednosti centra piksela (kutevi Θ i Φ), su mapirane u prostor teksture, a uzorak teksture je u tom trenutku iskorišten za određenje inteziteta piksela. No i ovaj način rješavanja dovodi do aliasing efekta. Zato Catmull uvodi djeljenje uzorka teksture zajedno s površinom.

2. Blin i Newellov algoritam

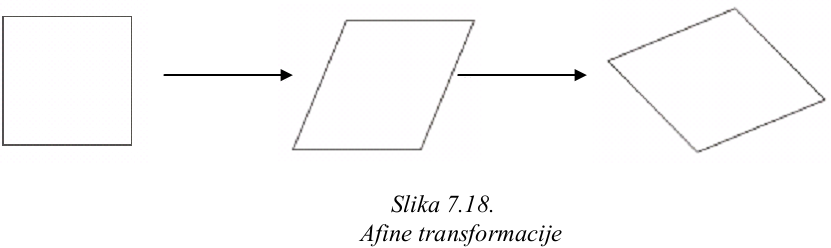
Isti kao i prethodno navedeni Catmullov algoritam samo sa boljim 2x2 piramidalnim, Bartlett antialiasing filterom. Prednost ovog algoritma je u tome što ne moramo znati inverzne transformacije iz prostora slike -> prostor objekta, ili dubinu poddjela (Z vrijednost) u prostoru slike. No jedan nedostatak je što se poddio može precizno prekriti 1 piksel u prostoru slike. Zaključno, precizno područje prekriveno pikselom u prostoru slike je transformirano u prostor teksture. Procedura je transformirati područje piksela iz prostora slike u prostor teksture.

Valja reći da je intezitet piksela u prostoru slike određen prosjekom inteziteta piksela u prostoru teksture.

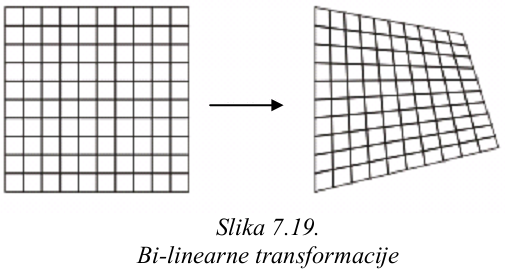
**FUNKCIJE MAPIRANJA**

Iako smo govorili samo o važnosti mapiranja, isto tako su važne karakteristike mapiranja ili transformacije. Tako dolazimo do afinih transformacija, čiji su specijalni slučaj linearne transformacije. Bi-linearne i dvo-dimenzionalne projekcijske transformacije su uglavnom upotrebljive u mapiranju tekstura.

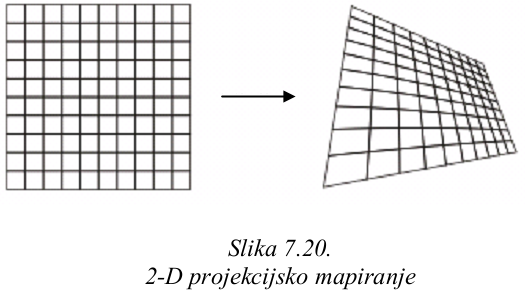
**Afine** transformacije mapiranja uključuju ili kombiniraju skaliranje, rotaciju, translaciju. One čuvaju paralelne linije i razmaknutost točke uzduž linija.



**Bi-linearne** transformacije mapiranja se upotrebljavaju u Gouraud i Phong sjenčenju. One mapiraju kvadrat na četverokut. Ovo mapiranje čuva vertikalne i horizontalne linije kao i razmaknutost točaka, ali ne i diagonalne linije. Kombinirano bi-linearne transformacije su bi-kvadratne. Inverzno bi-linearno mapiranje nije jednoznačno.



**2D projekcijsko mapiranje** ponekad i perspektivno mapiranje, koje zapravo mapira linije u beskonačnosti na stvarnu površinu. Ono čuva linije svih orijentacija, ali ne i razmaknutosti točaka.

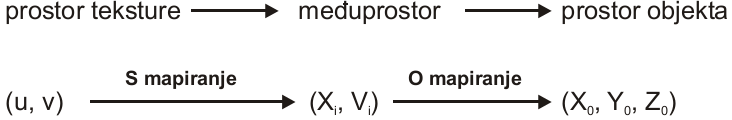


Heckbert i Moreton su kombinirali afine i perspektivne transformacije za dobivanje boljih rezultata, jer su izbjegli izračun nekih kompliciranih koeficijenata. No kod kombinacije ova dva mapiranja pozitivno je i to što su lako inverzna pa imamo sljedeće rješenje naših problema:



*Dvodjelno mapiranje tekstura*

Ova metoda služi za 3D površine koje 2D teksturu mapiraju na jednostavan 3D objekt. Dobra je i za prostorno mapiranje. U postupku mapiranja koristimo dva mapiranja:



Za najjednostavniju među površinu, ravninu, potrebno je uzorak teksture reorijentirati (da se poklapa sa ravninom), skalirati, 3 puta rotirati i 3 puta translatirati. Naš problem se ''vrti'' oko dobrog izbora među površine za završno projiciranje na 3D objekt, a to je upravo sljedeće.

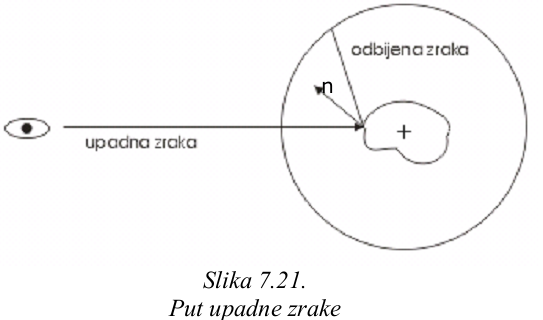
Primijenjuju se četiri tehnike mapiranja tekstura iz međupovršine u objekt:

1. reflektirana zraka (engl. reflected ray)
2. normala objekta
3. centroid objekta
4. normala među površine, ISN

O-mapiranje mora biti jednoznačno, invertibilno i kontinuirano. Bier i Sloan su zaključili da je prva metoda neupotrebljiva pa su promatrali samo zadnje 3 metode. Zaključili su da zadnja metoda izaziva najmanje distorzije.

*Prostorno mapiranje*

Imamo scenu, što odgovara objektima i svjetlu. Cilj je okolinu mapirati na unutrašnjost velike sfere, sa objektom koji se renderira unutar centra sfere. Slika okoline u objektu je dobivena praćenjem zrake iz oka do površine renderiranog objekta, reflektirajući zraku oko normale do površine i prateći odbijenu zraku do sfere. Green je predložio da se okolina mapira u kocku, kroz 6 rasterskih slika.



*Bump mapping*

Ideja je da se površina poligona učini grubom, hrapavom. U početku se to činilo tako da se fotografija hrapave teksture mapirala na površinu poligona, ali to nije bilo zadovoljavajuće jer je izgledalo kao hrapava tekstura naslikana na ravnoj površini (što u biti i jest). Razlog leži u tome što hrapave površine imaju mali slučajni faktor u normali površine i isto tako u smjeru odbijanja svjetlosti. Blinn je razvio metodu koja intezivno koristi derivacije i vektore. Govoreći laički, hrapavost se postiže korištenjem preturbacijske funkcije.



Preturbacijska funkcija može biti bilo koja funkcija čije su derivacije određene. Rezultat preturbacijske funkcije izaziva aliasing efekt, no on se uspješno otklanja prefiltriranjem.

**ANIMACIJA**

**UVOD**

ANIMIRATI = oživjeti.

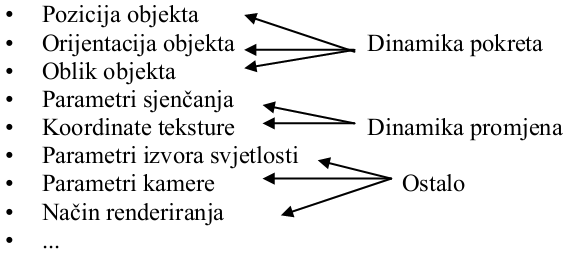
Animacija je uvriježeni sinonim za kretanje i pokriva sve promjene koje imaju vizalni učinak.

ANIMIRATI – stvoriti dojam o pokretu pomoću sekvence slika koje se prikazuju u brzom slijedu.

Animiranje može biti :

* u realnom vremenu (npr. video igre)
* off-line (npr. film ili televizija)

S obzirom na pitanje „Što se animira?“, animaciju možemo je podijeliti na tri osnovne grupe:



Primjene :

* Industrija zabave
* Obrazovanje
* Industrija (npr. u obliku upravljaĉkih sustava ili simulatora leta različitih letjelica)
* Znanstvena istraživanja
* Internet
* ...

**LJUDSKA** **PERCEPCIJA**

Naš vizualni sustav je evoluirao kako bi bolje registrirao pokrete.

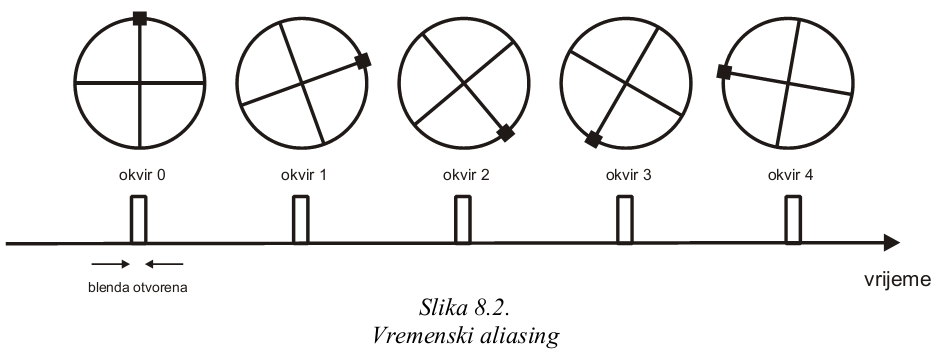
Sekvenca mirnih slika

* Tromost oka -> Perzistencija vida
* Ako je promjena dovoljno brza -> Interpretiramo je kao kontinuirane pokrete
* Film: 24 slike / s, PAL 25 slika / s, NTSC: 30 slika / s

Bljeskanje (Flicker) nastaje kad brzina osvježavanja nije dovoljno brza. Frekvencija titranja zavisi o osvjetljenju prostora ili udaljenosti od ekrana. Za smanjenje bljeskanja može se koristiti se Double Buffering metoda. Postupak:

1. Iscrta se nevidljivi pozadinski međuspremnik,
2. Kopira se vidljivi prednji međuspremnik. (Bitblt)==>vrlo brz ako imamo sklopovsku podršku

U nekim aspektima animacija se može mijenjati prebrzo u odnosu na broj prikazanih kadrova u sekundi pa dolazi do vremenskog aliasinga (TEMPORAL ALIASING) (npr. kola vagona izgledaju kao da se rotiraju unatrag umjesto unaprijed).



**VRSTE** **ANIMACIJE**

Vrste animacija su:

* Potpuno ručna metoda
* Key-frame animacija ili animacija ključnih kadrova
* Motion capture ili snimanje pokreta
* Proceduralna / simulacija (automatsko generiranje)
* Kombinacije (npr. snimanje pokreta + simulacija)

*Potpuno* *ručna* *metoda*

Svaku sliku se ručno nacrta. Moguće korištenje slojeva (layers).

Prednost:

* potpuna kontrola

Nedostatci

* zamorno, sporo, skupo



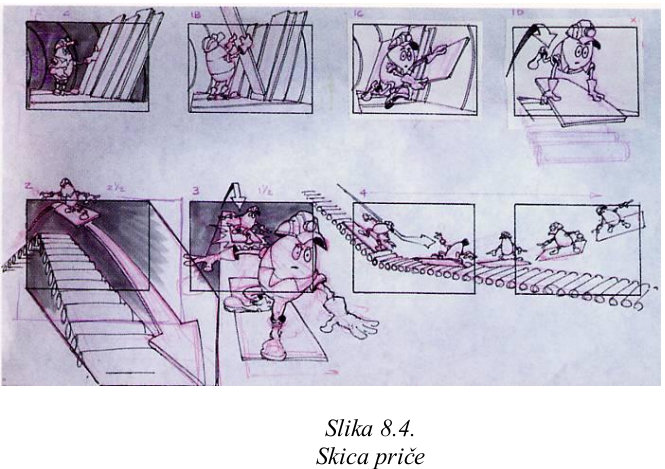
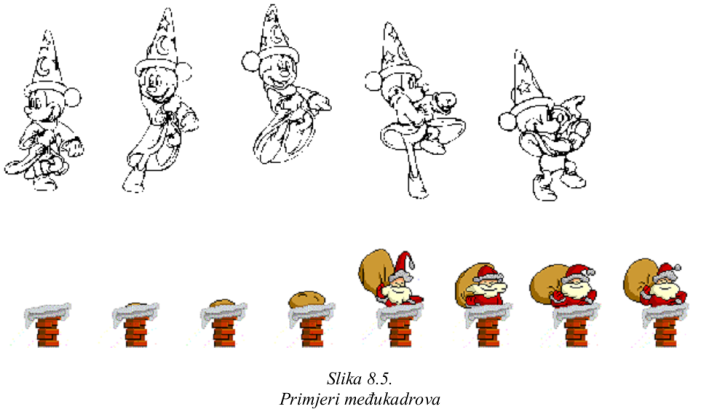
*Slojevi*

*Animacija ključnih kadrova (key-framing)*

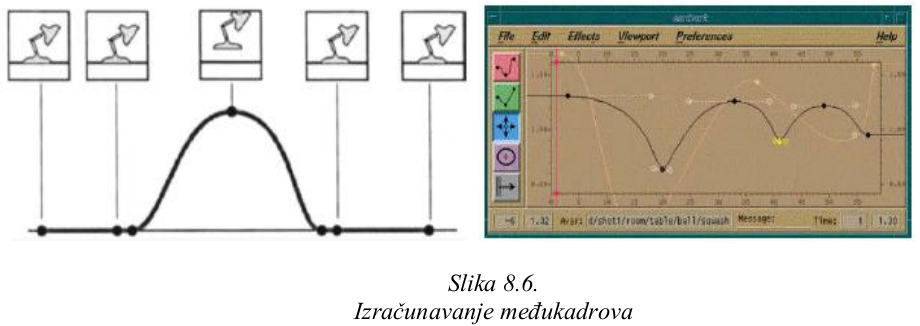
Također ručna metoda.

Postupak:

1. Prvo se stvori skica priče u obliku niza crteža (STORYBOARD).
2. Detaljno se razrade crteži za svaku scenu.
3. Trenutci u kojima se određeni zvukovi pojavljuju snimaju se po redu.
4. Korelira se zvuk i detaljni prikaz.
5. Izrađuju se KLJUČNI KADROVI -> kadrovi u kojima su animirani entiteti u ekstremnim ili karakteristiĉnim položajima i iz tih kadrova se onda mogu generirati međustanja

Pokreti objekata opisuju se u funkciji vremena iz skupa ključnih pozicija objekta. Ukratko, potrebno je izračunati međukadrove (*inbetweening*).



Proces proizvodnje animacije bi trebao biti sekvencijalan ali često je iterativan (pogotovo ako se vrši uz pomoć računala). Zvuk može uzrokovati potrebu blagog mijenjanja skice priče, eventualni izgled animacije može zahtjevati da se neke sekvence prošire, pa to zahtjeva nove segmente zvuka itd...

Mnoge korake konvencionalne animacije je moguće izraditi uz pomoć računala, posebice izradu međukadrova i bojenje. Prije toga je potrebno postojeće crteže digitalizirati (ili izraditi nove na računalu) te ih dodatno obraditi po potrebi.

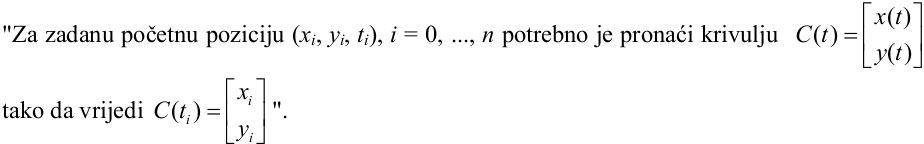
*Interpolacija*

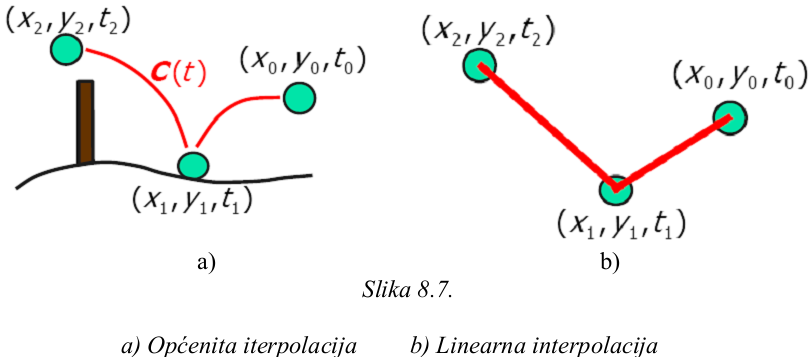
proces izvediv računalnim metodama, ali s nedostacima.

čovjek – predviđa okolnosti u kojima se nalazi objekt (npr. padajuća ili kotrljajuća lopta)

računalni sustav – dobije početni i konačni položaj

Općenito:

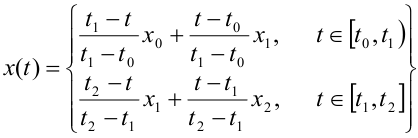




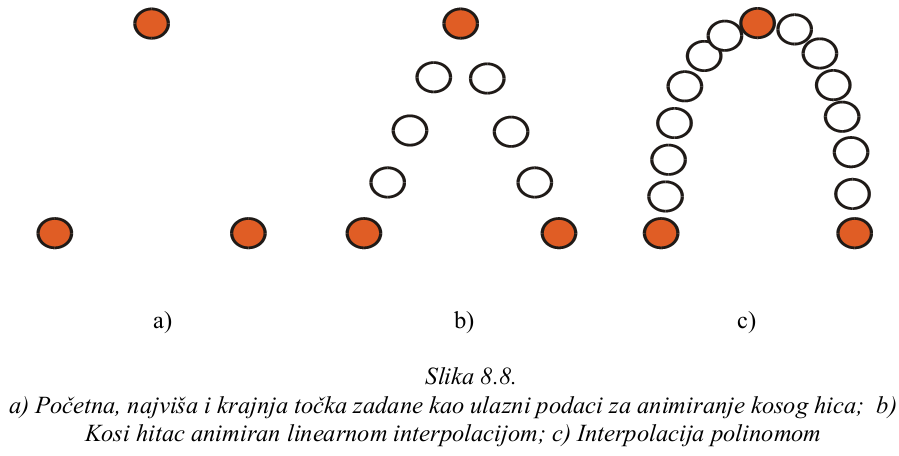
Najjednostavnija metoda : LINERANA INTERPOLACIJA (LERPING)

Uz pretpostavku : t0 = 0 i t1 =1; x(t) = x0(1-t) + x1t.

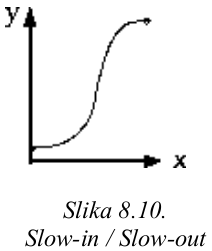
Linearna interpolacija koordinate x za cijelu krivulju na slici desno:



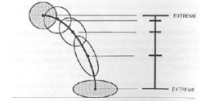
Linearna interpolacija: animacija može izgledati nerealistično, malo stvari u prirodi se kreće pravocrtno. Mogu se koristiti i polinomi n-tog stupnja za interpolaciju.



Da bi se izgladila interpolacija između ključnih okvira mogu se koristiti B-krivulje (spline-ovi). Mogu se koristiti za glatko variranje bilo kojeg parametra kao funkcije vremena. Da bi se dobio gladak početak i završetak promjene (tzv. slow-in i slow-out) i relativno konstantne brzine promjene, može se koristiti funkcija f(t): .



Slow in / slow out:



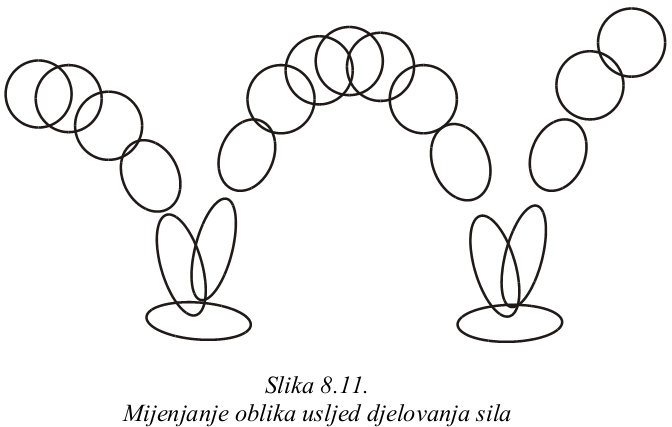
Jednaka udaljenost između slika = konstantna brzina. Bolje je imati postupno ubrzavanje pa usporavanje (kao u f(t)).

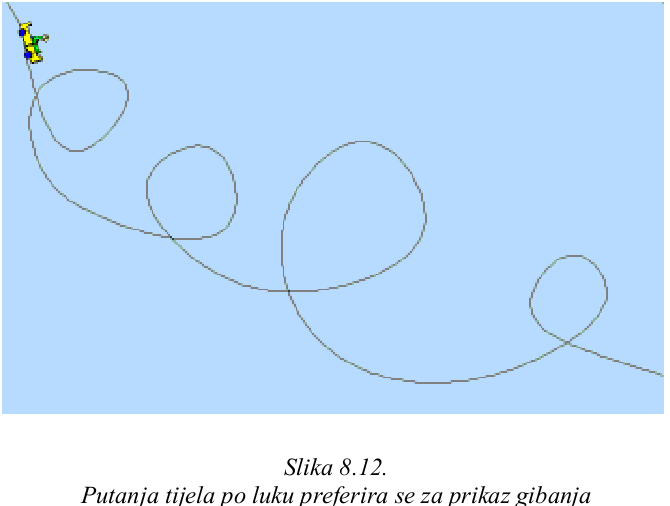
Vrijeme je ovisno o težini:

* Lakši objekt se giba brže
* Teži objekt se giba sporije

Faktor vremena vrlo mnogo utječe na interpretaciju pokreta. Zbog toga, animatori su često crtali vremensku skalu uz ključni kadar da bi naznačili kako treba generirati međukadrove.

Radi vjernije animacije (fizički model) često se primijenjuju metode mijenjanja oblika objekta uslijed udarca i djelovanja drugih sila.

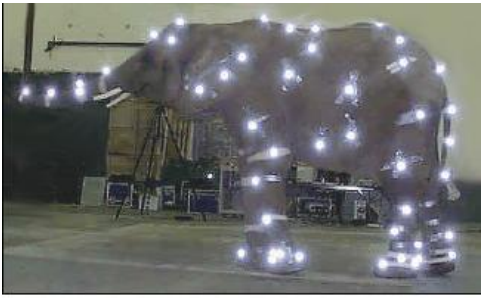




**SNIMANJE POKRETA (MOTION CAPTURE)**

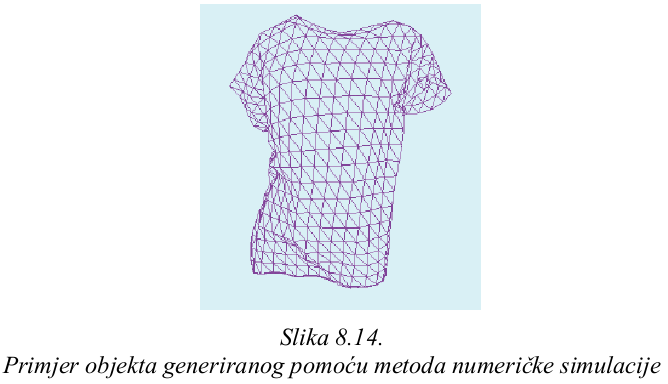
Dvije kategorije: optičke i ne-optičke.

* **Optičke** **metode** - specijalne kamere koje snimaju glumca (ili neki drugi objekt) koji se kreće (glumac nosi markere). \*optički sustavi bez markera, kod kojih glumci ne moraju nositi nikakve oznake jer se hvatanje pokreta temelji na računalom vidu i složenim algoritmima.
* **Ne**-**optičke** **metode** - ne moraju se koristiti kamere. Pokret se ovdje hvata korištenjem različitih inercijalnih senzora, mehaničkih sustava ili magnetskog polja (glumac nosi žiroskope, potenciometre ili magnetske prijemnike).

Markeri/senzori se postavljaju na subjekt. Vremenski zahtijevno postavljanje i skidanje. Dobar omjer kvaliteta / cijena. Algoritmi za upravljanje predmet su intenzivnog istraživanja.

**SIMULACIJA / PROCEDURALNA ANIMACIJA**

Pokreti objekata generiraju se pomoću metoda numeričke simulacije. Potrebna je točnost percepcije. Bitni faktori su i stabilnost, lakoća korištenja, brzina i robustnost.

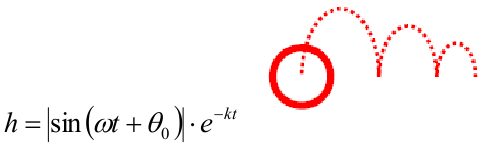


Pokreti se opisuju pomoću algoritama. Animacija se iskazuje funkcijom s malim brojem parametara.

Primjer: sat (kazaljke za sekunde, minute i sate):

* kazaljke trebaju rotirati zajedno
* iskaži kretnje kazaljki pomoću varijable “sekundi”
* sat je animiran variranjem parametra sekundi

Primjer: lopta koja skakuće



**ANIMACIJA TEMELJENA NA FIZIKALNIM ZAKONIMA**

Potrebno je pridružiti fizikalna svojstva svakom objektu (mase, sile, svojstva inercije). Ponašanje tijela u skladu s fizikalnim zakonima se simulira rješavanjem jednadžbi. Realistično, ali teško za upravljanje.

**ANIMACIJSKI JEZICI**

*Notacija linearnom listom*

Svaki događaj u animaciji se opisuje rednim brojem početnog i krajnjeg kadra i radnjom koja se treba izvršiti.

npr. 42,53, B ROTATE «PALM», 1, 30

Između 42. i 53. kadra rotiraj objekt koji se zove «PALM» oko osi 1 za 30º , određujući količinu rotacije za svaki kadar iz tablice B.

*Jezici opće namjene*

Vrijednosti varijabli u jeziku se mogu koristiti kao parametri za neke rutine koje generiraju animaciju, pa se viši jezici mogu koristiti za izradu simulacija koje onda kao nuspojavu generiraju animaciju. Takvi jezici imaju veliki potencijal, ali zahtjevaju značajno progamersko znanje i sposobnost korisnika.

**ANIMACIJSKI ALATI**

* 3ds Max (3D modeliranje i iscrtavanje) ~ 3500 $ /Autodesk/
* Maya (3D modeliranje, animacija, iscrtavanje, vizualni efekti) ~ 3500 $ /Autodesk/
* MotionBuilder (animacija likova u realnom vremenu) ~ 4000$ /Autodesk/
* Messiah Studio (3D animacija i iscrtavanje) ~ 600 $ /Messiah studio/
* Animata (Animacija u realnom vremenu) - opensource
* Blender (Modeliranje, iscrtavanje, animacija) – opensource

**PROIZVODNI POSTUPAK RAČUNALNE ANIMACIJE**

* Odjel priče: storyboards
* Umjetnički odjel: dizajn i boje
* Odjel modeliranja
* Odjel za razmještanje objekata
* Odjel sjenčanja
* Animacijski odjel
* Odjel osvjetljenja
* Odjel kamere: renderiranje

Cijena:

* 25 slika / sekundi
* animirani film od 80 minuta
* 120000 slika
* izrada 1 slika - 5 minuta
* ukupni utrošak vremena - približno 36\*106 sekundi
* (preko 1 godine uz pretpostavku da radimo 24 sata dnevno bez prekida!)