Zavod za telekomunikacije

Osnove virtualnih okruženja

Igor S. Pandžić

Uvod u 3D grafiku II: Iscrtavanje

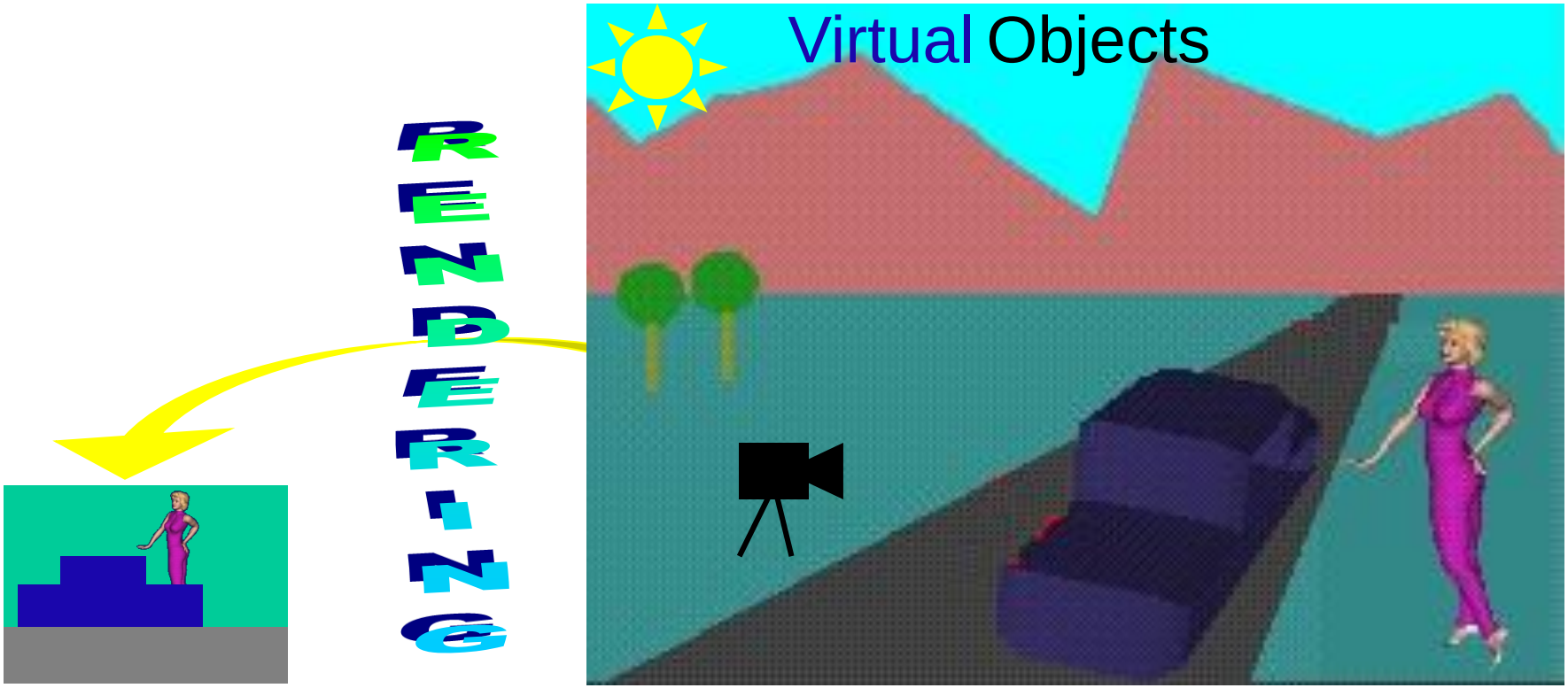
Iscrtavanje (engl. rendering)

Zavod za telekomunikacije

Virtual Light

Virtual Materials

Virtual Objects



Real Photo

Virtual Camera

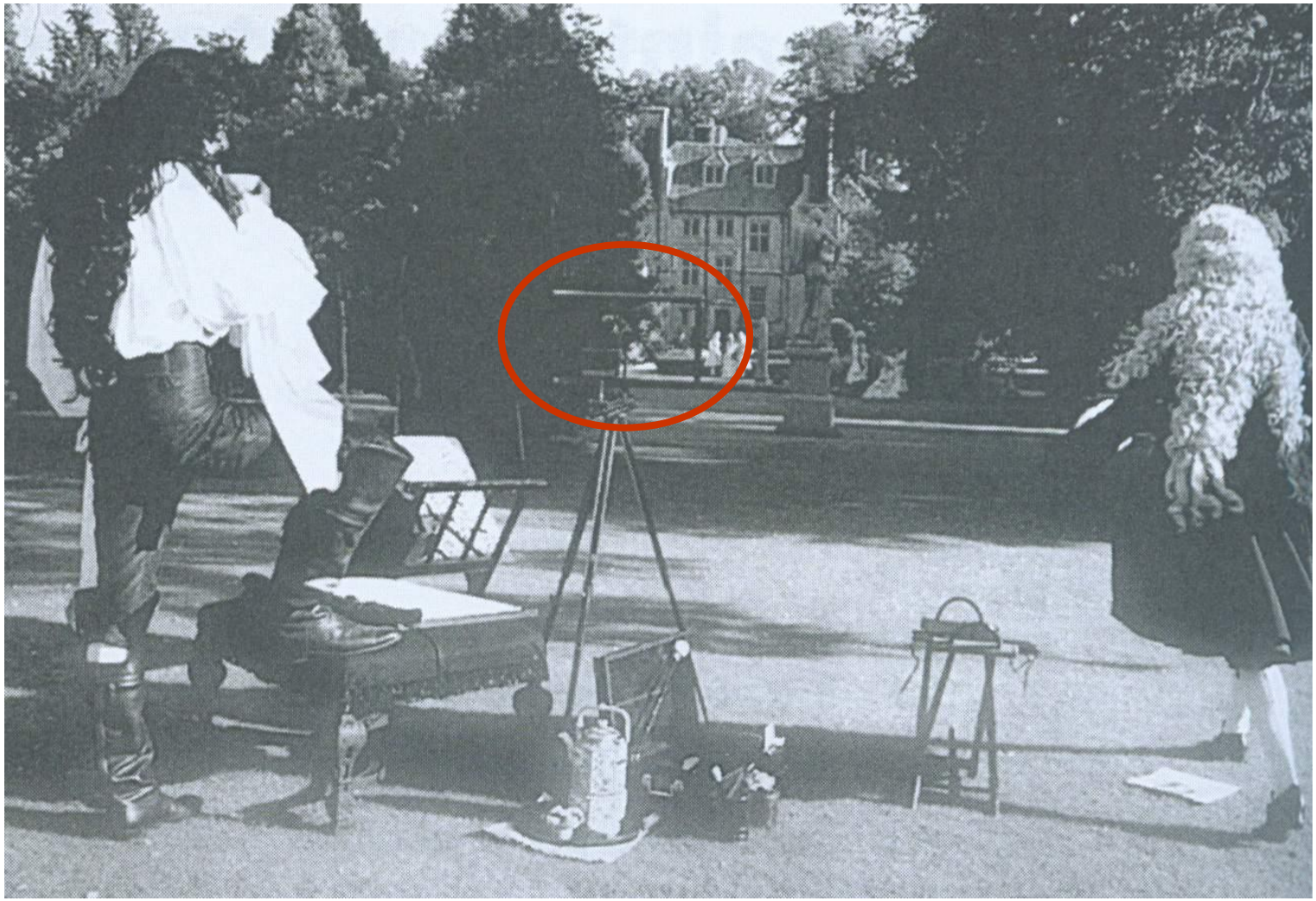
Uvod u 3D grafiku II: Iscrtavanje

- Ray tracing
- Grafički protočni sustav u realnom vremenu
  - Aplikacijska faza
  - Geometrijska faza
  - Faza rasteriziranja
- Osnovne tehnike
  - Sjenčanje
  - Teksturiranje
  - Određivanje vidljivosti
  - Prozirnost
  - Anti-aliasing

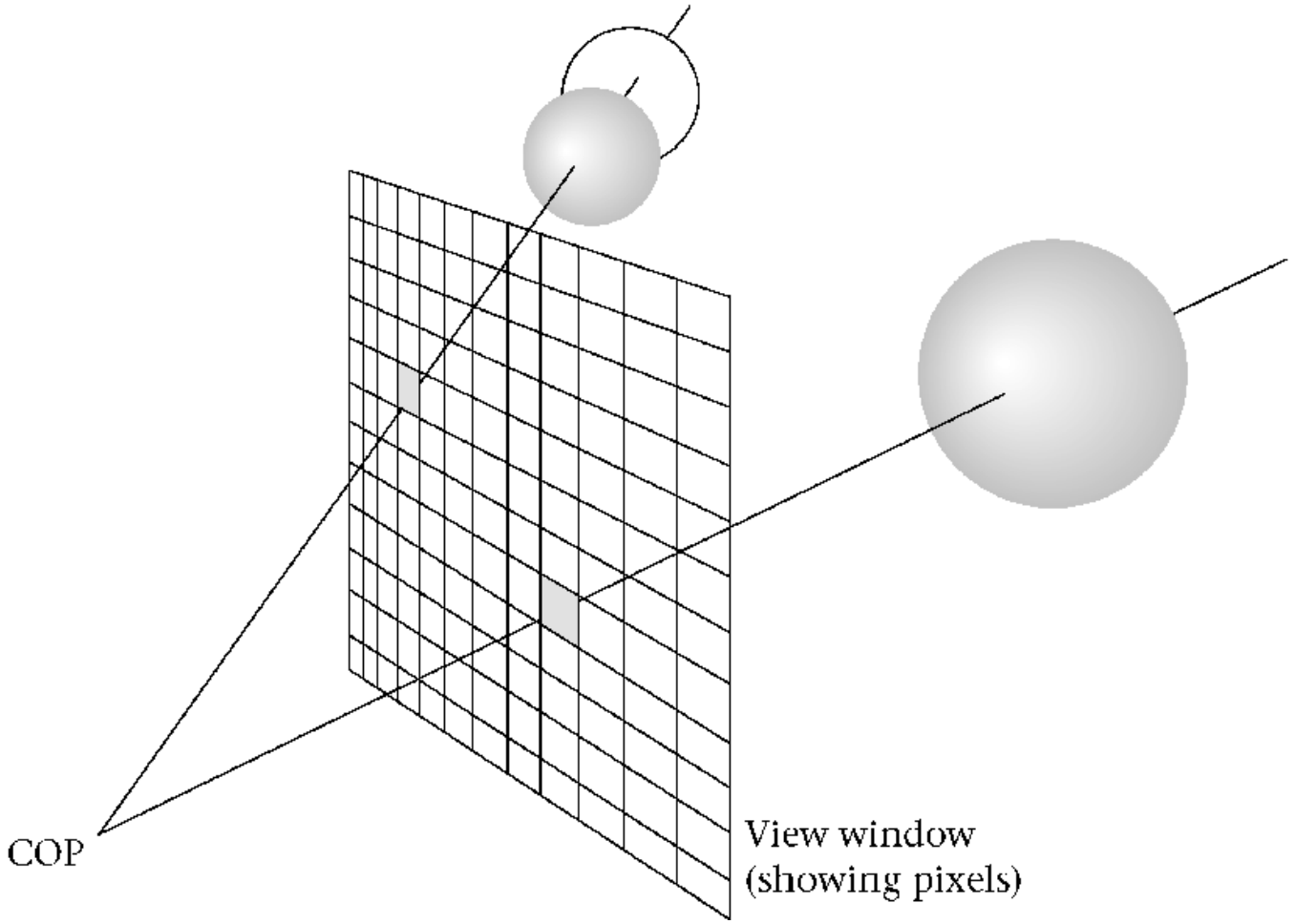
Uvod u 3D grafiku II: Iscrtavanje

- Programska sučelja za 3D grafiku
  - Programska sučelja niske razine
  - Programska sučelja visoke razine
- Ostale metode iscrtavanja
  - Isijavanje
  - Vizualizacija volumena

Kada slikari „varaju“...

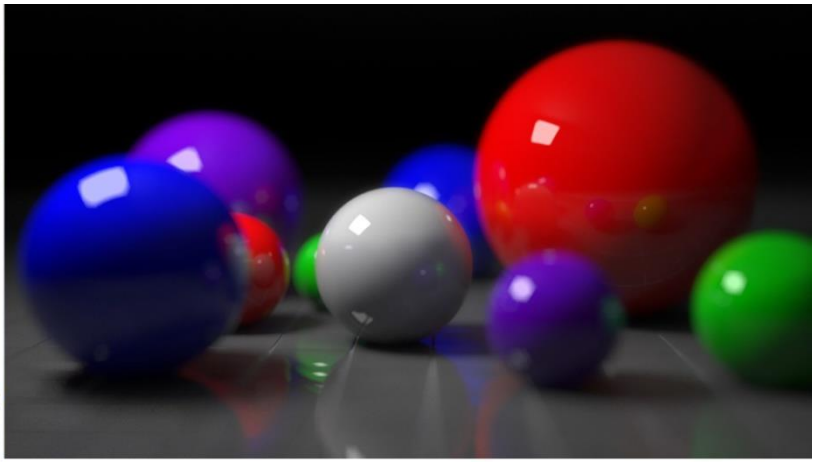


Projekcija



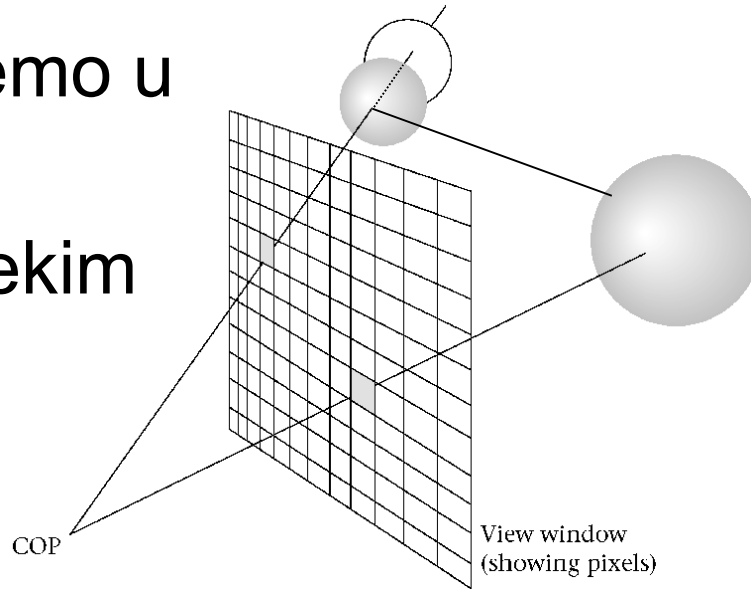
Praćenje zrake (engl. ray tracing)

- Klasična i široko upotrebljavana metoda
  - Odlično prikazuje refleksije, (oštre) sjene, prozirnost
- Nije podobno za realno vrijeme
- Relativno jednostavna metoda
  - Odlična za shvaćanje problema iscrtavanja

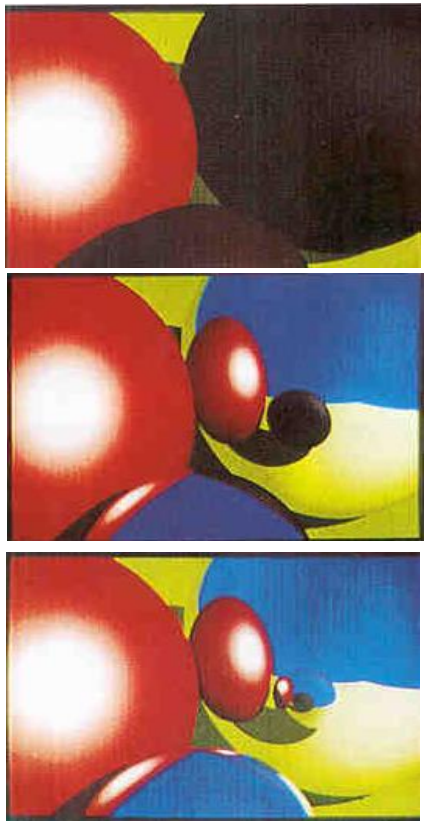


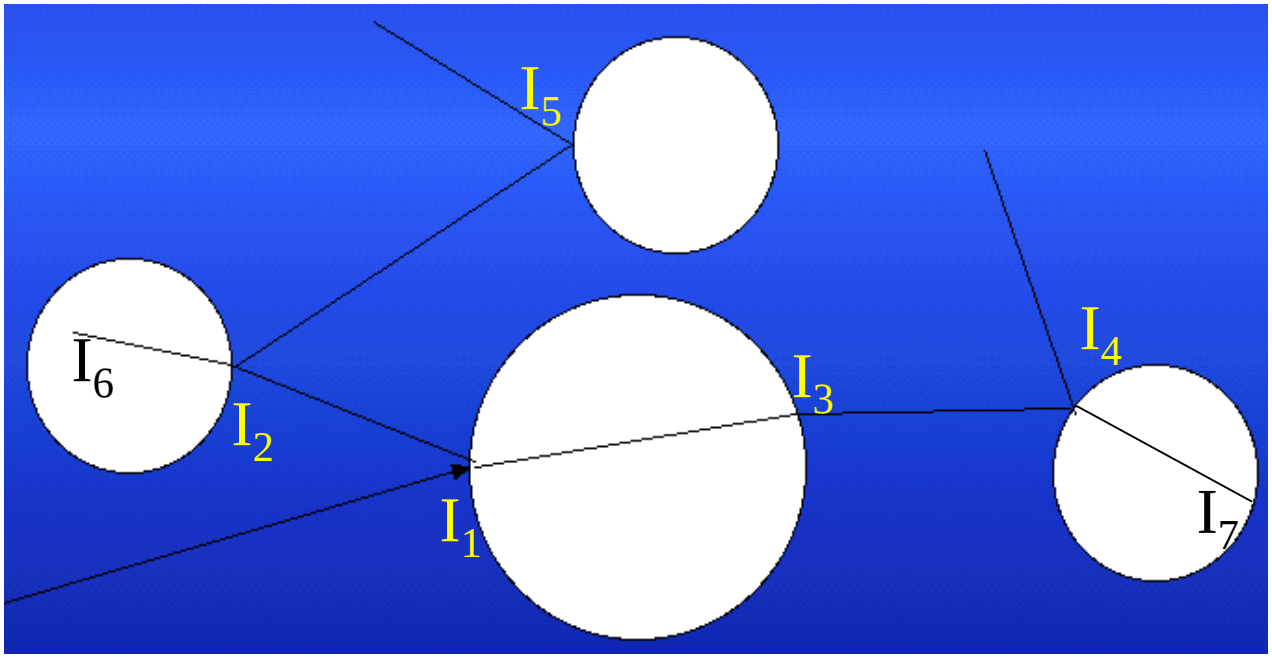
Princip praćenja zrake

- Za svaku točku ekrana šaljemo u scenu jednu zraku
- Na prvom sudaru zrake s nekim predmetom računamo osvjetljenje
- Zraka se lomi i odbija
  - Nastaju dvije nove zrake
  - Rekurzivno ponavljamo na njima postupak
  - Zbrajamo sve doprinose osvjetljenja
  - Rezultat je boja točke na ekranu



Praćenje zrake - primjer





$$I_1 = I_1^L + k_2 I_2 + k_3 I_3$$
$$I_2 = I_2^L + k_5 I_5 + k_6 I_6$$
$$I_3 = I_3^L + k_4 I_4 + k_7 I_7$$
$$\ldots$$

Malo više detalja

- Kako izračunati osnovnu, odbijenu i lomljenu zraku?
- Što je sa sjenama?
- Kada zaustaviti rekurziju?
- Kako naći presjek zrake i predmeta?

Kako izračunati osnovnu zraku?

- Sjetimo se modela kamere
  - Centar projekcije, projekcioni prozor...
- Projekcioni prozor dijelimo na broj točaka kolika je rezolucija ekrana
- Problem se svodi na pravac kroz dvije točke

Kako izračunati odbijenu zraku?

- Zrcaljenje oko normale na površinu u točki presjeka zrake s predmetom

$$R = -(2U \cdot N)N - U$$

  - U – ulazna zraka
  - R – odbijena zraka
  - N – normala u točki presjeka (jedinični vektor)

Kako izračunati lomljenu zraku?

- Snellov zakon

$$\eta_1 \sin \theta_1 = \eta_2 \sin \theta_2$$

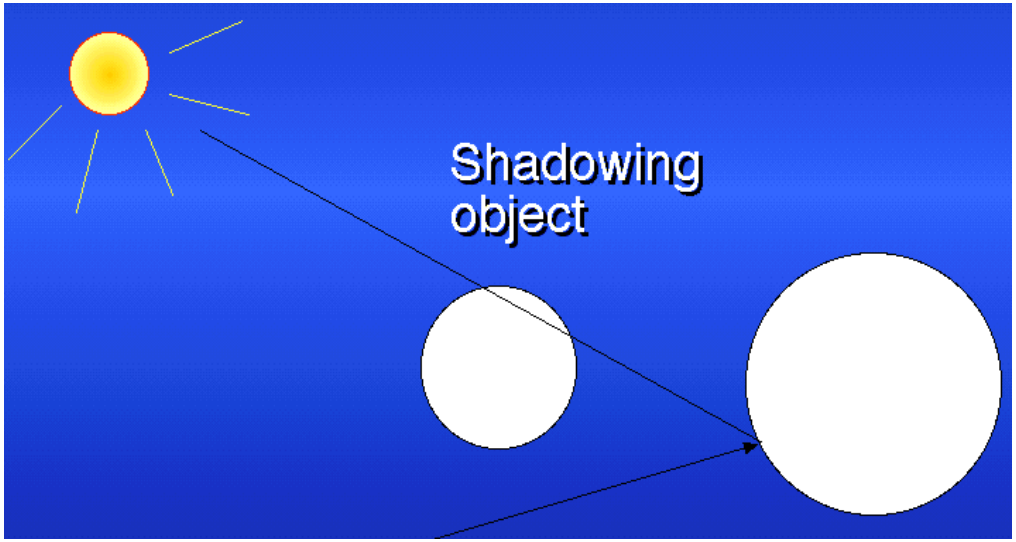
- Indeks loma

$$\eta = c / v$$

  - c: brzina svjetlosti u vakuum
  - v: brzina svjetlosti u promatranom mediju

Što je sa sjenama?

- Od mjesta presjeka šalje se zraka prema izvoru svjetlosti (shadow ray, shadow feeler)
  - Ako se zraka sječe s neprozirnim predmetom, lokalni doprinos osvjetljenja je nula
  - Mogu se uzeti u obzir prozirni predmeti



Kada zaustaviti rekurziju?

- Nakon n-te razine rekurzije
- Još bolje: kada doprinos zrake padne ispod zadanog praga
  - Doprinos pada jer se množi svim faktorima refleksije/prozirnosti preko kojih se dolazi do trenutno promatrane zrake

Kako naći presjek zrake i predmeta?

- E, to....
- Za svaku vrstu predmeta, drugačiji postupak
- Najčešći princip:
  - Uvrstiti jednadžbu zrake u jednadžbu predmeta
  - Zraka: $P = P_0 + \mu U$, $\mu > 0$
    - P_0 – izvor zrake (x_0, y_0, z_0)
    - U – vektor smjera zrake (Ux, Uy, Uz)
- Banalni primjer: kugla sa središtem u ishodištu
  - $x^2 + y^2 + z^2 = r^2$
  - $(x_0 + \mu Ux)^2 + (y_0 + \mu Uy)^2 + (z_0 + \mu Uz)^2 = r^2$

Praćenje zrake - zaključak

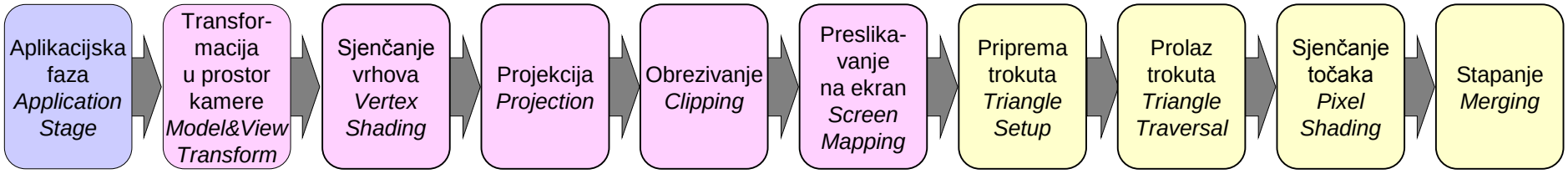
- Općenita metoda
  - Za bilo koju vrstu predmeta dovoljno je napisati novi algoritam presjeka
- Postoje proširenja metode za prirodnije efekte
  - Mekane sjene, defokusiranje, razliveni odsjaji, efekt mliječnog stakla...
- Najčešće korištena metoda za off-line grafiku

Grafički protočni sustav u stvarnom vremenu

- Engl. Graphics Rendering Pipeline
- Niz funkcija koje se izvode jedna za drugom, a koje virtualnu scenu pretvaraju u sliku
- Funkcije se izvode istovremeno, kao na pokretnoj traci; najsporija funkcija usko grlo
- Dio funkcija izveden u sklopovlju
- Neke funkcije se mogu paralelizirati
- Optimizirano za rad s trokutima

Osnovne faze grafičkog protočnog sustava

- Aplikacijska faza
  - Operacije specifične za aplikaciju; priprema primitiva (trokuta)
- Geometrijska faza
  - Transformacije, projekcije, osvjetljenje: što, gdje i kako iscrtati
- Faza rasteriziranja
  - Konačno iscrtavanje, tj. „punjenje“ točaka
- Osnovne faze dijele se na funkcijske faze
- U implementaciji, funkcijske faze se preslikavaju u implementacijske faze, ne nužno jedan na jedan



Aplikacijska faza (1/2)

- „Puni“ ostatak protočnog sustava elementima za iscrtavanje u odgovarajućem obliku
  - To su najčešće trokuti, točke i linije
  - Ovo je osnovni zadatak aplikacijske faze
- Sve „pametne“ stvari se rade ovdje:
  - Logika same aplikacije
  - Animacija
  - Simulacija
  - Ulaz/izlaz
  - Detekcija sudara
  - Ostalo...

Aplikacijska faza (2/2)

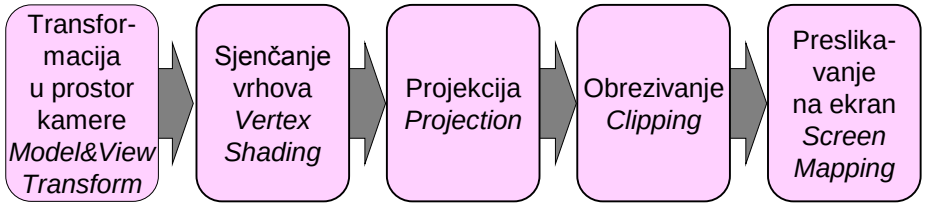
- Uvijek izvedena programski: najveća kontrola
- Najčešće nema pod-faza (osim možda korištenjem više procesora)
- Može jako utjecati na ukupnu brzinu iscrtavanja
  - Npr. odbacivanjem trokuta (engl. culling)

Geometrijska faza

- Ulaz: 3D trokuti, svjetla, kamera
- Izlaz: 2D trokuti u zaslonskim koordinatama
- Izvedba najčešće sklopovska, na grafičkom procesoru (eng. Graphics Processing Unit, GPU)

Geometrijska faza: funkcijske faze

- Transformacija u prostor kamere (engl. model & view transform)
- Sjenčanje vrhova (engl. vertex shading)
- Projekcija (engl. projection)
- Obrezivanje (engl. clipping)
- Preslikavanje na ekran (engl. screen mapping)



Tipični ulaz u geometrijsku fazu

- Niz trokuta
  - Određeni vrhovima (vertex)
  - Mogu se zadati normale za svaki vrh
  - Može se zadati fiksna boja za trokut ili za svaki vrh
  - Parametri materijala
- Većina operacija izvodi se na vrhovima, dakle 3D točkama (x,y,z)

Transformacija u prostor kamere (1/3)

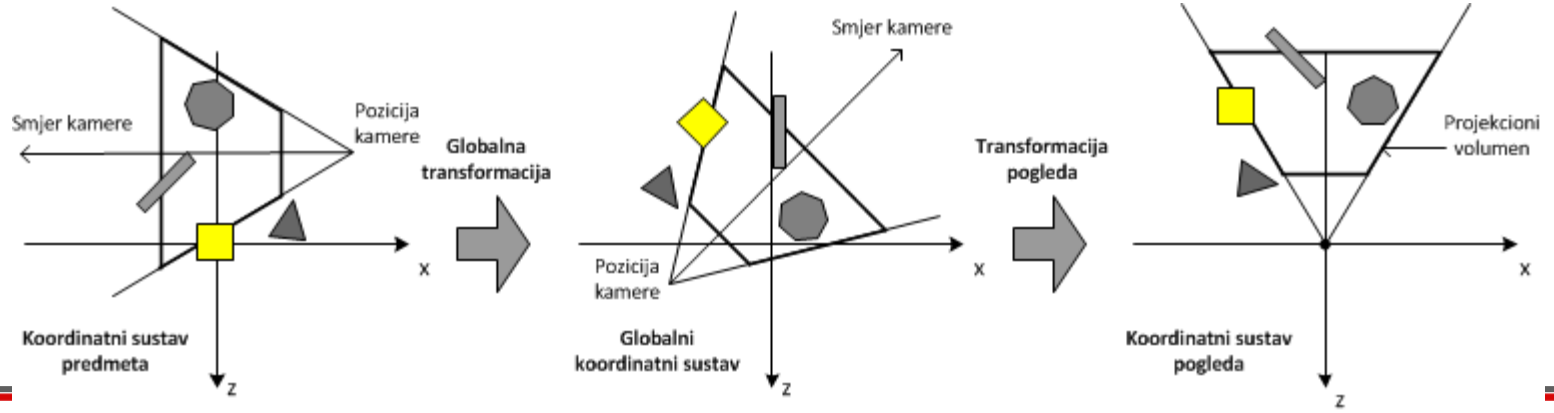
- Vrhovi su zadani u *koordinatnom sustavu predmeta*
- Svakom predmetu pridružena je *globalna transformacija*, koja određuje njegov položaj i orijentaciju u *globalnom koordinatnom sustavu* (zajednički za čitavu scenu)
- Kamera ima svoj *koordinatni sustav pogleda* – definira se *transformacija pogleda* koja transformira iz globalnog sustava u sustav pogleda

Transformacija u prostor kamere (2/3)

- KS pogleda:
  - Kamera je u ishodištu
  - Gledamo u smjeru –z osi
  - y os je gore, x os je desno

Transformacija u prostor kamere (3/3)

- Izvodi se kao 2 transformacije:
  1. Transformacija iz KS predmeta u globalni KS (korištenjem globalne transformacije)
  2. Transformacije iz globalnog KS u KS pogleda (korištenjem transformacije pogleda)
- Transformacije su zadane kao 4x4 matrice kojima množimo svaki vrh

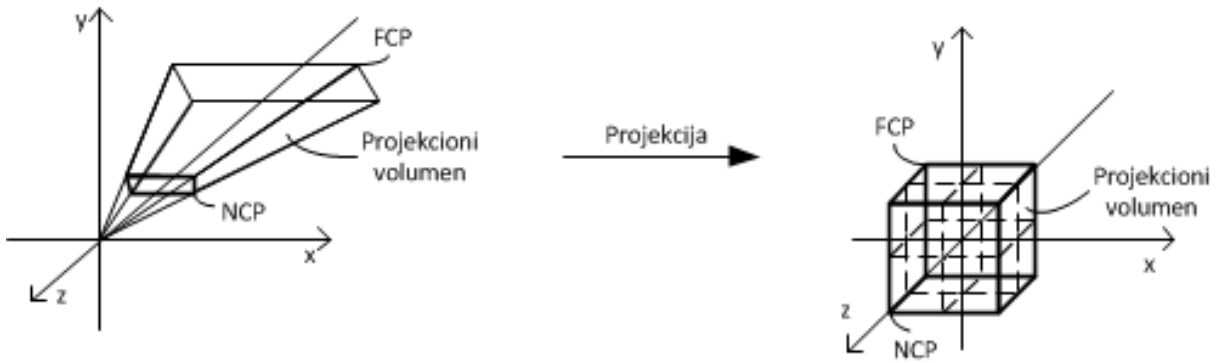


Sjenčanje vrhova (engl. vertex shading)

- U klasičnoj izvedbi, osvjetljenje se može računati npr. modelom s prošlog predavanja
  - Normala, materijal, svjetla, položaj kamere -> (jednadžba sjenčanja) -> boja vrha/trokuta
- Boja može biti i direktno zadana (nema osvjetljenja, slab 3D efekt)
- Moderan GPU – sjenčanje vrhova potpuno programabilno:
  - Proizvoljni modeli osvjetljenja i sjenčanja
  - Pomicanje/stvaranje/brisanje vrhova

Projekcija

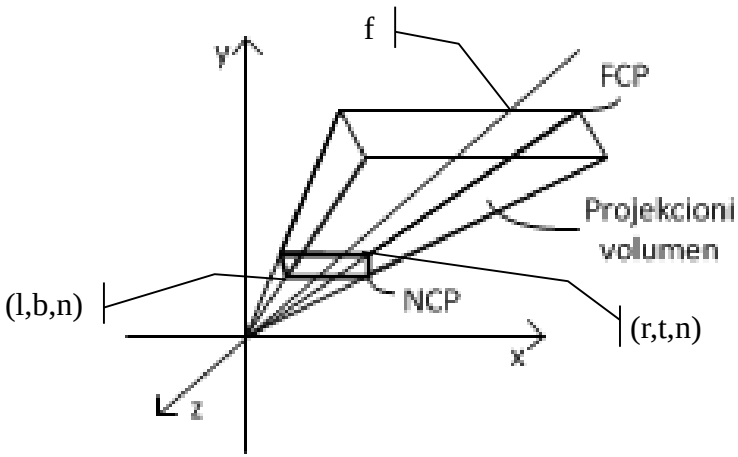
- Projekcioni volumen se transformira u jediničnu kocku (-1 -1 -1) (1 1 1):
  - x i y su normalizirane projicirane koordinate
  - z je norm. dubina, još će nam zatrebati



- Obavlja se tako da se vrhovi množe matricom projekcije

Matrica projekcije, vidni kut, *aspect ratio*

$$P_P = \begin{bmatrix} \frac{2n}{r-l} & 0 & 0 & 0 \\ 0 & \frac{2n}{t-b} & 0 & 0 \\ -\frac{r+l}{r-l} & -\frac{t+b}{t-b} & \frac{f+n}{f-n} & 1 \\ 0 & 0 & -\frac{2fn}{f-n} & 0 \end{bmatrix}$$



- Vidni kut (engl. *Field of View, FOV*)
  - Objekt širine *w* na udaljenosti *d* vidi se pod kutem:
  - Veza vidnog kuta i matrice Pp:
- Omjer širina/visina (engl. *aspect ratio*):

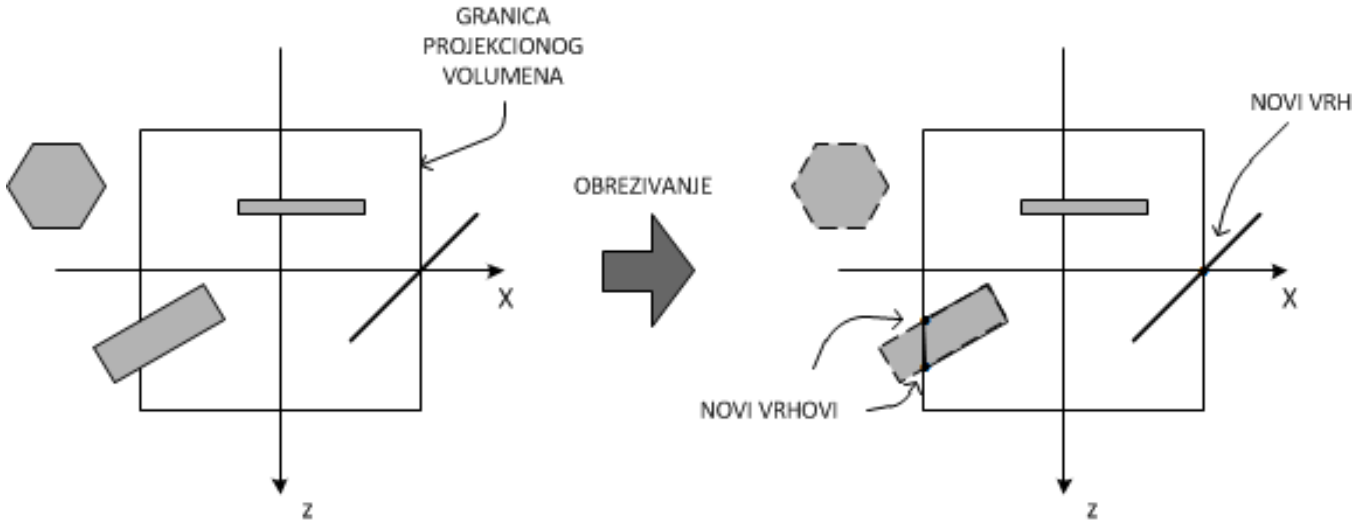
$$\phi = \arctan\left(\frac{w}{2d}\right)$$

$$\phi_h = \arctan\left(\frac{r-l}{2n}\right)$$

$$\mathrm{A} = \frac{r-l}{t-b} = \frac{\phi_h}{\phi_v}$$

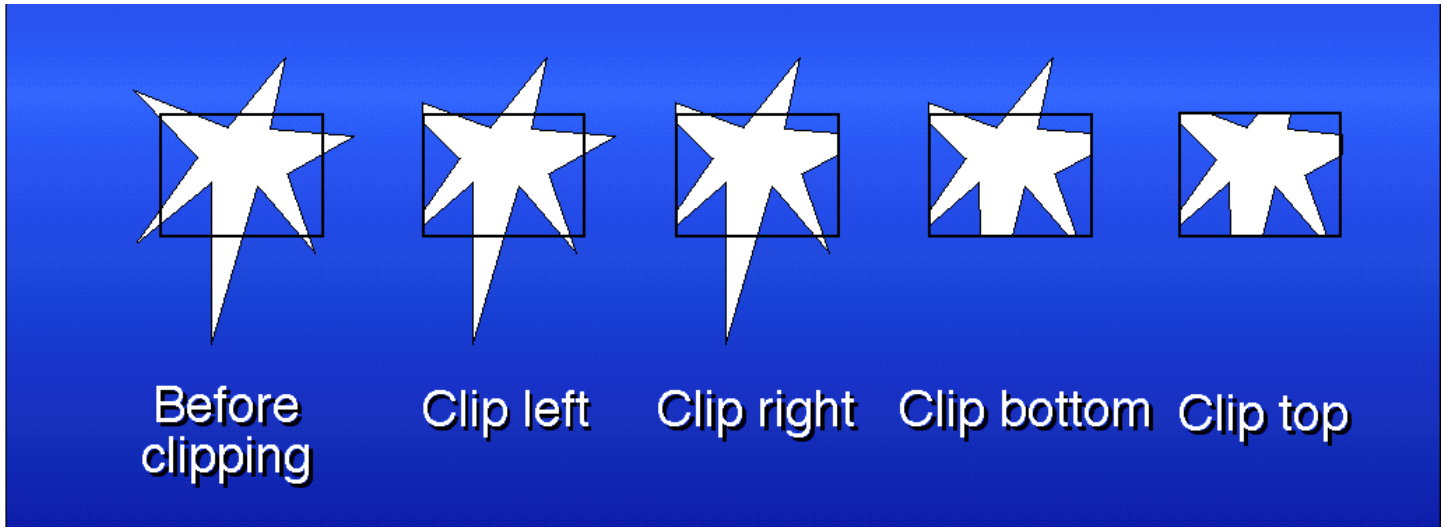
Obrezivanje (engl. clipping) (1/2)

- Odbacuju se trokuti koji su izvan jedinične kocke
- Od trokuta koji presijecaju granice jediničke kocke nastaju novi, krnji trokuti



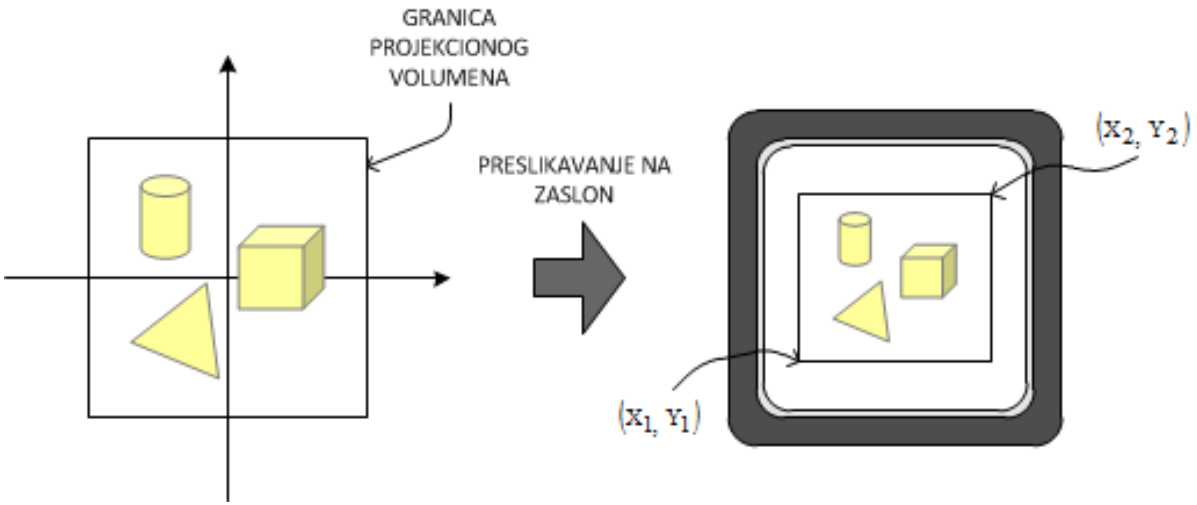
Obrezivanje (engl. clipping) (2/2)

- Sutherland-Hodgeman algoritam
  - Obrezuje se jedna po jedna stranica kocke



Preslikavanje na ekran

- x i y se transformiraju iz intervala -1,1 u koordinate ekrana
  - Jednostavna linearna operacija $x_s = ax_n + b$
- z ostaje nepromijenjen

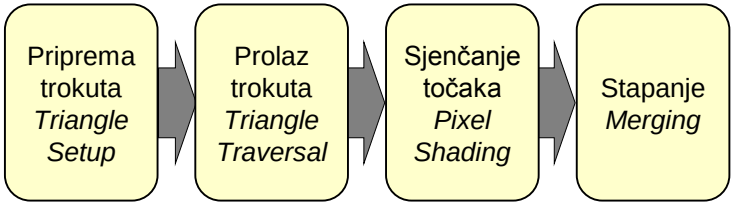


Faza rasteriziranja

- Engl. rasterisation, scan conversion
- Upisuje boju u pojedine točke zaslona (piksele)
- Određuje vidljivost točaka
- Obavlja razne dodatne funkcije (detalji kasnije)
  - Teksturiranje, prozirnost, anti-aliasing, efekti zamućenosti...
- Vrlo zahtjevna, gotovo uvijek izvedba na grafičkom procesoru

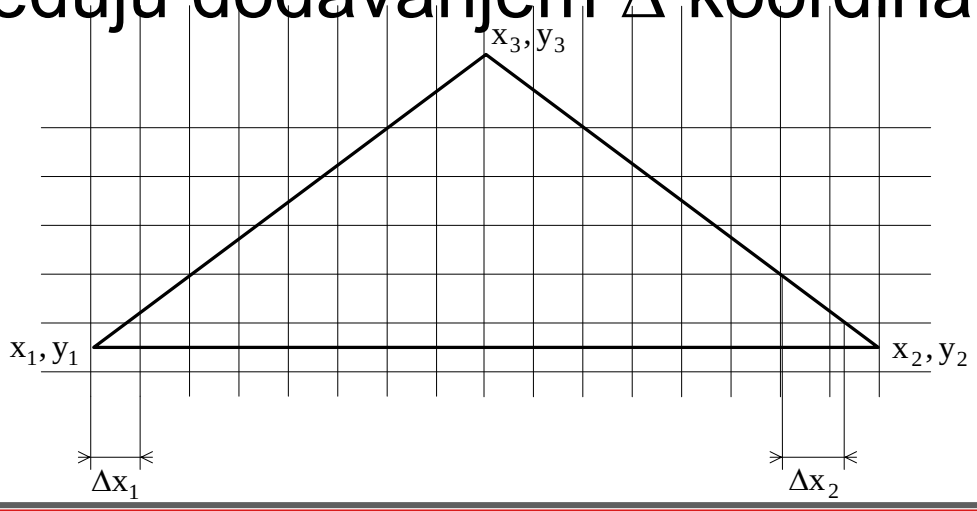
Faza rasteriziranja: funkcijske faze

- Priprema trokuta (engl. triangle setup)
- Prolaz trokuta (engl. triangle traversal)
- Sjenčanje točaka (engl. pixel shading)
- Stapanje (engl. merging)



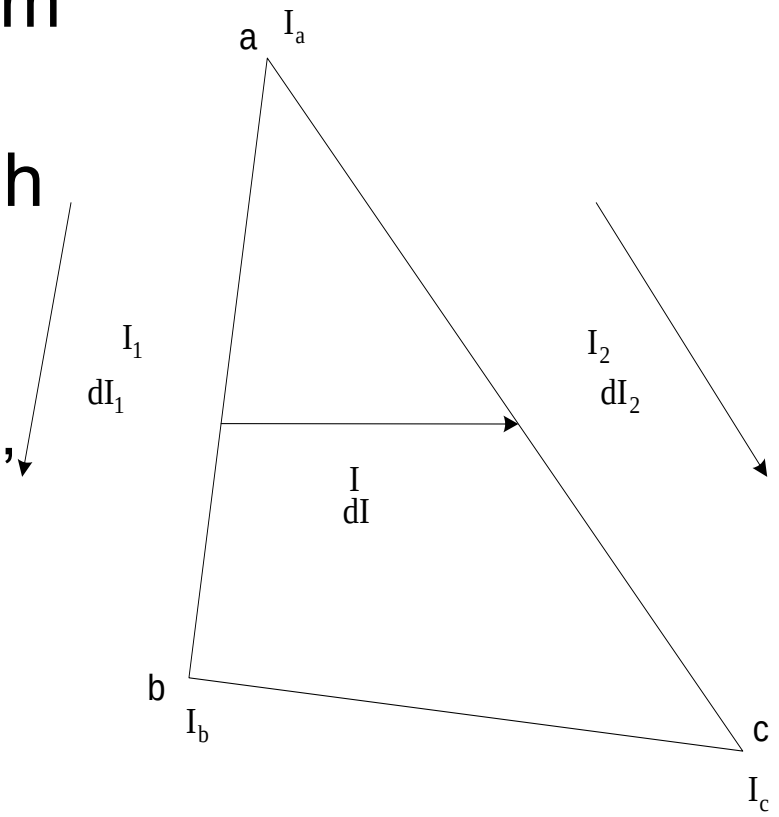
Priprema i prolaz trokuta

- Fazi pripreme – izračun Δ koordinata
- Faza prolaza – prekrivenost, izračun fragmenta
- Prolaz trokuta redak po redak, od lijevog do desnog ruba
- Pri prijelazu u novi redak, novi lijevi i desni rub se određuju dodavanjem Δ koordinata



Prolaz trokuta

- Svi podaci u vrhu interpoliraju se na prethodno objašnjenom principu (korištenjem Δ)
- Fragment: skup interpoliranih podataka potrebnih za sjenčanje jedne točke
  - To mogu biti koordinate točke, dubina, boja, koordinate teksture, normala, prozirnost, specifični podaci za pojedine efekte sjenčanja...



Sjenčanje točaka

- Iz interpoliranih podataka računa se boja točke
- Može se raditi kompletan proračun modela osvjetljenja, ili samo koristiti interpolirana boja
- Programabilna faza – gotovo neograničen broj efekata
- Teksturiranje – određuje se boja teksture u trenutnoj točki, te se kombinira s bojom sjenčanja, ili koristi kao parametar materijala...

Stapanje

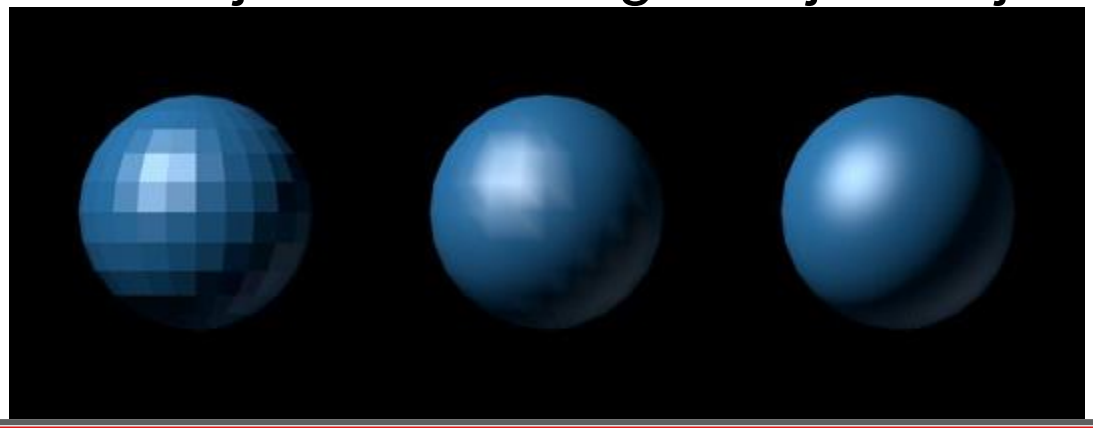
- Izračunata boja točke stapa se s postojećom bojom točke u spremniku boje
- Konfiguriranjem ove faze postižu se razni efekti:
  - ROP – mat. i log. operacije na spremnicima
  - Primjeri – efekt maske, prozirnost, anti-aliasing...
  - Npr. efekt maske – oblik iscrtan u zasebnom spremniku (spremniku maske) određuje područje ekrana u kojem se točke iscrtavaju
- Faza stapanja odgovorna je i za određivanje vidljivosti metodom Z-spremnika

Osnovne tehnike

- Standardni dio grafičkog protočnog sustava
- Najvažnije tehnike:
  - Sjenčanje
  - Preslikavanje tekstura
  - Određivanje vidljivosti
  - Prozirnost
  - Anti-aliasing

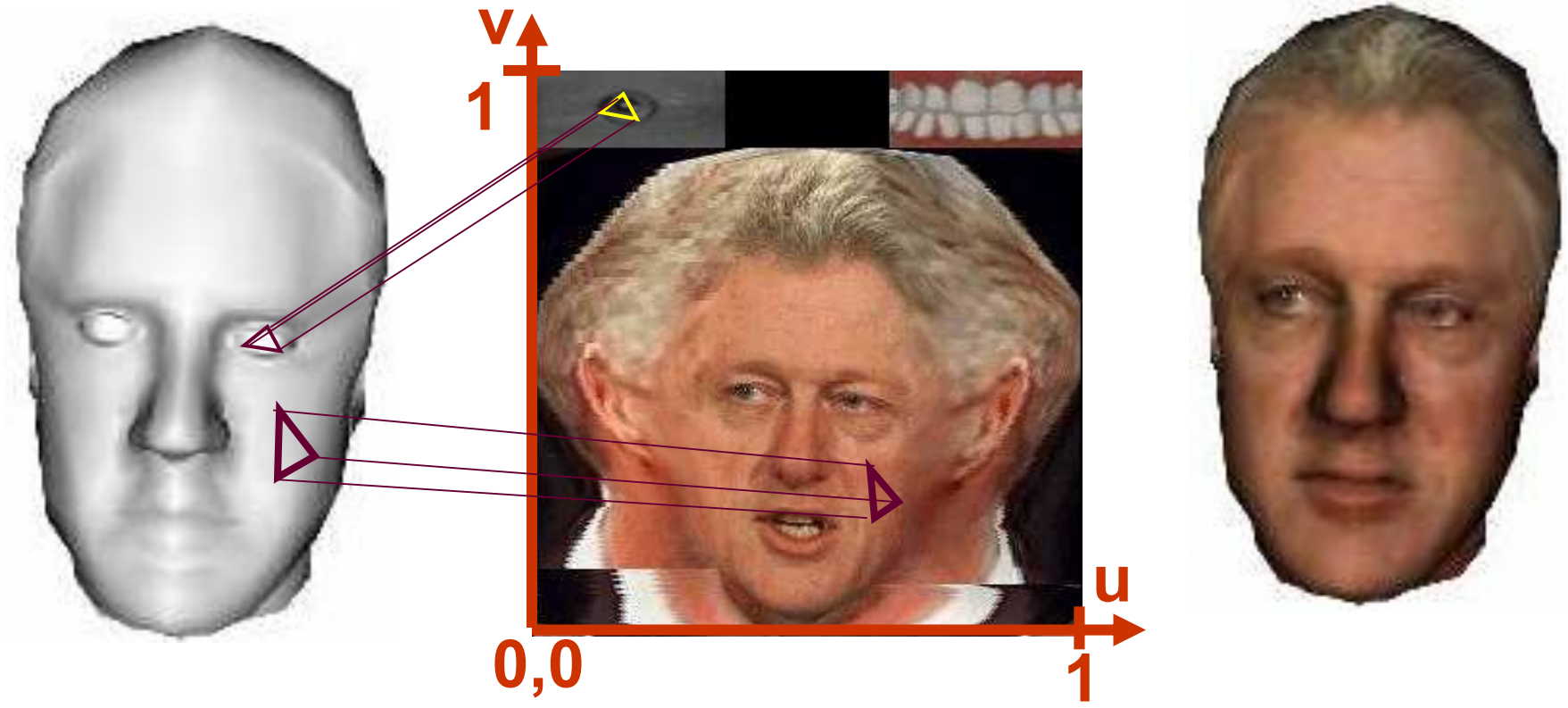
Sjenčanje

- Određivanje boje na temelju svojstava materijala i izvora svjetlosti, korištenjem jednadžbe sjenčanja
- Jednadžba sjenčanja može se evaluirati:
  - Jednom za svaki trokut – plošno sjenčanje
  - U svakom vrhu – Gouraudovo sjenčanje
  - U svakoj točki – Phongovo sjenčanje



Preslikavanje tekstura (1/2)

- U osnovi: „lijepljenje“ slike na geometriju
- Za svaki vrh imamo koordinate teksture u, v

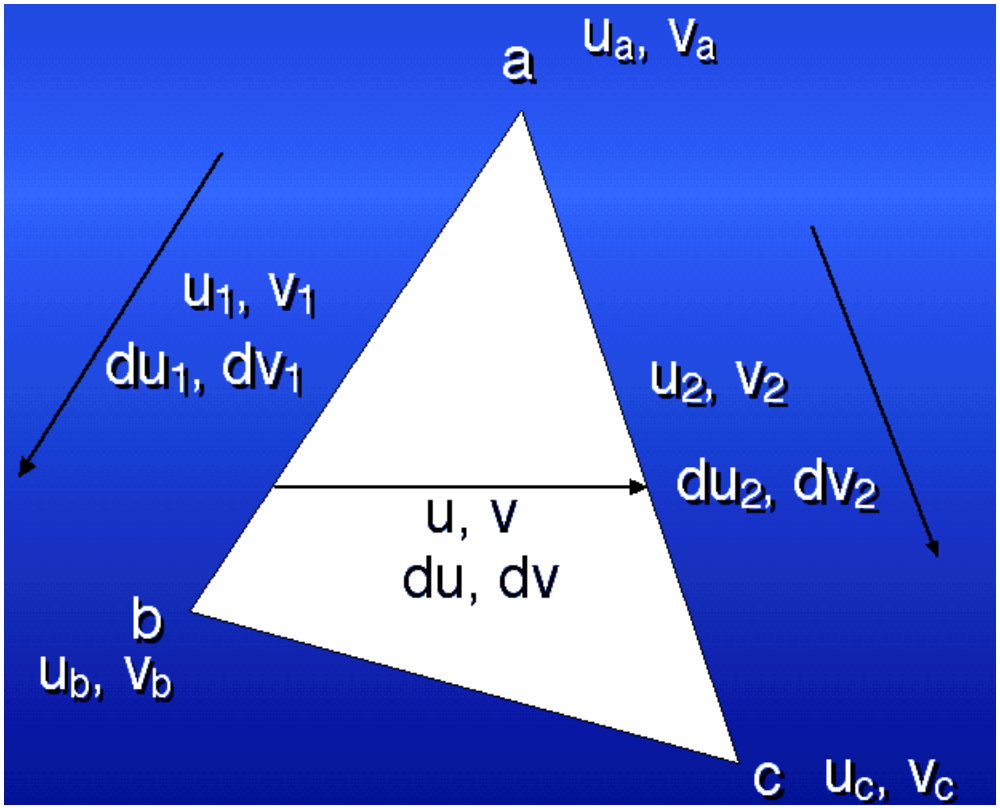


Preslikavanje tekstura (2/2)

- Izvedeno u fazi rasterizacije
- Faze pripreme i prolaza trokuta – teksturne koordinate se interpoliraju
- Faza sjenčanja točaka – uzorkovanje (dohvat boje) teksture u točki; dobivena boja naziva se teksel
- Teksel se koristi za dobivanje konačne boje točke (npr. težinskim miješanjem s bojom osvjetljenja)

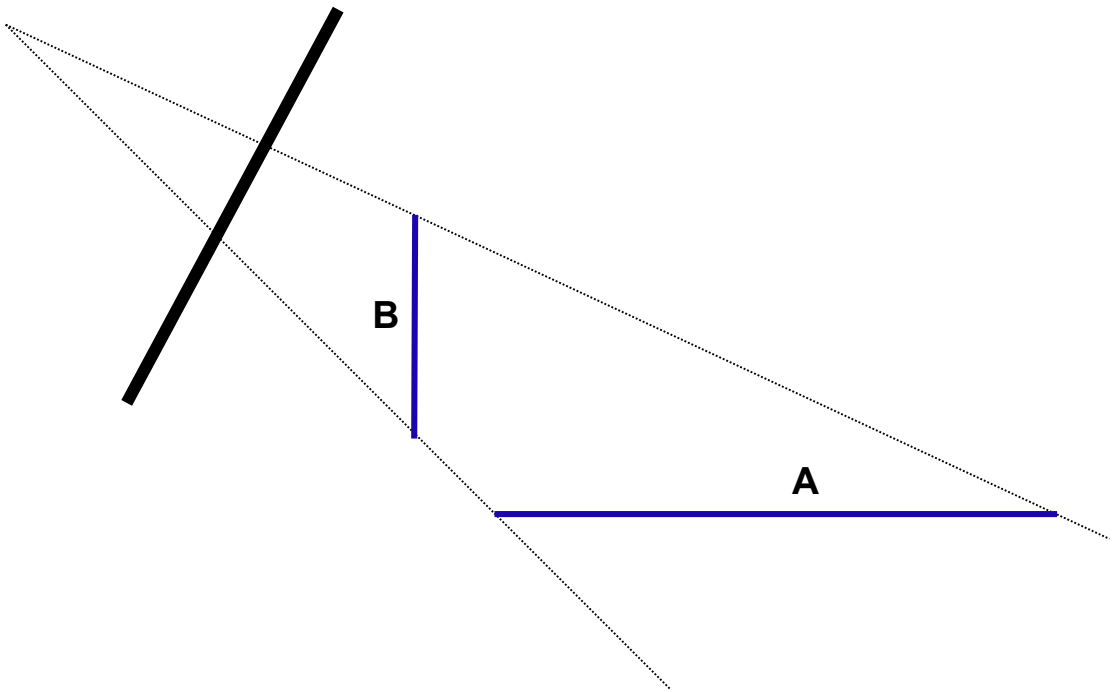
Interpolacija u,v koordinata (1/3)

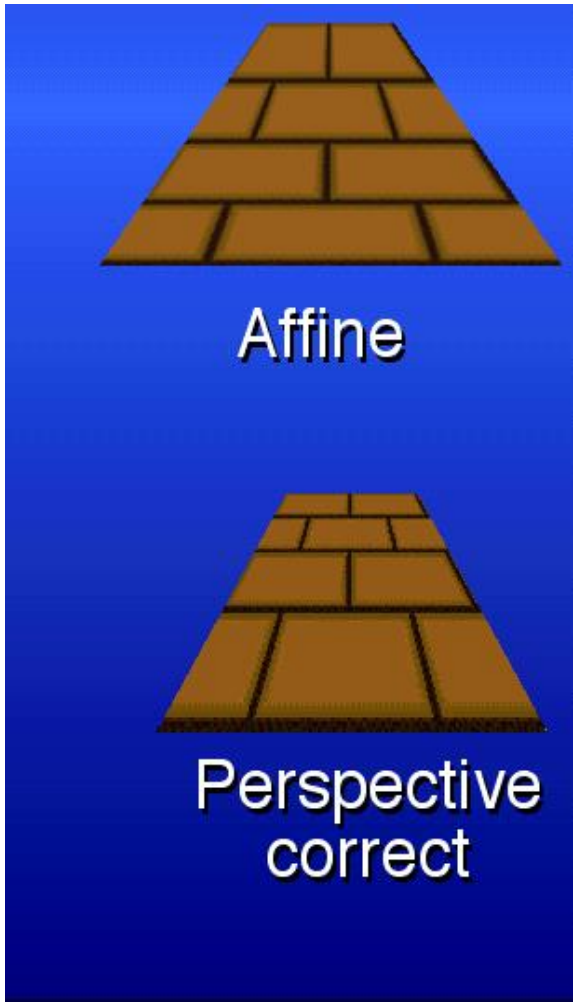
- Obavlja se u rasterizacijskoj fazi
- Naivni pristup:



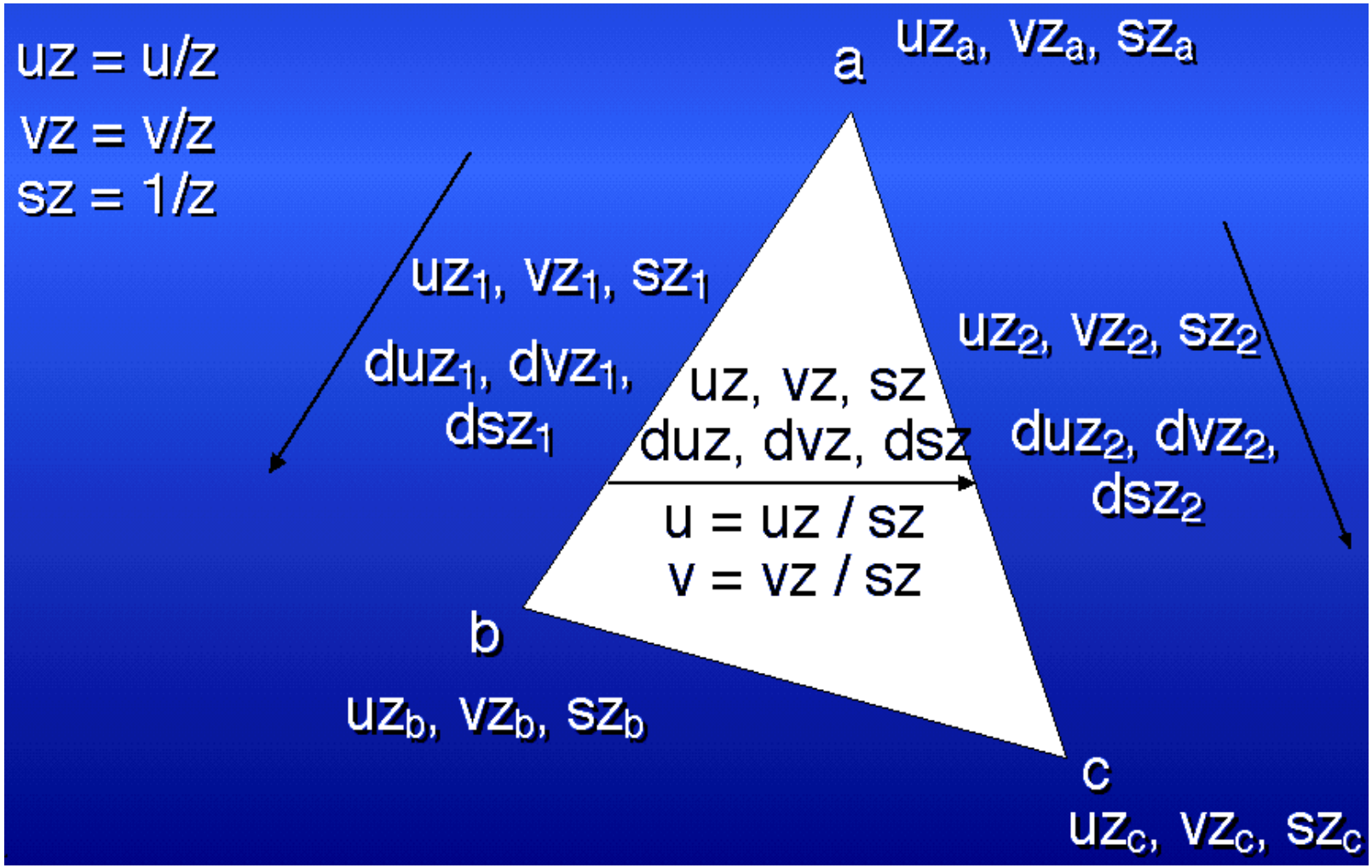
Interpolacija u,v koordinata (2/3)

- Problem: afino preslikavanje ne uzima u obzir perspektivu



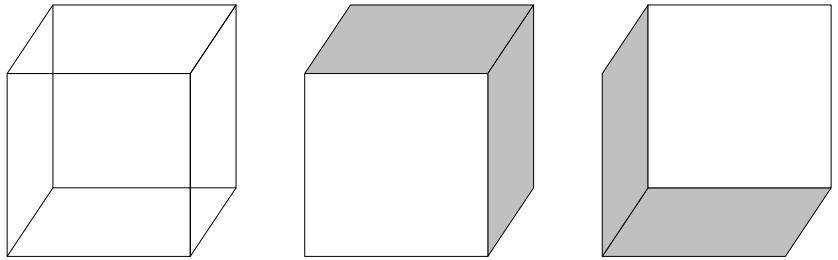


Interpolacija u,v koordinata (3/3)



Određivanje vidljivosti

- Klasičan problem 3D grafike
- Postoji niz mogućih rješenja
  - Slikarska tehnika
  - BSP stablo
  - Ray casting...



- Za realno vrijeme najpopularnija metoda Z-spremnika (Z-buffer)
  - Potrebna je dodatna memorija (Z-spremnik) u koju se sprema dubina objekta na svakoj točki ekrana
  - To je OK, jer memorija nije skupa

Metoda Z-spremnika

- Prije crtanja, sve vrijednosti u Z-spremniku postavljaju se na maksimalnu vrijednost (1.0)
- Prilikom crtanja svake točke svakog poligona:
  - Ako je dubina točke poligona koji crtamo veća od vrijednosti u Z-spremniku za točku ekrana, znači da je na ekranu već nacrtan poligon koji je bliži od ovoga kojeg upravo crtamo, dakle preskačemo
  - U protivnom upisujemo točku u spremnik boje („zaslon“), a dubinu u Z-spremnik

Prozirnost

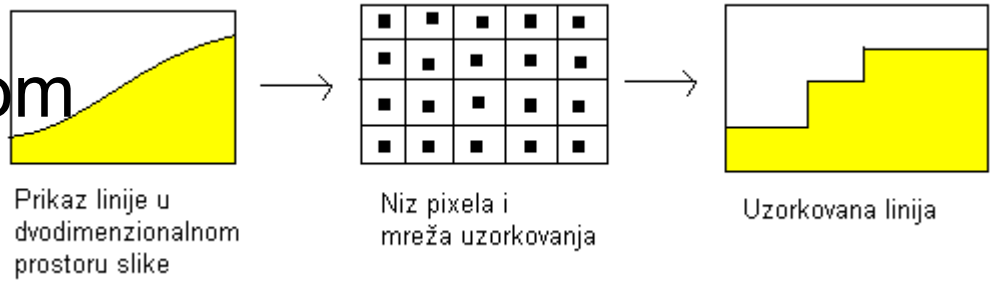
- U realnom vremenu samo ograničeni efekt
  - Nema refrakcije, apsorpcije, promjene prozirnosti u ovisnosti o kutu gledanja...
- Prozirnost u točki zadana faktorom prozirnosti α
  - $\alpha = 0.0 \rightarrow$ potpuno prozirno
  - $\alpha = 1.0 \rightarrow$ potpuno neprozirno
- α može biti zadan na 2 načina:
  1. Za čitav predmet (kao parametar materijala)
  2. Za svaku točku (korištenjem α-kanala teksture)

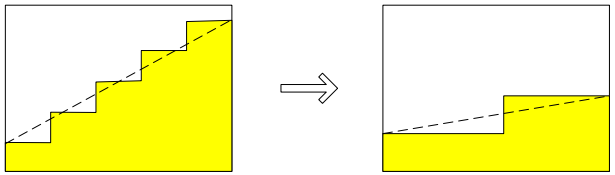
Postupak crtanja prozirne geometrije

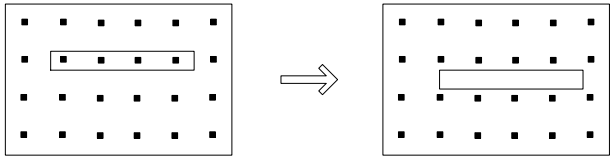
- Prvo iscrtati sve neprozirne trokute
- Iscrtati sve prozirne trokute redom od najdaljeg do najbližeg kameri
- Boja pixela se računa „over“ funkcijom:
  - $c = \alpha\, c_p + (1 - \alpha)\, c_b$
  - Cp = doprinos boje trokuta koji se trenutno crta
  - Cb = dosadašnja vrijednost piksela
  - Ova funkcija je ovisna o redoslijedu izvođenja, dakle ukoliko poligoni ne idu pravim redoslijedom nastaju pogreške

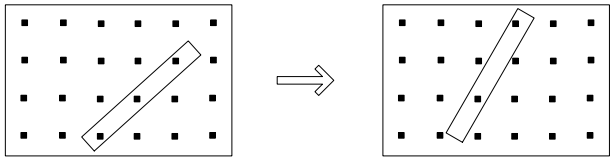
Anti-aliasing (1/2)

- Aliasing: efekt uzorkovanja nedovoljnom frekvencijom
- U grafici se javlja:
  1. Pri rasterizaciji (aliasing cijele slike)
  2. Pri uzorkovanju tekstura (aliasing teksture)
- Očituje se stepeničastim rubovima i drugim nepoželjnim efektima




Nazubljeni uzorci ruba se mijenjaju ovisno o orijentaciji linije


Dugi uski objekti mogu potpuno nestati


Dugi uski objekti se mogu nepredvidljivo slamati

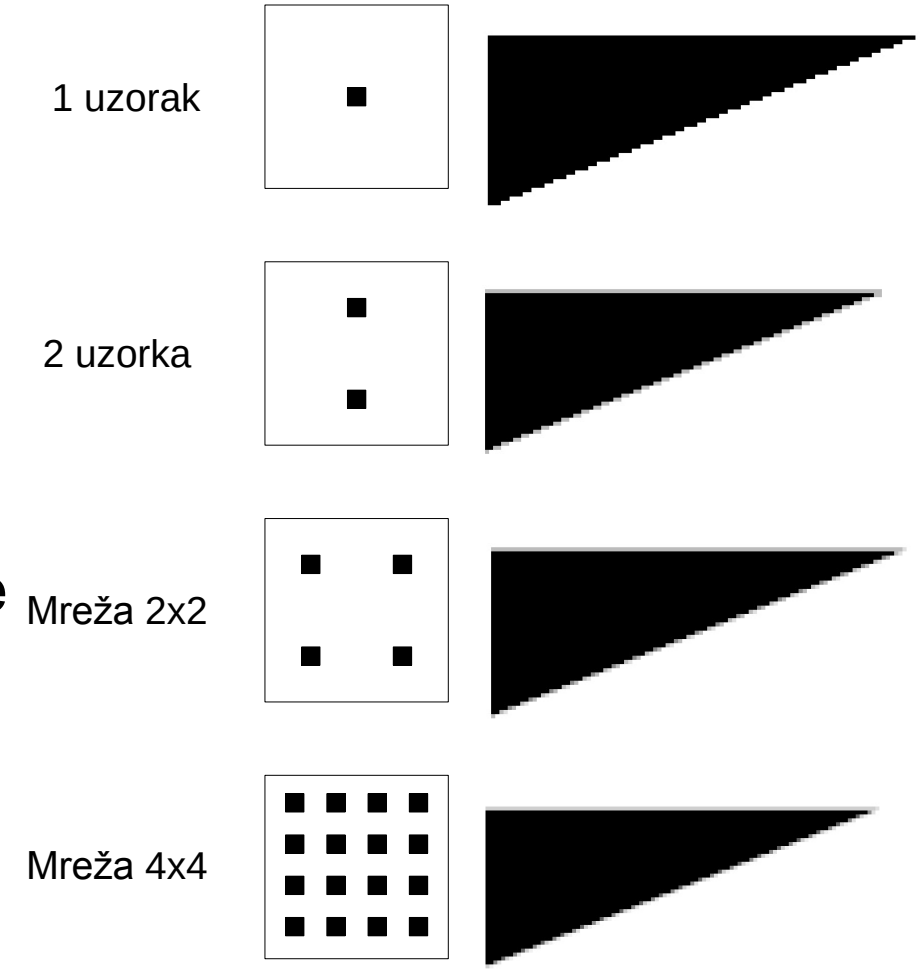
Anti-aliasing (2/2)

- Anti-aliasing – uzimanje većeg broja uzoraka i njihova interpolacija
  - U grafici to znači računanje više točaka nego što će biti prikazano, i zatim filtriranje na odgovarajuću razlučivost:

$$p(x,y) = \sum_{i=1}^{n} w_i c(i,x,y)$$

Sheme uzorkovanja

- Učinkovitost uvelike ovisi o shemi uzorkovanja (koliko se uzoraka uzima i gdje)
  - Veći broj uzoraka = bolji učinak, ali i slabije performanse

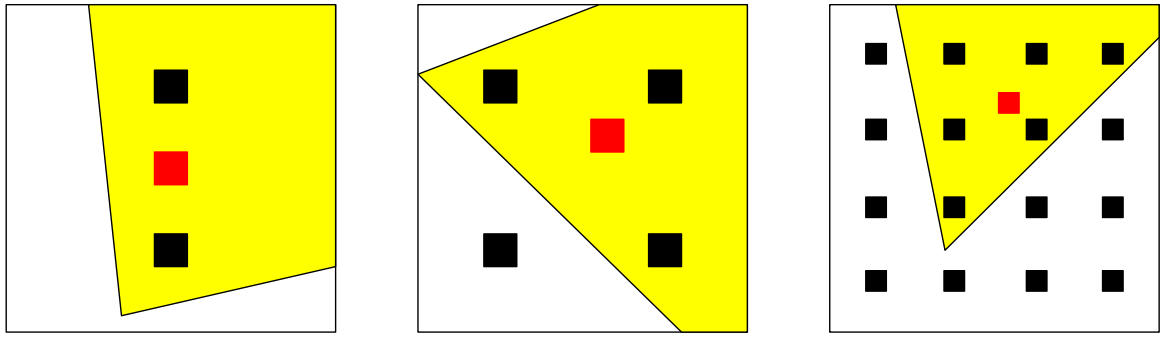


Postupci anti-aliasinga (1/2)

- Naduzorkovanje (engl. supersampling, SSAA)
  - Slika se iscrtava na Nx većoj rezoluciji, te zatim skalira
  - Svaki fragment se sjenča Nx – neučinkovito
- Akumulacijski spremnik
  - Slika se iscrtava Nx uzastopno, uz male pomake koordinata uzorkovanja
  - Uzastopne slike se miješaju (korištenjem ROP)

Postupci anti-aliasinga (2/2)

- Višestruko uzorkovanje (engl. multisample anti-aliasing, MSAA)
  - Samo koordinate se uzorkuju na većoj razlučivosti
  - Samo za jedan uzorak vrši se sjenčanje fragmenta
  - Najbolja sklopovska podrška, ali i velike potrebe za memorijom



Programska sučelja za 3D grafiku

- Veza između aplikacije i grafičkog protočnog sustava
- Podjela prema razini apstrakcije:
  - Prog. sučelja niske razine (DirectX, OpenGL)
  - Prog. sučelja visoke razine
- Podjela prema otvorenosti:
  - Vlasnički sustavi (DirectX)
  - Otvoreni sustavi (OpenGL)
  - Sustavi otvorenog kôda

Programska sučelja niske razine

- Izravno upravljanje funkcija graf. protočnog sustava
- Rade na razini vrhova, linija, trokuta
- Najpoznatija sučelja:
  - DirectX
  - OpenGL

DirectX

- Zatvoren, vlasnički sustav, samo na Windows platformi
- Primarna namjena – razvoj igara za PC i Xbox
- Objektno-orijentirano C++ sučelje, zasnovan na COM modelu
- Sve standardne funkcije protočnog sustava + pomoćne biblioteke D3DX i DXUT
- Često izdavanje novih verzija
- Od DirectX 10 nema unatražne kompatibilnosti sa starijim grafičkim sklopovljem

OpenGL (1/2)

- Otvoren, višeplatformski sustav (sve desktop platforme + inačica za mobilne i web platforme OpenGL ES)
- Pristup moguć iz raznih jezika – C, C++, Java, .NET jezici, Python, JavaScript...)
- Širok spektar primjena – razvoj igara, simulacija, znanstvene vizualizacije...

OpenGL (2/2)

- Prethodnik – vlasnički sustav IrisGL tvrtke Silicon Graphics (SGI)
- Razvojem upravlja nezavisni konzorcij OpenGL ARB
- Proceduralno sučelje, preko 250 funkcija (interno – stroj stanja)
- Pomoćne biblioteke GLU, GLUT
- Rijetko se izdaju nove verzije, no nova funkcionalnost se dodaje vrlo brzo mehanizmom proširenja

Programska sučelja visoke razine (1/2)

- Uvode koncept scene i grafa scene
  - Umjesto crtanja poligona jedan po jedan, definira se čitava scena sa predmetima, svjetlima, kamerom
  - Nakon što je definirana, scena se automatski iscrtava sa svim zadanim parametrima
- Obično postoje alati za učitavanje scene iz popularnih formata, kao i za osnovnu navigaciju
  - Osnovna vizualizacija scene s kretanjem ponuđena je praktički kao gotovo rješenje

Programska sučelja visoke razine (2/2)

- Obično koriste sučelje niske razine kao vezu prema protočnom sustavu
  - Prenosivost na razne platforme
- Optimalno iscrtavanje
  - Odbacivanje po grafu scene
  - Minimiziranje promjena stanja
  - Iznimno dobri programeri će za posebne primjene izvući bolje performanse sučeljem niske razine, no za dosta primjena se to ne isplati

Vrste sučelja visoke razine

- Sustavi za graf scene
  - Organizacija u graf scene, optimalno iscrtavanje
- Sustavi za iscrtavanje (*graphics rendering engine*)
  - Širi skup funkcija za organizaciju scene i iscrtavanje (npr. ubrzavanje, skeletalna animacija, efekti, mahanizam proširenja...)
- Pokretački sustavi za igre (*game engine*)
  - Većina funkcionalnosti potrebnih za igru (sudari, fizika, ulazni sustav, navigacija, umrežavanje...)

Primjeri sučelja visoke razine

- Sustavi za graf scene
  - IRIS Performer (povijest); OpenSceneGraph, OpenSG, Java3D
- Sustavi za iscrtavanje (*graphics rendering engine*)
  - OGRE, Irrlicht, Horde3D
- Pokretački sustavi za igre (*game engine*)
  - Unity, Torque, C4 Game Engine, Unreal Dev. Kit
  - Open source: Panda3D, Delta3D, Blender Game Engine

Grafičko sklopovlje (skraćeno)

- Arhitektura grafičkog sklopovlja
- Upravljačka jedinica videa
- Spremnici i memorija

1995.

2008.

Arhitektura grafičkog sklopovlja

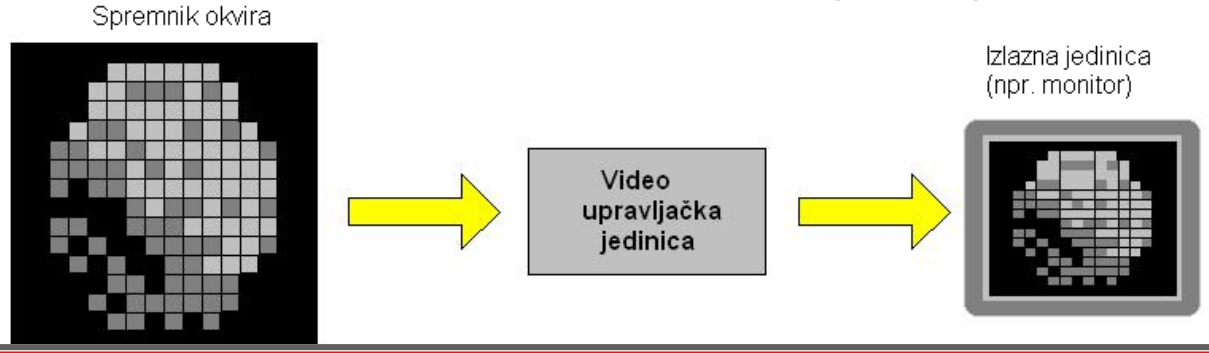
- Fizička izvedba:
  - Grafička kartica
  - Grafički čip na matičnoj ploči
- Dijelovi:
  - Grafički procesor
  - Memorija
  - Sabirnica
  - Upravljačka jedinica videa

Komunikacija s grafičkim sklopovljem

- Preko sabirnice (engl. bus)
- Kriteriji:
  - Što veća propusnost (engl. bandwidth)
  - Što manje kašnjenje (engl. latency)
- Danas se uglavnom koristi PCI Express (PCIe)
  - Propusnost varira od revizije do revizije
  - Npr. PCIe 3.0 – 1x 1 GB/s, 16x 16 GB/s

Upravljačka jedinica videa

- Protočni sustav iscrtava u spremnik boje
- Upravljačka jedinica videa – sklop koji sliku iz spremnika boje prenosi na zaslon
- Ako se koristi analogni prikaz, potrebno izvršiti digitalno-analognu konverziju (DAC)
- Iscrtavanje ovisi o tehnologiji zaslona:
  1. Katodni zasloni (CRT)
  2. Zasloni s tekućim kristalima (LCD)



Memorija

- Dijelovi:
  1. Spremnik okvira:
     I. Spremnik boje
     II. Z-spremnik
     III. Spremnik predloška
  2. Memorija za teksture
- Fizička izvedba:
  - Dio memorije sustava (npr. Intel GMA, Xbox)
  - Odvojena video memorija VRAM (npr. grafičke kartice, PS3)
  - Hibridni pristup (npr. Xbox 360) – teksture u memoriji sustava, spremnik okvira na zasebnom čipu

Spremnik boje (1/2)

- Sadrži sliku koja ide na zaslon
- Što više bitova, to više boja
- Boja visoke kvalitete (engl. high color)
  - 16 bpp ~ 64 tisuće boja
  - 5 bitova za crvenu i plavu, 6 bitova za zelenu komponentu – 32, odnosno 64 nivoa
  - Problem Machovih pruga

Spremnik boje (2/2)

- Istinska boja (engl. true color)
  - 3 ili 4 okteta po točki (24 ili 32 bpp) ~ 16.8 milijuna boja
  - 8 bitova za svaku komponentu (RGB)
  - RGBA – dodatnih 8 bitova za prozirnost (engl. alpha channel, alpha)
  - Problem Machovih pruga mnogo teže uočljiv
- Interno u graf. protočnom sustavu se koristi i veća preciznost (radi računskih operacija):
  - Današnje grafičko sklopovlje podržava spremnike visoke preciznosti (i do 32 bita po komponenti)
  - No, prikazni uređaji su uglavnom ograničeni na 8 bita, pa je potrebno izvršiti preslikavanje na manji raspon (engl. tone mapping)

Z-spremnik (Z-buffer)

- Koristi se za izračun vidljivosti
- Tipično 24 bpp
- Primjer
  - Razmak između bliske i daleke odrezujuće plohe 100 m u globalnim koordinatama
  - Uz 16 bpp, preciznost 100 m / 2^{16} ~ 1.5 mm
  - Zbog perspektivne projekcije, veća preciznost na bližim udaljenostima
- Za dobre rezultate vidljivosti treba maksimalno približiti daleku odr. plohu, i (pogotovo) maksimalno udaljiti blisku o.p.

Dvostruko spremanje

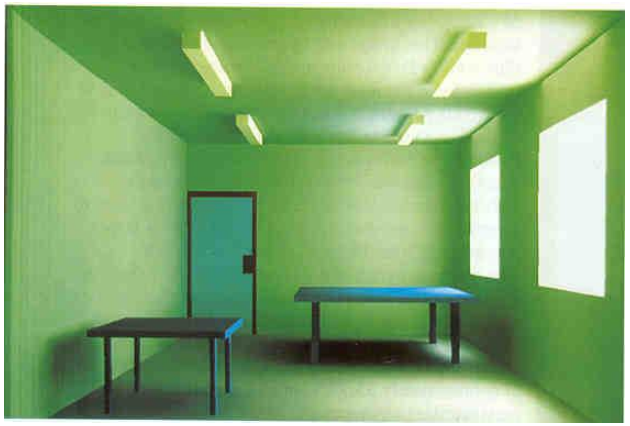
- Naivni pristup iscrtavanju
  - Rasterizator crta u spremnik boje (koji se svaki puta prethodno obriše)
  - Istovremeno video upravljačka jedinica iz spremnika crta na ekran
  - Dobivamo slike u raznim fazama dovršenosti, tj. nemirnu sliku
- Dva spremnika: prednji i stražnji (back, front)
  - Rasterizator crta u stražnji spremnik koji se ne vidi, jer v.u.j. crta na ekran uvijek prednji spremnik
  - Kada je slika spremna, izvršava se zamjena uloga spremnika (swap), tj. stražnji postaje prednji
  - U sljedećem iscrtavanju na ekran v.u.j. poslužuje novu sliku
  - Brzina iscrtavanja sinkronizira se s brzinom vert. osvježavanja v.u.j. (V-sync) – u protivnom dolazi do *deranja* (screen tearing)

Ostale metode iscrtavanja

- Isijavanje
- Iscrtavanje volumena
- Nisu u centru pažnje ovog kolegija, samo radi informiranosti

Isijavanje (engl. radiosity)

- Zasniva se na teoriji isijavanja energije
- Odlično modelira difuzne efekte svjetla
  - Efekt nezavisan od kuta gledanja: jednom izračunat rezultat može se promatrati u realnom vremenu



Vizualizacija volumena

- Reprezentacija voxel-ima →
- Prikaz uz različite stupnjeve prozirnosti pojedinih slojeva