Računarska grafika
Teksture

Vesna Marinković

Kako modelovati antičku piramidu?

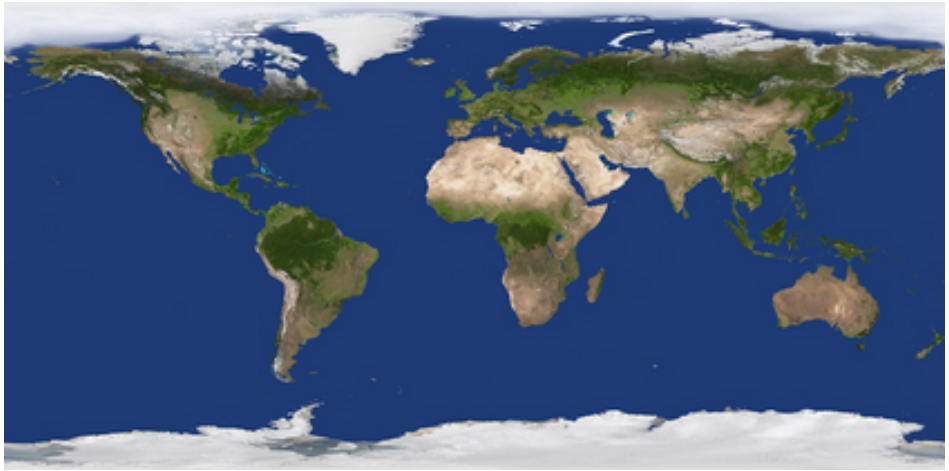
- Možemo napraviti detaljniju geometriju objekta – veći broj manjih trouglova
  - Prednost: realistično modelovanje osvetljenja
  - Mana: teško se generiše, duže se renderuje, memorijskih zahtevno
- Da li možemo da proizvedemo složene vizualne elemente bez menjanja geometrije objekta?

Motivacioni primer: modelovanje Zemlje

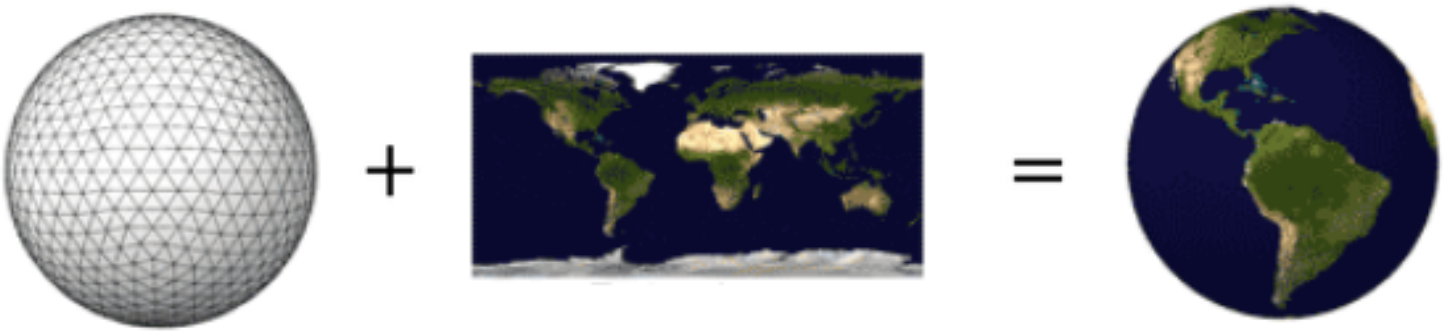
- Zemlju, gledano iz svemira, možemo aproksimirati sferom
- Ne želimo da modelujemo geometrijske detalje koji odgovaraju planinama, rekama, obalama, . . .

Preuzeto sa slajdova kursa Univerziteta Brown

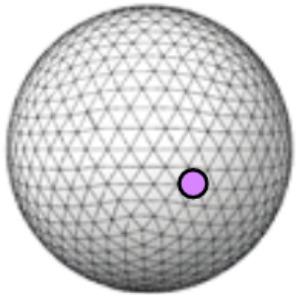
Motivacioni primer: modelovanje Zemlje



- Imamo na raspolaganju mapu Zemlje: sliku kako izgleda Zemljina površina kada se razvije u ravan
- Oko sfere možemo omotati sliku mape i dobiti željeni vizualni efekat

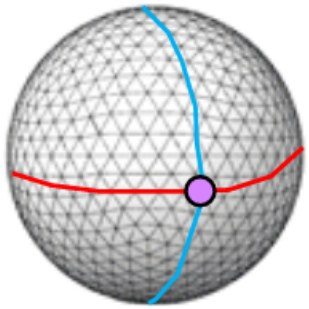


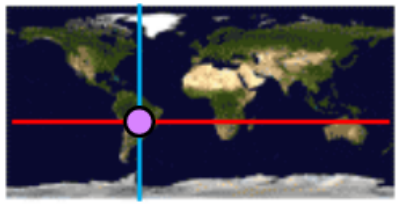
Motivacioni primer: modelovanje Zemlje



- Kako za proizvoljnu tačku na sferi odrediti kojoj tački sa mape odgovara?

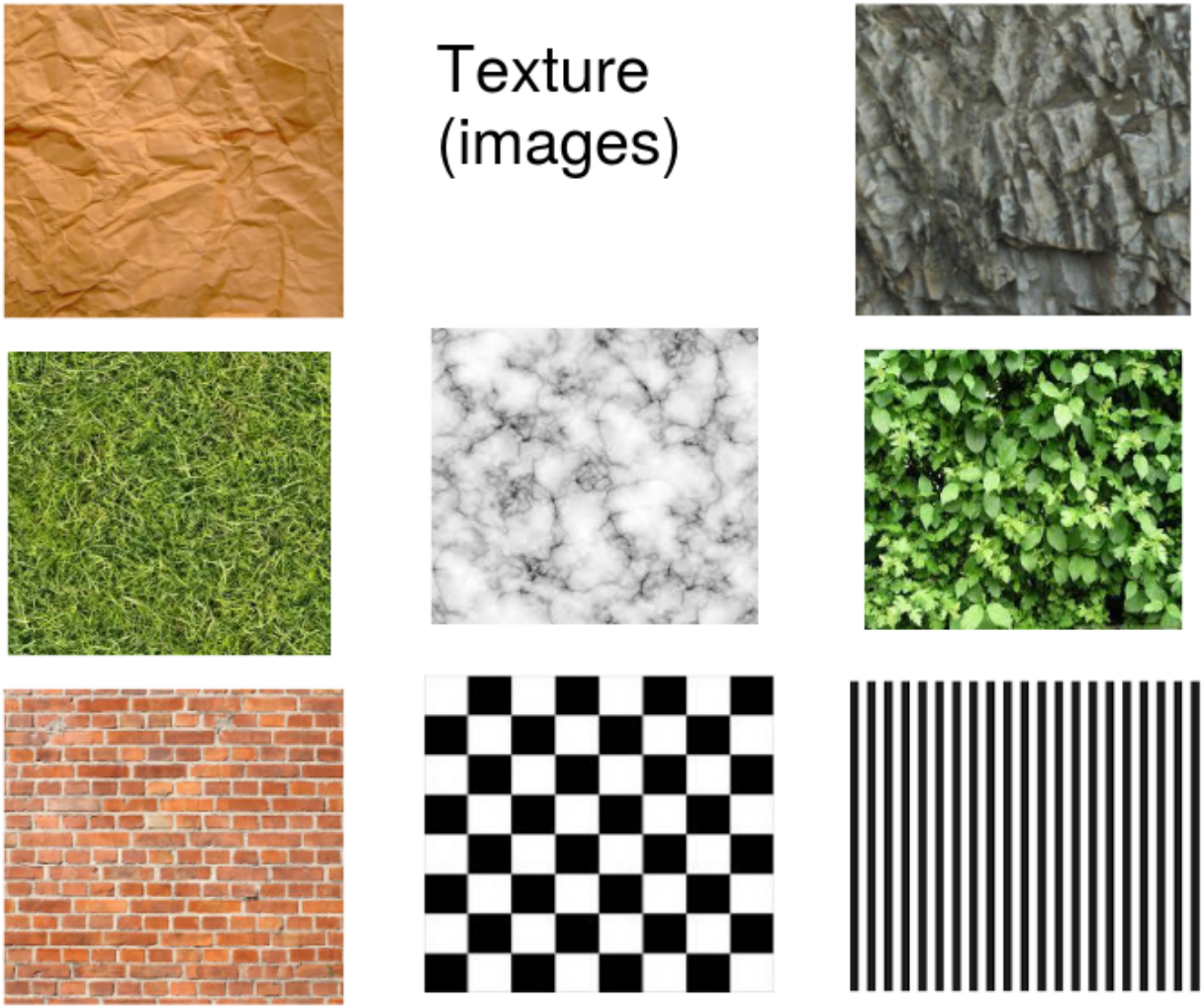
Motivacioni primer: modelovanje Zemlje





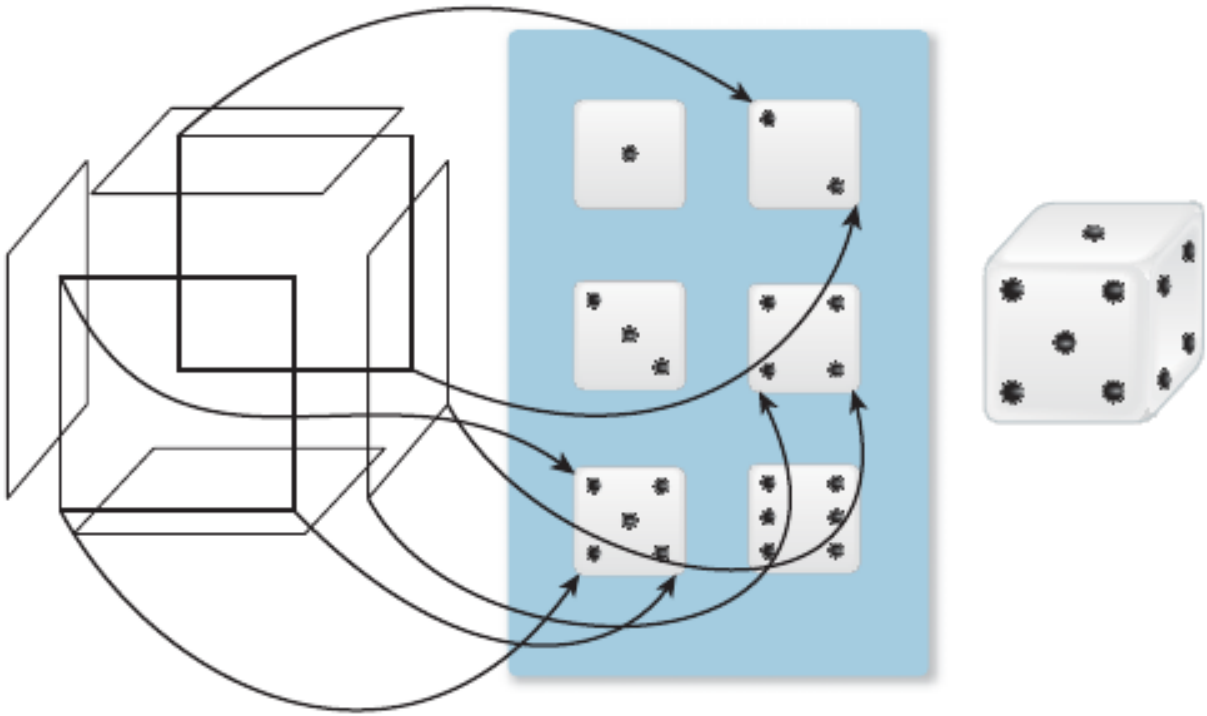
- Svaka tačka sa površine Zemlje ima svoju geografsku dužinu i geografsku širinu
- Koristimo ih za dobijanje informacije gde se na mapi nalazi odgovarajuća tačka

Teksture



Preslikavanje tekstura

- Preslikavanje (mapiranje) tekstura – svakoj tački date površi (domen) dodeljujemo vrednost sa slike teksture (kodomen) pretragom kroz sliku teksture

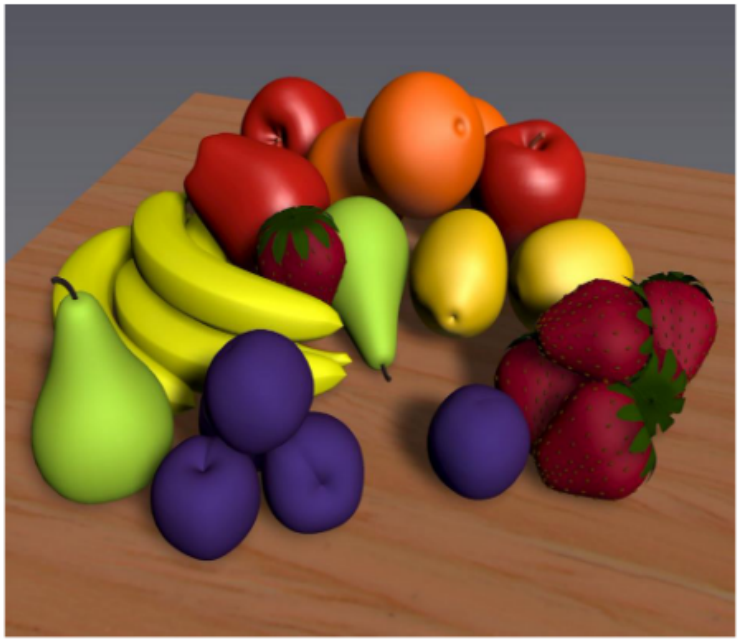


Zašto su nam potrebne teksture?

- Teksture su pogodne za prikaz:
  - grubih materijala
  - složenih scena
  - objekata koji imaju repetitivnu strukturu
- Iako grafičke kartice mogu da renderuju preko 10^7 poligona u sekundi, to nije dovoljno za pojave kao što su oblaci, trava, zemljište, koža

Zašto su nam potrebne teksture?

- Teksture popravljaju realističnost prikaza

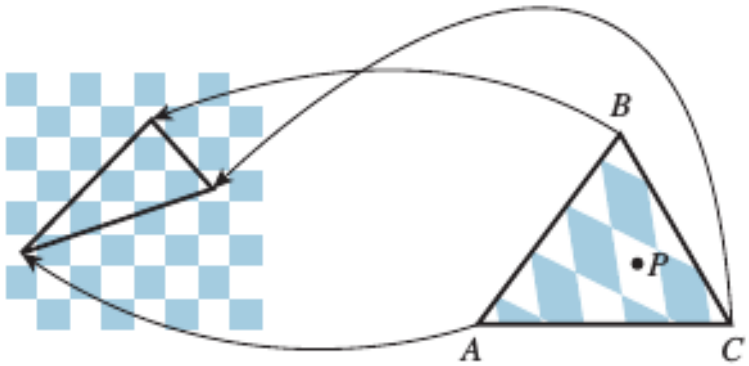


Prednosti i nedostaci preslikavanja tekstura

- Prednosti:
  - mogu se jednom sačuvati i koristiti veći broj puta
  - renderuju se jako brzo
  - posebno korisne za daleke objekte poput reljefa, neba
- Nedostaci:
  - gruba aproksimacija stvarnog života
  - površi sa nalepljenim teksturama i dalje deluju glatko jer njihova geometrija ostaje nepomenjena

Teksturne koordinate

- Mapa teksture je najčešće slika pravougaonog oblika
- Na tačke sa slike teksture referišemo korišćenjem teksturnih koordinata u i v čije su vrednosti iz intervala $[0, 1]$
- Temenima trougla pridružuju se lokacije sa mape teksture
- Boja tačke P trougla ABC određuje se pregledanjem lokacija na mapi teksture



Preslikavanje tekstura

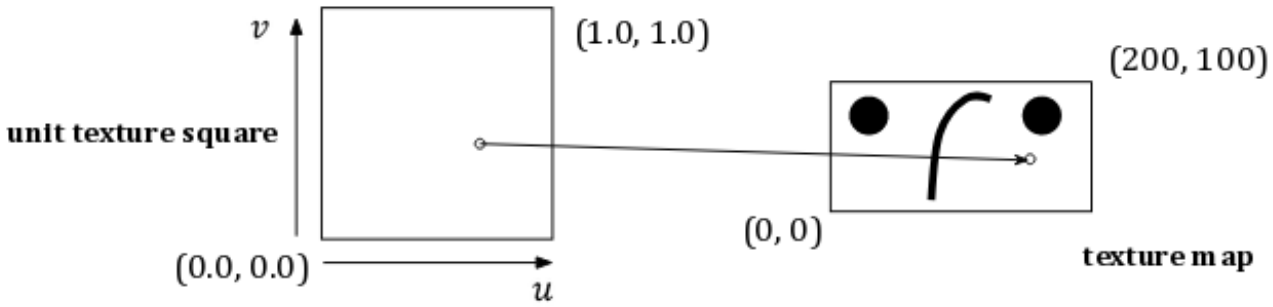
- Preslikavanje tekstura se sastoji od dve komponente:
  - Parametrizacija površi: preslikavanje 3D površi u ravan – ,,odmotavanje“ 3D objekta
    - formalno, traženje funkcije $f(x, y, z) = (u, v)$ koja opisuje površ
    - traženje vrednosti u i v za svaku tačku površi
  - Teksturisanje: dodeljivanje vrednosti svakoj tački (u, v) ravni
    - vrednosti mogu biti boje, drugi vizualni parametri (npr. eksponent spekularne refleksije), čak i normale površi
- Postoji veći broj različitih načina na koje se ove komponente mogu realizovati

Teksturisanje

- Teksturisanje je često jednostavnija komponenta procesa preslikavanja tekstura
- Razlikujemo dve vrste teksturisanja:
  - teksturisanje slikom – koordinate (u, v) se koriste za indeksiranje pozicije na slici koja sadrži teksturu
  - proceduralno teksturisanje – koordinate (u, v) su ulaz u neku funkciju kojom se računa tekstura

Teksturisanje slikom

- Koordinate (u, v) se koriste za indeksiranje pozicije na slici koja sadrži teksturu
- Preslikavanje tačke (u, v) jediničnog kvadrata na teksturu širine w i visine h
  - Odgovarajuća tačka sa mape teksture je proporcionalna po svakoj osi

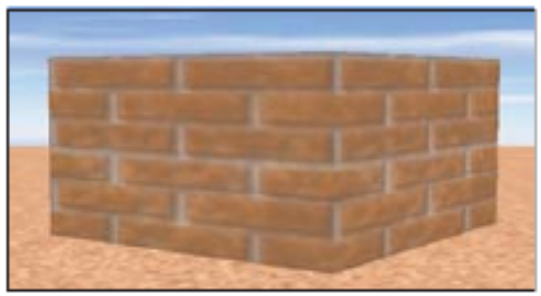
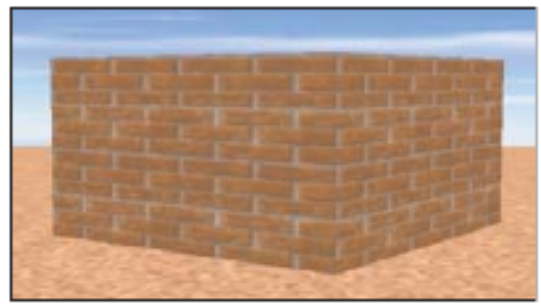


  - U ovom primeru: $(0.0, 0.0) \rightarrow (0, 0)$, $(1.0, 1.0) \rightarrow (200, 100)$, $(0.7, 0.45) \rightarrow (140, 45)$
  - Kada dobijemo koordinate sa slike teksture, dovoljno je pročitati vrednost teksture na datim koordinatama
  - Koordinate teksture nisu uvek celobrojne jer se preslikavaju iz neprekidnog u, v prostora, pa je nekada potrebno izračunati prosek susednih elemenata teksture

Nanošenje tekstura

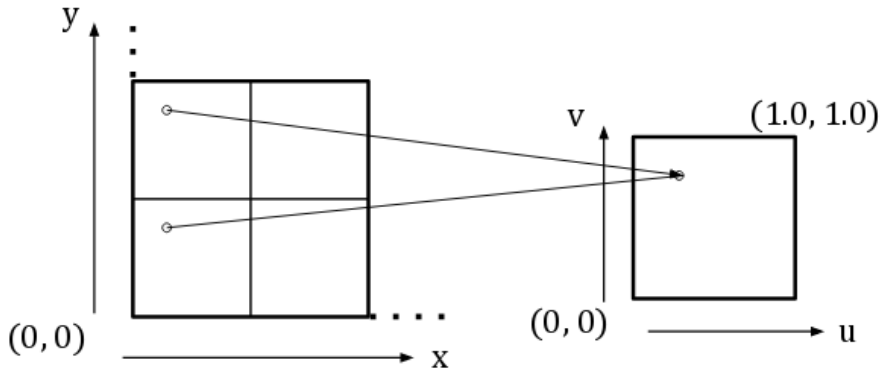
- Skaliranje – koristi se kao zamena za jako složeni model, ciljna površina se pokriva skaliranjem slike teksture


- Popločavanje – koristi se za simulaciju materijala konzistentnog izgleda, ciljna površina se pokriva ponavljanjem slike teksture

Popločavanje tekstura: implementacija

- Razmotrimo popločavanje proizvoljno velikog pravougaonika pravougaonicima dimenzije (w, h)



- Za tačku sa koordinatama (x, y) sa velikog pravougaonika koordinate (u, v) jediničnog kvadrata mogu se izračunati po formuli:

$$(u, v) = \left(\frac{x\%w}{w}, \frac{y\%h}{h}\right)$$

- Operacijom % se vrednosti (x, y) svode na opseg $[0, w - 1)$, odnosno $[0, h - 1)$, a deljenjem sa w i h na opseg $[0, 1)$

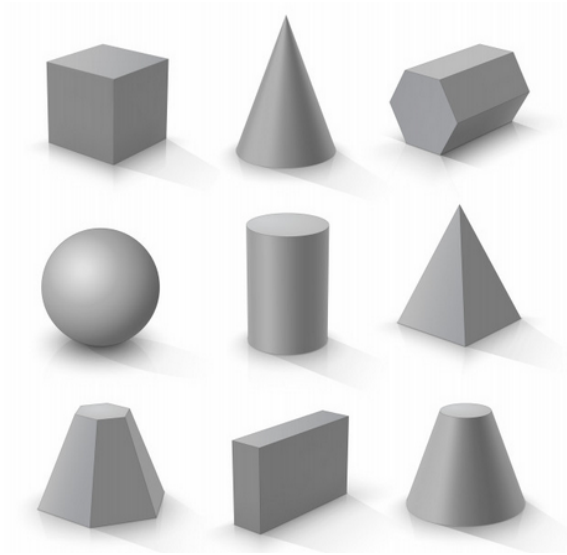
Parametrizacija površi

- Prva komponenta procesa preslikavanja tekstura
- Treba odrediti funkciju $f(x, y, z) = (u, v)$ kojom se opisuje površ
- Ovde leži najveća težina preslikavanja tekstura
- Nalaženje dobre parametrizacije, tj. one koja neće mnogo izobličiti teksturu kada se primeni na proizvoljnu površ je i dalje otvoren istraživački problem
- Mi ćemo razmatrati dva slučaja:
  - jednostavan slučaj – primitiva koju je lako parametrizovati
  - teži slučaj – parametrizacija proizvoljne mreže trouglova

Parametarsko definisanje površi

- Određene 3D primitive se prirodno definišu parametarskom funkcijom $g(u, v) = (x, y, z)$: valjak, kupa, sfera, ...
- Potrebno je odrediti tri funkcije:

$$\begin{aligned} x &= x(u, v) \\ y &= y(u, v) \\ z &= z(u, v) \end{aligned}$$



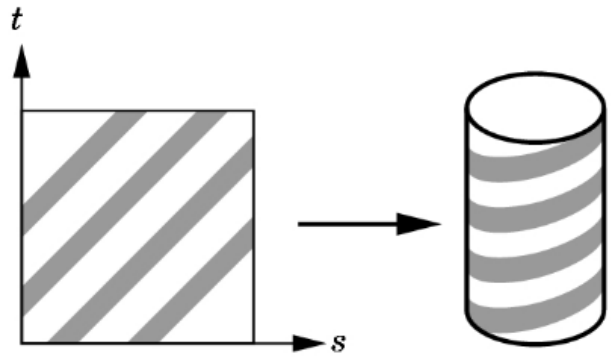
- Podsetimo se kako se parametarski mogu zadati valjak i sfera

Parametarsko zadavanje valjka

- Preslikavanje pravougaonika parametrizovanog sa u i v na omotač valjka poluprečnika r i visine h čiji je centar osnove u koordinatnom početku može se zadati kao:

$$\begin{aligned} x &= r \cdot \cos 2\pi u \\ y &= v \cdot h \\ z &= r \cdot \sin 2\pi u \end{aligned}$$

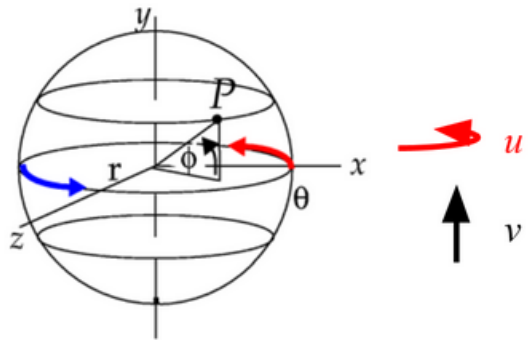
- Sa $\theta = 2\pi u$ zadat je ugao koji projekcija tačke P na ravan Oxz zaklapa sa x osom
- Koordinatom y zadaje se visina tačke P



Parametarsko zadavanje sfere

- Preslikavanje pravougaonika parametrizovanog sa u i v na sferu poluprečnika r sa centrom u koordinatnom početku može se zadati kao:

$$\begin{aligned} x &= r \cdot \sin 2\pi u \cdot \cos 2\pi v \\ y &= r \cdot \cos 2\pi u \\ z &= r \cdot \sin 2\pi u \cdot \sin 2\pi v \end{aligned}$$



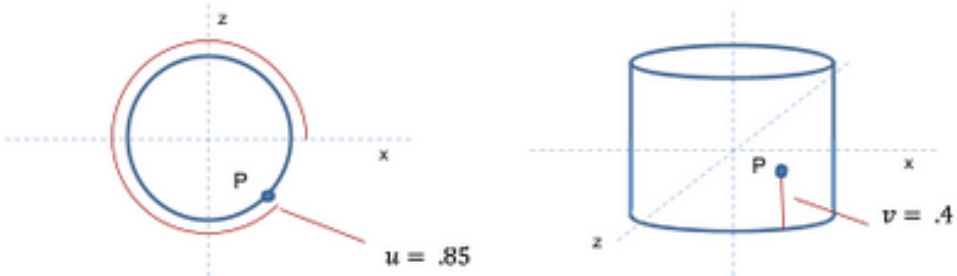
- Sa $\phi = 2\pi u$ zadat je ugao koji tačka P zaklapa sa svojom projekcijom na ravan Oxz, a sa $\theta = 2\pi v$ ugao koji ta projekcija tačke P zaklapa sa x osom
- Pri ovom preslikavanju javlja se distorzija na polovima

Preslikavanje teksture na valjak

- Nama je potreban obrnuti smer: da za datu tačku sa valjka/sfere odredimo kojoj tački teksture odgovara
- Potrebno je odrediti preslikavanje oblika:

$$\begin{aligned} u &= u(x, y, z) \\ v &= v(x, y, z) \end{aligned}$$

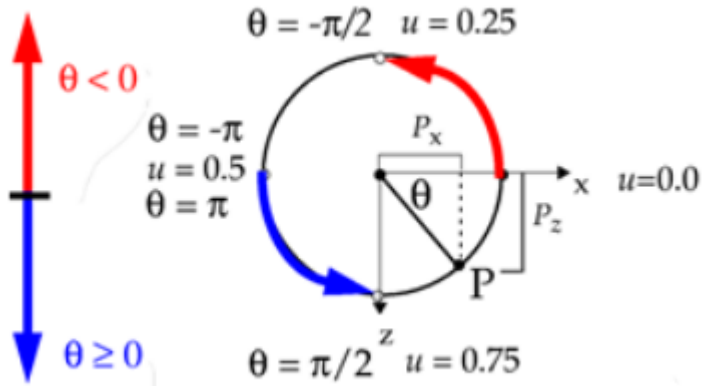
- Razmotrimo tačku P koja pripada omotaču valjku
  - koristimo poziciju tačke u odnosu na obim kruga da izračunamo koordinatu u
  - koristimo visinu tačke da izračunamo koordinatu v



Preslikavanje teksture na valjak (2)

- Koordinatu v dobijamo dodavanjem vrednosti 0.5 na y koordinatu: opseg $[-0.5, 0.5]$ slikamo u $[0, 1]$
- Treba preslikati sve tačke sa kruga u opseg $[0, 1]$, u CCW smeru
- Koordinatu u dobijamo kao procenat obima koji je tačka P obišla od $z = 0$
- Najjednostavnije je izračunati $\theta = \arctan \frac{z}{x}$, a zatim $u = \frac{\theta}{2\pi}$, međutim $\arctan \frac{z}{x}$ daje vrednost $\theta \in (-\frac{\pi}{2}, \frac{\pi}{2})$ i preslikava dve dijametralno suprotne tačke na krugu u isti ugao θ
- Problem se rešava razmatranjem slučajeva:

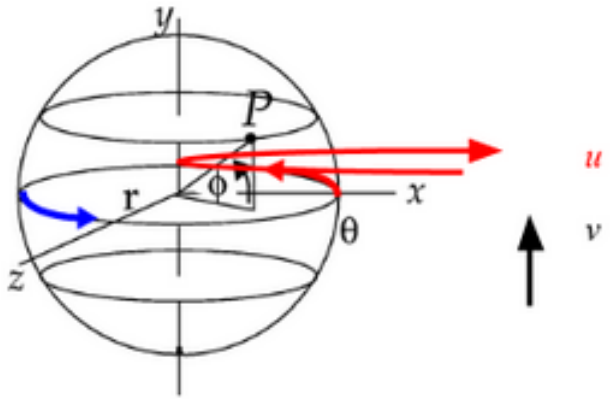
$$
\begin{aligned}
\theta &= \arctan(z/x) \\
u &= \begin{cases} \frac{\theta}{2\pi} \text{ za } \theta < 0 & [0.0, 0.5] \\ 1 - \frac{\theta}{2\pi} \text{ za } \theta \geq 0 & [0.5, 1] \end{cases} \\
v &= y + 1/2
\end{aligned}
$$



Preslikavanje teksture na sferu

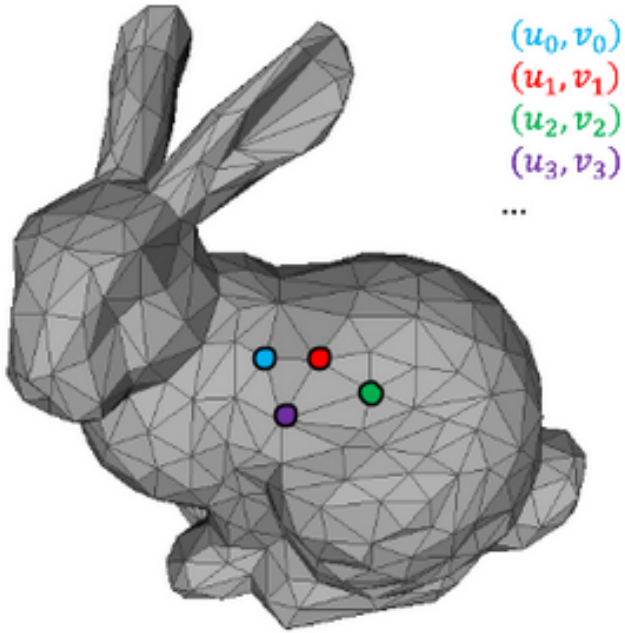
- Koordinatu u dobijamo isto kao kod valjka: kao procenat obima koji je tačka P obišla od $z = 0$
- Koordinatu v dobijamo kao funkciju geografske širine ϕ: računamo ugao ϕ koji tačka P zahvata sa ravni Oxz
- Na polovima, za $v = 0$ i $v = 1$ javljaju se singularnosti i u se postavlja na neku predefinisanu vrednost, npr. $u = 0.5$

$$
\begin{aligned}
\theta &= \arctan \frac{z}{x} \\
u &= \begin{cases} \frac{\theta}{2\pi} \text{ za } \theta < 0 \\ 1 - \frac{\theta}{2\pi} \text{ za } \theta \geq 0 \end{cases} \\
\phi &= \arcsin \frac{y}{r}, \quad -\frac{\pi}{2} \leq \phi \leq \frac{\pi}{2} \\
v &= \frac{\phi}{\pi} + \frac{1}{2}
\end{aligned}
$$



Preslikavanje tekstura u slučaju proizvoljne mreže trouglova

- U opštem slučaju ovakva preslikavanja nije lako odrediti
- Za proizvoljnu mrežu trouglova, nemamo pristup nekoj finoj funkciji $g(u, v) = (x, y, z)$ koja definiše oblik
- Umesto toga, cilj je definisati (u, v) koordinate za svako teme mreže
- Videćemo uskoro da ako to znamo, lako je odrediti (u, v) koordinate za svaku tačku mreže

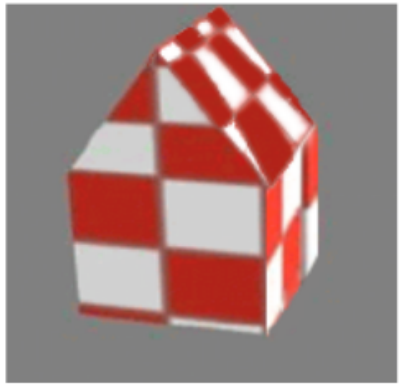


Preslikavanje složenih geometrijskih tela (1. način)

- Svaku poligonalnu stranu možemo nezavisno teksturisati
- Ovakav vid preslikavanja može rezultovati ružnim spojevima tamo gde se strane sustiču

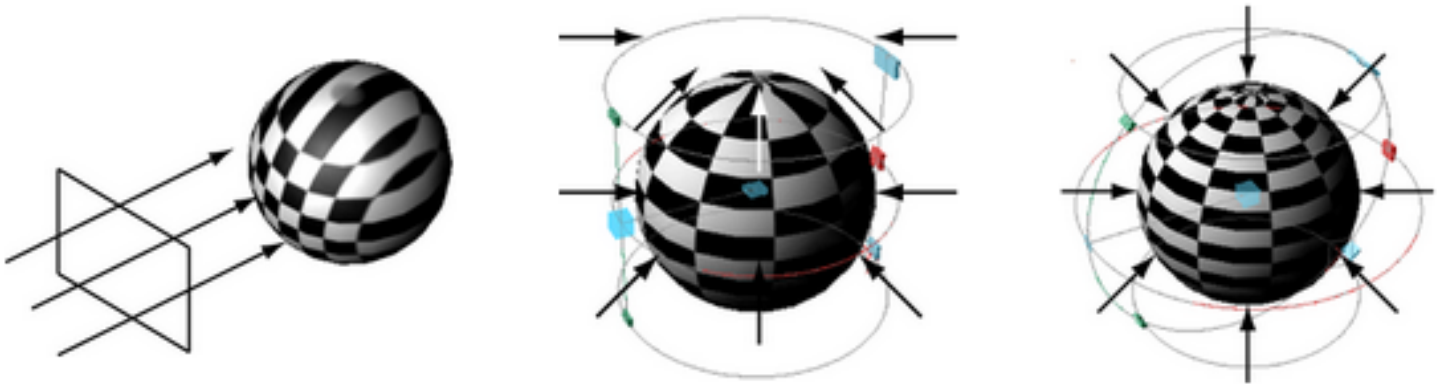
Input shape

Independent face textures



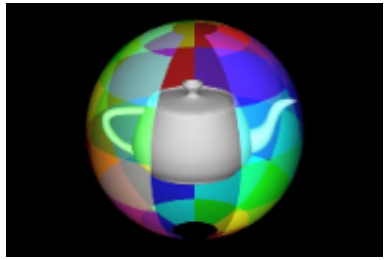
Preslikavanje složenih geometrijskih tela (2. način)

- Ukoliko je objekat približno ravanski, cilindrični ili sferni možemo koristiti formule za parametrizaciju ovih primitivnih oblika

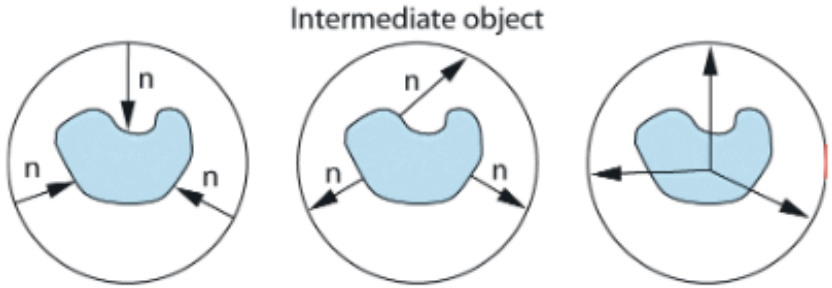


Preslikavanje složenih geometrijskih tela (2. način)

- Oko objekta se opisuje granični opseg



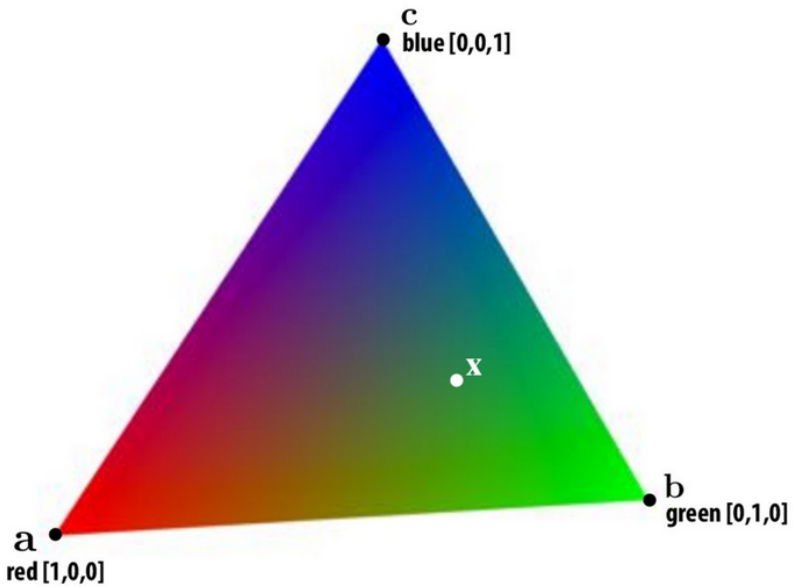
- Izvodimo dvostepeno preslikavanje u dva koraka:
  - vrši se preslikavanje teksture na granični opseg
  - vrši se preslikavanje sa graničnog opsega na ciljni objekat
    - određuje se presek normala graničnog opsega sa ciljnom površi
    - određuje se presek normala ciljne površi sa graničnim opsegom
    - određuje se presek vektora iz centra ciljne površi sa graničnim opsegom



Ručno preslikavanje tekstura

- Objekti sa kojima se radi u video-igrama i na filmu su, po pravilu, složeniji od prethodno razmatranih
- Potrebna nam je precizna kontrola kako slika teksture izgleda na objektu
- Iz tog razloga se za preslikavanje tekstura koriste programi za 3D modelovanje kao Blender, Maya,...

Uzorkovanje boje u tački trougla

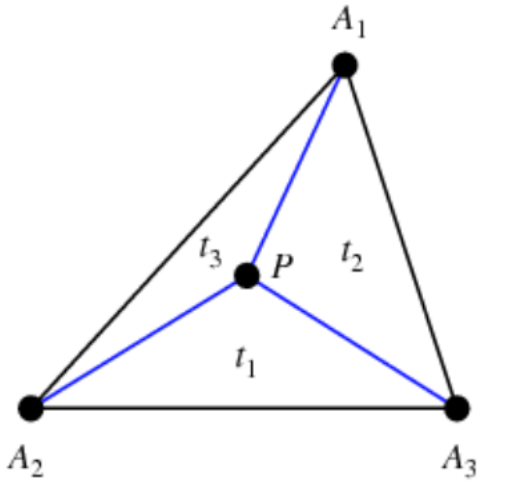


- Kako odrediti boju u tački x u unutrašnjosti trougla?
- Potrebno je izvršiti interpolaciju vrednosti boje u temenima
- Interpolacija u trouglu se vrši korišćenjem baricentričnih koordinata
  - najpre definišemo tačku trougla u terminima njegova tri temena
  - koristimo ovu reprezentaciju da izračunamo boju te tačke

Linearna interpolacija u 1D

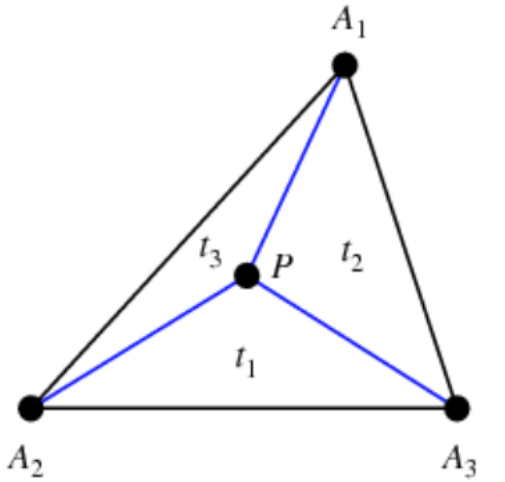
- Neka je data vrednost neke funkcije f u tačkama A_1 i A_2 i želimo da nađemo vrednost funkcije f u proizvoljnoj tački Q duži A_1A_2
- Proizvoljnu tačku Q duži A_1A_2 možemo parametarski predstaviti kao $Q = (1-t)A_1 + tA_2$, gde je $0 \leq t \leq 1$
- Drugačije rečeno važi: $Q = t_1A_1 + t_2A_2,\ t_1 + t_2 = 1,\ t_1, t_2 \geq 0$
- t_1 i t_2 su baricentrične koordinate duži A_1A_2
- Duž je konveksna linearna kombinacija njenih krajnjih tačaka
- Parametre t_1 i t_2 možemo videti kao težine, pa je duž težinski prosek krajnjih tačaka duži
- Vrednost proizvoljnog svojstva koje se linearno menja duž duži A_1A_2 može se izraziti kao
  $f(Q) = t_1 \cdot f(A_1) + t_2 \cdot f(A_2),\ t_1 + t_2 = 1,\ t_1, t_2 \geq 0$

Cilj baricentričnih koordinata



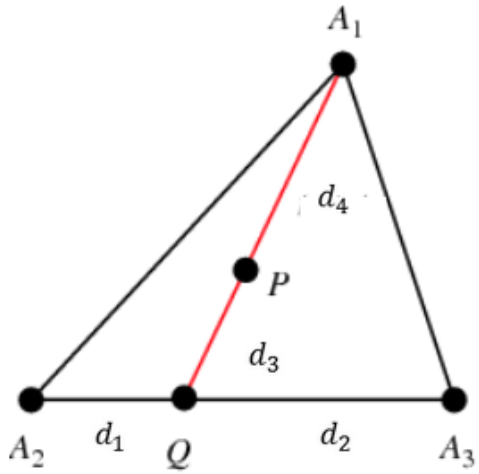
- Baricentrične koordinate se mogu uopštiti na trougao
- Neka je dat $\triangle A_1A_2A_3$ i proizvoljna tačka P trougla $A_1A_2A_3$
- Koje težine treba staviti na svako od temena trougla tako da trougao bude savršeno izbalansiran u datoj tački?

Cilj baricentričnih koordinata



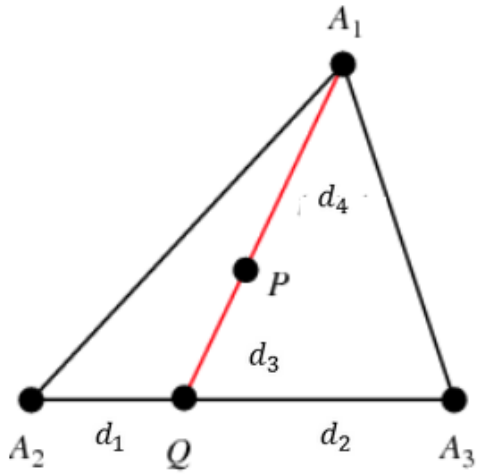
- Baricentrične koordinate se mogu uopštiti na trougao
- Neka je dat $\triangle A_1A_2A_3$ i proizvoljna tačka P trougla $A_1A_2A_3$
- Koje težine treba staviti na svako od temena trougla tako da trougao bude savršeno izbalansiran u datoj tački?
- Što je tačka P bliža temenu A_1 uticaj tačke A_1 je veći, a što je udaljenija od temena A_1 njen doprinos je manji

Linearna interpolacija u 2D



- Neka je Q presek prave A_1P i prave A_2A_3
- Zapišimo tačku Q kao težinsku sumu tačaka A_2 i A_3, a tačku P kao težinsku sumu tačaka Q i A_1

Linearna interpolacija u 2D

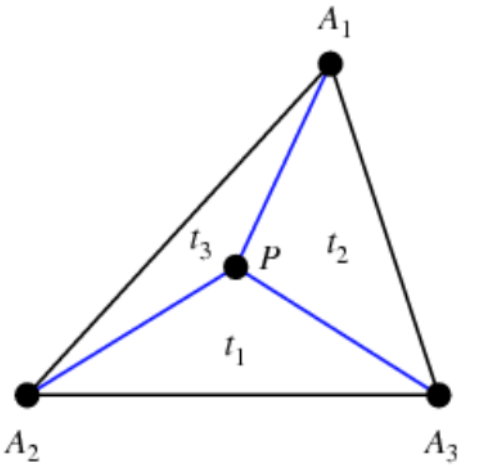


- Neka je Q presek prave A_1P i prave A_2A_3
- Zapišimo tačku Q kao težinsku sumu tačaka A_2 i A_3, a tačku P kao težinsku sumu tačaka Q i A_1

  $Q = (1-t)A_2 + tA_3, \quad t = d_1/(d_1+d_2)$
  $P = (1-s)Q + sA_1, \quad s = d_3/(d_3+d_4)$
- Zamenom prve jednačine u drugu dobijamo

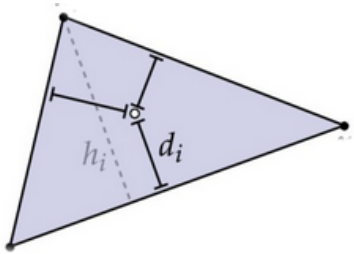
  $P = sA_1 + (1-s)(1-t)A_2 + (1-s)tA_3$
- Koeficijenti uz A_1, A_2 i A_3 su nenegativni i njihova suma jednaka je 1

Linearna interpolacija u 2D



- Tačka P trougla $A_1A_2A_3$ može se predstaviti kao $t_1A_1 + t_2A_2 + t_3A_3$, gde je $t_1 + t_2 + t_3 = 1$ i $t_1, t_2, t_3 \geq 0$
- Brojeve t_1, t_2 i t_3 nazivamo baricentričnim koordinatama tačke P u odnosu na $\triangle A_1A_2A_3$
- Teksturne koordinate u i v tačke P možemo na ovaj način odrediti na osnovu u i v koordinata tačaka A_1, A_2 i A_3

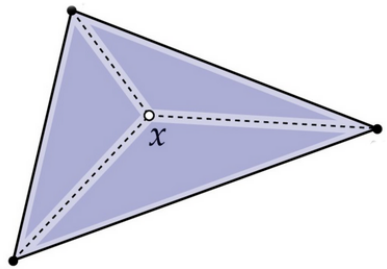
Interpretacija baricentričnih koordinata



- Baricentrične koordinate tačke P možemo da tumačimo i putem rastojanja tačke od stranica trougla:
  - težina uz teme A_1 treba da bude proporcionalna rastojanju tačke P do stranice A_2A_3
- Baricentrične koordinate tačke P dobijamo tako što rastojanja tačke P do stranica A_1A_2, A_2A_3 i A_3A_1 podelimo dužinama visina trouglova iz temena A_3, A_1 i A_2 redom:

$$t_i = \frac{d_i}{h_i}$$

Interpretacija baricentričnih koordinata

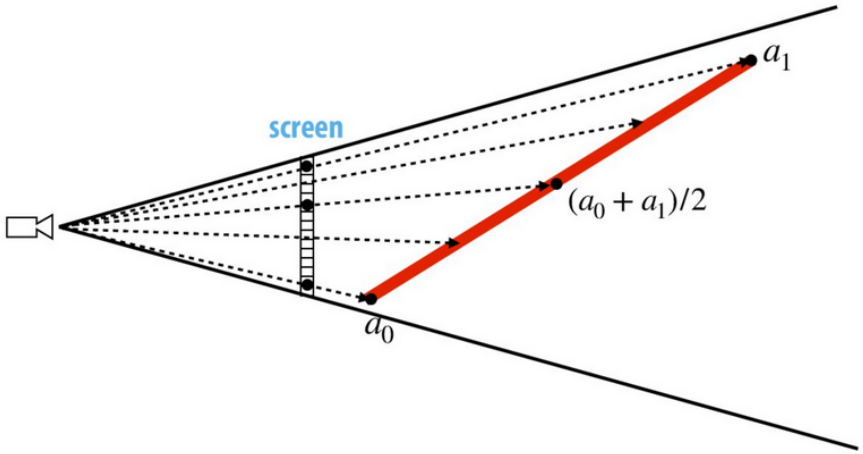


- Baricentrične koordinate tačke P možemo da tumačimo i putem površina trouglova:
  - težina uz teme A_1 treba da bude proporcionalna površini $\triangle PA_2A_3$
- Baricentrične koordinate tačke P dobijamo tako što površine $\triangle PA_1A_2$, $\triangle PA_2A_3$ i $\triangle PA_3A_1$ podelimo površinom $\triangle A_1A_2A_3$:

$$t_i = \frac{p(PA_1A_2)}{p(A_1A_2A_3)}$$

Perspektivno nekorektna interpolacija

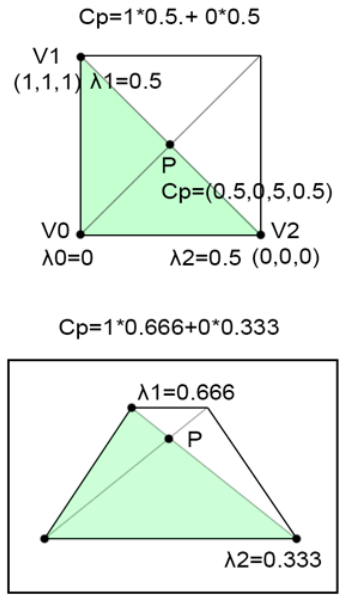
- Neka je cilj interpolirati neku vrednost nad trouglom u 3D
- Ako smo izračunali projekcije temena trougla i vršimo interpolaciju vrednosti u temenima trougla korišćenjem baricentričnih koordinata dobijenih za 2D trougao, ne radimo ispravnu interpolaciju vrednosti za 3D trougao
- Naime, zbog perspektivne projekcije, baricentrična interpolacija vrednosti na trouglu sa različitim dubinama ne daje dobre rezultate



- Želimo da interpoliramo vrednosti atributa linearno u 3D prostoru objekta, a ne u prostoru slike!

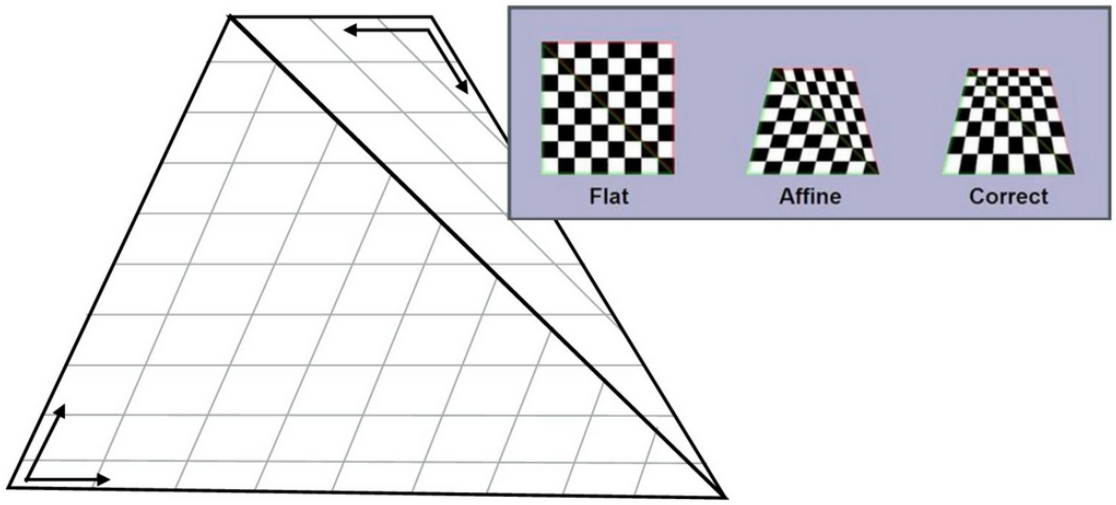
Razlika u baricentričnim koordinatama u 3D prostoru objekta i u prostoru ekrana

- Neka je dat kvadrat u 3D prostoru koji nije paralelan ravni projekcije i neka se on projektuje u paralelogram prikazan na donjoj slici
- Razmotrimo položaj tačke P u 3D prostoru objekta i 2D prostoru projekcije
- U kvadratu važi: $P = 0.5 \cdot V_1 + 0.5 \cdot V_2$
- U paralelogramu važi: $P' = 0.666 \cdot V_1' + 0.333 \cdot V_2'$



Primer: perspektivno nekorektna interpolacija

- Razmotrimo kvadrat u 3D prostoru koji se projektuje u četvorougao na slici
- Delimo ga na dva trougla koja treba teksturisati
- Ako preslikamo teksture korišćenjem baricentričnih koordinata za 2D (projektovane) koordinate, gubimo neprekidnost u interpolaciji na mestu gde je četvorougao izdeljen na trouglove



Perspektivno korektna interpolacija

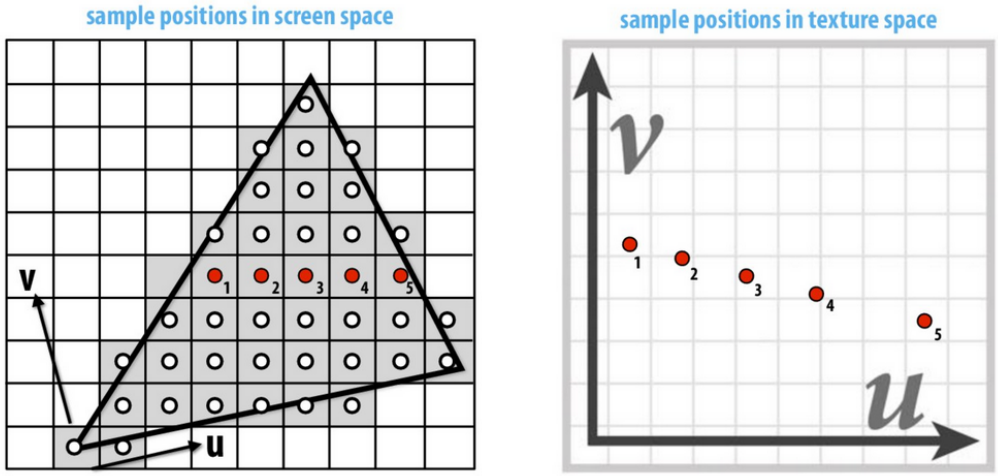
- Cilj: interpolirati neke atribute f date u temenima trougla
- Osnovni recept:
  - izračunati dubinu z u svakom od temena
  - izračunati vrednost $Z = 1/z$ i $g = f/z$ u svakom od temena
  - sada možemo izvršiti interpolaciju vrednosti Z i g korišćenjem standardnih (2D) baricentričnih koordinata
  - u svakom od fragmenata, podeliti interpolisanu vrednost funkcije g interpolisanom vrednošću funkcije Z da bismo dobili finalnu vrednost funkcije f fragmenta

Više detalja i izvođenja dostupni su na:

- `https://www.scratchapixel.com/lessons/3d-basic-rendering/rasterization-practical-implementation/visibility-problem-depth-buffer-depth-interpolation`
- `https://www.scratchapixel.com/lessons/3d-basic-rendering/rasterization-practical-implementation/perspective-correct-interpolation-vertex-attributes`

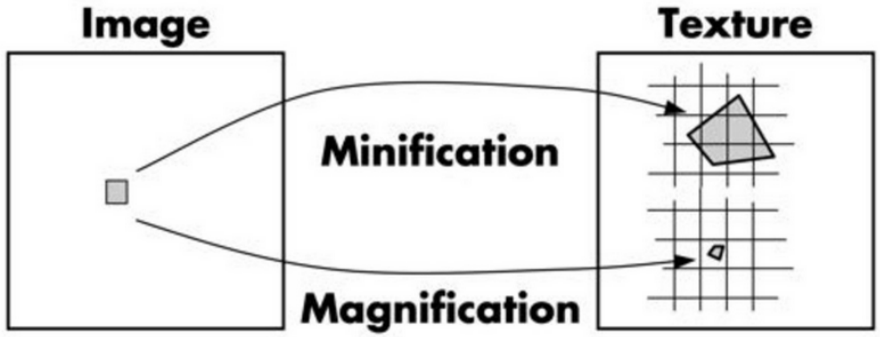
Vizualizacija uzoraka teksture

- Trouglovi se projektuju iz 3D prostora u ravan, pa će pikseli u prostoru ekrana odgovarati regionima promenljive veličine i lokacije na teksturi
- Lokacije u kojima vršimo uzorkovanje u prostoru ekrana imaju ravnomernu raspodelu
- Međutim, lokacije u kojima vršimo uzorkovanje u prostoru tekstura nemaju ravnomernu raspodelu



- Nepravilan obrazac uzorkovanja rezultuje alijasingom

Uvećanje vs. umanjenje teksture



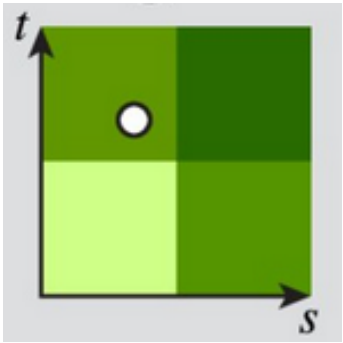
- Uvećanje teksture (magnifikacija) – jednostavnije:
  - npr. kada je kamera jako blizu objektu sa scene
  - pojedinačni piksel ekrana se preslikava u veoma mali region teksture
  - može se interpolacijom dobiti vrednost u centru piksela ekrana
- Umanjenje teksture (minifikacija) – teže:
  - npr. kada je objekat jako daleko
  - pojedinačni piksel ekrana se preslikava u veliki region teksture
  - potrebno je izračunati prosečnu vrednost dela teksture pokrivenog pikselom da bismo izbegli aliasing efekat

Uvećanje teksture (1. pristup)

- Piksel ekrana se preslikava u mali region teksture
- Kako naći vrednost teksture u necelobrojnoj lokaciji (u, v)?

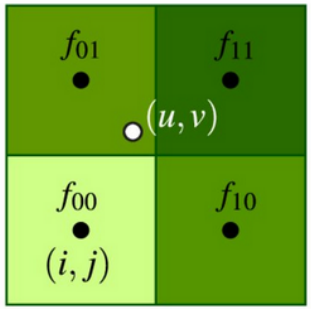
Uvećanje teksture (1. pristup)

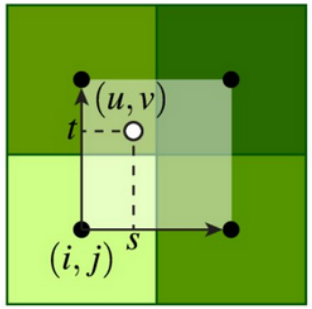
- Piksel ekrana se preslikava u mali region teksture
- Kako naći vrednost teksture u necelobrojnoj lokaciji (u, v)?
- Prvi pristup: metoda najbližeg suseda – pokupimo vrednost najbližeg piksela teksture
- Efikasno, ali ružno



Uvećanje teksture (2. pristup)

- Drugi pristup: bilinearna interpolacija – najpre vršimo linearnu interpolaciju po horizontali, pa po vertikali





$$i = \lfloor u - \tfrac{1}{2} \rfloor \qquad s = u - \left(i + \tfrac{1}{2}\right) \in [0,1]$$

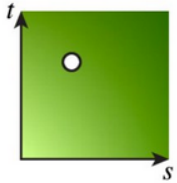
$$j = \lfloor v - \tfrac{1}{2} \rfloor \qquad t = v - \left(j + \tfrac{1}{2}\right) \in [0,1]$$

linear (each row)

$$(1-s)f_{01} + sf_{11}$$

$$(1-s)f_{00} + sf_{10}$$

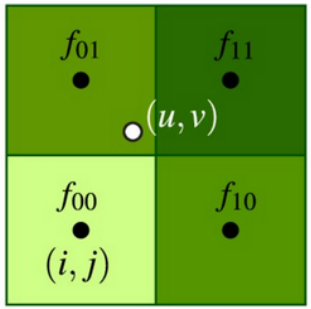
bilinear

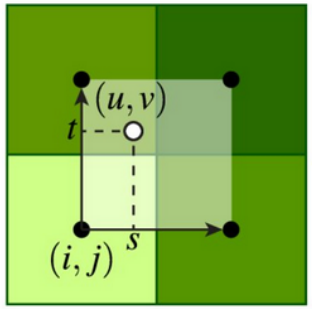


$$(1-t)\left((1-s)f_{00} + sf_{10}\right) + t\left((1-s)f_{01} + sf_{11}\right)$$

Uvećanje teksture (2. pristup)

- Drugi pristup: bilinearna interpolacija – najpre vršimo linearnu interpolaciju po horizontali, pa po vertikali





$$i = \lfloor u - \tfrac{1}{2} \rfloor \qquad s = u - \left(i + \tfrac{1}{2}\right) \in [0,1]$$

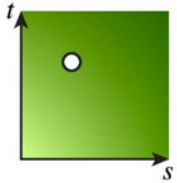
$$j = \lfloor v - \tfrac{1}{2} \rfloor \qquad t = v - \left(j + \tfrac{1}{2}\right) \in [0,1]$$

linear (each row)

$$(1-s)f_{01} + sf_{11}$$

$$(1-s)f_{00} + sf_{10}$$

bilinear

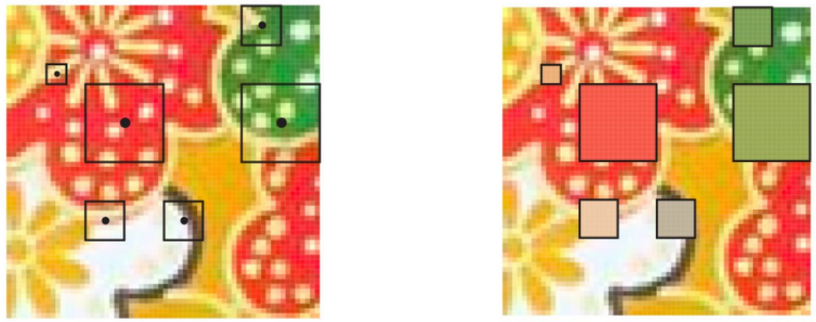


$$(1-t)\left((1-s)f_{00} + sf_{10}\right) + t\left((1-s)f_{01} + sf_{11}\right)$$

- Da li bismo dobili istu vrednost da smo prvo vršili interpolaciju po vertikali, pa po horizontali?

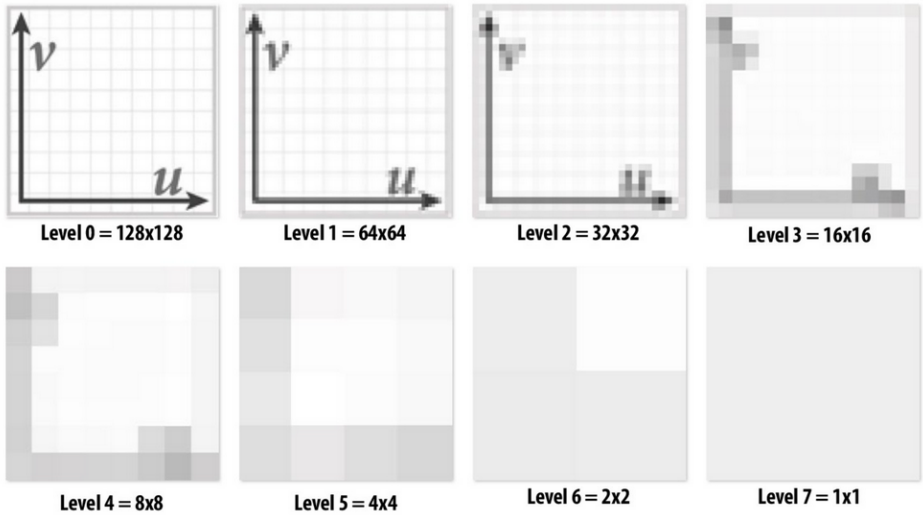
Umanjenje teksture

- Piksel ekrana se preslikava u veliki region teksture
- Ako bismo samo uzeli vrednost teksture u centru piksela, dobili bismo teksturni aliasing (boju koja se menja ako se uzorak vrlo malo pomeri)
- Idealno bi bilo uzeti prosečnu vrednost teksture, ali je ovo skupo izračunati



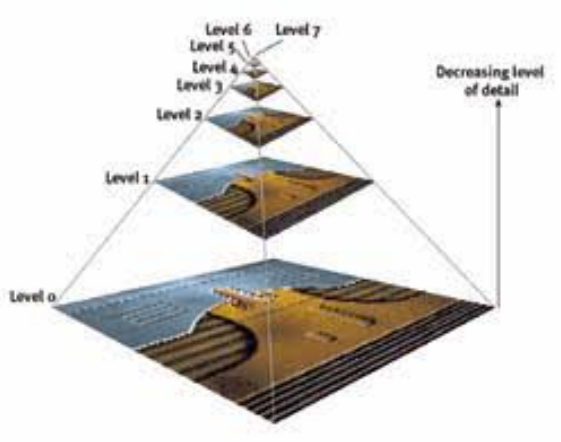
- Umesto toga, možemo koristiti prefiltriranje: unapred izračunati proseke (jednom) i u vreme izvršavanja (veliki broj puta) koristiti dobijene vrednosti
- Koje proseke treba čuvati? Ne možemo ih sve unapred izračunati
- Za objekte koji su blizu trebaju nam slike visoke rezolucije, ali su one neefikasne za udaljene objekte

MIP mape



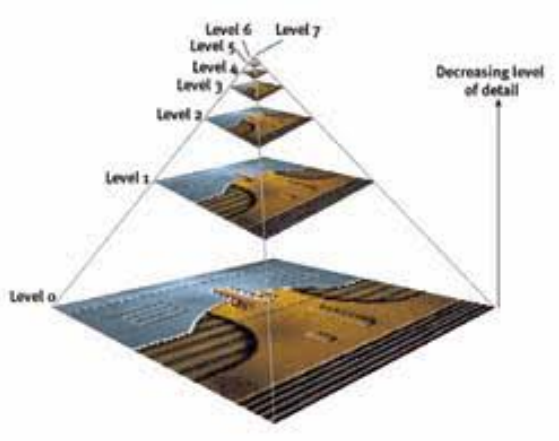
- MIP mape (lat. multum in parvo, eng. much in little)
- Osnovna ideja: sačuvati unapred smanjenu sliku u "svakoj mogućoj razmeri"
- Tekseli na višem nivou čuvaju proseke tekstura sa prethodnog nivoa
- Za udaljene objekte koristićemo teksture manje rezolucije
- Moguće je pogledati samo jedan piksel MIP mape odgovarajućeg nivoa
- Ovo je primer LOD (level of detail) principa u računarskoj grafici

MIP mape

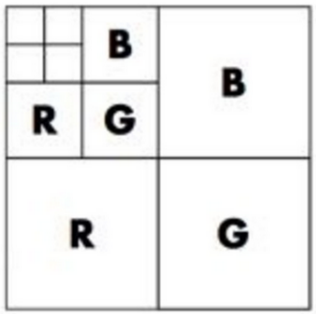


- Koliki su troškovi skladištenja MIP mape?

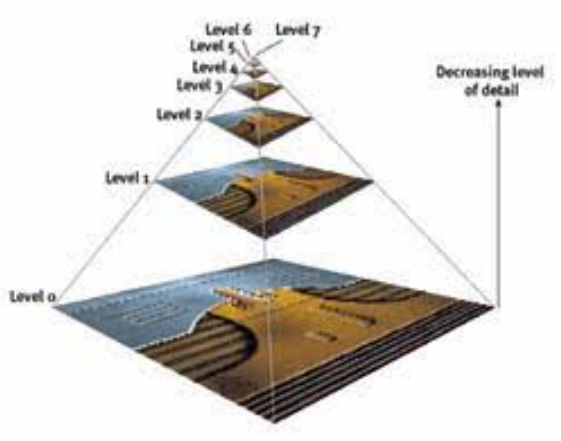
MIP mape



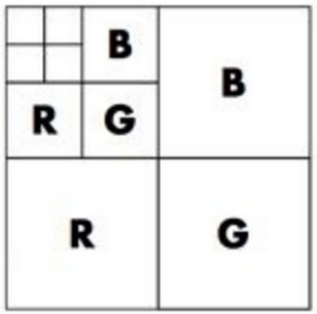
- Koliki su troškovi skladištenja MIP mape?



MIP mape



- Koliki su troškovi skladištenja MIP mape?



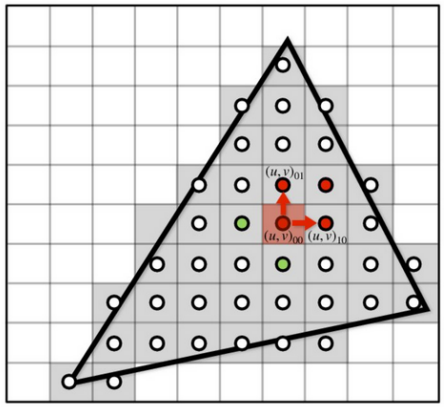
- $\sum_{i=1}^{\infty} \frac{1}{4^i} = \frac{1}{3}$

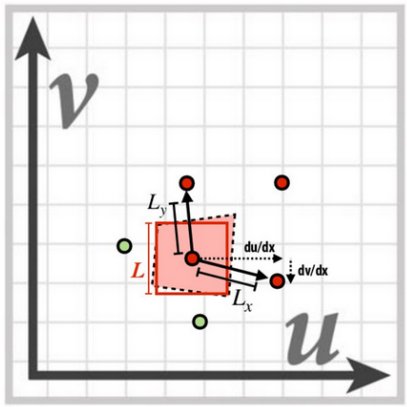
Odabir nivoa MIP mape

- Računamo kako se brzo menjaju u i v vrednosti po x i y osi – računamo razlike vrednosti teksturnih koordinata susednih piksela

$$\frac{du}{dx} = u_{10} - u_{00}, \quad \frac{dv}{dx} = v_{10} - v_{00}$$
$$\frac{du}{dy} = u_{01} - u_{00}, \quad \frac{dv}{dy} = v_{01} - v_{00}$$
$$L_x^2 = \left(\frac{du}{dx}\right)^2 + \left(\frac{dv}{dx}\right)^2 \quad L_y^2 = \left(\frac{du}{dy}\right)^2 + \left(\frac{dv}{dy}\right)^2$$

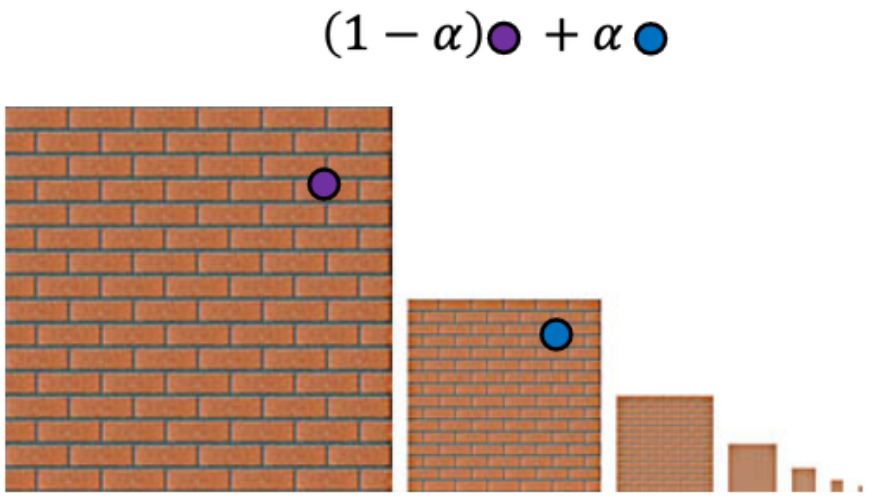
- Dužine vektora L_x i L_y nam daju neku procenu koliki region teksture pokriva jedan piksel: $L = \sqrt{\max(L_x^2, L_y^2)}$
- Pošto je kod MIP mape svaka naredna slika teksture duplo manje dimenzije, nivo mipmape dobijamo kao: $d = \log_2 L$





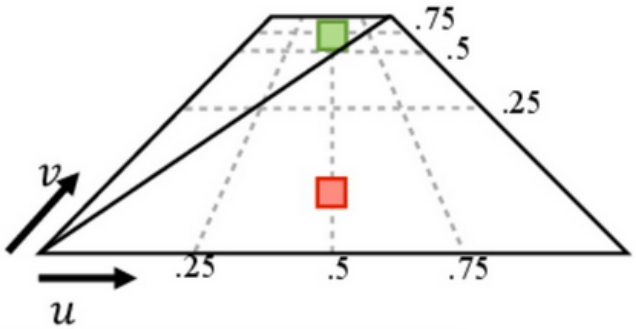
Problemi pri korišćenju MIP mapa

- Ako samo koristimo najbliži nivo MIP mape, možemo dobiti efekat da nivo "skoči" – prikaz se naglo menja od detaljnijeg ka mutnijem
- Umesto da biramo jedan nivo MIP mape koji odgovara najbližem celom broju, možemo da koristimo neprekidnu vrednost nivoa d
- Pošto unapred računamo fiksan broj nivoa MIP mape, možemo vršiti interpolaciju između dva nivoa MIP mape
- Ovde dolazimo do trilinearne interpolacije: mešamo teksture sa dva susedna nivoa MIP mape



Anizotropno filtriranje

- Trilinearno filtriranje pretpostavlja da se uzorci smanjuju istom brzinom duž u i v koordinate
- Bilinearno i trilinearno filtriranje su primeri izotropnog filtriranja
- Razmotrimo primer ravni koja je zarotirana u odnosu na kameru – promene duž koordinate v su veće nego duž u
- Kod anizotropnog filtriranja se pravi nova mapa teksture koja nezavisno smanjuje sliku po x i y osi: $(d_x, d_y) = (\log_2 \sqrt{L_x^2}, \log_2 \sqrt{L_y^2})$

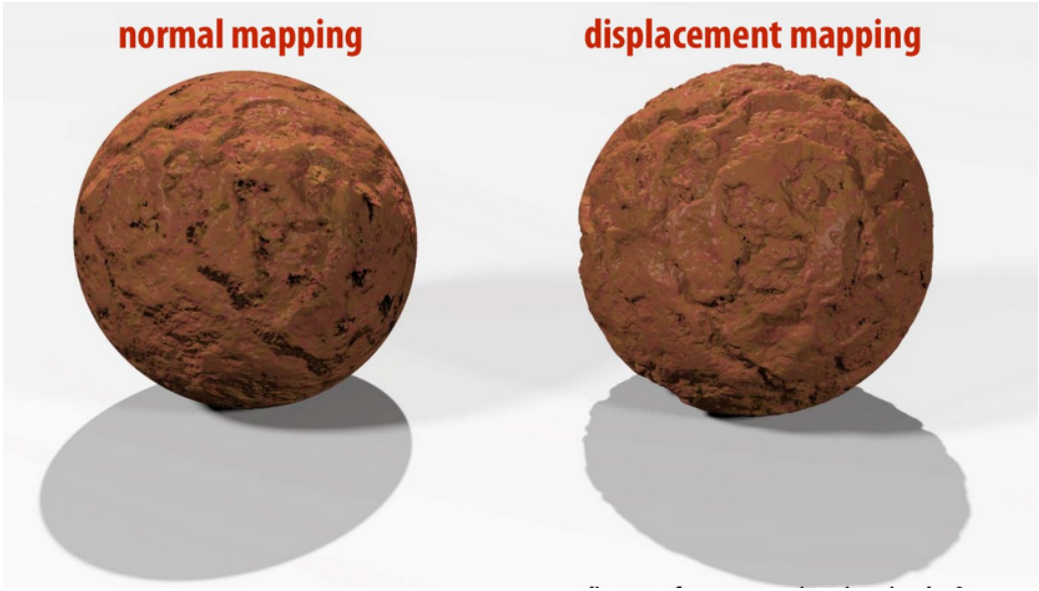


Druge vrste preslikavanja

- Ideja preslikavanja kodiranog slikom može se preneti mnogo šire nego samo za dobijanje boje površi
- Mnogi atributi u procesu renderovanja mogu se kodirati slikom:
  - razne osobine materijala: sjajnost, izgrebanost,...
  - normale površi, transparentnost, refleksivnost,...
  - boja svetlosti, intenzitet svetlosti,...
  - pozicija objekta
- Teksture mogu sadržati različite vrste informacija
- Može se koristiti i kombinacija većeg broja mapa

Druge vrste preslikavanja

- Preslikavanje normala – vrednosti teksture se koriste da izmene normale površi kako bi se dobio privid neravne površi
- Preslikavanje pomeraja – deli geometriju površi na male trouglove i menja im pozicije u skladu sa vrednostima teksture



Druge vrste preslikavanja

- Preslikavanje okruženja – koristi sliku okruženja za preslikavanje tekstura, omogućava simulaciju površi koje imaju visoku refleksiju
- Preslikavanje neravnina – omogućava izmenu vektora normala tokom renderovanja