Mapování textur

Pavel Strachota

FJFI ČVUT v Praze

7. dubna 2020

Obsah

Obsah

Textura

- **textura** = popis vlastností povrchu - struktury, barvy, kvality
  - pravidelná či nepravidelná
- pro „dobrý dojem“ stačí mnohdy jednoduchá geometrie a detailní textura
- textura může obsahovat informace o
  - barvě povrchu
  - odrazu světla (změna zrcadlové složky barvy materiálu m_S - viz *osvětlování*)
  - změně normálového vektoru - opticky mění tvar povrchu $\Longrightarrow$ hrbolatost (metoda *bump mapping*)
  - průhlednost
  - optických vlastnostech nad povrchem - tzv. hypertextura

Typy textur

rozdělení podle dimenze

- jednorozměrné - definice opakujících se podélných vzorků, generování přerušovaných čar, vrstevnic
- dvourozměrné - mapovány na povrch tělesa
- trojrozměrné (objemové) - simulace objektů vyříznutých z bloku materiálu (dřevo, mramor, ...)
- čtyřrozměrné - animace

rozdělení podle reprezentace

- tabulka (obrázek, pole voxelů)
- volání funkce (procedurální textury)

Inverzní mapování textury

- úloha: nanést dvourozměrnou texturu na povrch tělesa
- textura popsána parametricky: $T = T(u, v)$, $T : D_T \mapsto H_T$
  $\Longrightarrow$ *texturové souřadnice* u, v
- (inverzní) **mapování textury** je transformace

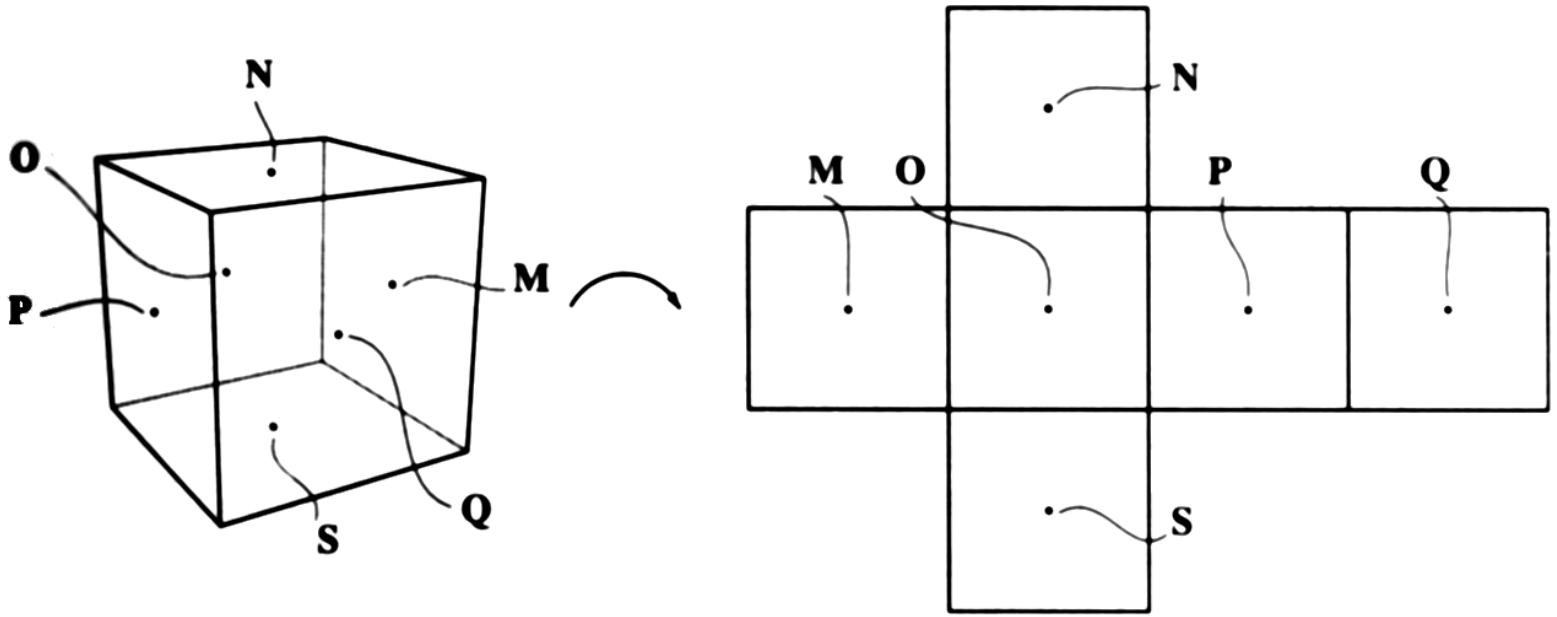
$$M : D_M \mapsto D_T,$$

  kde $D_M \subset \mathbb{R}^3$ je množina bodů tvořící povrch tělesa
- důvod názvu: je-li plocha popsaná parametricky funkcí $\boldsymbol{Q}(s, t)$, lze M volit jako $M = \boldsymbol{Q}^{-1}$
  (pokud inverzní zobrazení existuje)
  - jinak: s, t lze při vykreslování povrchu použít přímo jako texturové souřadnice (resp. po škálování)
- hodnota textury v bodě $\boldsymbol{x} \in D_M$ je $(T \circ M)(\boldsymbol{x})$
- zborcené povrchy (nelze bez deformace rozvinout do roviny) $\Longrightarrow$ zkreslení textury

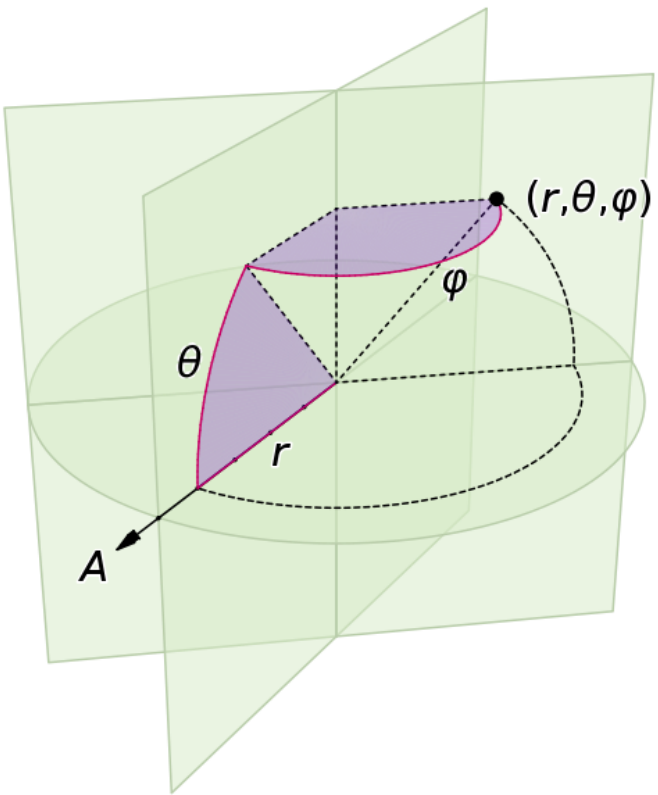
Texturové souřadnice

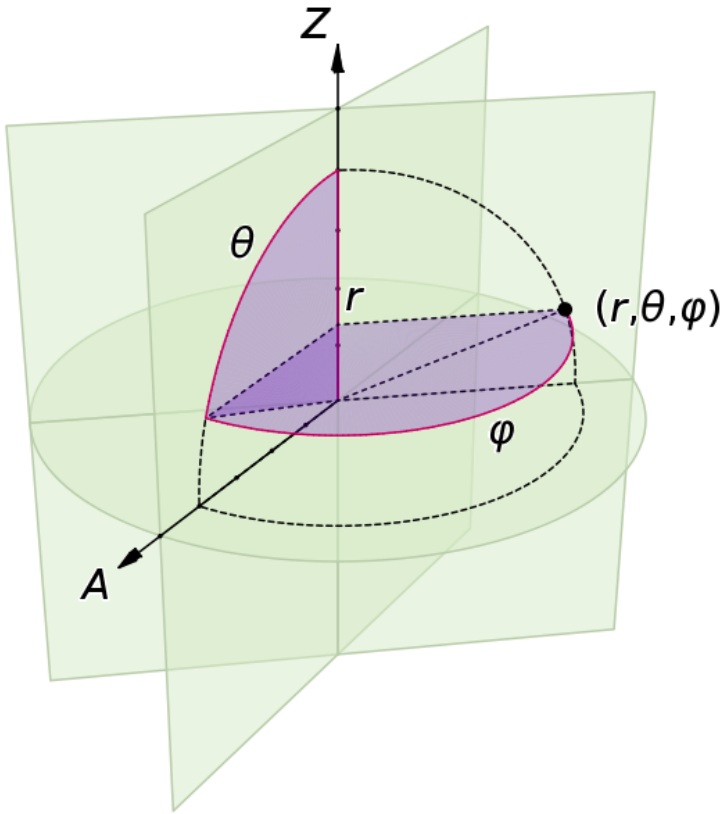
- polygonální síť - ve vrcholech $\boldsymbol{v}_i$ uloženy mimo jiné texturové souřadnice u, v, tj. hodnota $M(\boldsymbol{v}_i)$
- pro bod $\boldsymbol{x}$ uprostřed trojúhelníku najdeme $M(\boldsymbol{x})$ interpolací hodnot ve vrcholech
- texturové souřadnice uloženy při modelování objektu podle způsobu generování sítě
  - dělení parametrických kubických povrchů - z parametrů s, t
  - mapování na povrch krychle
  - mapování na kulovou plochu - **sférická projekce**
  - mapování na povrch válce - **cylindrická projekce**
  - ...

Mapování na povrch krychle



Sférické souřadnice





Sférická projekce 1/2

- koule umístěna do počátku souř. soustavy
- sférické souřadnice

$$\begin{aligned} x &= r\cos\varphi\cos\theta, \\ y &= r\sin\varphi\cos\theta, \\ z &= r\sin\theta, \end{aligned}$$

  kde $\varphi \in [0, 2\pi]$, $\theta \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right]$.
- smysl parametrů u, v budou mít úhly φ, θ (případně po škálování)
- zpětná transformace:

$$\varphi = \begin{cases} \arccos \frac{x}{\sqrt{x^2+y^2}} & y \geq 0 \\ 2\pi - \arccos \frac{x}{\sqrt{x^2+y^2}} & y < 0 \end{cases}$$

$$\theta = \arcsin \frac{z}{r} = \arcsin \frac{z}{\sqrt{x^2+y^2+z^2}}$$

Sférická projekce 2/2

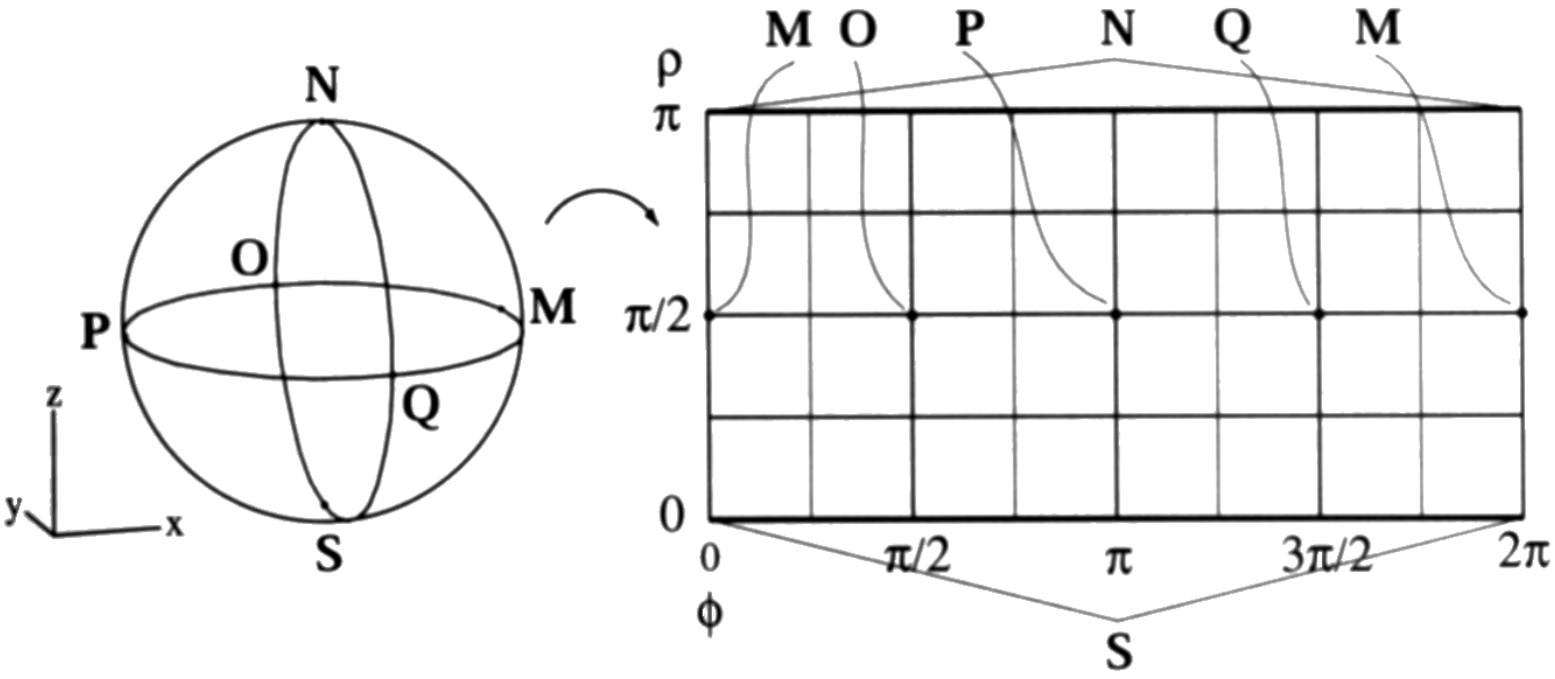
Alternativní výpočet úhlu φ

- polární souřadnice $x = r\cos\varphi, y = r\sin\varphi$
- výpočet $\varphi \in [-\pi, \pi]$:

$$\varphi = \begin{cases} \arctan\frac{y}{x} & x > 0 \\ \arctan\frac{y}{x} + \pi & x < 0, y > 0 \\ \arctan\frac{y}{x} - \pi & x < 0, y < 0 \\ \frac{\pi}{2} & x = 0, y > 0 \\ \frac{3\pi}{2} & x = 0, y < 0 \\ 0 & x = y = 0 \end{cases}$$

- toto je definice funkce `atan2(y,x)` v různých programovacích jazycích
  (C, Fortran, Java, C#, Perl, Lisp, ...)
  - chování v $x = y = 0$ se někdy liší

Mapování textury na kouli



$$\phi = \varphi, \rho = \theta + \frac{\pi}{2}$$

Cylindrická projekce

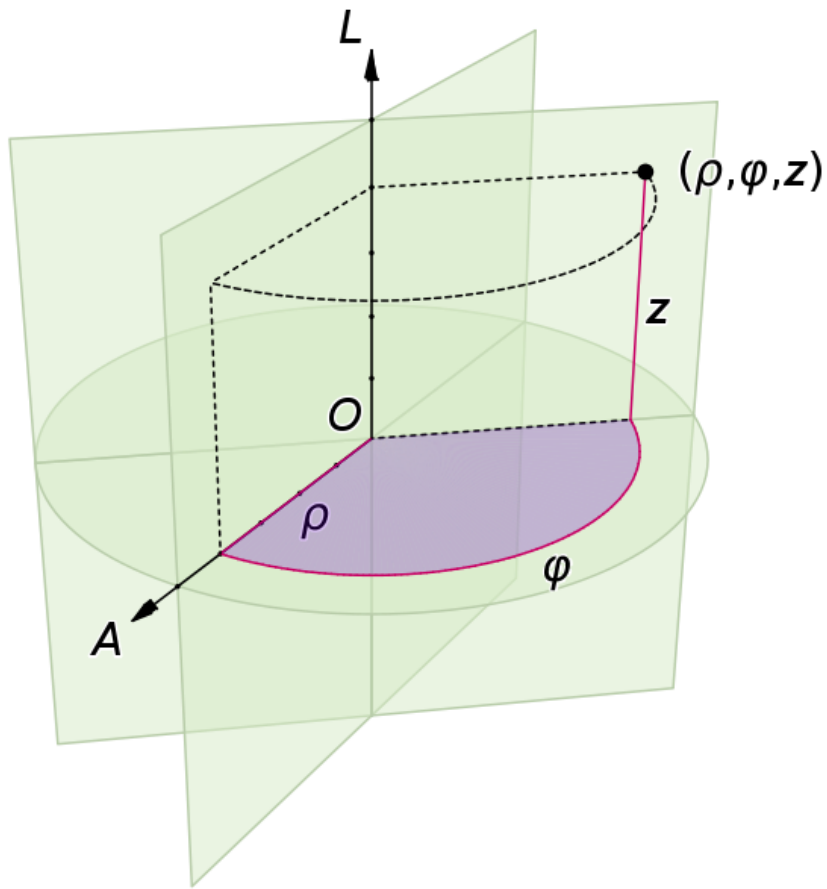
- cylindrické souřadnice

$$\begin{aligned} x &= r\cos\varphi \\ y &= r\sin\varphi \\ z &= z \end{aligned}$$

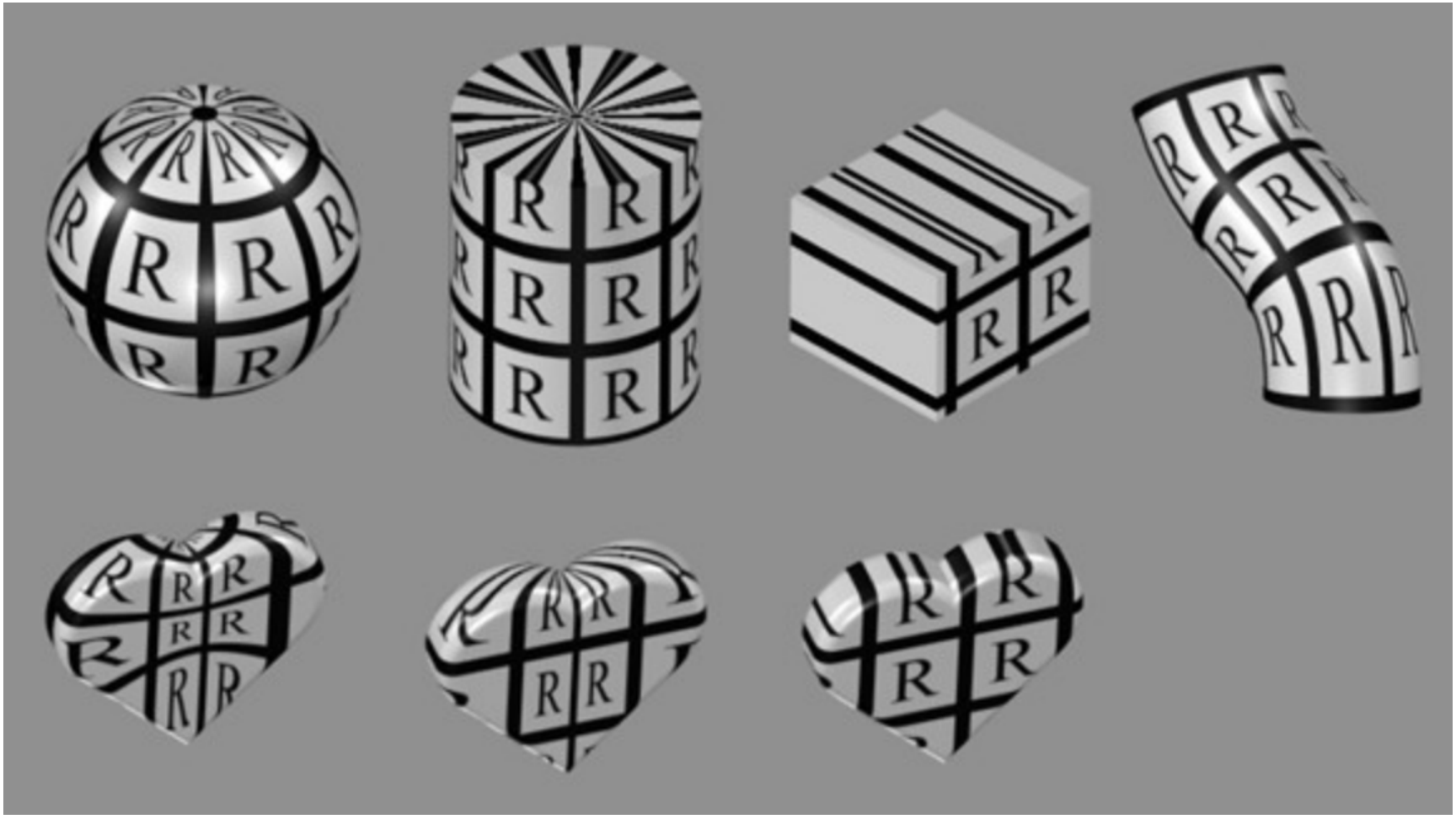
- zpětná transformace

$$\varphi = \operatorname{atan2}(y, x)$$

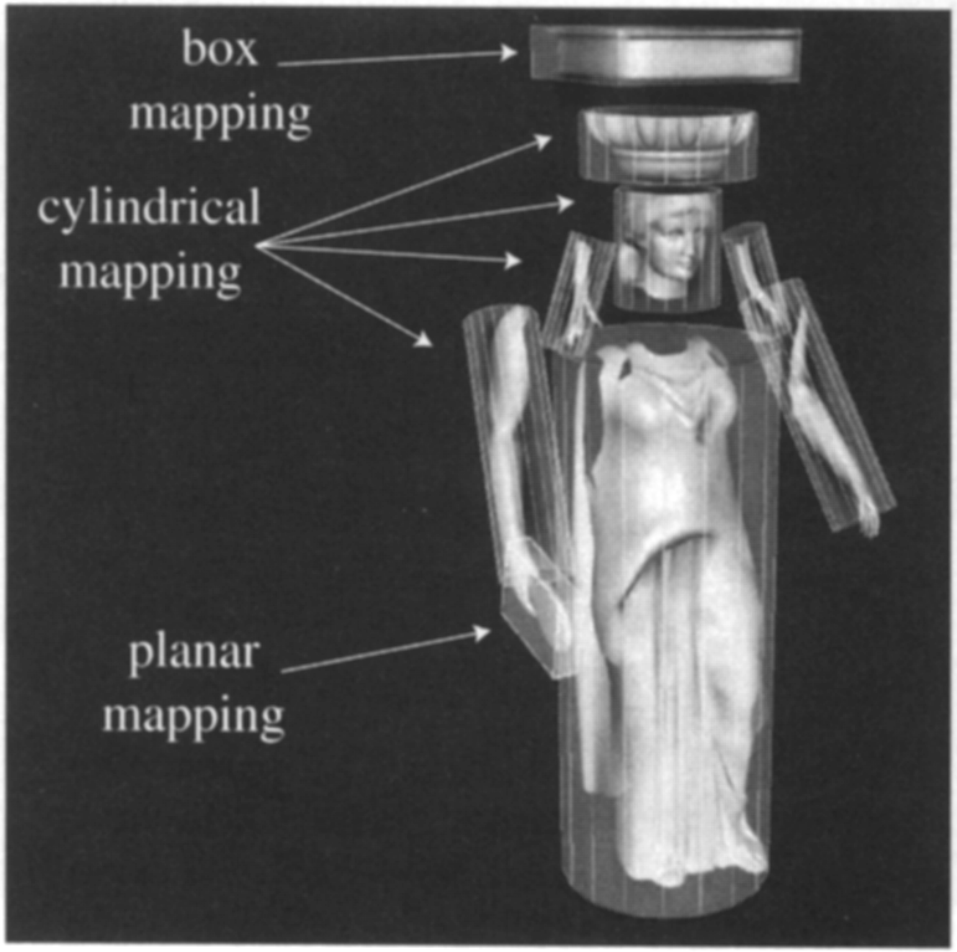
- smysl texturových souřadnic u, v mají φ a z (příp. po škálování)



Příklady



Příklady

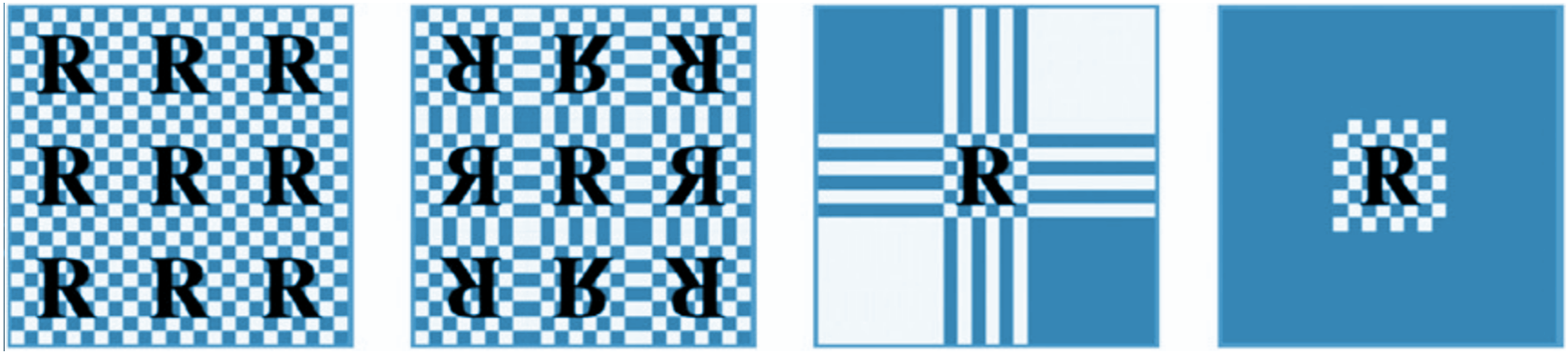


Korespondenční funkce

- předzpracování texturových souřadnic funkcí C
- zobecněné schéma: $M : D_M \mapsto D_C$, $C : D_C \mapsto D_T$, $T : D_T \mapsto H_T$
- D_C ... dvourozměrný *parametrický* prostor
- $D_T = [0, 1]^2$ texturové souřadnice omezené do jednotkového čtverce (OpenGL, DirectX)
- hodnota textury v bodě $\boldsymbol{x} \in D_M$ je $(T \circ C \circ M)(\boldsymbol{x})$
- **význam**: posunutí, rotace, zkosení textury, dodefinování mimo definiční obor textury

Dodefinování textury

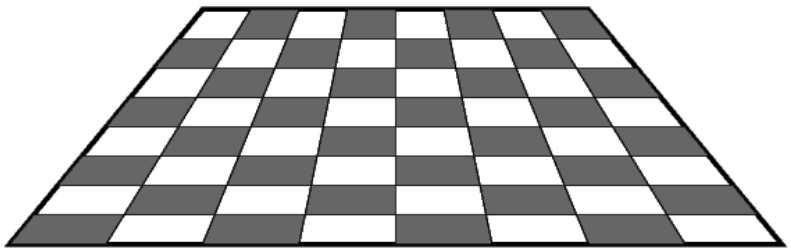
- ***wrap*** (*tile, repeat*) - opakování textury:
  $C(u, v) = (\text{frac}(u), \text{frac}(v))$
- ***mirror*** - zrcadlení textury (opakování se zrcadlením v každém směru)
- ***clamp*** (*clamp to edge*) - $C(u, v)$ ořízne hodnoty u, v na interval $[0, 1]$
- ***border*** (*clamp to border*) - mimo D_T je dána pevná barva hranice



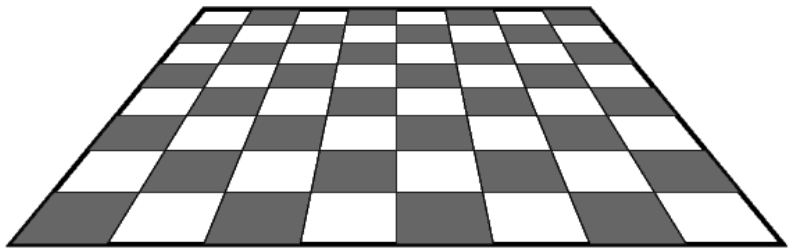
Perspektivně korektní mapování

- vykreslení obrysu trojúhelníku: promítnou se vrcholy a spojí se čarami (vlastně lineární interpolace poloh vrcholů na obrazovce)
- vyplněné trojúhelníky kreslíme po pixelech - barvu, resp. texturové souřadnice (definované ve vrcholech) **lineárně** interpolujeme v rámci $\triangle$:
  1. vzhledem k poloze na obrazovce - **špatně** (v perspektivě délky závisí na vzdálenosti od pozorovatele)
  2. vzhledem ke skutečné poloze pixelu v trojúhelníku ve 3D - nutnost zpětné transformace do souřadnic scény (WCS - viz promítání)
- nejefektivnější řešení: **racionálně lineární** (**hyperbolická**) interpolace texturových souřadnic
  - v homogenních souřadnicích VCS (souřadnicích v průmětně) - lineárně se interpoluje i čtvrtá souřadnice W
  - v realtimových systémech lze volitelně zapnout (OpenGL)

Příklady perspektivně (ne)korektního mapování

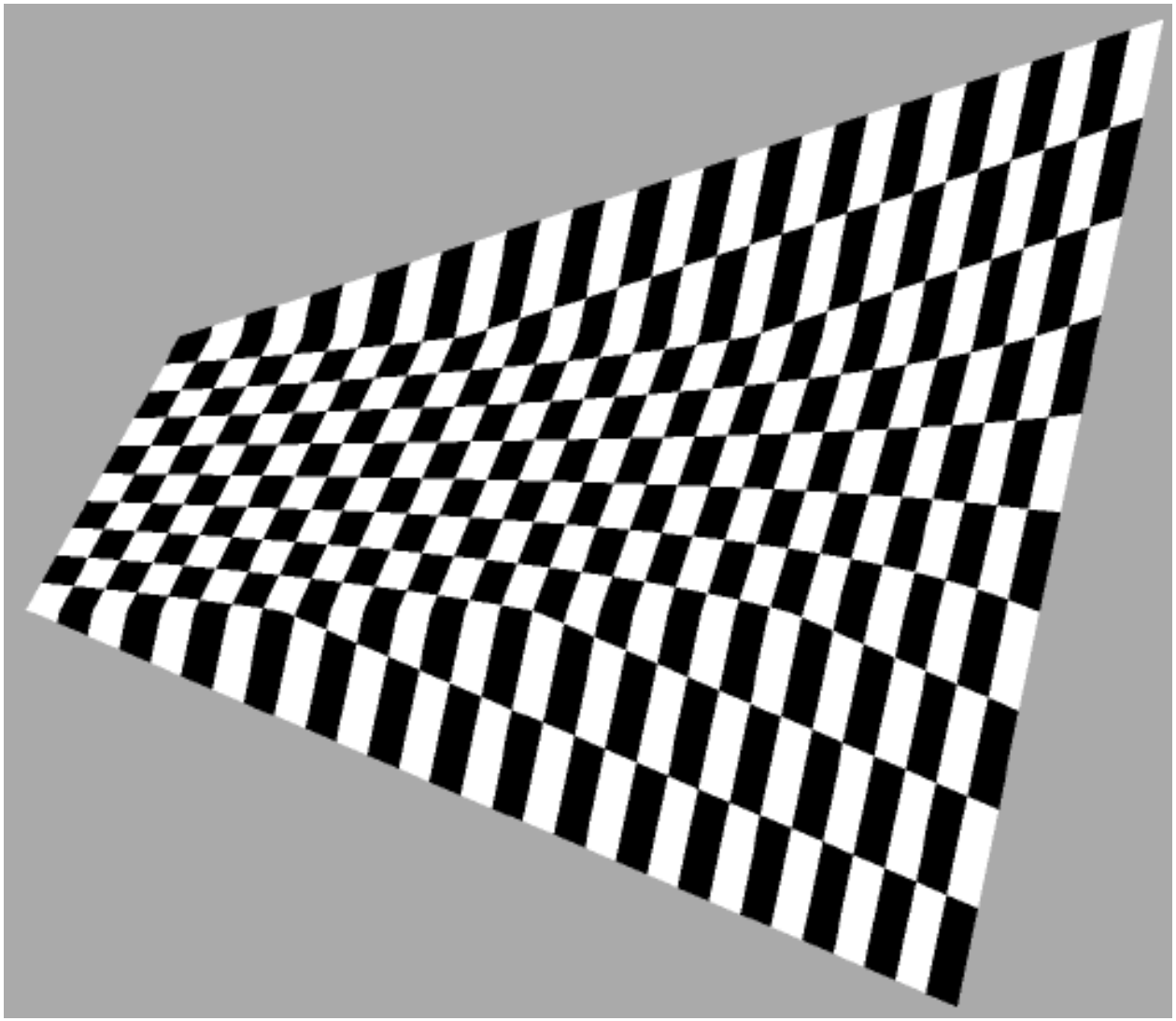


Nekorektní
(lineární interp. ve VCS)



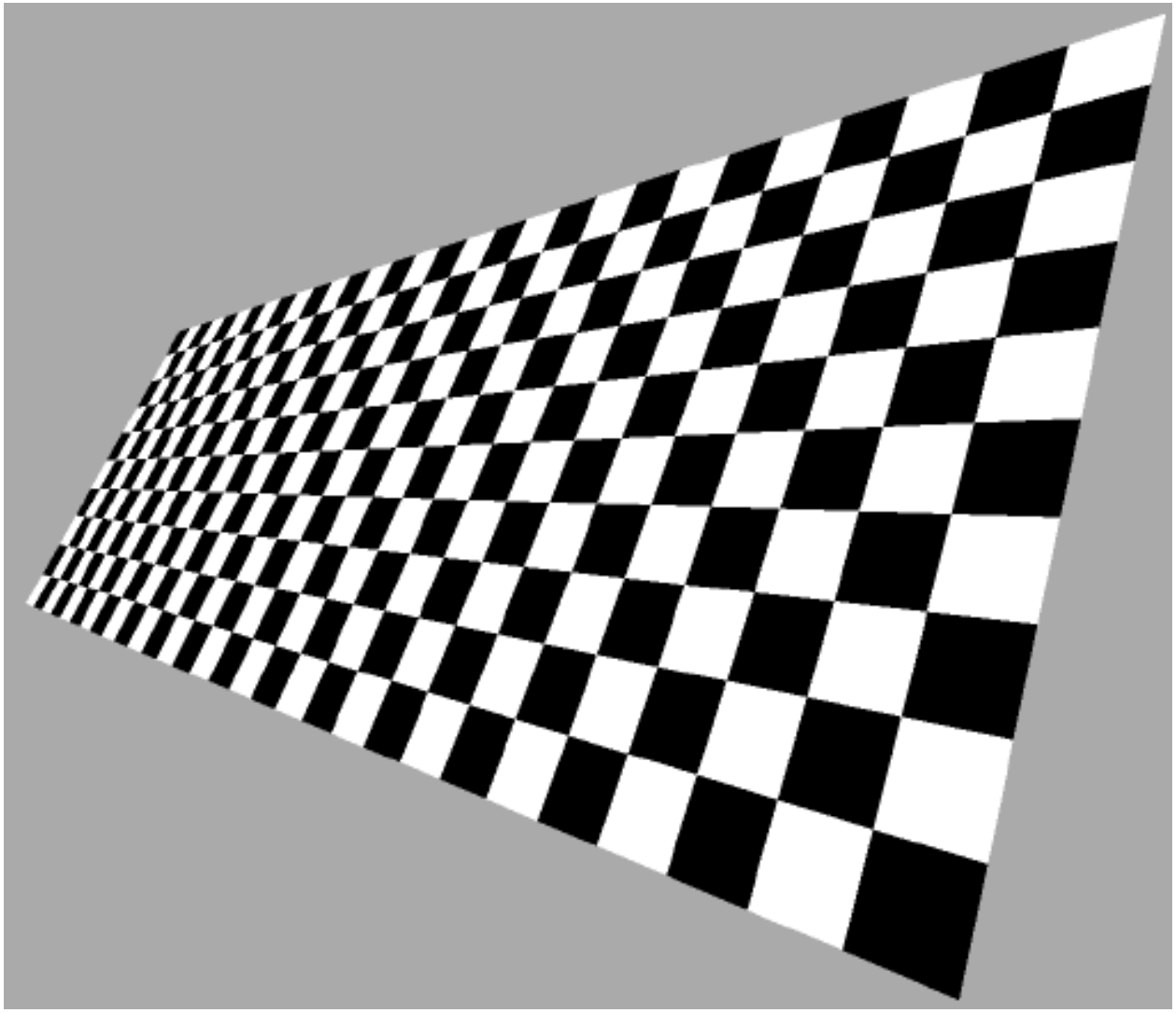
Korektní
(hyperbolická interp. ve VCS)

Příklady perspektivně (ne)korektního mapování



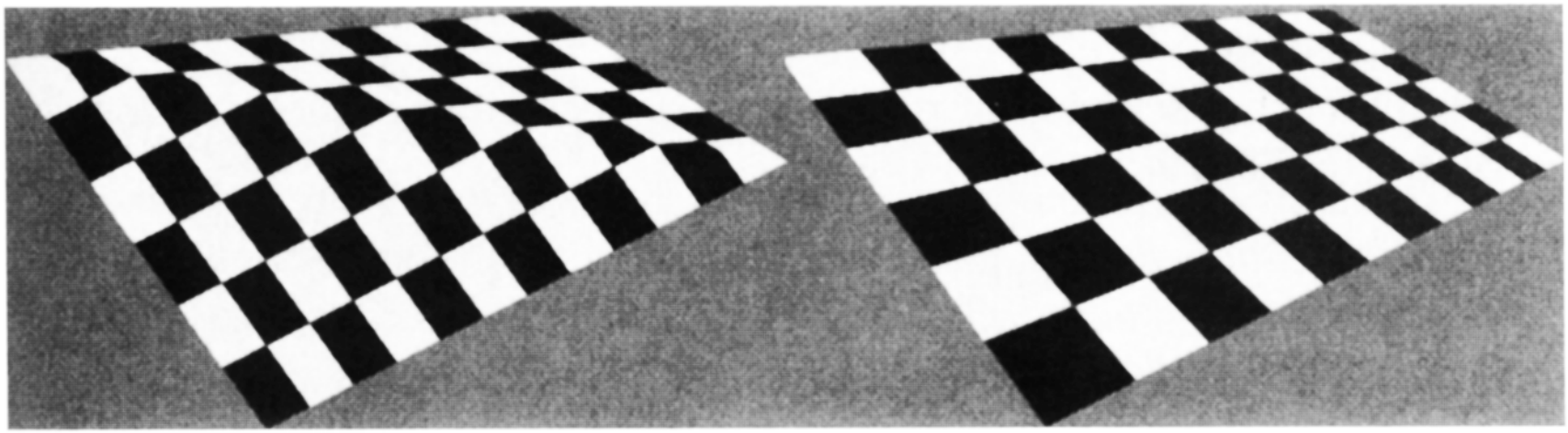
Nekorektní mapování

Příklady perspektivně (ne)korektního mapování



Korektní mapování

Příklady perspektivně (ne)korektního mapování



Vlevo: triangulace obdélníku a lineární interpolace na každém trojúhelníku zvlášť

Obsah

Modifikace vlastností povrchu

- povrch má sám o sobě materiálové vlastnosti (barvy m_A, m_D, m_S, exponent zrcadlového odrazu atd.), definované obvykle ve vrcholech a interpolované v $\triangle$
- pomocí textury tyto vlastnosti nahradíme nebo modifikujeme
- při Gouraudově stínování modifikujeme přímo výslednou interpolovanou barvu:
  - *replace* (*glow texture*) - textura nahradí barvu získanou na základě „globálních“ vlastnosti materiálu, odstraní vliv osvětlování na povrch
  - *decal* (potisk) - textura obsahuje kanál alfa, původní barva povrchu je barvou „pozadí“ pro průhlednost
  - *modulate* - barva povrchu se vynásobí barvou textury (někdy je realističtější modulovat pouze difuzní složku m_D - např. i černý povrch se leskne)

„Ne-barevné“ složky textury

- ***light mapping*** - textura obsahující informace o osvětlení, jestliže je osvětlení statické
  - možnost předpočítat metodou globálního osvětlování (radiozita apod.)
  - stačí menší rozlišení než obrazová textura
- ***gloss mapping*** - modifikuje příspěvek zrcadlové složky (rozliší matné a lesklé části povrchu)

Environment mapping 1/4

- Blinn, Newell
- generování odlesků na zakřivených objektech
- paprsky vedené od pozorovatele k lesklému objektu
  - tolik paprsků, kolik je pixelů v průmětu objektu
- s pomocí normály povrchu se vypočítá směr odraženého paprsku
- nezkoumá se průsečík odraženého paprsku s ostatními objekty ve scéně (raytracing), ale průsečík s texturou okolí
  - objekt je obklopen tělesem (koulí, krychlí), na níž je namapována textura s obrazem okolní scény

Environment mapping 2/4

Metoda pole Blinna a Newella

- $\boldsymbol{e}$ ... vektor paprsku od pozorovatele, $|\boldsymbol{e}| = 1$
- $\boldsymbol{r} = \boldsymbol{e} - 2\,(\boldsymbol{n} \cdot \boldsymbol{e})\,\boldsymbol{n}$ je vektor odrazu
- $\boldsymbol{r} = \left(r_x, r_y, r_z\right)$, $|\boldsymbol{r}| = 1$
- transformace do sférických souřadnic
  - $\phi \in [0, 2\pi]$ ... zeměpisná délka
  - $\rho \in [0, \pi]$ ... zeměpisná šířka

$$\begin{aligned} \phi &= \operatorname{atan2}\left(r_y, r_x\right) \\ \rho &= \arccos\left(-r_z\right) \end{aligned}$$

- textura musí navazovat na krajích ve směru ϕ (zem. délky) a musí se eliminovat zkreslení na pólech

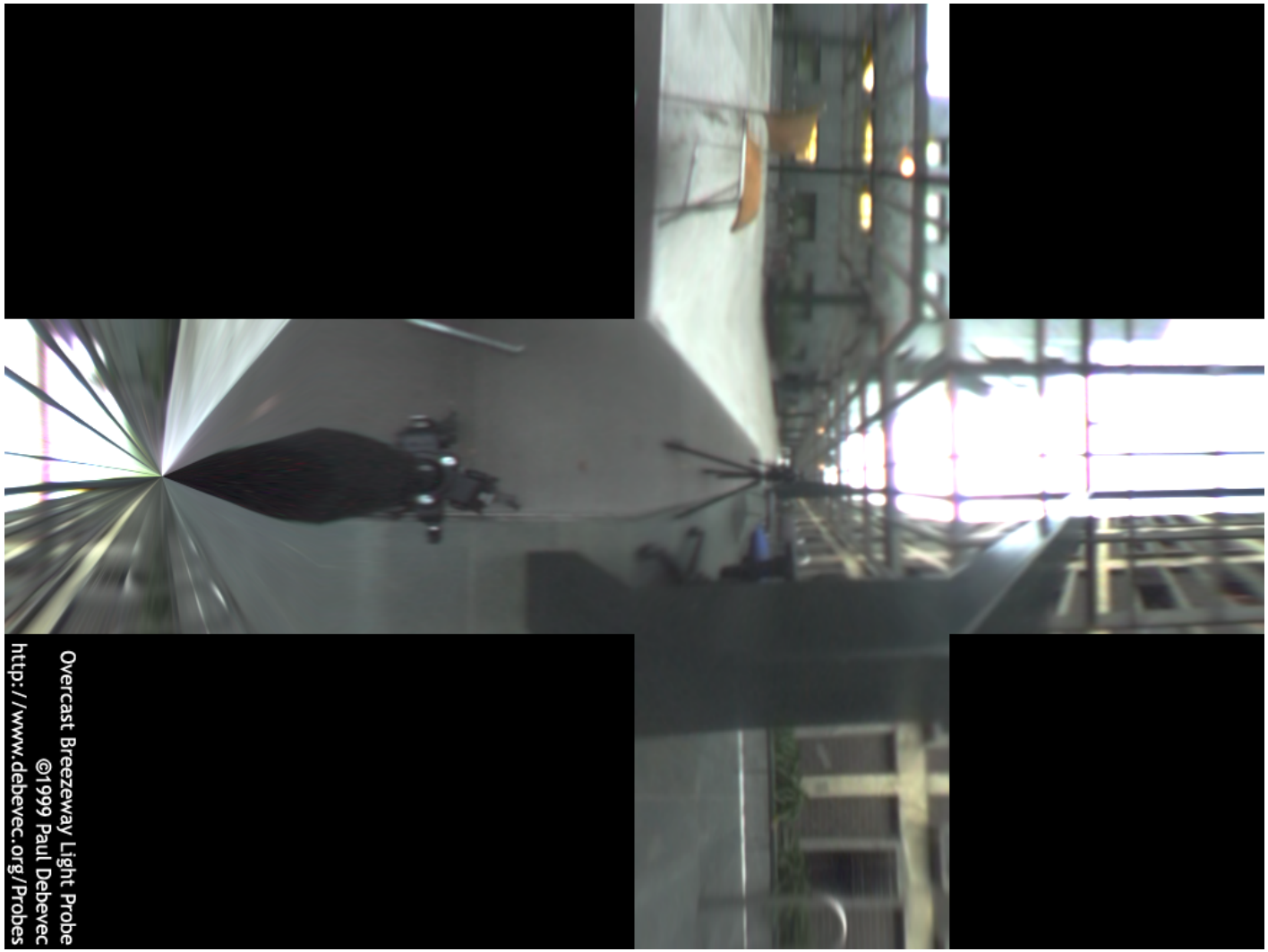
Environment mapping 3/4

Metoda podle Greena

- „obalové“ těleso je krychle se středem v poloze kamery
- obrazy na stěnách krychle = mapy okolí
- rychlá, jednoduchá metoda, HW implementace
- rovnoměrnější rozdělení vzorků (u Blinnovy metody potřeba vyšší hustoty vzorků v okolí pólů)
- nefunguje dobře pro ploché objekty (je vidět efekt obklopující krychle), nebo pro rovnoběžné promítání

Environment mapping

Příklady sítí krychle

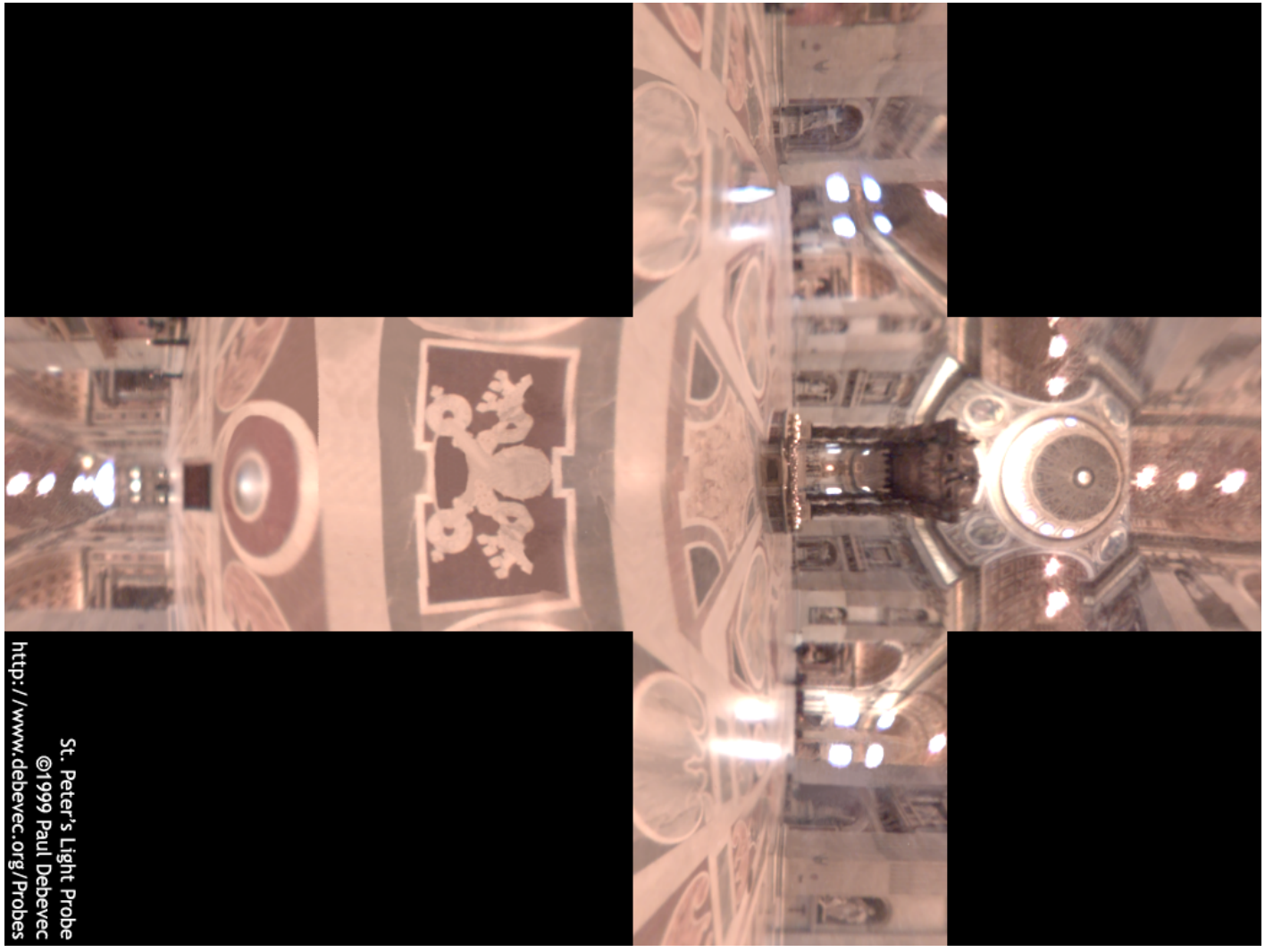


Environment mapping

Příklady sítí krychle

Environment mapping

Příklady sítí krychle



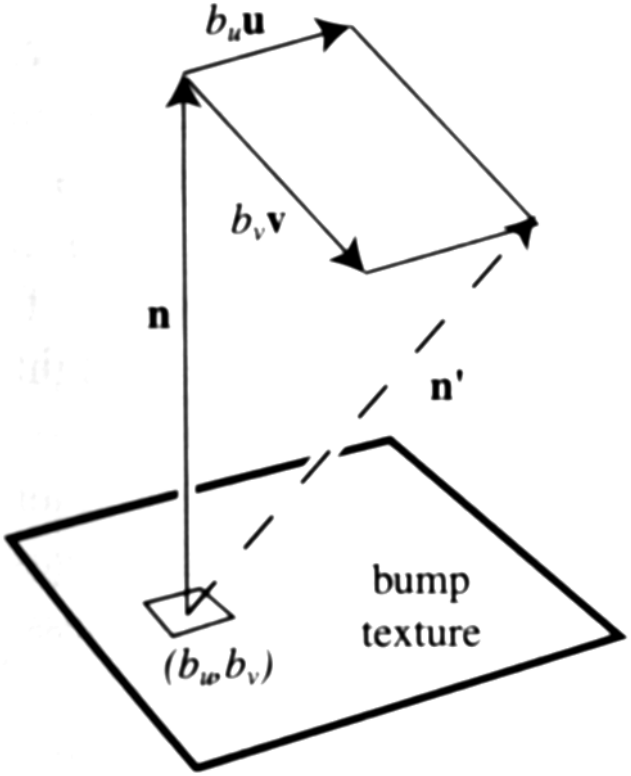
Obsah

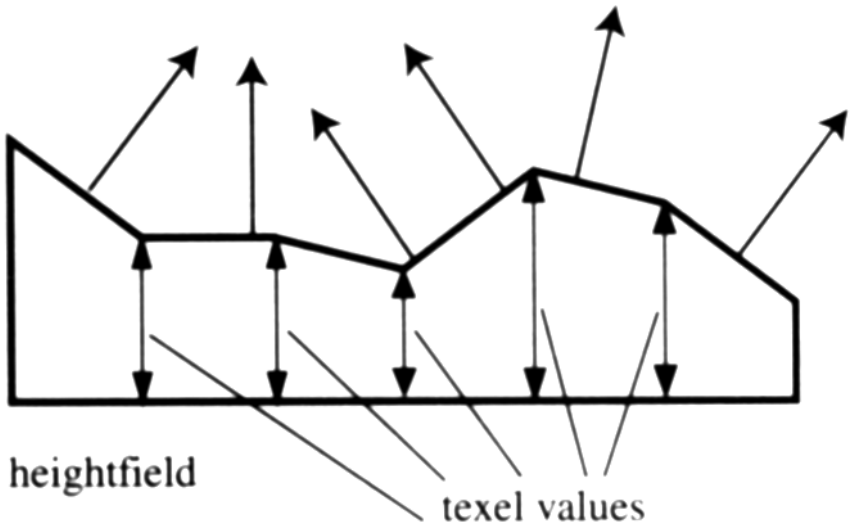
Bump mapping 1/4

- technika pro iluzi hrbolatých, zvrásněných či jinak lokálně nerovných povrchů
- hodnoty textury modifikují směr normálového vektoru v každém bodě
  $\Longrightarrow$ iluze nerovnosti se projeví při nasvícení, povrch je však stále rovný
- možnosti uložení textury:
  - 2 hodnoty pro každý bod určující posunutí $\boldsymbol{n}$ ve dvou navzájem kolmých směrech (v rovině trojúhelníku)
  - skalární textura - výškové pole
  - přímo pole normálových vektorů

Bump mapping 2/4

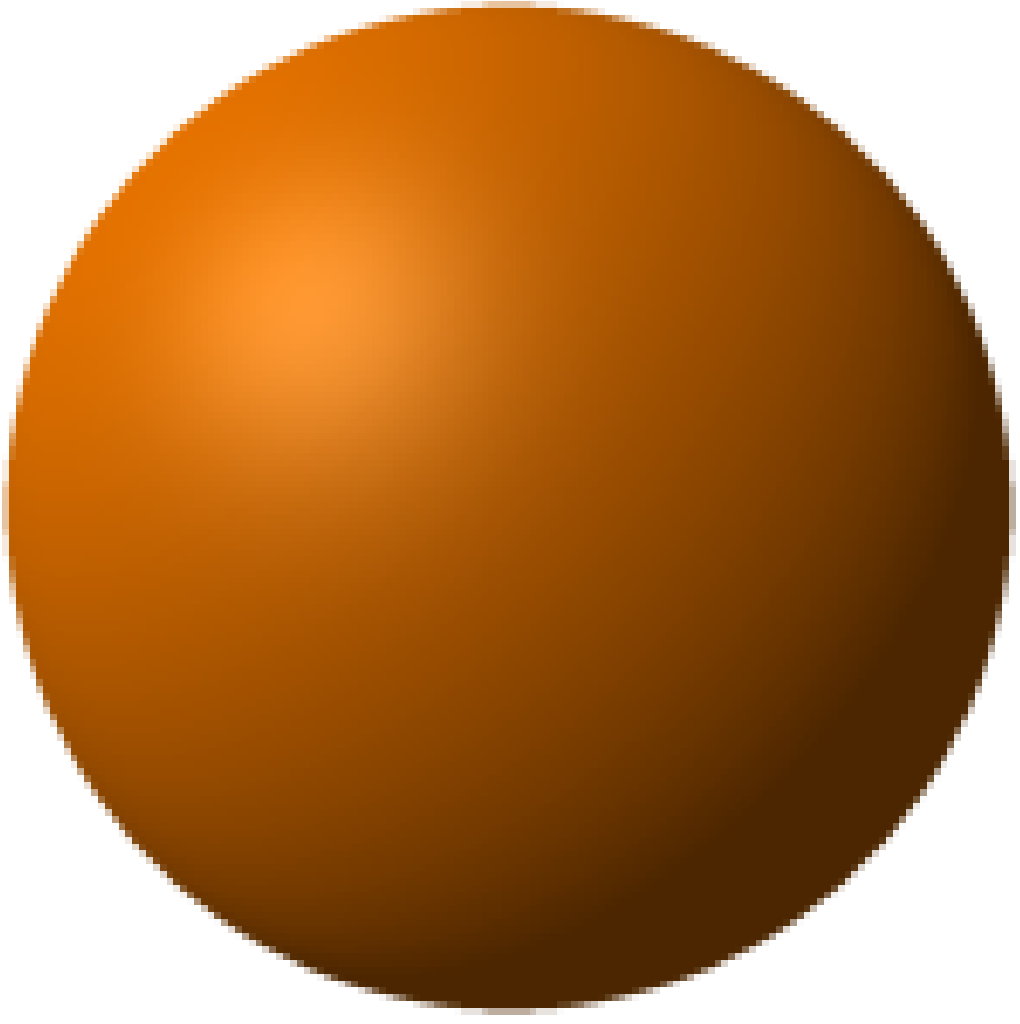
Schéma metod modifikace normály

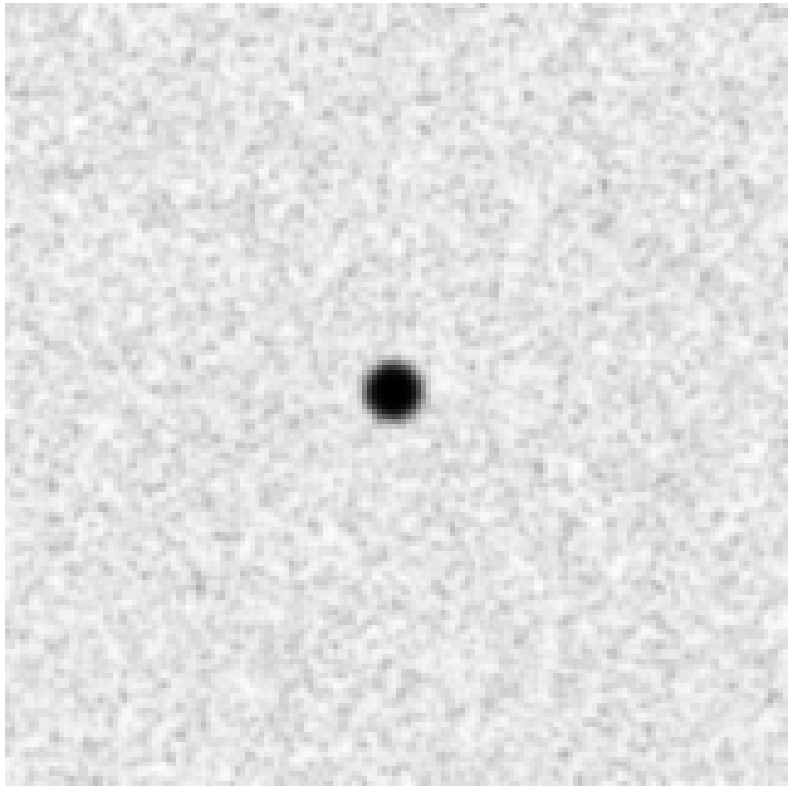




Bump mapping 3/4

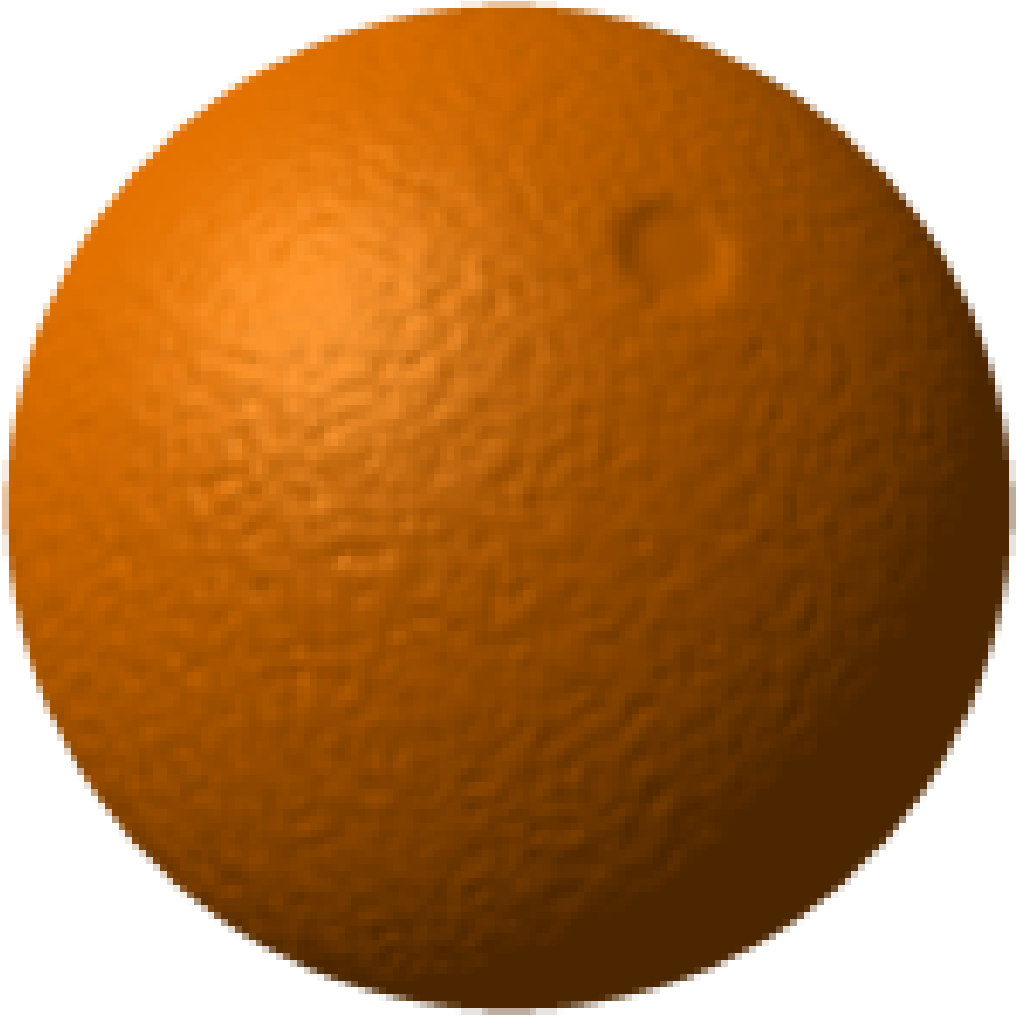
Schéma aplikace výškového pole

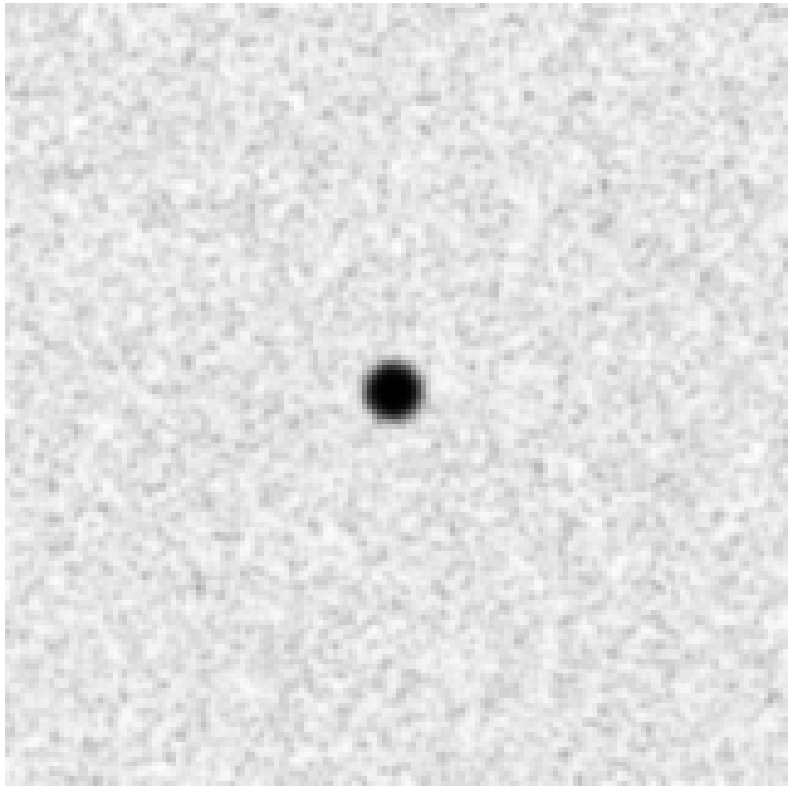




Bump mapping 3/4

Schéma aplikace výškového pole





Bump mapping 4/4

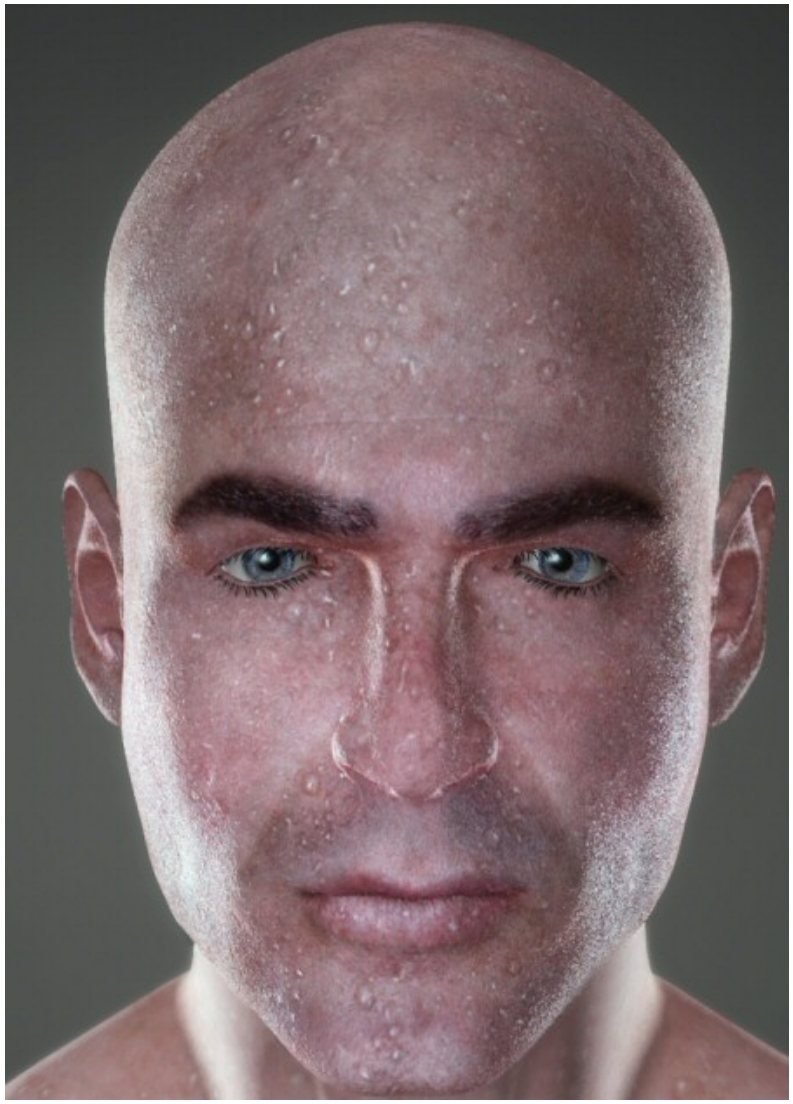
Další příklady aplikace

Bump mapping 4/4

Další příklady aplikace

Bump mapping 4/4

Další příklady aplikace



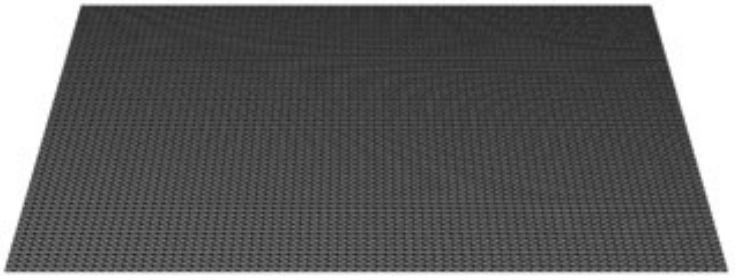
„Sweaty Simon“
SkinVue rendering system,
Poser

Displacement mapping

- nevýhody bump mappingu
  - na obrysech objektu je stále vidět jeho původní (rovný) tvar
  - nerovnosti nevrhají stín
- řešení: texturu výšek použít ke skutečné modifikaci povrchu - mapa posunutí (*displacement*)
- metody posunutí
  - posunutí vrcholů původní polygonální sítě
  - rozdělení sítě na mikropolygony, jejichž velikost je určena adaptivně, aby v daném pohledu nepřekročily velikost pixelu

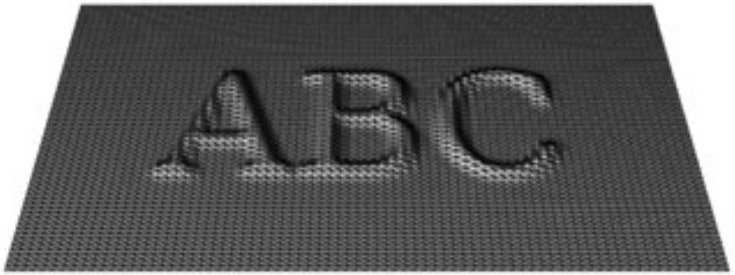
Displacement mapping

Schéma aplikace



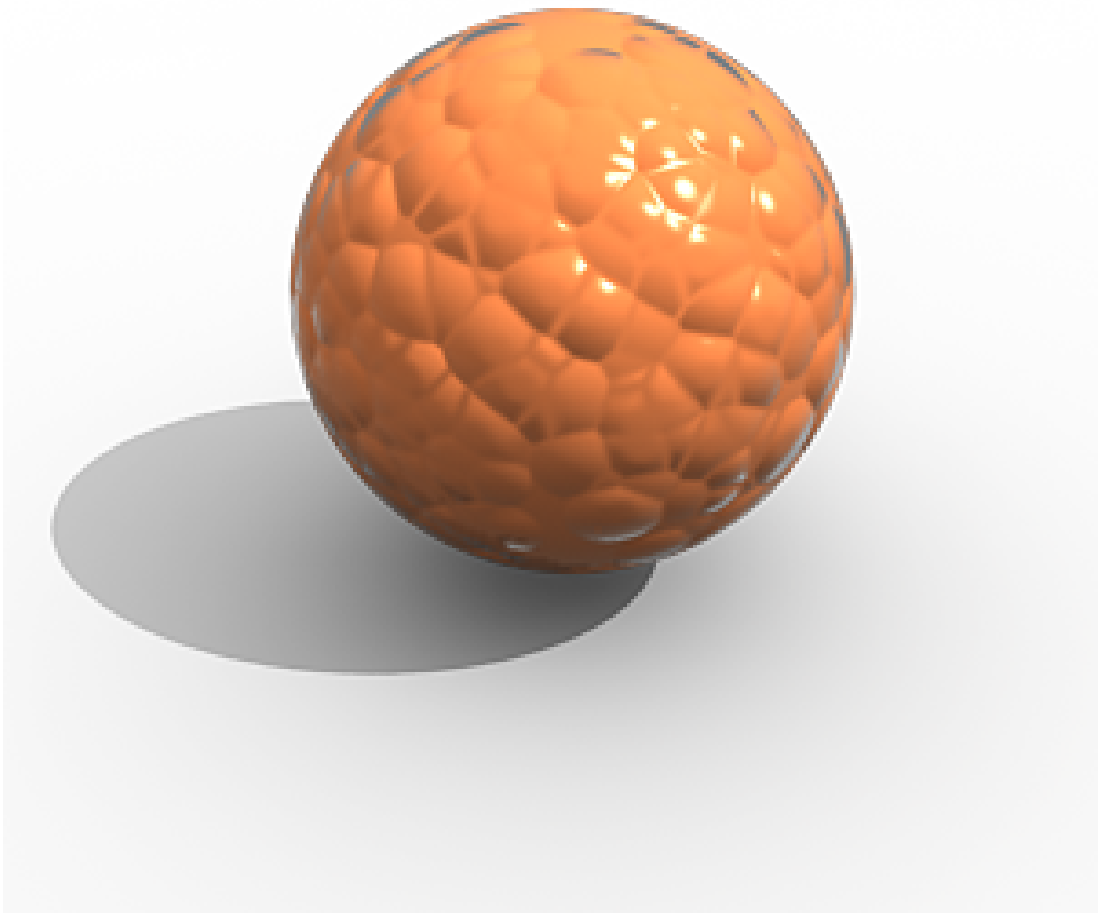
ORIGINAL MESH

DISPLACEMENT MAP

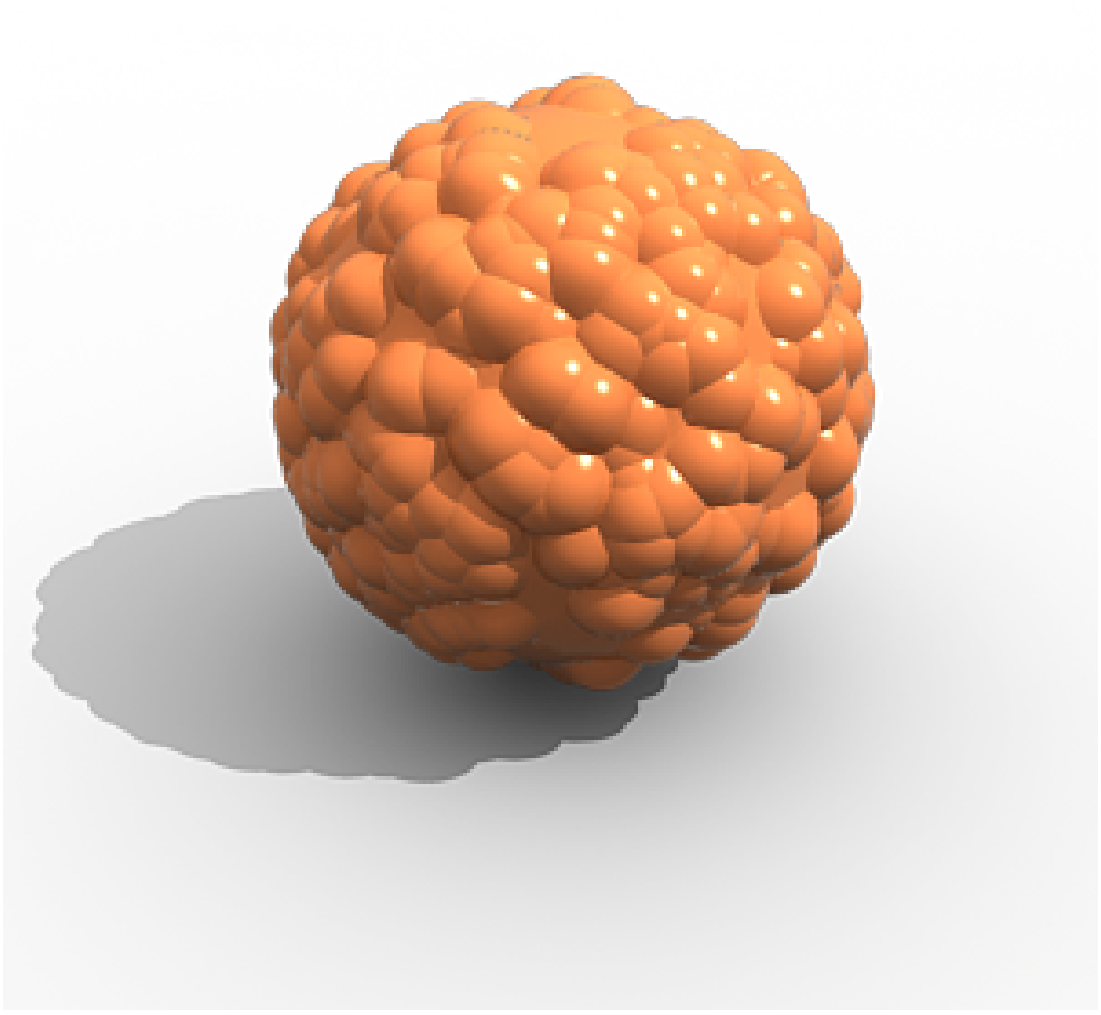


MESH WITH DISPLACEMENT

Displacement mapping vs. bump mapping



Displacement mapping vs. bump mapping



Obsah

Motivace

- *texel* = texture element (analogie pixelu)
- standardní rozměry obrazové textury jsou $2^m \times 2^m$, $m \in \mathbb{N}$ - typicky 512×512 texelů
- při mapování na objekt (povrch) se používá bilineární interpolace - pixel ovlivněn max. 4 texely
  - přílišné zvětšení (přiblížení objektu) $\Longrightarrow$ jeden texel ovlivní barvu mnoha pixelů
  - dostatečné zmenšení (oddálení) $\Longrightarrow$ více texelů se (teoreticky) zobrazí do jednoho pixelu
    pokud je to více než 4, **vzniká aliasing i přes použití interpolace**
- odstranění aliasingu - filtrování (např. průměrování) textury přes celou oblast ovlivňující daný pixel
  (low pass filtr - snížení max. frekvence v textuře)

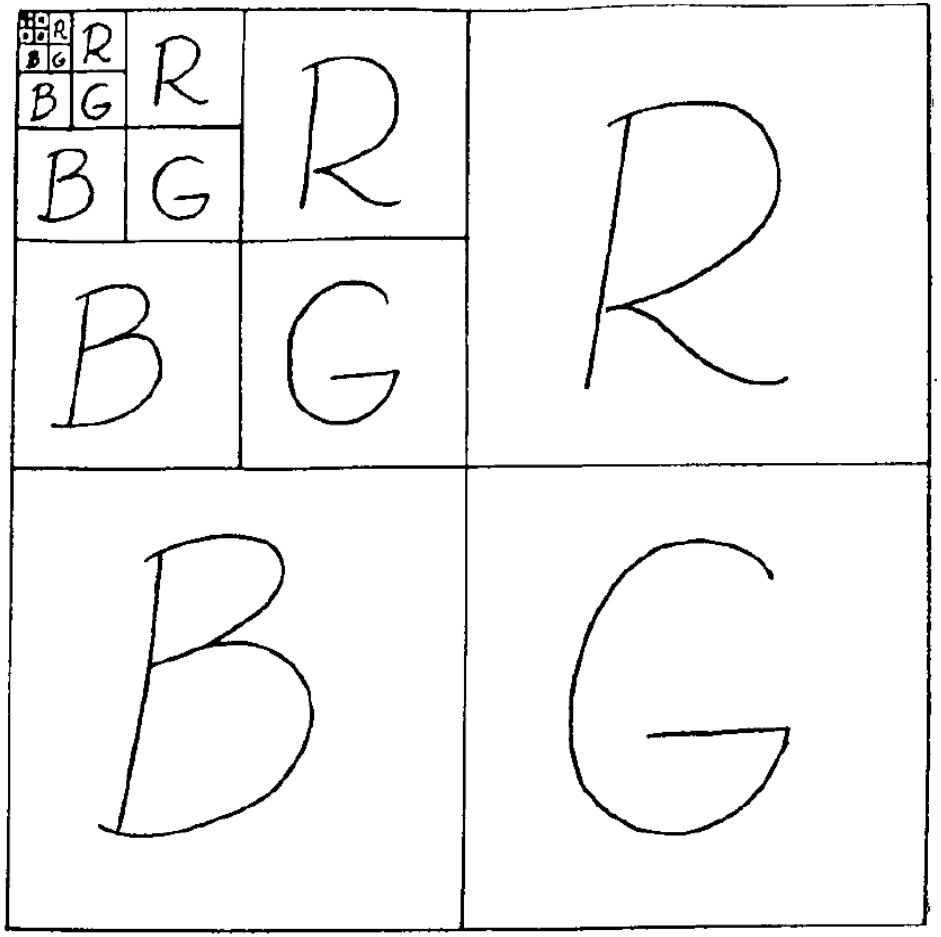
Mipmapping

- filtrování - velká výpočetní náročnost, problém při HW zpracování (streamline)
- **mipmapping** - předpočítání zmenšených filtrovaných kopií textury a jejich uložení v paměti
  - při mapování textury se pak jen zvolí vhodná kopie (aby 1 texel $\approx$ 1 pixel - viz dále)
- **MIP** = lat. **multum in parvo** , tj. mnoho v málu

Mipmapping

Způsob uložení v paměti

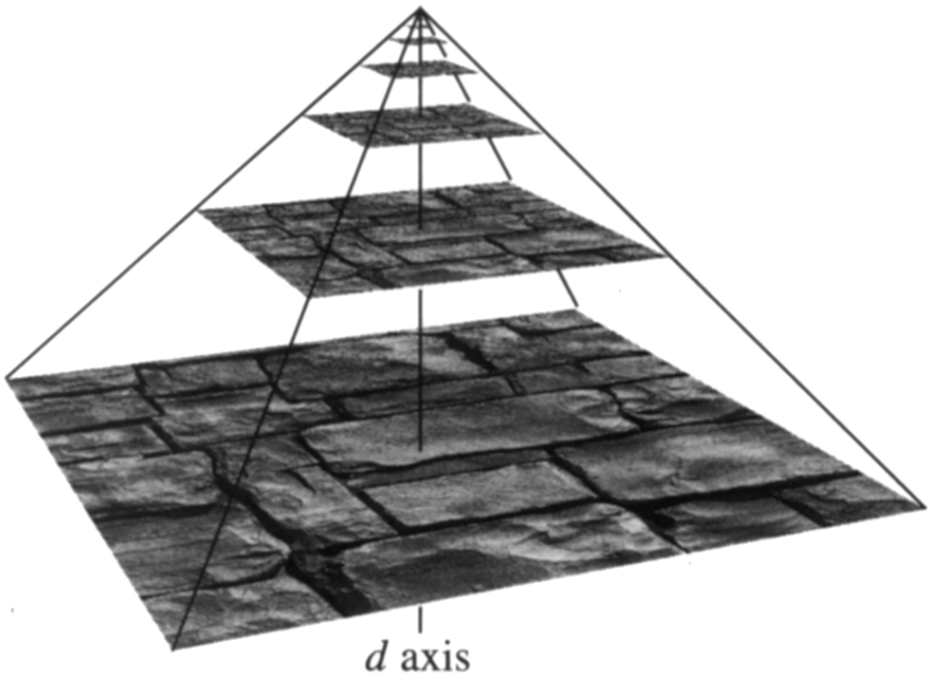
- rozklad na R,G,B složky, celkem se použije o $\frac{1}{3}$ více paměti než na původní texturu
- zmenšené kopie na $\frac{1}{2}$ v každém směru až do velikosti 1×1 pixel $(2^n, n = m, m-1, \ldots, 0)$



Mipmapping

Trilineární interpolace

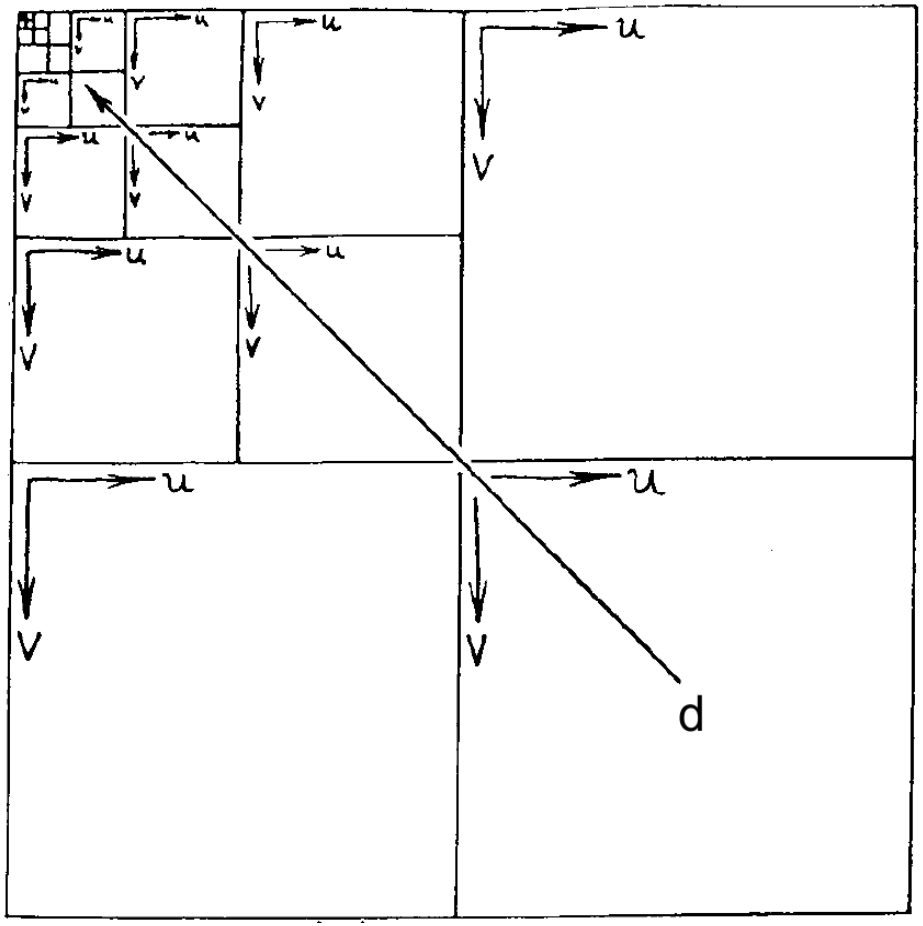
- kopie textury lze uspořádat do pomyslné pyramidy
- kromě texturových souřadnic u, v vypočítáme ještě parametr d - „výšku“ v pyramidě, tj. úroveň zmenšení
- vybereme buď kopii, která je nejblíže d, nebo interpolujeme mezi dvěma nejbližšími kopiemi $\Longrightarrow$ trilineární interpolace v u, v, d



Mipmapping

Trilineární interpolace

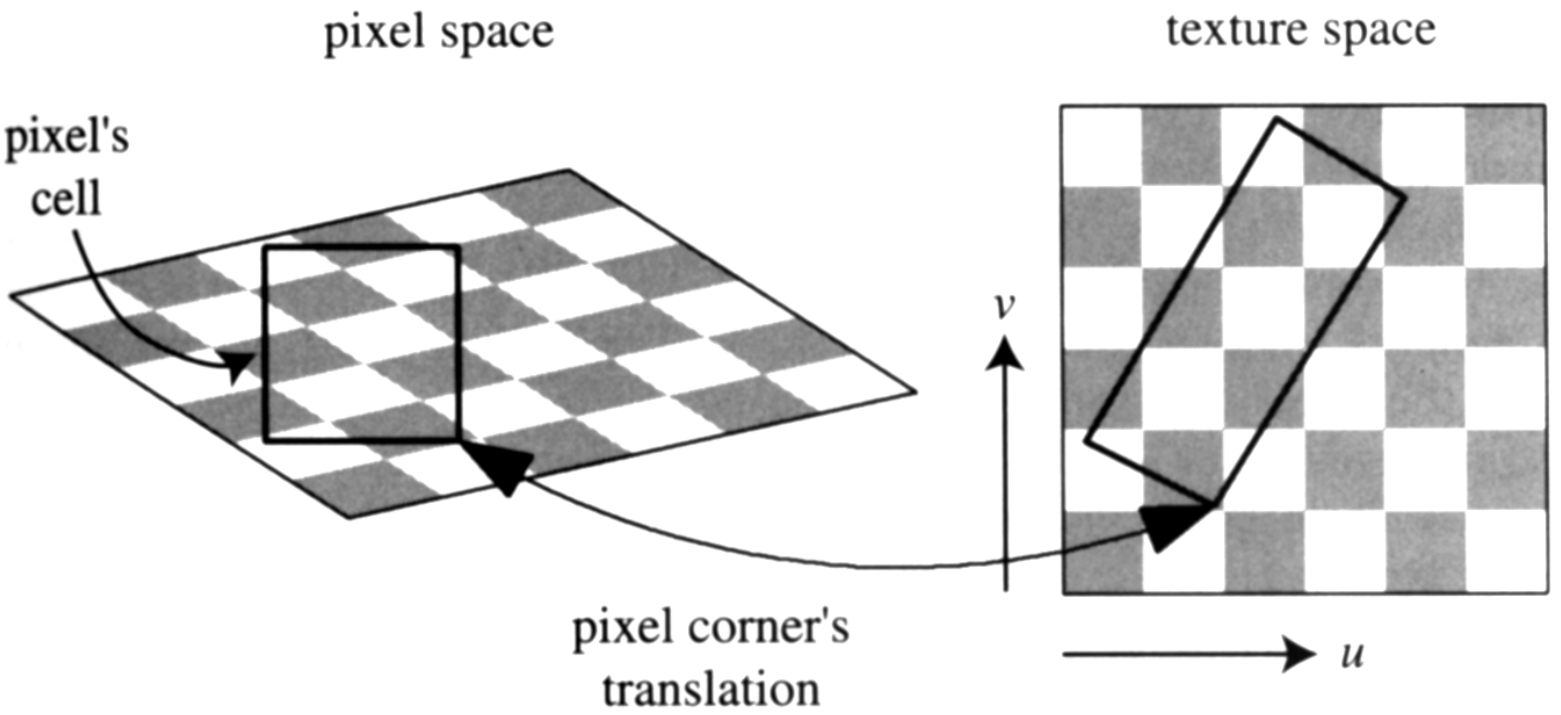
- kopie textury lze uspořádat do pomyslné pyramidy
- kromě texturových souřadnic u, v vypočítáme ještě parametr d - „výšku“ v pyramidě, tj. úroveň zmenšení
- vybereme buď kopii, která je nejblíže d, nebo interpolujeme mezi dvěma nejbližšími kopiemi $\Longrightarrow$ trilineární interpolace v u, v, d



Mipmapping

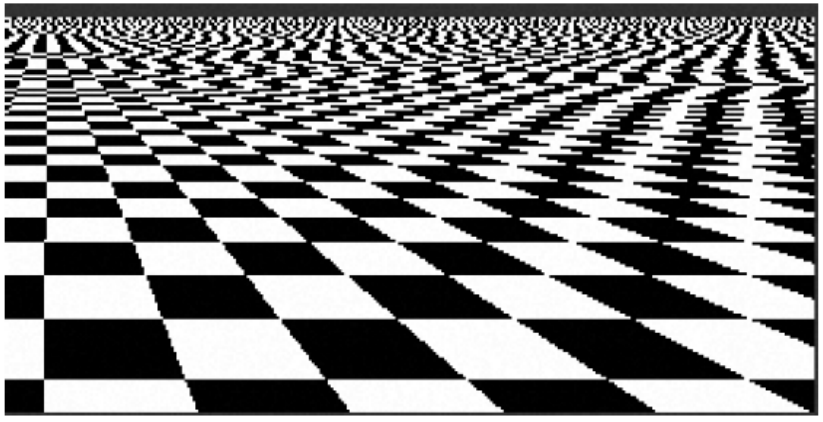
Výpočet úrovně zmenšení *d*

- 1 texel má přibližně odpovídat 1 pixelu
- obvykle hodnota *d* dána delší stranou čtyřúhelníku vzniklým zpětným mapováním pixelu (přes texturovaný polygon) do texturových souřadnic - pokud je to v původní textuře *t* texelů, je třeba zmenšit $\log_2 t$ - krát

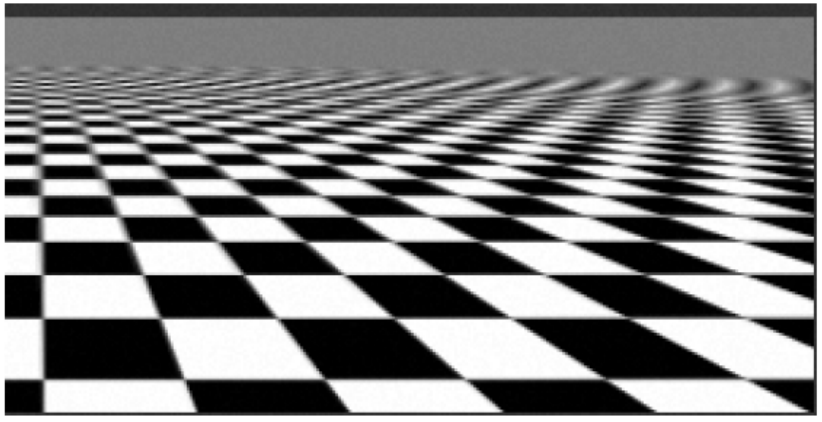


Mipmapping

Srovnání s přímým mapováním textury



Bodové vzorkování



Trilineární interpolace

Mipmapping

Vlastnosti

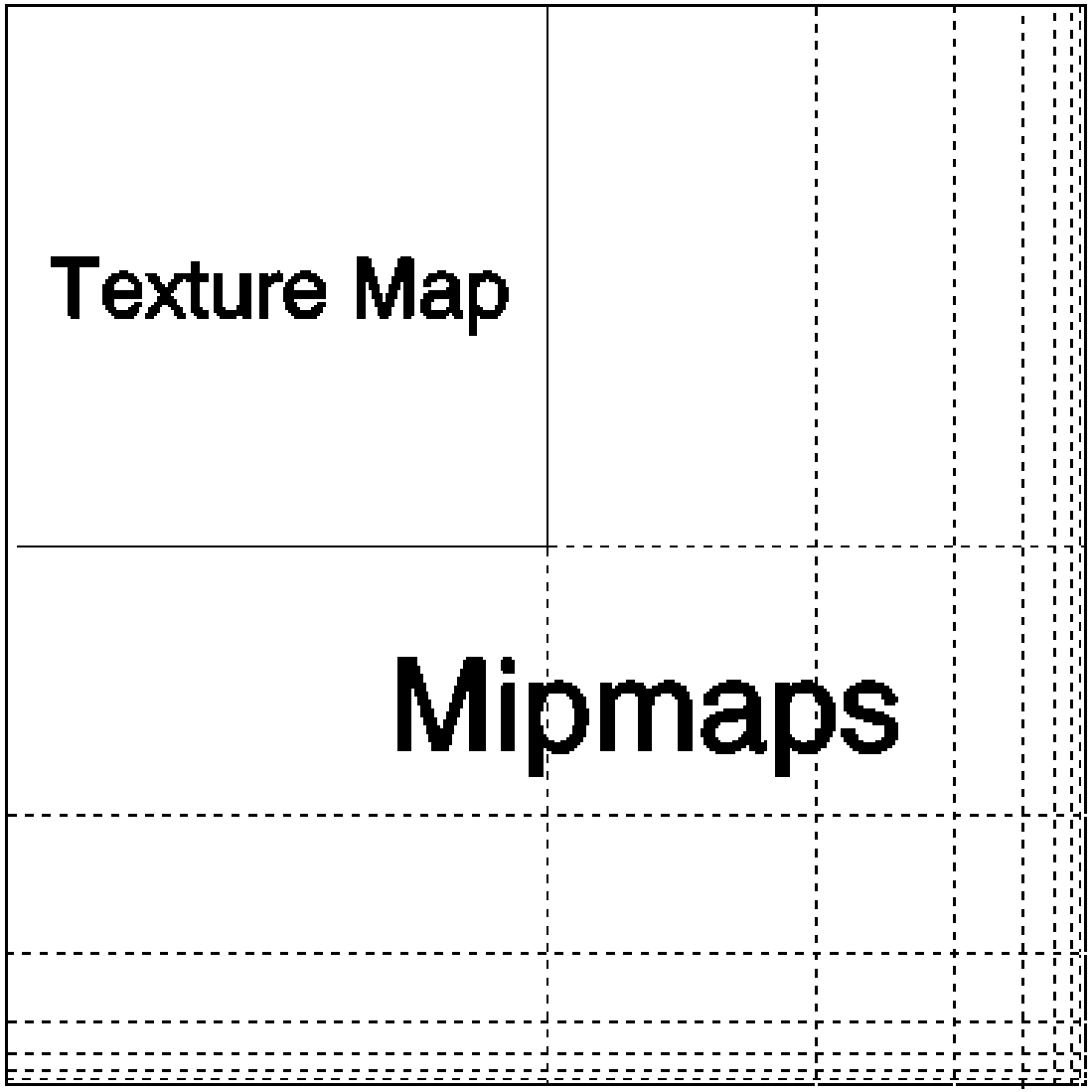
- **výhoda** mipmappingu - předpočítání textur (filtrování) lze provést kvalitně (a klidně pomalu), vlastní vykreslování je rychlé
- **nevýhoda** - při pohledu na polygon z velkého úhlu (od normály) dochází k rozmazání textury v obou směrech (nejen ve směru od pozorovatele)

řešení přílišného rozmazání

- rozlišit směr, ve kterém se textura „komprimuje“, od směru na něj kolmého
- ***ripmapping***
- **anizotropní filtrování**

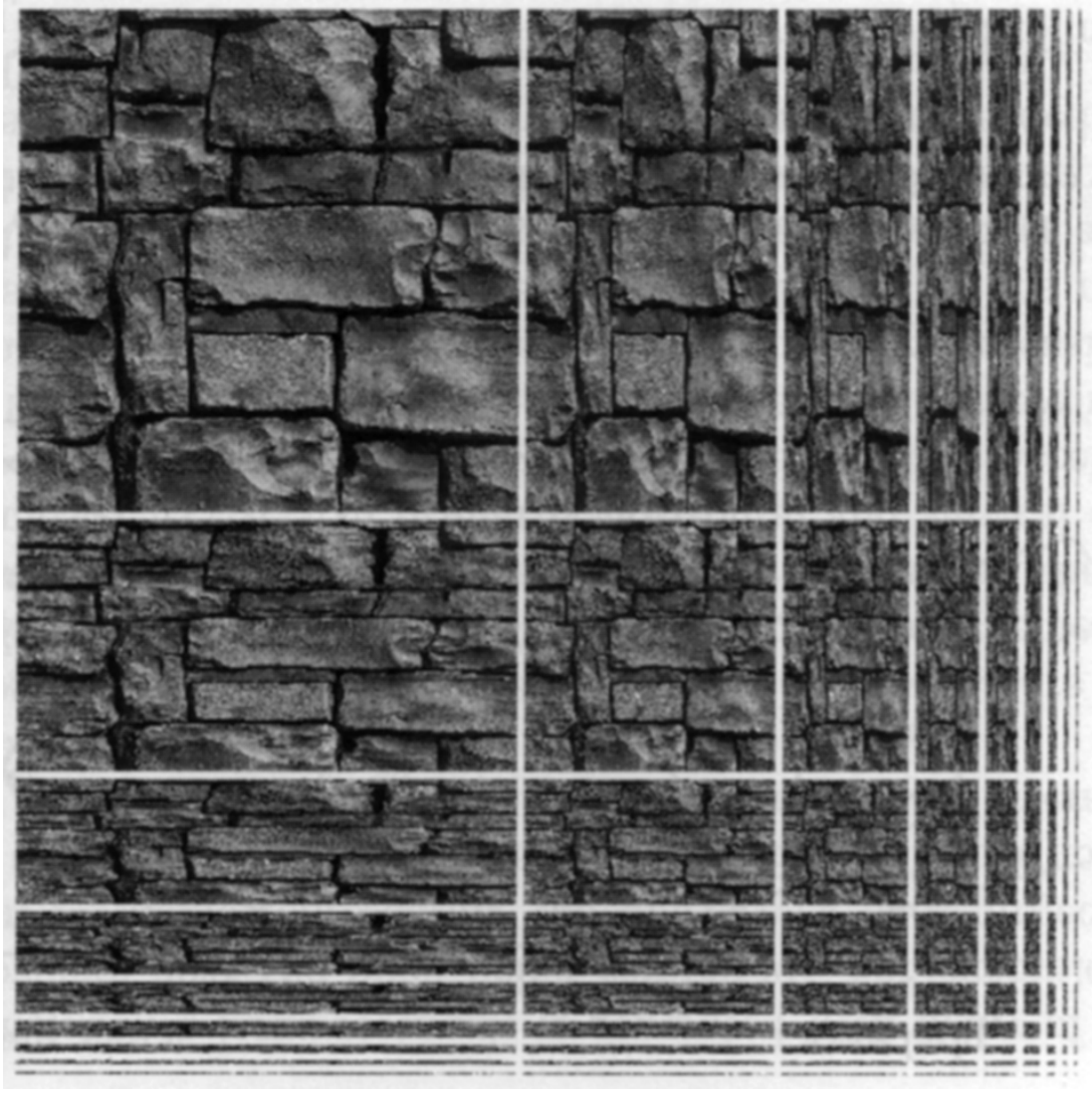
Ripmapping

- akcelerátory HP, 90. léta
- ukládáme nejen zmenšené kopie v obou směrech zároveň, ale také v každém ze směrů zvlášť
- dobře funguje jen pro orientace polygonů ve shodě s osami textury



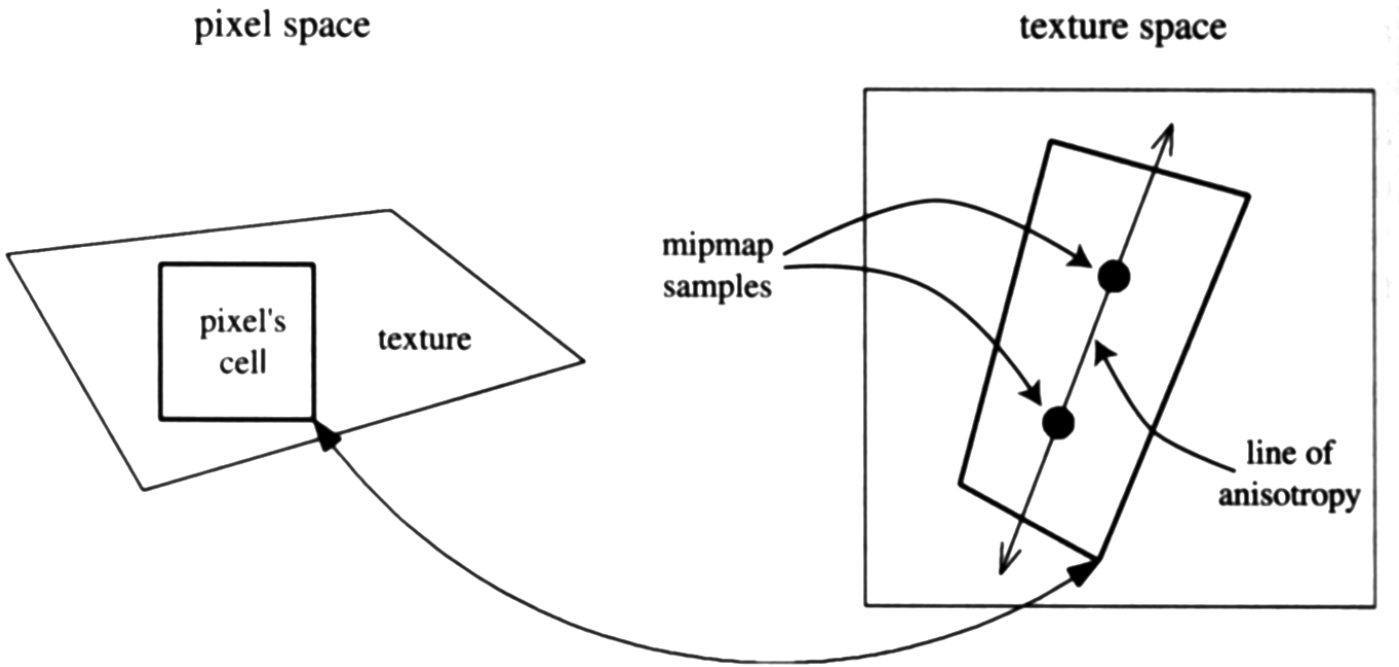
Ripmapping

- akcelerátory HP, 90. léta
- ukládáme nejen zmenšené kopie v obou směrech zároveň, ale také v každém ze směrů zvlášť
- dobře funguje jen pro orientace polygonů ve shodě s osami textury



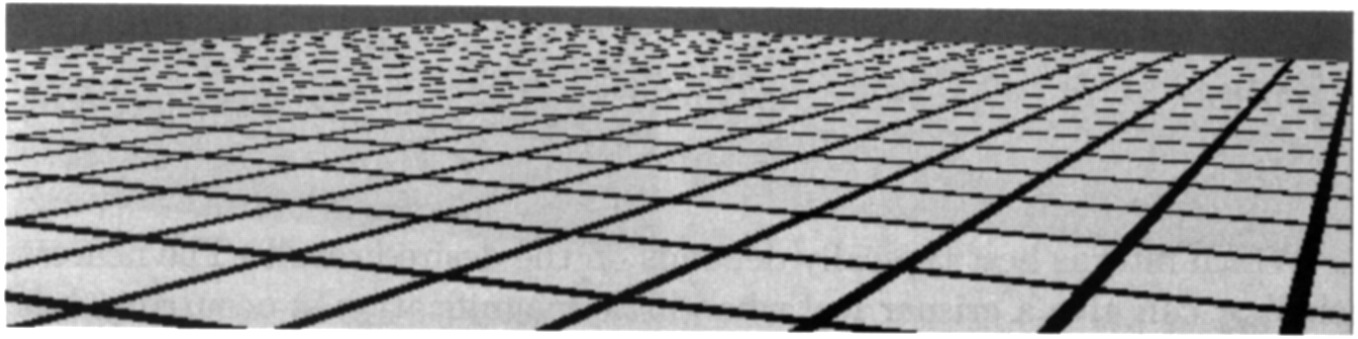
Anizotropní filtrování

- jako mipmapping (včetně trilineární interpolace), ale pro výpočet *d* se bere **kratší** strana promítnutého pixelu
- směr delší strany definuje *osu anizotropie*
- v mip-mapě (úrovni zmenšení) určené *d* se průměruje (nebo jinak filtruje) několik vzorků podél osy
  - jejich počet závisí na míře anizotropie = poměru delší a kratší strany promítnutého pixelu

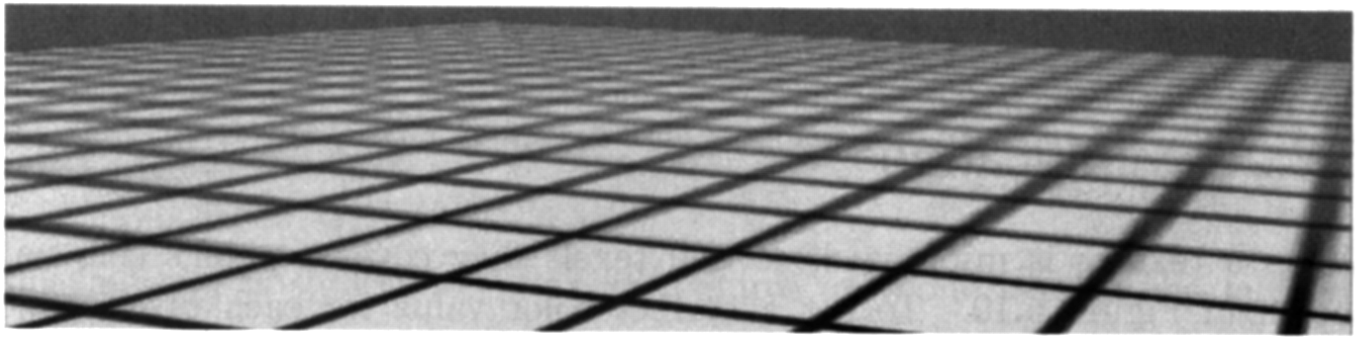


Anizotropní filtrování

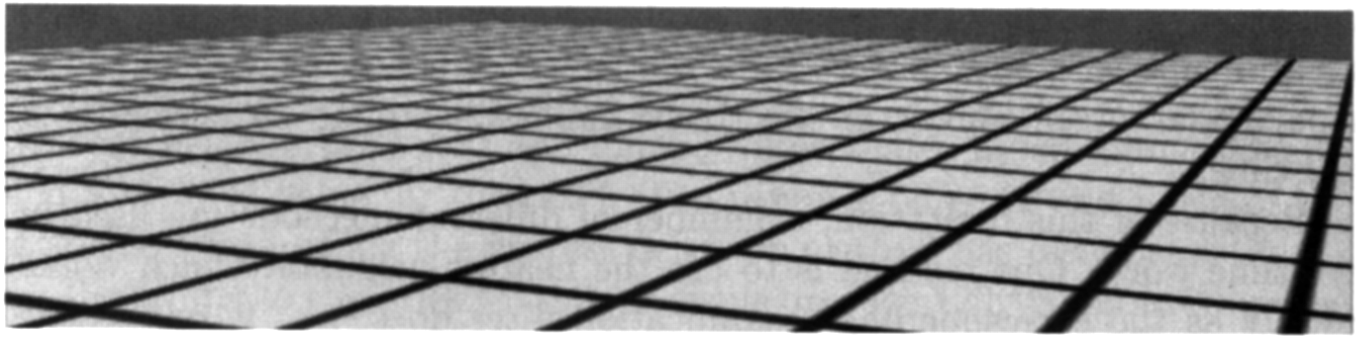
Příklady



Bodové vzorkování



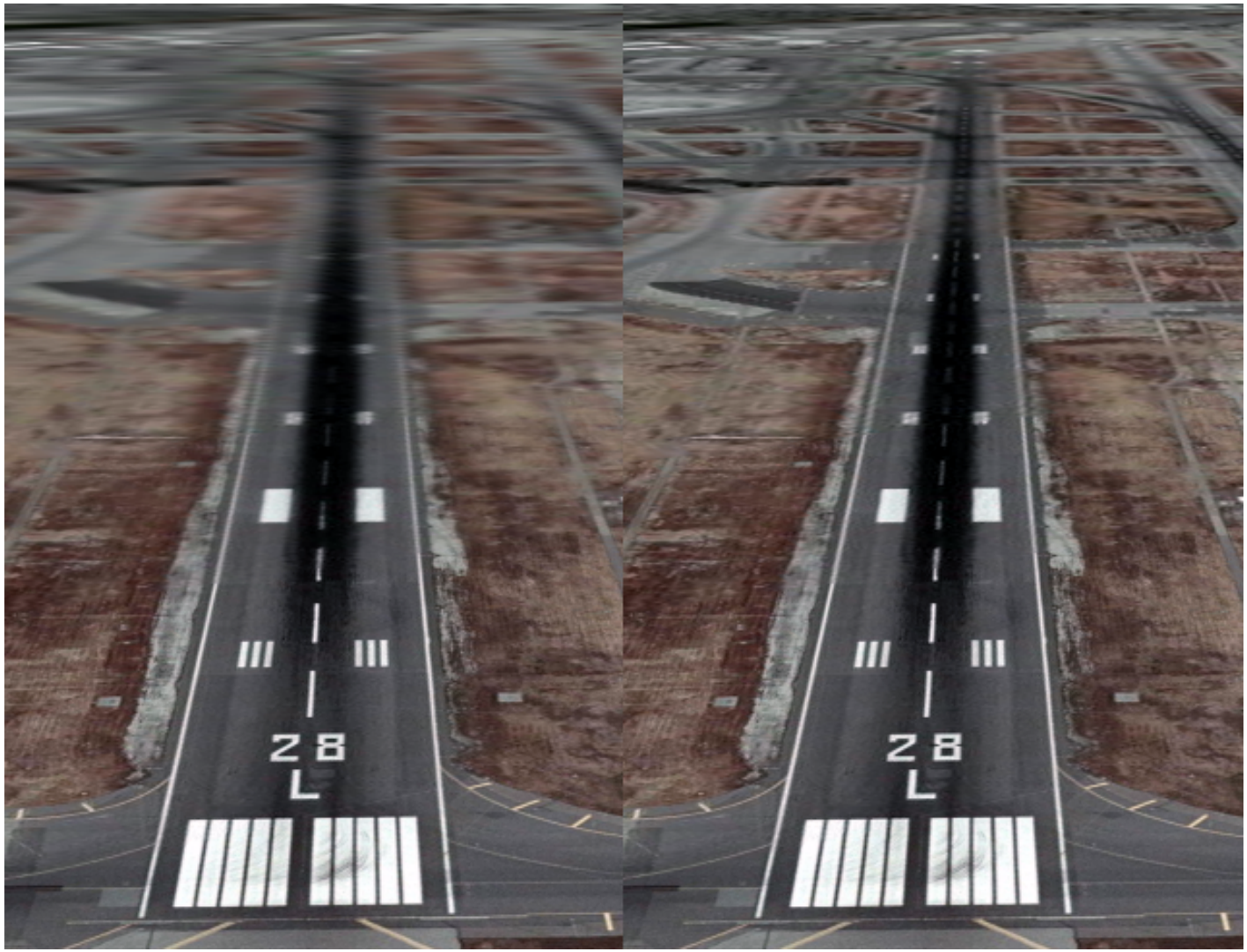
Mipmapping



Anizotropní filtrování

Anizotropní filtrování

Příklady



Literatura 1/2

- Žára, Beneš, Sochor, Felkel: *Moderní počítačová grafika*. Computer Press, 2005.
- J. Blinn: *Hyperbolic interpolation*. IEEE Computer Graphics and Applications, 1992.
- J. Blinn: *Simulation of wrinkled surfaces*. SIGGRAPH 1978, (1978) 286293.
- J. Blinn, M. Newell: *Texture and reflection in computer generated images*. Communications of the ACM, 19 (1976), 542547.
- S. R. Buss: *3D Computer Graphics: A Mathematical Introduction with OpenGL*. Cambridge University Press, 2003.

Literatura 2/2

N. Greene: *Environment mapping and other applications of world projections*. IEEE Computer Graphics and Applications, 6 (1986), 2129.

L. Williams: *Pyramidal Parametrics*. ACM SIGGRAPH Computer Graphics 7 (1983), 111.