KATEDRA
INFORMATIKY
UNIVERZITA PALACKÉHO V OLOMOUCI

Reprezentace
KMI/3DG

Mgr. Markéta Trnečková, Ph.D.

Těleso

- **Vágní definice**: Spojitý útvar skládající se z bodů v prostoru. Body patřící tělesu – vnitřní a hraniční
- **Vnitřní bod**
- **Hraniční bod** – sousedí alespoň s 1 vnitřním, 1 hraničním a 1 vnějším bodem
- **Vnější bod**
- **Sousednost** – zatím nepotřebujeme definovat, stačí intuitivní chápání

Příklad

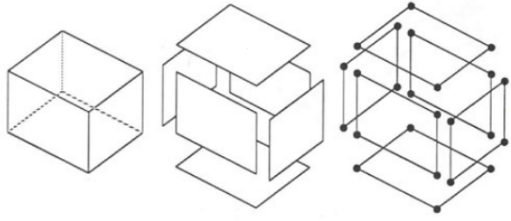
Jsou křivky tělesa? A co plochy?

Geometrie tělesa

- Tělesa můžeme reprezentovat pomocí
  - **Povrchu** (hraniční reprezentace):
    - Přesná reprezentace
    - Aproximace tvaru
  - **Postupu vytváření** – CSG stromem
  - **Objemová reprezentace**

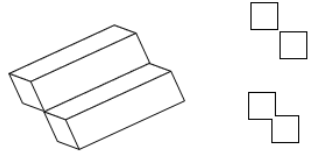
Hraniční reprezentace

- Nejrozšířenější
- Těleso popsáno jen pomocí hranice (hraničních bodů) – boundary representation **B-rep**
- Hranice může být zadána:
  - Přesně – pomocí ploch, plátů, ...
  - Pomocí polygonů – aproximují tvar tělesa



Hraniční reprezentace

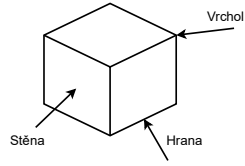
- Popis pomocí ploch umožňuje popsat objekt, který není tělesem (není možné ho v reálném světě vytvořit)
- Problémy:
  - nekonečně tenké hranice
  - části dotýkající se v jednom bodě, jedné úsečce a jiné
- Důležité pojmy:
  - **manifold** – každá hrana inciduje právě se 2 plochami, hrany neprotínají jiné plochy, osamocený vrchol nespojuje dvě části tělesa
  - **watertight** těleso – uzavřené těleso



Non-manifold

Mnohostěn

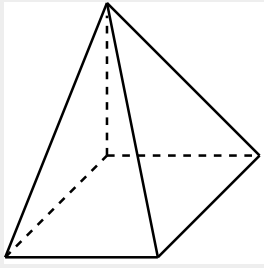
- Velká třída těles
- Stěny jsou tvořeny mnohoúhelníky
- Každou hranu sdílí sudý počet stěn (v případě 2-manifoldu právě 2)
- Splňuje další podmínky
- **Jednoduchý mnohostěn** – volnými deformacemi se dá převést na kouli (nemá díry)
- **Eulerova rovnost**: $F + V = E + 2$
- F = počet stěn, V = počet vrcholů, E = počet hran
- Pouze definice množin F, E, V nezaručuje, že se jedná o mnohostěn, hrany a stěny se nesmí dotýkat a každá hrana propojuje 2 vrcholy

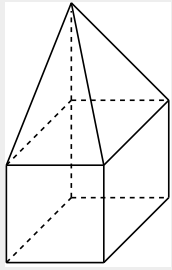


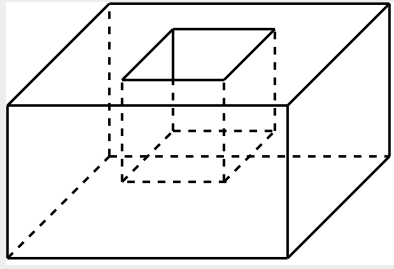
Mnohostěn

Příklad

Jsou následující tělesa mnohostěny?



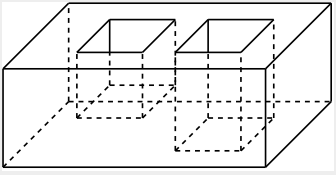




Mnohostěn

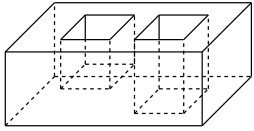
Příklad

Je následující těleso mnohostěn?



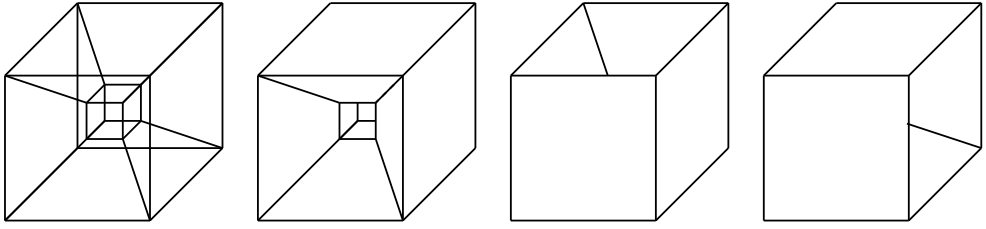
Mnohostěn

- **Mnohostěn s dírami**
- **Zobecněná Eulerova rovnost**:
  $F + V = E + 2(C - H) + R$
- F = počet stěn, V = počet vrcholů, E = počet hran
- C = počet komponent (samostatných oblastí)
- H = počet otvorů
- R = počet vnitřních smyček (ve stěně)



Hranová reprezentace

- Pouze hrany a vrcholy (drátové modely těles)
- 1 seznam vrcholů, 1 seznam hran (2 ukazatele do seznamu vrcholů)
- Nelze jednoznačně interpretovat
- Praktické použití – náhledy

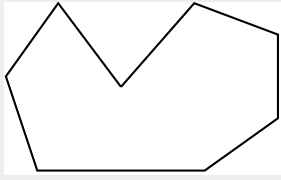


Jednoduchá plošková

- Předchozí datová struktura rozšířená o plochy – obecné polygony (ukazatele do seznamu vrcholů)
- Z implementačního hlediska je lepší používat trojúhelníky
- **Proč?**
- Euler dokázal, že kažný n-úhelník je možné triangulovat $n - 2$ trojúhelníky

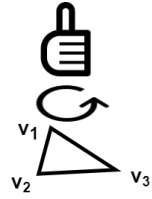
Příklad

Triangulujte:



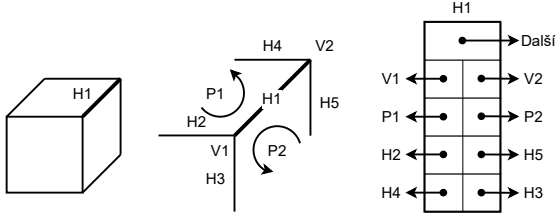
Jednoduchá plošková

- Pořadí vrcholů se volí dle orientace stěny tělesa
- Pravidlo pravé ruky – normálový vektor ukazuje ven
- V této reprezentaci nemáme informaci, které stěny spolu sousedí



Strukturovaná plošková reprezentace

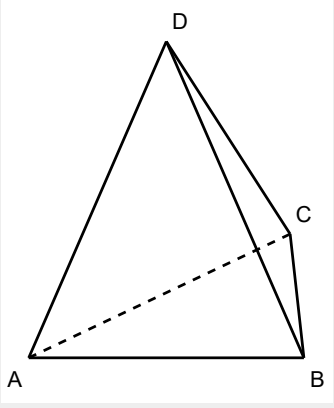
- **Okřídlená hrana** – struktura
- S každou hranou uchováváme i informaci o vrcholech, hranách a stěnách, které s ní sousedí (ukazatele)
- 3 seznamy – vrcholy (souřadnice), stěny (ukazatel na libovolnou hranu) a okřídlené hrany



Strukturovaná plošková reprezentace

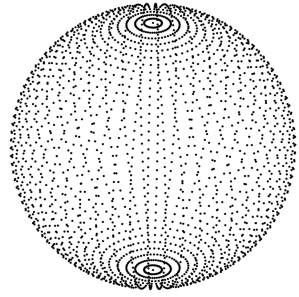
Příklad

Pro čtyřstěn napište, jak by vypadala strukturovaná plošková reprezentace.



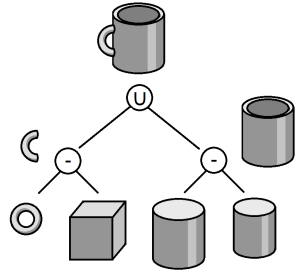
Bodová reprezentace

- Zvláštní případ hraniční
- Povrch = množina povrchových bodů
- Např. pomocí 3D skeneru, nebo vypočítané
- Každý bod nese informaci o části povrchu + barva a jiné



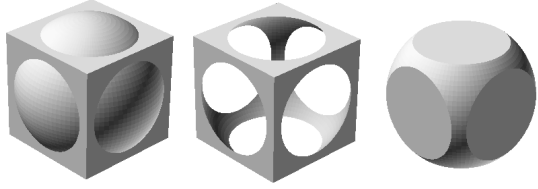
Konstruktivní geometrie

- **Constructive solid geometry** (CSG)
- Popisuje postup konstruktéra při navrhování tvaru tělesa (zejména v CAD)
- Těleso se reprezentuje **CSG stromem**
- Listy stromu = geometrická primitiva (kvádr, koule, ale i NURBS plochy a jiné)
- Ostatní uzly = transformace a množinové operace, které se aplikují na potomky
- Kořen stromu = výsledné těleso



Konstruktivní geometrie

- **Transformace**: posunutí, otočení, změna měřítka ...
- **Operace**: sjednocení, rozdíl, průnik ...



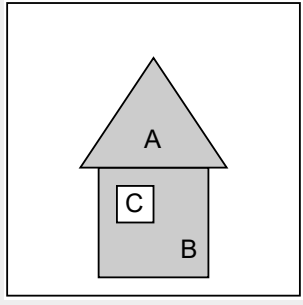
- Někdy se transformace provádí hned s primitivy a v uzlech jsou jen operace

Konstruktivní geometrie

- Tato reprezentace není vhodná na zobrazování – neobsahuje zobrazitelné prvky (plochy)
- Metody: převod na hraniční reprezentaci, sledování paprsku, upravený algoritmus pro paměť hloubky
- Vyhodnocení CSG stromu (nalezení hraniční reprezentace kořene) je složitý proces

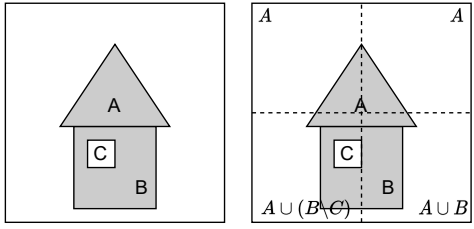
Příklad

Jak by vypadal konstrukční strom pro následující obrázek?



Konstruktivní geometrie

- Pokud zobrazujeme jen část prostoru, primitiva, která nejsou jeho součástí neovlivní zobrazený výsledek
- Není potřeba počítat výsledek pro celý strom, ale jen prořezaný strom



Příklad

Jak vypadají prořezané stromy pro jednotlivé části v předchozím obrázku?

Objemová reprezentace

- Nemáme geometrický popis tělesa, jen sadu vzorků v určitém objemu
- **Vzorek**: hodnotu (jas, barvu, hustotu, ...)
- Dělení objemových dat:
  - **Rozptýlená data** – každý vzorek má navíc souřadnice
  - **Pravidelné/nepravidelné mřížky**
- Např. CT – data v pravidelné pravoúhlé mřížce
- Např. při simulaci proudění – nepravidelné mřížky
- Např. meteorologická měření – rozptýlená data

Mřížky

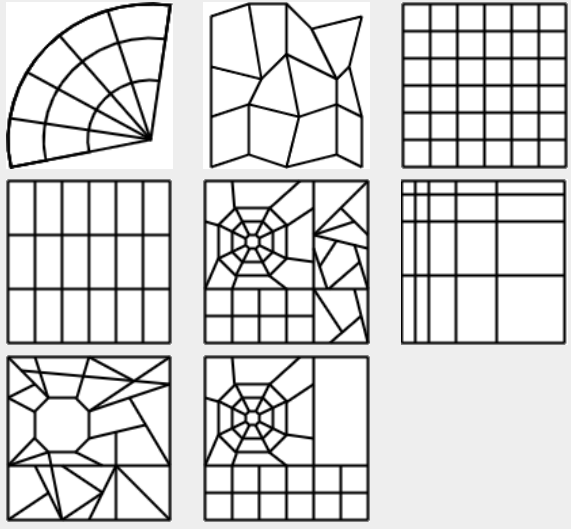
- Důležitý je tvar mřížky (doména)
- Pravidelné mřížky, pravidelné mřížky se zdeformovaným tvarem
- Nestrukturované musí mít uloženou topologii v dalším poli
- Každá buňka obsahuje odkazy na vrcholy
- Mřížka může mít libovolný počet vrcholů (dimenzionalita mřížky)

Mřížky

Příklad

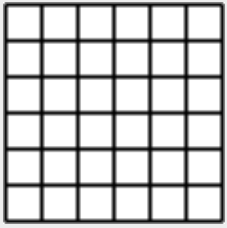
Přiřaďte názvy mřížek k obrázkům.

- Kartézská
- Pravidelná
- Zakřivená pravidelná (curvy linear)
- Pravoúhlá
- Strukturovaná
- Nestrukturovaná
- Blokově strukturovaná
- Hybridní

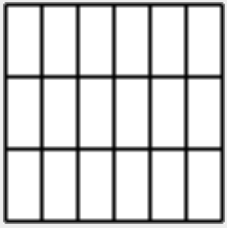


Mřížky

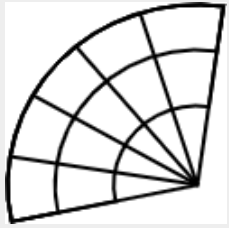
Příklad (Řešení)



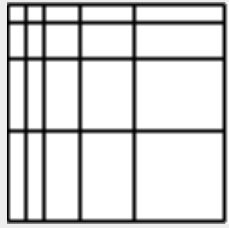
Kartézská



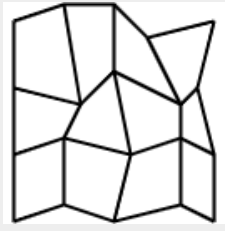
Pravidelná



Zakřivená pravidelná

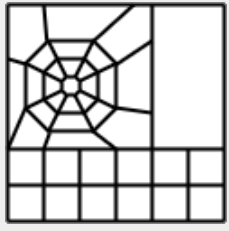


Pravoúhlá

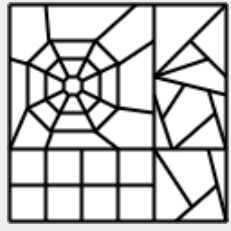


Strukturovaná

Nestrukturovaná



Blokově strukturovaná



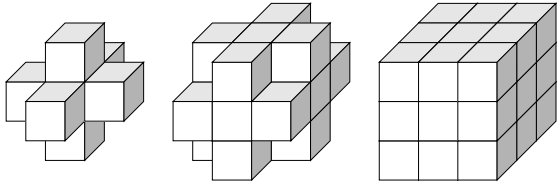
Hybridní

3D objekty v diskrétní mřížce

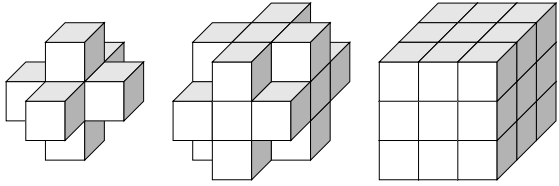
- Objemová data = velké paměťové nároky
- Jelikož jsou diskrétní, špatně se s nimi provádí transformace (otáčení o jiný než pravý úhel, změna velikosti)
- **Výhody**:
  - Snadná práce s daty – logické a blokové operace
  - Pracujeme s celým objemem najednou (bez ohledu an složitost scény)
  - Pro zobrazení stačí 1 algoritmus
- Dále se omezíme jen na skalární data v pravidelné prostorové mřížce

Voxel

- Voxel = nejmenší element ve 3D diskrétním prostoru – analogie pixelu (2D)
- Má tvar krychle (kvádru)
- Voxely jsou uspořádány v mřížce
- Voxel má v celém svém objemu konstantní hodnotu
- Je potřeba definovat sousednost (6-ti, 18-ti, 26-ti)



Voxel – sousednost

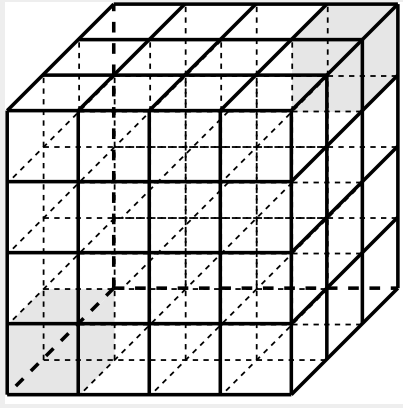


- Sousednost je důležitá pro hledání průsečíků, správné zobrazení (metoda sledování paprsku)
- Paprsek musí být 6-ti spojitý v případě 18-ti a 26-ti spojitých objektů (a naopak)
- Spojitost paprsku ovlivňuje jeho délku – čím je spojitost vyšší, tím je paprsek kratší (při výpočtu zpracováváme méně voxelů)

Voxel – sousednost

Příklad

Jaká je vzdálenost šedých voxelů pokud bereme v úvahu 6-ti, 18-ti a 26-ti spojitost?



Objemová reprezentace

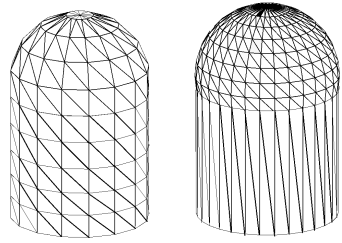
- **Zobrazení**:
  - Přímé zobrazení objemu
  - Hledání povrchu – vytvoření hraniční reprezentace

Trojúhelníky a sítě trojúhelníků

- Proč trojúhelník?
- Vrcholy jsou vždy v rovině
- Je konvexní
- Existují rychlé algoritmy pro jejich vykreslení, ale i jiné operace
- **Síť trojúhelníků** (triangle mesh) - síť trojúhelníků, které sdílí své hrany
- 2 části:
  - **Geometrická část** – souřadnice vrcholů
  - **Topologická část** – informace o tom, které vrcholy tvoří trojúhelník (případně, které spolu sousedí)
- Toto rozdělení je výhodné – transformace se provádí jen nad vrcholy (nezávisle na topologii)
- Snaha o co nejúspornější reprezentaci, ale zároveň se se sítí dobře pracovalo.

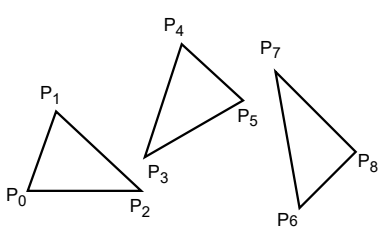
Trojúhelníky a sítě trojúhelníků

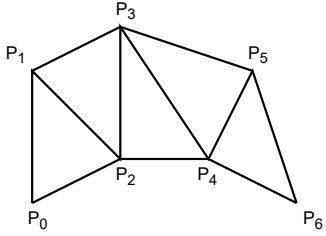
- Síť trojúhelníků většinou nepopisuje tvar přesně
- Při aproximaci povrchu trojúhelníky chceme, co nejpřesnější tvar a zároveň co nejmenší počet trojúhelníků

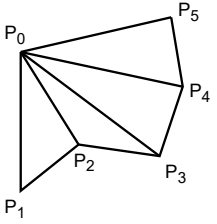


Trojúhelníky a sítě trojúhelníků

- Chceme provádět co nejméně operací s trojúhelníky (vrcholy)
- Trojúhelníky chceme mít v jedné struktuře
- **Struktury**:
  - Seznam trojúhelníků (triangle soup, triangle list)
  - Pás trojúhelníků – struktura je dána jen posloupností vrcholů; každý je zpracováván jen jen jednou; každý trojúhelník tvoří vrchol spolu se dvěma předchozími
  - Vějíř trojúhelníků – Vzniká např. při triangulaci mnohoúhelníku







Trojúhelníky a sítě trojúhelníků

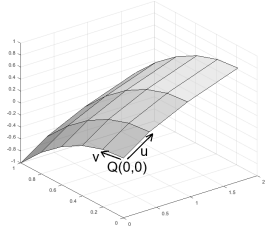
- Hledání optimálního vyjádření není snadná ani jednoznačná úloha
- Není snadné mapování textur
- **Geometrický alias** – projevuje se při změně měřítka, při chybě zaokrouhlení – dotýkající se trojúhelníky se nedotýkají

Plochy

- Popis přesné hranice objektu
- **Je dobré si zopakovat** – křivky z POGR
  - Zadávání křivek – implicitní, explicitní, parametrické
  - Tečna, směrový vektor
  - Segment křivky, uzel
  - Spojitost uzlů – parametrická, geometrická
  - Modelování křivek – interpolační a aproximační křivky
  - Interpolační – Hermitovské
  - Aproximační – Beziérovy

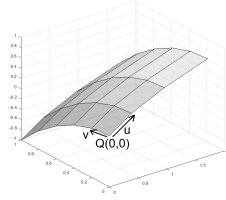
Parametrické plochy

- **Bodová rovnice plochy**:
  $Q(u, v) = [x(u, v), y(u, v), z(u, v)]$
- $x(u, v)$, $y(u, v)$, $z(u, v)$ – funkce parametrů u a v; $u, v \in \langle 0, 1 \rangle$
- Obvykle polynomiální (výhodné vlastnosti při modelování a navazování)
- Bod Q o souřadnicích $[x, y, z]$ v trojrozměrném kartézském prostoru má souřadnice $[u, v]$ v parametrickém prostoru
- Používáme shodné matematické prostředky jako u polynomiálních parametrických křivek



Parametrické plochy

- **Tečný vektor**: $\overrightarrow{q_u}(u,v)$, $\overrightarrow{q_v}(u,v)$
- Tečné vektory k ploše $Q(u,v)$ ve směru parametru
- $\overrightarrow{q_u}(u,v) = \dfrac{\partial(Q(u,v))}{\partial u} = \left(\dfrac{\partial x(u,v)}{\partial u}, \dfrac{\partial y(u,v)}{\partial u}, \dfrac{\partial z(u,v)}{\partial u}\right)$
- $\overrightarrow{q_v}(u,v) = \dfrac{\partial(Q(u,v))}{\partial v} = \left(\dfrac{\partial x(u,v)}{\partial v}, \dfrac{\partial y(u,v)}{\partial v}, \dfrac{\partial z(u,v)}{\partial v}\right)$
- **Parametrická rovnice tečné roviny**: $T(r,s) = Q(u,v) + r \cdot \overrightarrow{q_u}(u,v) + s \cdot \overrightarrow{q_v}(u,v);\ r,s \in \mathbb{R}$



Parametrické plochy

- **Plát** – část plochy, jako segment u křivek
- **Plátování** – spojení plátů
- Při navazování hrají roli **zkruty** (zkrutové vektory) charakterizující vyklenutí plochy v místech, kde se pláty napojují
- $\overrightarrow{q_{uv}}(u,v) = \dfrac{\partial^2(Q(u,v))}{\partial u\,\partial v} = \left(\dfrac{\partial^2 x(u,v)}{\partial u\,\partial v}, \dfrac{\partial^2 y(u,v)}{\partial u\,\partial v}, \dfrac{\partial^2 z(u,v)}{\partial u\,\partial v}\right)$
- **Normála v bodě**: $\overrightarrow{n}(u,v) = \frac{\overrightarrow{q_u}(u,v) \times \overrightarrow{q_v}(u,v)}{||\overrightarrow{q_u}(u,v) \times \overrightarrow{q_v}(u,v)||}$

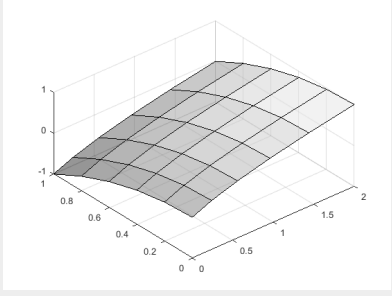
Parametrické plochy

Pojmy

- **Hlavní křivka plochy ve směru** u – každá křivka určená $Q(u, k)$ pevného parametru k
- **Hlavní křivka plochy ve směru** v – analogicky
- **Rohy** – body $Q(0, 0)$, $Q(0, 1)$, $Q(1, 0)$, $Q(1, 1)$
- **Strany** – hlavní křivky ve směru u s parametrem $v = 0$ nebo $v = 1$ a hlavní křivky ve směru v s parametrem $u = 0$ a $u = 1$
- **Okraj** – všechny strany plochy

Příklad

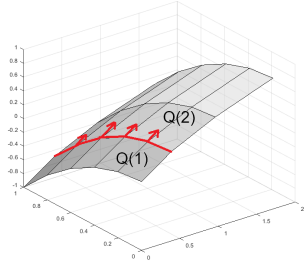
Zakreslete pojmy do obrázku.



Parametrické plochy

Plátování

- **Plátování** – napojení více plátů
- **Spojitost**:
  - Parametrická – C^i
  - Geometrická – G^i
- C^0 (G^0) = pláty mají společnou stranu, která je křivkou třídy alespoň C^0 (G^0)
- C^1 = pláty mají společnou stranu a shodné příčné parciální derivace ve všech bodech společné strany obou plátů ($Q(1)$ a $Q(2)$)
- G^1 = pláty mají společnou stranu, která je alespoň G^1 křivkou a příčné parciální derivace ve všech bodech společné strany obou plátů ($Q(1)$ a $Q(2)$) jsou lineárně závislé (s kladným koeficientem, který se spojitě mění podél společné stěny)



Zadávání ploch

- Řídícími body a bázovými funkcemi
- **Interpolační plochy**
  - Interpolace ve vyšších dimenzích než 2 je složitá
  - Ve 3D se používá interpolace ploškami – většinou trojúhelníky (**Surface fitting**)
- **Aproximační plochy**
  - Řídící body určují tvar, ale nemusí jimi plocha procházet
  - Složitější napojování – nutno znát tečné podmínky na stranách ploch
- Bázové funkce ploch jsou většinou polynomy, ne nutně stejného stupně v obou směrech
- Pro pojmenování se používá stupeň polynomu – např. bikubická plocha, kvadraticko-lineární plocha
- Nejčastěji se používají polynomy stupně 3 – Polynomy menšího stupně nelze určit tak, aby měl při procházení 2 body vhodné vlastnosti tečen
- Kubika je nejmenší stupeň, který umožňuje C^1 a C^2 spojitost
- Vyšší stupně = složitější výpočty

Vlastnosti ploch

- Invariance k lineárním transformacím – při aplikaci transformace na řídící body a následné vykreslení plochy dá stejný výsledek, jako aplikace transformace na plochu po jejím vykreslení
- Konvexní obálka
  - Silná – plocha leží v konvexní obálce všech svých řídících bodů
  - Slabá – část plochy (plát) leží v konvexní obálce některých svých řídících bodů (plát leží v konvexní obálce svých řídících bodů)
- Lokalita změn – změnou polohy řídícího bodu se mění jen část plochy, nikoli plocha celá
- Plocha může procházet krajními body sítě řídících bodů

Interpolační plochy

- Obtížná úloha
- Mějme $(m+1) \times (n+1)$ řídících bodů P_{ij}
- Interpolační plocha: $P(u, v)$
- Existuje řešení rovnic:
  $P(u, v) = P_{ij}$ pro $\forall i, j$
- Tedy prochází body, kterými je zadaná
- Při interpolaci polynomy: $P(u, v) = \sum_{i=0}^{m} \sum_{j=0}^{n} b_{ij} u^i v^j$
- Maticově:
  $P(u, v) = U \cdot B \cdot V^T$
  kde $u = [u^m, u^{m-1}, \ldots u, 1]$ a $v = [v^n, v^{n-1}, \ldots v, 1]$
- Což je $(m+1) \cdot (n+1)$ rovnic o stejném počtu neznámých

Příklad

Kolik bodů potřebujeme pro jednoznačné zadání bikubické interpolační plochy?

Interpolační plochy

Příklad

Vypočítejte rovnici bilineární plochy, která interpoluje body $P_{i,j}$ zadané následující tabulkou.

		v	
		0	1
u	0	1	2
	1	2	1

Aproximační plochy

- Plochy jsou zadány body, případně tečnými vektory nebo zkruty a polynomy modelujícími plochy
- Pro zjednodušení si zavedeme následující značení:
  - $\cdot$ bod
  - $\rightarrow$, $\uparrow$ tečné vektory (v jednotlivých osách)
  - $\nearrow$, $\nwarrow$, $\searrow$,$\swarrow$ zkruty

Hermitovské plochy

- Zadány rohy (body) a tečnými vektory v nich (případně ještě zkruty)
- Tvar je řízen Hermitovskými polynomy třetího stupně
- **Definice Hermitovského polynomu**:
  $F_1(t) = 2t^3 - 3t^2 + 1$
  $F_2(t) = -2t^3 + 3t^2$
  $F_3(t) = t^3 - 2t^2 + t$
  $F_4(t) = t^3 - t^2$

Dvanáctivektorové plochy

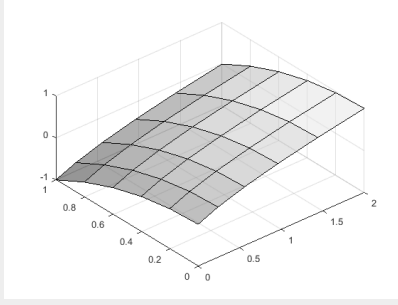
- 4 body (P_{00}, P_{01}, P_{10} a P_{11})
- Tečnými vektory v nich v obou směrech ($\overrightarrow{p_u}(0,0)$, $\overrightarrow{p_v}(0,0)$, $\ldots$, $\overrightarrow{p_u}(1,1)$, $\overrightarrow{p_v}(1,1)$)
- **Definice**:
  $Q(t) = [F_3(u), F_1(u), F_2(u), F_4(u)] \cdot M \cdot [F_3(v), F_1(v), F_2(v), F_4(v)]^T$
- F_i Hermitovské polynomy
- M mapa plochy

$$\begin{bmatrix} & \uparrow & \uparrow & \\ \rightarrow & \cdot & \cdot & \rightarrow \\ \rightarrow & \cdot & \cdot & \rightarrow \\ & \uparrow & \uparrow & \end{bmatrix} \begin{bmatrix} 0 & \overrightarrow{p_u}(0,0) & \overrightarrow{p_u}(0,1) & 0 \\ \overrightarrow{p_v}(0,0) & P_{00} & P_{01} & \overrightarrow{p_v}(0,1) \\ \overrightarrow{p_v}(1,0) & P_{10} & P_{11} & \overrightarrow{p_v}(1,1) \\ 0 & \overrightarrow{p_u}(1,0) & \overrightarrow{p_u}(1,1) & 0 \end{bmatrix}$$

Dvanáctivektorové plochy

Příklad

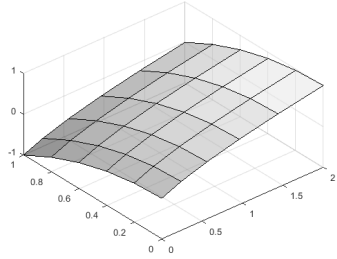
Zakreslete do obrázku body P_{00}, P_{01}, P_{10} a P_{11} a vektory $\overrightarrow{p_u}(0,0)$, $\overrightarrow{p_v}(0,0)$, $\ldots$, $\overrightarrow{p_u}(1,1)$, $\overrightarrow{p_v}(1,1)$



Dvanáctivektorové plochy

Napojení

- Krajní body leží na ploše
- Pokud se krajní body rovnají – C^0
- Pokud se rovnají tečné vektory – C^1 (sloupce 1 a 4 případně řádky 1 a 4)

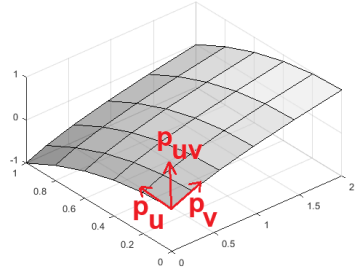


Šestnáctivektorové plochy

- Ve 12ti vektorové matici nahradíme 0 zkruty $\overrightarrow{p_{uv}}(0,0)$, $\overrightarrow{p_{uv}}(0,1)$, $\overrightarrow{p_{uv}}(1,0)$, $\overrightarrow{p_{uv}}(1,1)$
- Mapa plochy

$$\begin{bmatrix} \nwarrow & \uparrow & \uparrow & \nearrow \\ \rightarrow & \cdot & \cdot & \rightarrow \\ \rightarrow & \cdot & \cdot & \rightarrow \\ \swarrow & \uparrow & \uparrow & \searrow \end{bmatrix}$$

- Zkruty řídí vyklenutí rohu v plátu
- Navazování plátů stejné jako u 12ti vektorové
- Díky zkrutům je možné zajistit i G^1 mezi 4 pláty dotýkajících se v rozích



Plochy spojující dvě křivky

- Úkolem je vytvořit plochu tak, aby spojovala jiné plochy
- Propojit jejich strany (strany ploch = křivky)
- Přímková plocha, kubická plocha

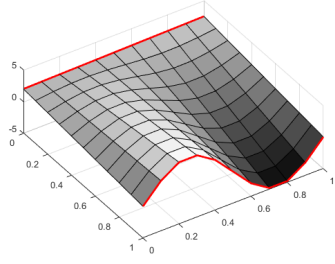
Přímková plocha

- Plochy spojuje přímkami
- Hledaná plocha je zadána dvěma okrajovými křivkami – $P(0, v)$ a $P(1, v)$, $v \in \langle 0, 1 \rangle$ (případně $P(u, 0)$ a $P(u, 1)$, $u \in \langle 0, 1 \rangle$)
- Rovnice plochy spojující křivky ($u \in \langle 0, 1 \rangle$): $Q(u, v) = (1 - u) \cdot P(0, v) + u \cdot P(1, v)$
- Snadno ověříme, že plocha spojuje okrajové křivky (Jak?)
- Jiný zápis

$$Q(u, v) = [1 - u, u] \cdot \begin{bmatrix} P(0, v) \\ P(1, v) \end{bmatrix}$$

- Mapa plochy:

$$\begin{bmatrix} \cdot \\ \cdot \end{bmatrix}$$



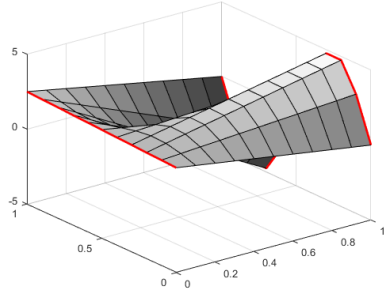
Kubická plocha

- Hledaná plocha je zadána:
  - Dvěma okrajovými křivkami – $P(0, v)$ a $P(1, v)$, $v \in \langle 0, 1 \rangle$
  - Dvěma funkcemi pro tečné vektory podél okrajových křivek $\overrightarrow{p_u}(0, v)$ a $\overrightarrow{p_u}(1, v)$, $v \in \langle 0, 1 \rangle$
- Vektory můžeme převzít z ploch, které touto plochou spojujeme – tím zajistíme spojitost C^1
- Rovnice plochy:
  $Q(u, v) = F1(u) \cdot P(0, v) + F2(u) \cdot P(1, v) + F3(u) \cdot \overrightarrow{p_u}(0, v) + F4(u) \cdot \overrightarrow{p_u}(1, v)$
- $F1$, $F2$, $F3$, $F4$ = Hermitovské polynomy 3. stupně (viz slide 43)

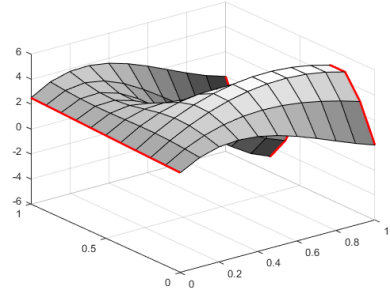
Příklad

Jak vypadá maticový zápis plochy?
Jak vypadá mapa plochy?

Kubická plocha



Přímková plocha



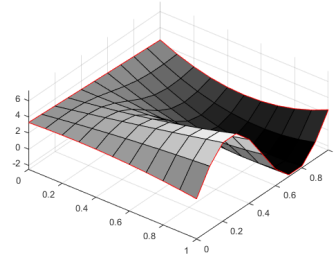
Kubická plocha

Plochy zadané okrajem

- Hledáme plochy, které jsou zadány 4 stranami (celým okrajem)
- Plochy:
  - Bilineární Coonsova plocha
  - Bikubická plocha
  - Obecná bikubická plocha (Coonsova)

Bilineární Coonsova plocha

- Hledaná plocha je zadána 4 okrajovými křivkami – $P(u,0)$, $P(u,1)$, $P(0,v)$ a $P(1,v)$, $u,v \in \langle 0,1 \rangle$
- Křivky musí tvořit uzavřený okraj (Co musí platit?)
- Rovnice plochy spojující křivky (implicitní zadání):
  $[1-u,-1,u] \cdot C \cdot [1-v,-1,v]^T = 0$
  $$C = \begin{bmatrix} P_{00} & P(0,v) & P_{01} \\ P(u,0) & Q(u,v) & P(u,1) \\ P_{10} & P(1,v) & P_{11} \end{bmatrix}$$
- Implicitní zadání není vhodné pro generování ploch



Příklad

Jak vypadá rovnice bilineární Coonsovy plochy?

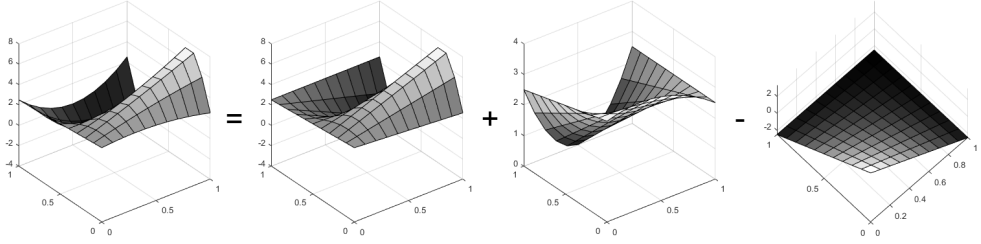
Bilineární Coonsova plocha

Příklad

Jak vypadá rovnice bilineární Coonsovy plochy?

- Řešení:

$$Q(u,v) = [1-u, u]\cdot\begin{bmatrix} P(0,v) \\ P(1,v) \end{bmatrix} + [P(u,0), P(u,1)]\cdot\begin{bmatrix} 1-v \\ v \end{bmatrix} - [1-u, u]\cdot\begin{bmatrix} P_{00} & P_{01} \\ P_{10} & P_{11} \end{bmatrix}\cdot\begin{bmatrix} 1-v \\ v \end{bmatrix}$$

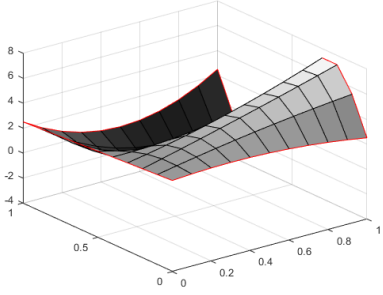


Bikubická plocha

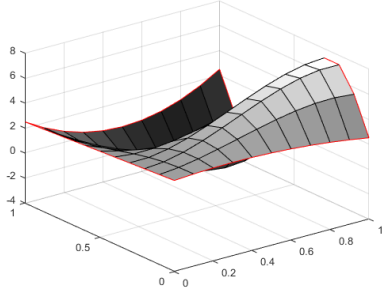
- Hledaná plocha je zadána 4 okrajovými křivkami – $P(u,0)$, $P(u,1)$, $P(0,v)$ a $P(1,v)$, $u,v \in \langle 0,1 \rangle$
- Křivky musí tvořit uzavřený okraj (Co musí platit?)
- Rovnice plochy spojující křivky (implicitní zadání):
  $[F1(u), -1, F2(u)] \cdot C \cdot [F1(v), -1, F2(v)]^T = 0$
  $$C = \begin{bmatrix} P_{00} & P(0,v) & P_{01} \\ P(u,0) & Q(u,v) & P(u,1) \\ P_{10} & P(1,v) & P_{11} \end{bmatrix}$$
- $F1$ a $F2$ Hermitovské polynomy (viz slide 43)
- Porovnejte s Bilineární Coonsovou plochou
- Explicitní vyjádření:
  $$Q(u,v) = [F1(u), F2(u)] \cdot \begin{bmatrix} P(0,v) \\ P(1,v) \end{bmatrix} + [P(u,0), P(u,1)] \cdot \begin{bmatrix} F1(v) \\ F2(v) \end{bmatrix} - [F1(u), F2(u)] \cdot \begin{bmatrix} P_{00} & P_{01} \\ P_{10} & P_{11} \end{bmatrix} \cdot \begin{bmatrix} F1(v) \\ F2(v) \end{bmatrix}$$

Bikubická plocha

- Nepoužívá se pro plátování – těžko se zachovává spojitost C^1 i G^1 (Proč?)



Bilineární Coonsova plocha



Bikubická plocha

Obecná bikubická plocha – Coonsova

- Je možné zajistit C^1 spojitost
- Je nutné zadat tečné vektory u okraje
- Je určena:
  - Okrajovými křivkami – $P(u,0)$, $P(u,1)$ $P(0,v)$ a $P(1,v)$, $u,v \in \langle 0,1 \rangle$
  - Funkcemi pro tečné vektory podél okrajových křivek – $\overrightarrow{p_u}(0,v)$, $\overrightarrow{p_u}(1,v)$, $\overrightarrow{p_v}(u,0)$ a $\overrightarrow{p_v}(u,1)$, $u,v \in \langle 0,1 \rangle$
  - Zkruty v rozích – $\overrightarrow{p_{uv}}(0,0)$, $\overrightarrow{p_{uv}}(0,1)$, $\overrightarrow{p_{uv}}(1,0)$, $\overrightarrow{p_{uv}}(1,1)$
- Jsou potřeba:
  - Rohové body P_{00}, P_{10}, P_{01} a P_{11}
  - Hodnoty tečných příčných vektorů v bodech okraje P_{0v}, P_{1v}, P_{u1} a P_{u1}
    Získáme je dosazením $u=0$, $u=1$, $v=0$ a $v=1$ do $\overrightarrow{p_v}(u,0)$, $\overrightarrow{p_v}(u,1)$, $\overrightarrow{p_u}(0,v)$ a $\overrightarrow{p_u}(1,v)$
  - Plocha prochází okrajem
  - Má hladká napojení díky zkrutům a tečným vektorům

Obecná bikubická plocha – Coonsova

- $[F3(u), F1(u), F2(u), F4(u)] \cdot C \cdot [F3(v), F1(v), F2(v), F4(v)]^T = 0$
- $F1$, $F2$, $F3$ a $F4$ Hermitovské polynomy (viz slide 43)
- $C = \begin{bmatrix} \overrightarrow{p_{uv}}(0,0) & \overrightarrow{p_u}(0,0) & \overrightarrow{p_u}(0,v) & \overrightarrow{p_u}(0,1) & \overrightarrow{p_{uv}}(0,1) \\ \overrightarrow{p_v}(0,0) & P_{00} & P_{0v} & P_{01} & \overrightarrow{p_v}(0,1) \\ \overrightarrow{p_v}(u,0) & P_{u0} & Q(u,v) & P_{u1} & \overrightarrow{p_v}(u,1) \\ \overrightarrow{p_v}(1,0) & P_{10} & P_{1v} & P_{11} & \overrightarrow{p_v}(1,1) \\ \overrightarrow{p_{uv}}(1,0) & \overrightarrow{p_u}(1,0) & \overrightarrow{p_u}(1,v) & \overrightarrow{p_u}(1,1) & \overrightarrow{p_{uv}}(1,1) \end{bmatrix}$
- Mapa plochy

$$\begin{bmatrix} \nwarrow & \uparrow & \uparrow & \uparrow & \nearrow \\ \rightarrow & \cdot & \cdot & \cdot & \rightarrow \\ \rightarrow & \cdot & \cdot & \cdot & \rightarrow \\ \rightarrow & \cdot & \cdot & \cdot & \rightarrow \\ \swarrow & \uparrow & \uparrow & \uparrow & \searrow \end{bmatrix}$$

Beziérovy plochy

- Zobecnění Beziérových křivek (jsou speciálním případem Beziérových ploch)
- Operace nad nimi jsou snadné (tyto plochy se snadno modelují i renderují)
- V současné době jsou spíše nahrazeny NURBS plochami
- Beziérova plocha $n \times m$ tého stupně je určena $(n+1) \cdot (m+1)$ body P_{ij} a vztahem $Q(u,v) = \sum_{i=0}^{n} \sum_{j=0}^{m} P_{ij} \cdot B_i^n(u) B_j^m(v)$
- B_i^k jsou bázové funkce – Bernsteinovy polynomy k-tého stupně

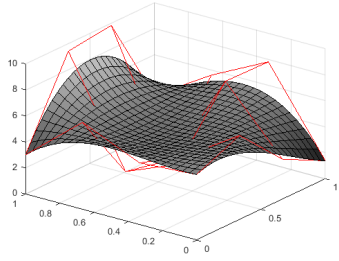
$$B_i^k(t) = \binom{k}{i} t^i (1-t)^{k-i}$$

Příklad

Jak vypadají Bernsteinovy polynomy 3. stupně?

Beziérovy plochy

- Dosadíme-li za u a v 0 respektive 1 zjistíme, že okraje jsou Beziérovy křivky
- Rohy plochy jsou totožné s body P_{00}, P_{01}, P_{10} a P_{11} řídící sítě
- **Tečný vektor** v rohu $Q(0,0)$:
  $\overrightarrow{q_u} = n \cdot (P_{10} - P_{00})$
  $\overrightarrow{q_v} = m \cdot (P_{01} - P_{00})$
  (dosazením $u = 0$ a $v = 0$ do derivace)
- **Normála v bodě** $Q(u,v)$ – vektorový součin tečných vektorů v tomto bodě



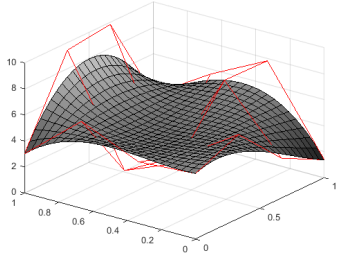
Beziérovy plochy

Vlastnosti

- Bernsteinovy polynomy jsou nezáporné a jejich součet je roven 1 $\rightarrow$ plocha leží v konvexní obálce svých řídících bodů
- Při změně jednoho bodu (jeho polohy), změní celý svůj tvar

Příklad

Viz matlab.



Beziérovy plochy

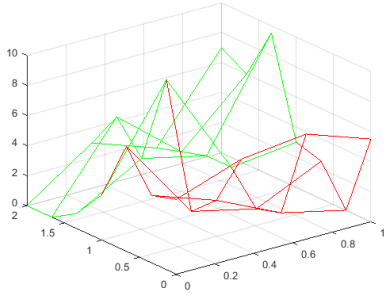
Navazování

- Pláty $Q(1)$, $Q(2)$
- Řídící body $Q(1)$: $P_{ij}^{(1)}$, $i = 0, \ldots s$, $j = 0, \ldots m$
- Řídící body $Q(2)$: $P_{ij}^{(2)}$, $i = 0, \ldots t$, $j = 0, \ldots m$
- Počet bodů je ve směru v shodný (navazujeme ve směru u)
- Požadujeme, aby jejich stupeň byl v tomto směru alespoň 3 ($s, t \leq 3$)

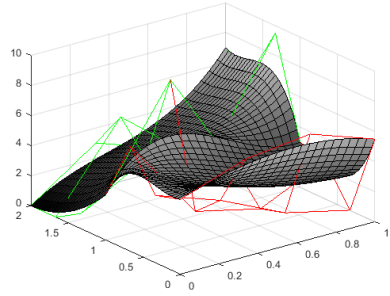
Beziérovy plochy

Navazování

- C^0 spojitě navázány ve směru u = mají stejné řídící body podél společné strany $P^{(1)}_{sj} = P^{(2)}_{0j}$, $\forall j$
- Vzniká ostrá hrana



Řídící polynom

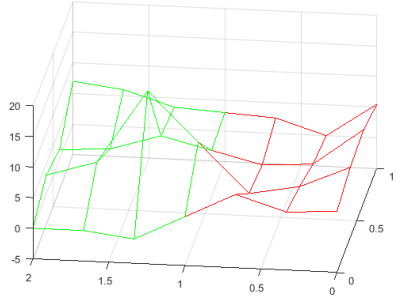


Navázání

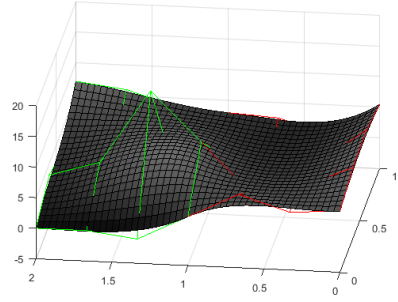
Beziérovy plochy

Navazování

- C^1 spojitosti dosáhneme:
  - Body na společné straně jsou shodné
    $P_{sj}^{(1)} = P_{0j}^{(2)}, \forall j$
  - Jsou shodné tečné vektory ve směru u podél této strany
    $s \cdot (P_{sj}^{(1)} - P_{s-1j}^{(1)}) = t \cdot (P_{1j}^{(2)} - P_{0j}^{(2)})$



Řídící polynom



Navázání

Beziérovy plochy

Navazování

Příklad

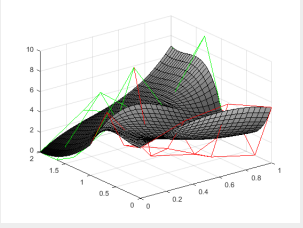
V následujících tabulkách máme řídící body dvou Beziérových plátů. Upravte řídící body druhého plátu tak, aby na sebe navazovali se spojitostí C^1.

j \ i	0	1	2	3
0	5.0	4.5	7.0	3.0
1	6.5	3.0	1.5	9.5
2	7.0	1.0	0.5	2.5
3	5.5	0.0	2.5	3.0

$P^{(1)}$

j \ i	0	1	2	3
0	3.0	1.0	0.0	0.0
1	9.5	3.5	5.5	3.0
2	2.5	2.5	2.0	1.5
3	3.0	9.5	6.0	7.0

$P^{(2)}$



Beziérovy plochy

Navazování

- G^1 spojitosti dosáhneme:
  - Body na společné straně jsou shodné
    $P_{sj}^{(1)} = P_{0j}^{(2)}$, $\forall j$
  - Tečné vektory ve směru u podél této strany nemusí být shodné, ale stačí lineárně závislé
    $s \cdot (P_{sj}^{(1)} - P_{s-1j}^{(1)}) = t \cdot (P_{1j}^{(2)} - P_{0j}^{(2)})$
- Hladké navazování je svazující podmínka – změna polohy bodu jednoho řídícího polygonu ovlivňuje polohu bodů v řídícím polygonu sousedního plátu
- V případě roku, dokonce 3 plátů (tedy celkem 9 bodů)

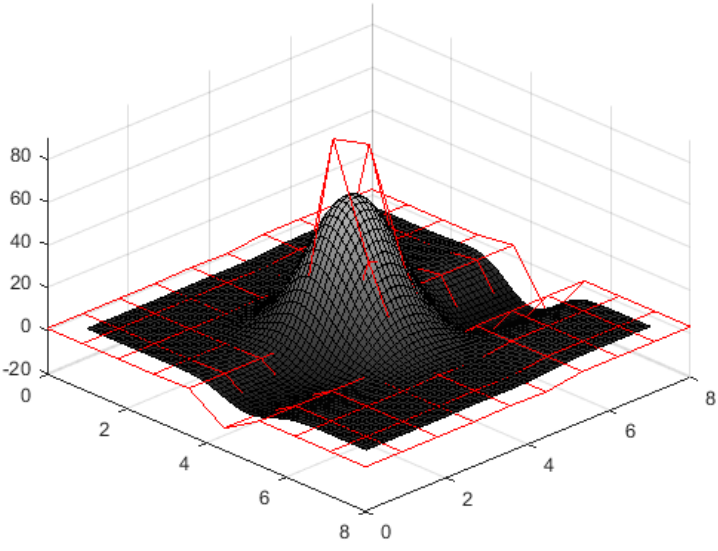
B-spline plochy

- Snadno se navazují – B-spline plochy n-tého stupně zaručují C^{n-1} spojitost
- Není potřeba omezovat podmínkami řídící body sousedních plátů
- Změnou 1 řídícího bodu měníme tvar pouze části B-spline plochy
- Navazování – každý plát se definuje $m \times (n-1)$ řídícími body předchozího plátu a přidáním m bodů
- **Vlastnosti**:
  - Celá plocha leží v konvexní obálce řídících bodů
  - Při změně polohy bodu se změní jen pláty, které jsou tímto bodem definovány
  - Obecně plocha neprochází krajními body řídící sítě
  - Aby plocha procházela krajními body, použijeme tyto body několikanásobně
  - Jsou invariantní k lineárním transformacím (posun, otočení, změna měřítka, ...)
- B-spline křivka k-tého stupně popsána $n+1$ body se skládá z $n-k+1$ Beziérových křivek

Bikubický B-spline

- Racionální, uniformní (Δu a Δv vždy stejné)
- Jsou C^2 spojité
- Nejsou invariantní k perspektivnímu promítání
- **Rovnice**:
  $Q(u,v) = \sum_{i=0}^{3} \sum_{j=0}^{3} P_{i,j} \cdot B_i(u) \cdot B_j(v)$
- B_i bázové polynomy:
  $B_0(t) = (-t^3 + 3t^2 - 3t + 1)/6$
  $B_1(t) = (3t^3 - 6t^2 + 4)/6$
  $B_2(t) = (-3t^3 + 3t^2 + 3t + 1)/6$
  $B_3(t) = t^3/6$
- U kubických ploch je každý krajní bod zadán $4\times$ – tím zajistíme, že prochází těmito body
- Body pak rozdělíme na pláty a postupně je napojujeme

Bikubický B-spline



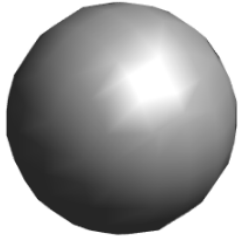
NURBS

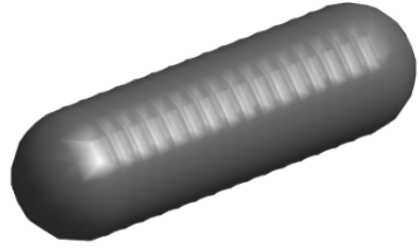
- Non-uniform rational B-spline
- Zobecnění – krok v jednotlivých směrech je neuniformní
- **Rovnice**:
  $$Q(u,v) = \frac{\sum_{i=0}^{n}\sum_{j=0}^{m} w_{ij}\cdot P_{i,j}\cdot N_i p(u)\cdot N_j q(v)}{\sum_{i=0}^{n}\sum_{j=0}^{m} w_{ij}\cdot N_i p(u)\cdot N_j q(v)}$$
- w_{ij} váhy bodů P_{ij} (pokud je roven 0 bod nemá žádnou váhu, ∞ plocha bodem prochází)
  m, n udávají počet řídících bodů
  p, q stupně polynomů
  $N_{ip}(u)$, $N_{iq}(v)$ normalizované B-spline bázové funkce určené vztahem
  $$N_{i1}(t) = \begin{cases} 1 & \text{pro } t_i \leq t \leq t_{i+1} \\ 0 & \text{jinak} \end{cases}$$
  $$N_{ik1}(t) = \frac{t-t_i}{t_{i+k}-t_i}\cdot N_{ik-1}(t) + \frac{t_{i+k}-t}{t_{i+k}-t_{i+1}}\cdot N_{i+1k-1}(t) \text{ pro } 0 \leq i \leq n$$
- Pro členy, kde se jmenovatel rovná 0 položíme celý zlomek roven 0
- Jsou invariantní k lineárním transformacím i k perspektivnímu promítání

Implicitní plochy

- Množina bodů P v prostoru, pro které platí $Q(P) = konstanta$
- Např. $Q(P) = x^2 + y^2 + z^2 = r^2$ – koule
- Základem je tzv. **kostra** – skládá se z generátorů prvků kostry
- Pro všechny body P v dosahu generátoru je funkce $d = d(P)$, která určuje vzdálenost bodu P od generátoru
- Pro každý generátor je definována **potenciálová funkce** $F(d)$, která určuje vliv generátoru na místa, která jsou ve shodné vzdálenosti
- Myšlenka: používáme jednoduché prvky (body, úsečky, polygony a jiné), vyhodnotíme jejich potenciálové funkce a plocha vznikne kombinací příspěvků od generátorů
- Příklad potenciálové funkce: $F_a(d) = (1 - \frac{d^2}{R^2})^2$ (d vzdálenost bodu od generátoru)
- Potenciálová funkce závisí pouze na vzdálenosti od generátoru – kostra určuje tvar
- Koule – 1 generátor (bod)

Implicitní plochy





Převod ploch na síť trojúhelníků

Naivní metoda

- Zvolíme pevný krok Δu a Δv a postupně dosazujeme do polynomů
- Získáme čtyřúhelníky a ty můžeme dále rozdělit na 2 trojúhelníky
- Normálové vektory získáme snadno – pomocí vektorového součinu
- Nerespektujeme zakřivení plochy $\rightarrow$ v místech, kde se křivka hodně mění je lepší hustší síť

Převod ploch na síť trojúhelníků

Patch splitting

- Rekurzivní dělení plátu – **Patch splitting**
- Založené na algoritmu de Castlejau (znáte z POGR)
- **Vstup**: matice řídících bodů P

Algoritmus

1. Je-li odchylka od roviny dostatečně malá, rozděl čtyřúhelník na 2 trojúhelníky a skonči.
2. Rozděl matici P ve směru u (2 matice P_L a P_R)
3. Rozděl matici P_L ve směru v (2 matice P_{LT} a P_{LB})
4. Rozděl matici P_R ve směru v (2 matice P_{RT} a P_{RB})
5. Spusť algoritmus s pláty P_{LT}, P_{LB}, P_{RT} a P_{RB}

Převod ploch na síť trojúhelníků

Patch splitting

- Pro dělení na dvě části použijeme algoritmus de Castlejau pro dělení křivky v bodě $t = 1/2$ – tzn. rozdělíme křivky určené řídícími body v jednotlivých řádcích matice P na 2 části
- Pro každý řádek získámedvě skupiny nových řídících bodů – rozdělili jsme plochu na dvě plochy reprezentované maticemi P_L a P_R
- Stejný postup aplikujeme na sloupce matic P_L a P_R
- Výsledkem jsou 4 matice (každá obsahuje tolik bodů, kolik měla původní matice

Příklad

Kolik máme matic po n děleních?

Převod ploch na síť trojúhelníků

Patch splitting

- Po n krocích dělení dostaneme 4^n ploch, které reprezentují stejnou plochu složenou z více částí
- Rohy každé matice (reprezentující řídící síť) leží na ploše
- Pokud je (křivá) plocha dostatečně rovná, můžeme jí nahradit rovnou plochou (2 trojúhelníky jejichž vrcholy leží v rozích)
- Algoritmus můžeme skončit i po pevném počtu kroku dělení – je znám počet trojúhelníků
- I v tomto případě ale aproximujeme stejným počtem trojúhelníků rovné a více zaoblené části

Příklad

Kolik vznikne trojúhelníků po po n děleních?

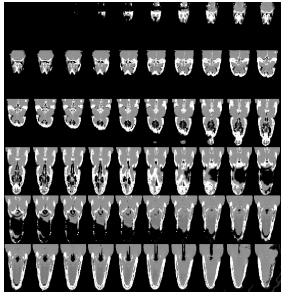
Převod prostorových dat na síť trojúhelníků

- Zobrazení prostorových dat:
  - Nalezneme povrch – nejčastěji síť trojúhelníků (ale ne vždy)
  - Přímé zobrazení – později

Převod prostorových dat na síť trojúhelníků

Prostorová data

- V pravidelné mřížce, skalární hodnoty
- Metody pro hledání povrchu – musíme znát hraniční hodnotu



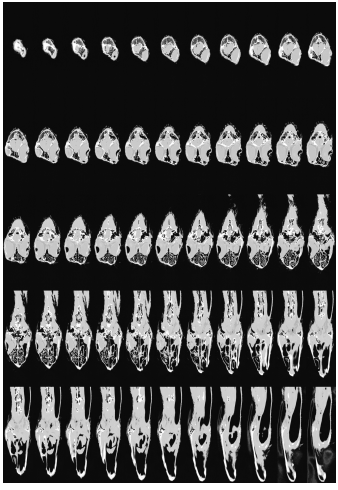
Převod prostorových dat na síť trojúhelníků

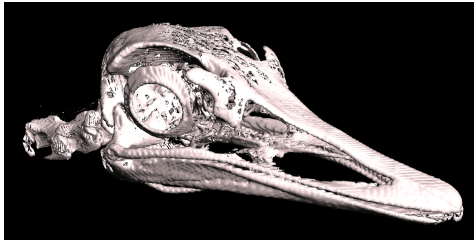
Metody pro převod prostorových dat do hraniční reprezentace

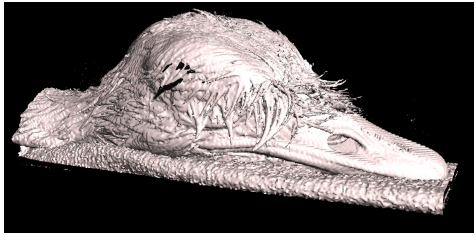
- Pochodující kostky a její varianty
- Pochodující čtyřstěny
- Dělené kostky – nevznikne síť trojúhelníků
- Opláštění kontur

Pochodující kostky

- **Marching Cubes**
- Vstup: data v prostorové mřížce, prahová hodnota
- Výstup: síť trojúhelníků





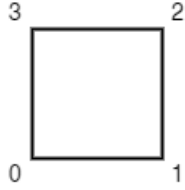


Pochodující čtverce

- **Marching squares** – 2D varianta Pochodujících kostek
- Prahová *prah* hodnota určuje hodnotu obrysu objektu
- Mřížku (ve které jsou data) rozdělíme na čtverce – vrcholy odpovídají sousedním vzorkům v mřížce
- Vrcholy si označíme
- Ohodnotíme vrcholy

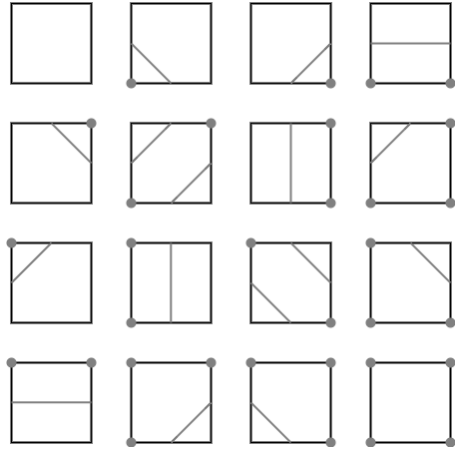
$$h_i = \begin{cases} 1 & \text{pro vnitřní bod tj. } \mathit{hodnota} \geq \mathit{prah} \\ 0 & \text{jinak} \end{cases}$$

- Vypočítáme index čtverce $\mathit{index} = \sum_{i=0}^{3} h_i \cdot 2^i$



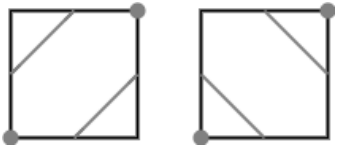
Pochodující čtverce

- Index čtverce $index = \sum_{i=0}^{3} h_i \cdot 2^i$
- 16 možností
- V pomocné tabulce uchováváme informaci, kudy prochází hranice objektu (pokud je jeden vrchol hrany uvnitř a druhý vně, hranice bude touto hranou procházet)
- Přesná poloha bodu na hraně – střed hrany, lineární interpolace
- Jelikož poloha bodu záleží jen na hodnotách ve vrcholech – křivka bude vždy spojitá



Pochodující čtverce

- Nejednoznačnost metody

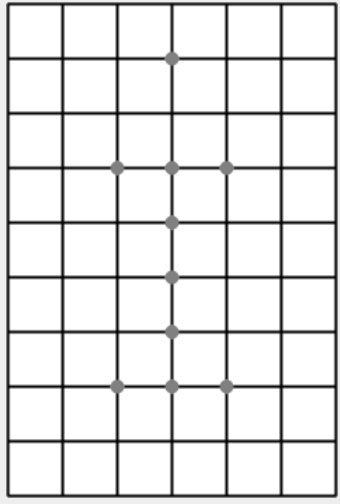


- 1. varianta = větší souvislé celky
- 2. varianta = menší objekty
- Volba záleží na aplikaci, je nutné používat jen jednu z variant

Pochodující čtverce

Příklad

Simulujte algoritmus pochodujících čtverců na následujícím obrázku. Šedé vrcholy jsou vnitřní. Obrys bude procházet ve středu mřížky.



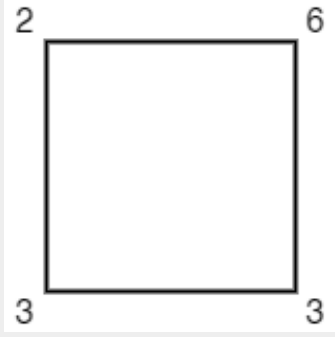
Pochodující čtverce – Algoritmus

- **Vstup**: Dvourozměrná mřížka s indexy i a j (odpovídající krokům na ose x a y) s rozestupem h, $i = 0, \ldots m-1$, $j = 0, \ldots n-1$
- $f_{ij} = f(x_0 + ih, y_0 + jh)$, hodnota v jednotlivých bodech mřížky
- f_0 hodnota, pro kterou hledáme křivku (okraj)
- Každému čtverci mřížky spočítáme jeho index a z tabulky hran zjistíme, na kterých hranách leží koncové body úseček
- Každé úsečce interpolujeme polohu krajních bodů na základě hodnot v mřížce
- Vykreslíme hrany

Pochodující čtverce

Příklad

Lineární interpolací určete, kudy prochází křivka $f_0 = 4$.

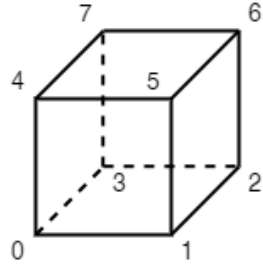


Pochodující krychle

- **Marching cubes** – 3D mřížka
- Výstupem je série trojúhelníků, jejichž vrcholy leží na hranách krychlí
- Prahová *prah* hodnota určuje hodnotu hranice objektu
- Mřížku (ve které jsou data) rozdělíme na krychle – vrcholy odpovídají sousedním vzorkům v mřížce
- Vrcholy si označíme
- Ohodnotíme vrcholy

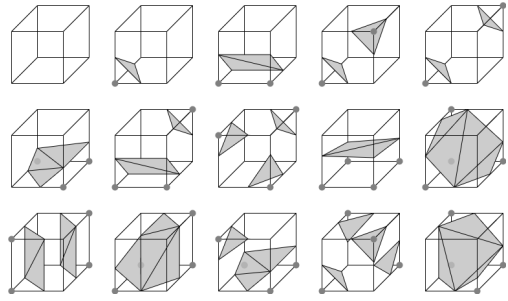
$$h_i = \begin{cases} 1 & \text{pro vnitřní bod tj. } \mathit{hodnota} \geq \mathit{prah} \\ 0 & \text{jinak} \end{cases}$$

- Vypočítáme index krychle $\mathit{index} = \sum_{i=0}^{7} h_i \cdot 2^i$



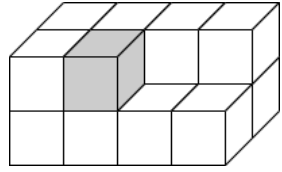
Pochodující krychle

- Index čtverce $index = \sum_{i=0}^{7} h_i \cdot 2^i$
- 256 možností
- V pomocné tabulce uchováváme informaci, kde leží vrcholy trojúhelníků
- Přesná poloha bodu na hraně – střed hrany, lineární interpolace
- Velké množství trojúhelníků – pro každou krychli až 4



Pochodující krychle

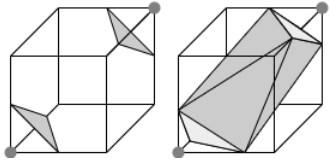
- 4 krychle mají společnou hranu, pokud na ní leží vrcholy trojúhelníků, pak jejich polohu (interpolaci) počítáme pro každou hranu znovu
- Snížení počtu výpočtů – pamatujeme si vrcholy již zpracovaných krychlí (bílé) a počítáme jen vrcholy na nezpracovaných hranách
- **Filtrování malých částí** – hledáme komponenty (ve smyslu 6 sousednosti) a ty, které jsou menší než zvolený práh do výpočtu povrchu neuvažujeme

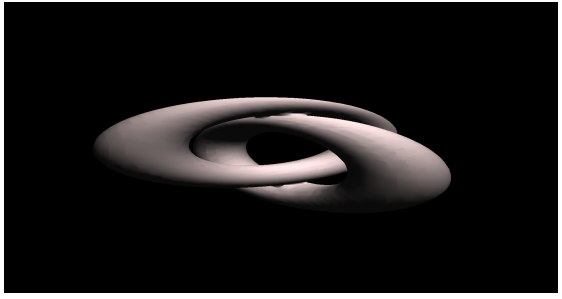


Pochodující krychle

Nejednoznačnosti

- Stejně jako u pochodujících čtverců – nejednoznačnost topologická
- Pokud zachováme jeden styl, nic se nestane

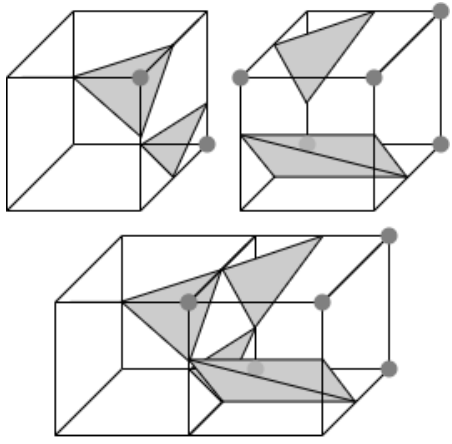




Pochodující krychle

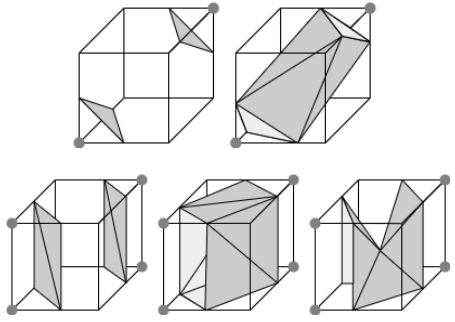
Nejednoznačnosti

- Počítáme každou krychli zvlášť, nebereme v potaz globální pohled na těleso
- Vznikají díry



Pochodující krychle 33

- **Marching cubes 33**
- Místo 15 případů máme rozšířenou tabulku (33 případů)
- Některé případy mají více možností
- Kterou zvolit závisí na okolních krychlích
- Pro jednu krychli může vzniknout až 12 trojúhelníků

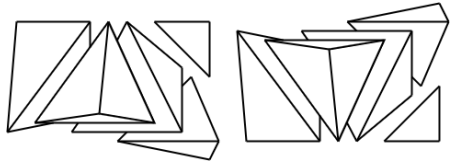


Pochodující krychle
Varianty

- Dual contouring
- Dual Marching cubes

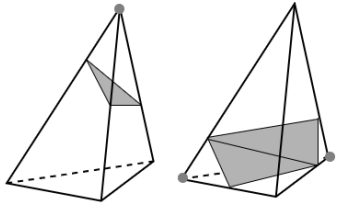
Pochodující čtyřstěny

- **Marching tetrahedra** – 3D mřížka
- Výstupem je série trojúhelníků, jejichž vrcholy leží na hranách čtyřstěnů
- Prahová *prah* hodnota určuje hodnotu hranice objektu
- Mřížku (ve které jsou data) rozdělíme na krychle a ty dále na 5 čtyřstěnů (jejich vrcholy odpovídají vrcholům krychle)
- 2 způsoby, jak je rozdělit



Pochodující čtyřstěny

- Dle toho, které vrcholy jsou uvnitř a které vně, mohou nastat jen 2 případy



- Problém děr se tímto vyřešil

Příklad

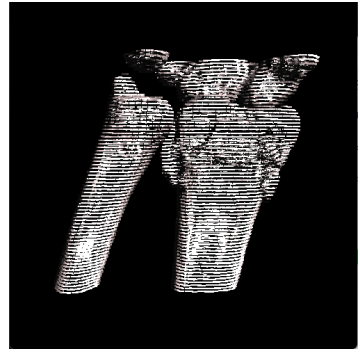
Kolik může v krychli vzniknout trojúhelníků?

Pochodující čtyřstěny

- Počet trojúhelníků narostl
- Je nutno střídat dělení – řešení = dělíme na 6 nebo 24 čtyřstěnů (ještě více trojúhelníků)
- V praxi se moc nepoužívá

Dělené kostky

- **Dividing cubes**
- Rasterizace velkého množství malých plošek → pomalé
- Výstupem algoritmu jsou pouze povrchové body s normálou (pro osvětlení)
- Velikost povrchového bodu = velikost obrazového bodu
- Zrychlení vykreslování
- Nevýhoda – nemáme informaci, ke které ploše bod patří, nemožnost přiblížení tělesa
- Pokud je bod menší než zobrazovací bod – díry
- Obdobně dobrá kvalita, jako pochodující kostky, nevykresluje nezobrazitelné detaily
- Velký počet bodů



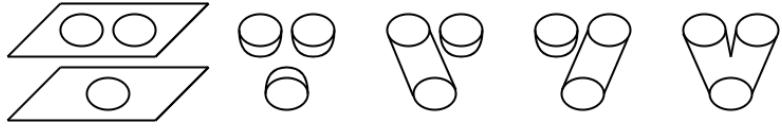
Dělené kostky

Algoritmus

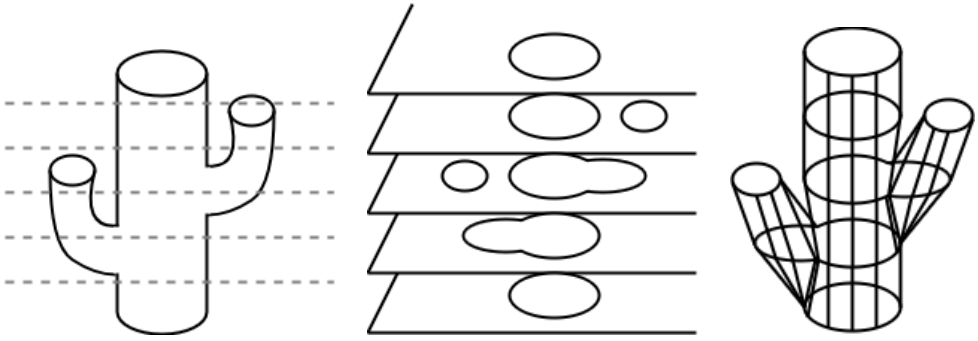
- **Vstup**: Data v 3D mřížce a prahová hodnota
- Do paměti načteme 4 sousedící řezy
- Vytvoříme krychli, kterou tvoří 8 bodů dvou sousedních řezů
- Ve všech vrcholech spočítáme vektor gradientu (jednotlivé složky spočítáme jako rzdíl mezi předchozím a následujícím sousedem ve směru každé osy)
- Ohodnotíme každou krychli:
  - vnitřní – intenzita všech vrcholů je větší rovna prahové hodnotě
  - vnější – intenzita všech vrcholů je menší než prahová hodnota
  - protíná povrch tělesa
- Hraniční krychle rozdělíme na $a \times b \times c$ subkrychlí, které jsou velké jako zobrazovací bod
- Hodnotu každého vrcholu nových subkrychlí vypočítáme lineární interpolací
- Najdeme subkrychle, které leží na hranici tělesa, interpolujeme pro ně gradient
- Vykreslíme

Opláštění kontur

- Těleso chápeme jako kolekci kontur (jeho hranic) v několika řezech
- Z objemových dat nejprve vypočítáme kontury tělesa (řezy jsou kolmé na jeden rozměr)
- Kontury pak propojujeme
- Největší problém – jak k sobě přiřadit kontury
- Propojení aproximujeme sítí trojúhelníků
- Jelikož nemáme informaci mezi řezy – pouze odhadujeme tvar tělesa

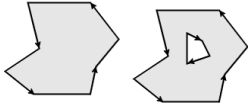


Opláštění kontur



Opláštění kontur

- **Data** = série snímků
- **Sada řezů** = množina řezů S, každý řez se skládá z množiny kontur
- **Kontury**:
  - odpovídají průniku hranice telesa a roviny řezu
  - je to orientovaný jednoduchý uzavřený polygon $c_i = p_1, \ldots, p_n$, i je index kontury v rámci řezu, p_k vrcholy polygonu
  - kontury se v rámci řezu neprotínají
  - 2 typy – vnější kontura a díra



  - vnější kontury jsou orientovány proti a díry po směru hodinových ručiček

Opláštění kontur

Výpočet řezu

- hledání průniku
- sestavení kontur
- nastavení orientace

Hledání průniku

- Například pomocí Marching squares
- Výsledkem je série úseček, které nenesou informaci o tom, ke které kontuře patří

Opláštění kontur

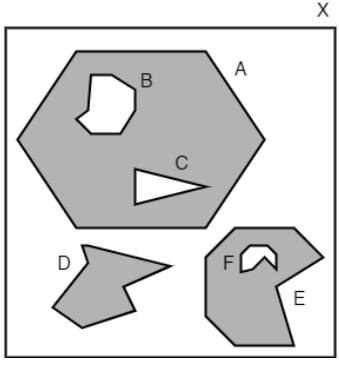
Sestavení kontur

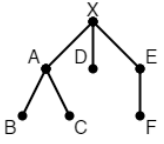
- Máme množinu úseček $s = s_1, \ldots s_n$, ze kterých chceme sestavit kontury
- Konturu sestavíme tak, že z množiny s přidáváme úsečky (a zároveň se z množiny s odstraňujeme)
- Úsečku přidáme k mnohoúhelníku, který právě vytváříme, pokud některý její krajní bod inciduje s jedním ze dvou stávajících krajních bodů polygonu
- Pokud už nemůžeme další úsečku přidat a s není prázdná, začneme sestavovat další konturu

Opláštění kontur

Nastavení orientace kontur

- Vyžaduje znalost hierarchie kontur v rámci řezu
- Po sestavení kontur nevíme, zda se jedná o vnější konturu nebo díru
- Vytváříme **strom vnoření**

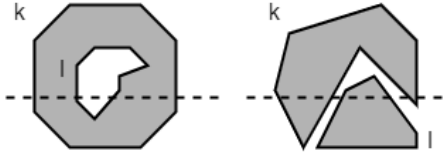




Opláštění kontur

Nastavení orientace kontur

- Je nutno nejprve určit, zda se jedná o vnější konturu nebo o díru
- Kontura je dírou, leží-li uvnitř jiné kontury
- Využijeme principu **ray crossingu**
- Najdeme nejlevější průnik přímky q s konturou l (P_l) a pak hledáme počet průsečíků s konturou k nalevo od P_l



- l je uvnitř k, pokud je počet průsečíků lichý

Opláštění kontur

- **Vstup**: množina kontur
- Ve své podstatě se jedná o hledání plochy spojující dvě křivky – tu aproximujeme trojúhelníky
- **Konstrukce pláště** (není to triviální úloha) – Objemové metody, Povrchové metody

Objemové metody

- Vzdálenost mezi řezy je shodná s hustotou vzorků (vzorky jsou chápány jako prostorová data)
- Pro výpočet se používají dříve zmíněné metody (např. Marching cubes)
- Čím jsou od sebe kontury (řezy) vzdálenější, tím více tyto metody selhávají

Povrchové metody

- Korespondující kontury spojují pásem trojúhelníků
- Klíčový problém – nalezení korespondujících kontur
- Více možností, záleží na datech (jiné přístupy se hodí pro stromové struktury, jiné pro menší kruhovité)

Opláštění kontur

Korespondence kontur – heuristiky

Plocha překrytí kontur

- Nejjednodušší a nejčastěji používaná
- Kontury spolu korespondují, pokud se při promítnutí do jedné roviny překrývají dostatečně velkou plochou
- Pokud nejsou řezy dostatečně husté – dochází k rozpadání na menší objekty

Příklad

Jak by mohla vypadat například definice „překrývání dostatečně velkou plochou“?

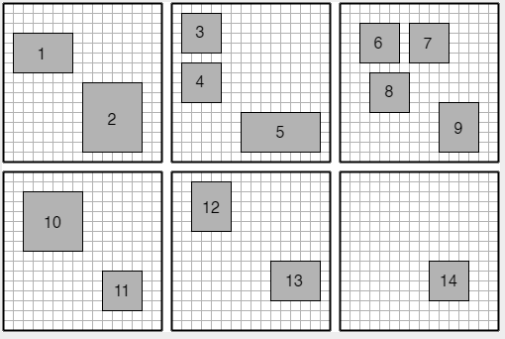
Opláštění kontur

Korespondence kontur – heuristiky

Plocha překrytí kontur

Příklad

Předpokládejme, že kontury spolu korespondují, pokud se překrývají alespoň 75% plochy menší z kontur. Určete, které kontury z následujícího obrázku spolu korespondují.



Na obrázku jsou kontury z 6 řezů.

Opláštění kontur

Korespondence kontur – heuristiky

Zobecněné válce

- Pokud má těleso stromovou strukturu kruhového, či oválného tvaru, předchozí přístup není vhodný
- Tato metoda používá globálnější pohled na těleso (jeho strukturu)
- Válce tvoří řezy ve tvaru elips jejichž středy leží přibližně na přímce (odchylují se o méně, než je zadaná chyba)

Opláštění kontur

Korespondence kontur – heuristiky

Strom minimálního pokrytí grafu kontur

- Předchozí metoda může vést k nechtěné propagaci chyby při větším vychýlení prvních dvou kontur
- Tato metoda pracuje ještě globálněji
- Nejprve sestavíme graf možných spojnic kontur tak, aby každé kontuře odpovídal jeden uzel každé možné spojnici (každá možná dvojice sousedních kontur) jedna hrana
- Tyto hrany jsou ohodnoceny podle vzdálenosti kontur
- Poté hledáme strom minimálního pokrytí (minimální kostru)
- Lepší výsledky, než předchozí heuristiky
- Může propojovat objekty, které jsou blízko sebe
- Nevhodné pro tělesa s cykly

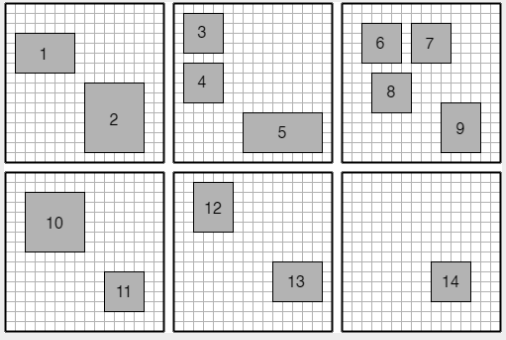
Opláštění kontur

Korespondence kontur – heuristiky

Strom minimálního pokrytí grafu kontur

Příklad

Dle metody minimálního pokrytí grafu spočítejte, které kontury spolu korespondují. Jako vzdálenost počítejme vzdálenost středů kontury.



Opláštění kontur

Propojení kontur

- Máme spočítanou vzájemnou korespondenci kontur
- Chceme kontury propojit – vytvořit síť trojúhelníků z jejich bodů
- Hledáme trojúhelníky, které opláští celou konturu a vzájemně se nepřekrývaly
- Problém nastává u větvení kontur a jejich propojování (symetrické problémy)
- Celá řada metod, některé umožňují jen propojení 1:1 (nepovolují větvení)

Opláštění kontur

Propojení kontur – heuristiky

Toroidní graf

- Využívá teorii grafů
- Uzly grafu odpovídají všem možným úsečkám propojující sousední kontury
- Trojúhelníky = dvojice úseček se společným vrcholem
- Hledáme cyklus s nejmenší cenou
- Grafové algoritmy dávají optimální výsledky, ale jsou výpočetně náročné
- Využíváme spíše heuristiky, které nezohledňují celý plášť, ale pracují lokálně

Opláštění kontur

Propojení kontur – heuristiky

Minimální povrch

- Nejjednodušší
- Počítáme obsah trojúhelníků
- Nehodí se v případě, že jsou kontury vůči sobě posunuty

Směr přiřazení

- Preferuje ty spojnice, jejichž směr se moc neliší od směrnice těžišť kontur

Maximální objem

- Objem klínu, který tvoří trojúhelníková záplata a spojnice těžišť kontur

Minimální délka spoje

- Délka úsečky, která spojuje sousední kontury

Opláštění kontur

Korespondence kontur – heuristiky

Příklad

Najděte propojení kontur 1 a 3 (bez větvení) z předchozího příkladu. Pro výběr trojúhelníku použijte kritéria:

- minimální povrch,
- směr přiřazení.