Modelování pevných těles

Pavel Strachota

FJFI ČVUT v Praze

12. března 2015

Obsah

Obsah

Proč nestačí povrchy

snaha vytvořit model pevného tělesa, protože:

- potřeba odlišit vnitřek a vnějšek objektu
- potřeba odlišit vnější a vnitřní *povrch*, orientaci povrchu
- modelování optických vlastností materiálu - v objemu
- fyzikální využití modelu
  - fyzikální vlastnosti - objem, těžiště
  - CAD/CAM - model budoucího výrobku (součástky stroje apod.)
  - automatizace výroby - obrábění atd. podle modelu

Vlastnosti ideální reprezentace pevného tělesa

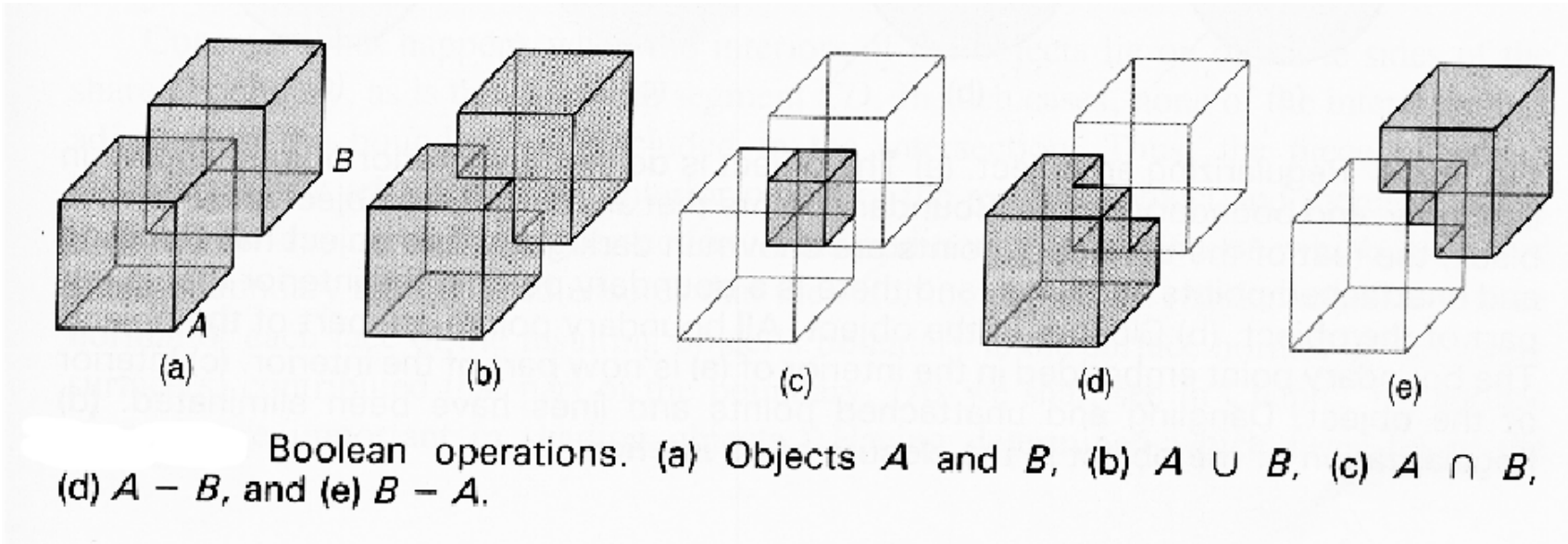
- schopnost reprezentovat širokou škálu těles
  - absence omezení typu: jen konvexní tělesa, jen tělesa bez děr apod.
- *jednoznačnost*
  1. daná reprezentace vyjadřuje jediný objekt (tzv. *úplnost*)
  2. jeden objekt lze reprezentovat jen jedním způsobem
- *přesnost* - modelování objektu bez aproximací
- „*platnost*“ , resp. korektnost - nemožnost reprezentovat objekt, který není 3D těleso (např. jeho část má nulový objem - křivka, plocha,...)
- uzavřenost vůči transformacím (rotace, škálování)
- efektivita z hlediska
  1. vytvoření reprezentace
  2. kombinace objektů (viz CSG)
  3. paměťových nároků
  4. složitosti vykreslování

Typy reprezentace

1. *objemová* (výčtová) - objekt vyjádřen pomocí specifikace objemových buněk prostoru
   - prostor rozdělen uniformně či neuniformně
2. *hraniční* (povrchová) - popis povrchu objektu pomocí sjednocení „plátů“ - obvykle variet
3. *tahová* (sweep representation, extrusion rep.) - vznik tělesa pohybem objektu (2D i 3D) po dané trajektorii

Regularizované booleovské operace 1/2

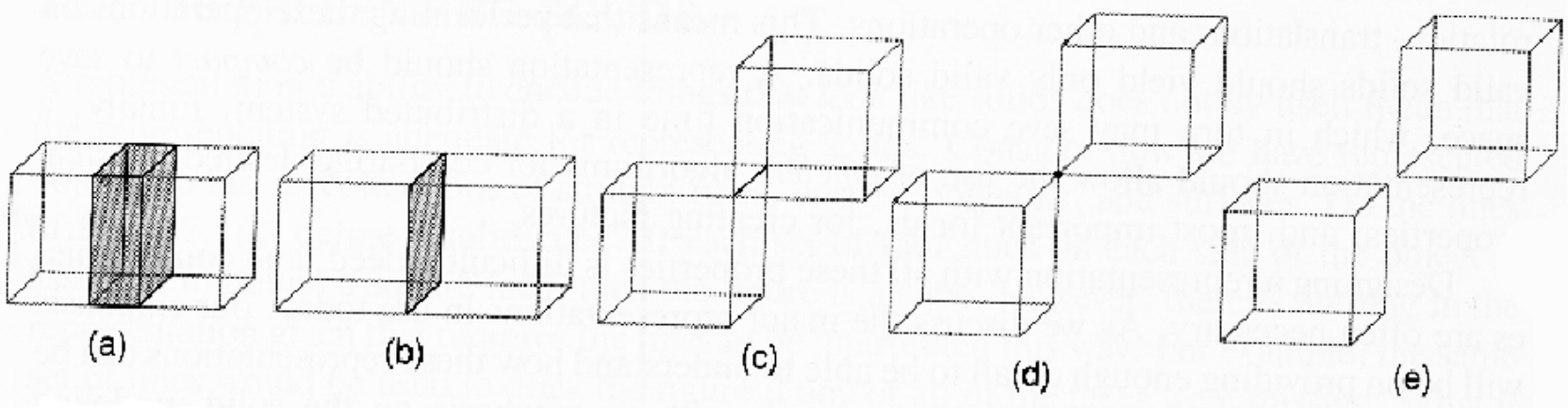
- základ tzv. konstruktivní geometrie pevných těles (CSG - Constructive Solid Geometry)
- operace s tělesy jako s množinami ve 3D: sjednocení, průnik, rozdíl
- výsledkem těchto operací musí být opět 3D těleso nebo prázdná množina



Boolean operations. (a) Objects A and B, (b) $A \cup B$, (c) $A \cap B$, (d) $A - B$, and (e) $B - A$.

Regularizované booleovské operace 1/2

- základ tzv. konstruktivní geometrie pevných těles (CSG - Constructive Solid Geometry)
- operace s tělesy jako s množinami ve 3D: sjednocení, průnik, rozdíl
- výsledkem těchto operací musí být opět 3D těleso nebo prázdná množina



The ordinary Boolean intersection of two cubes may produce (a) a solid, (b) a plane, (c) a line, (d) a point, or (e) the null set.

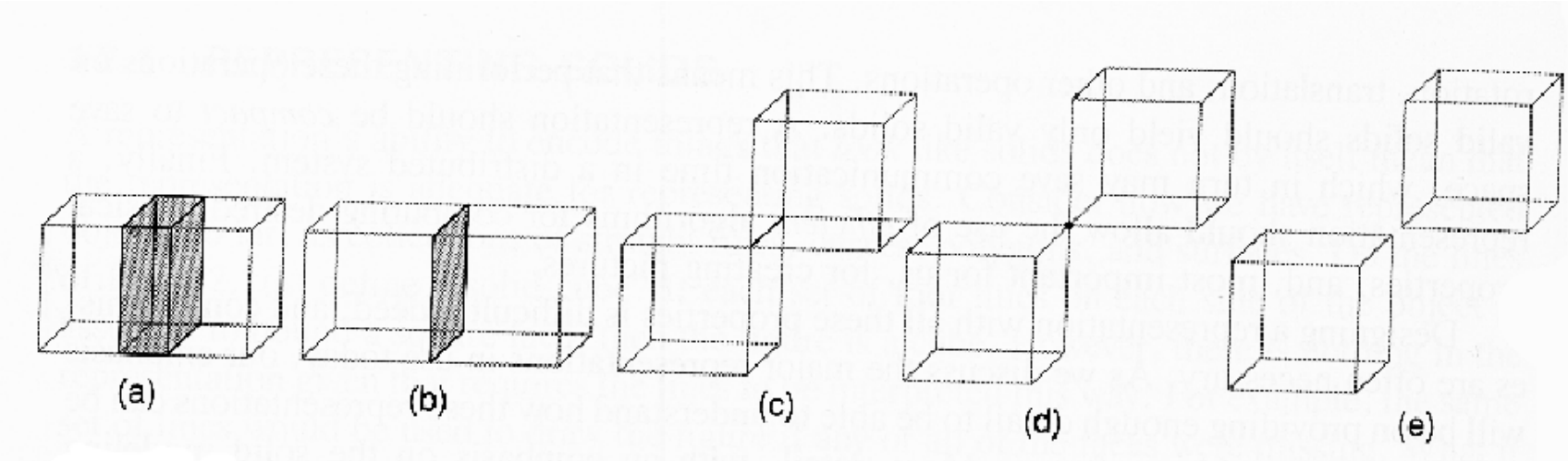
Regularizované booleovské operace 2/2

- *regulární* těleso - je rovné uzávěru svého vnitřku: $A = \overline{A^0}$
- regularizované operace:

$$A \bullet^\star B := \overline{(A \bullet B)^0},$$

kde $\bullet \in \{\cup, \cap, -\}$

- $A \cap^\star B = \emptyset$ v případech (b), (c), (d), (e)



Obsah

Objemová reprezentace

- prostor rozdělen do buněk (voxelů)
- reprezentace tělesa pomocí obsazenosti voxelů
- omezená přesnost $\Longrightarrow$ použití spíše jako pomocných datových struktur

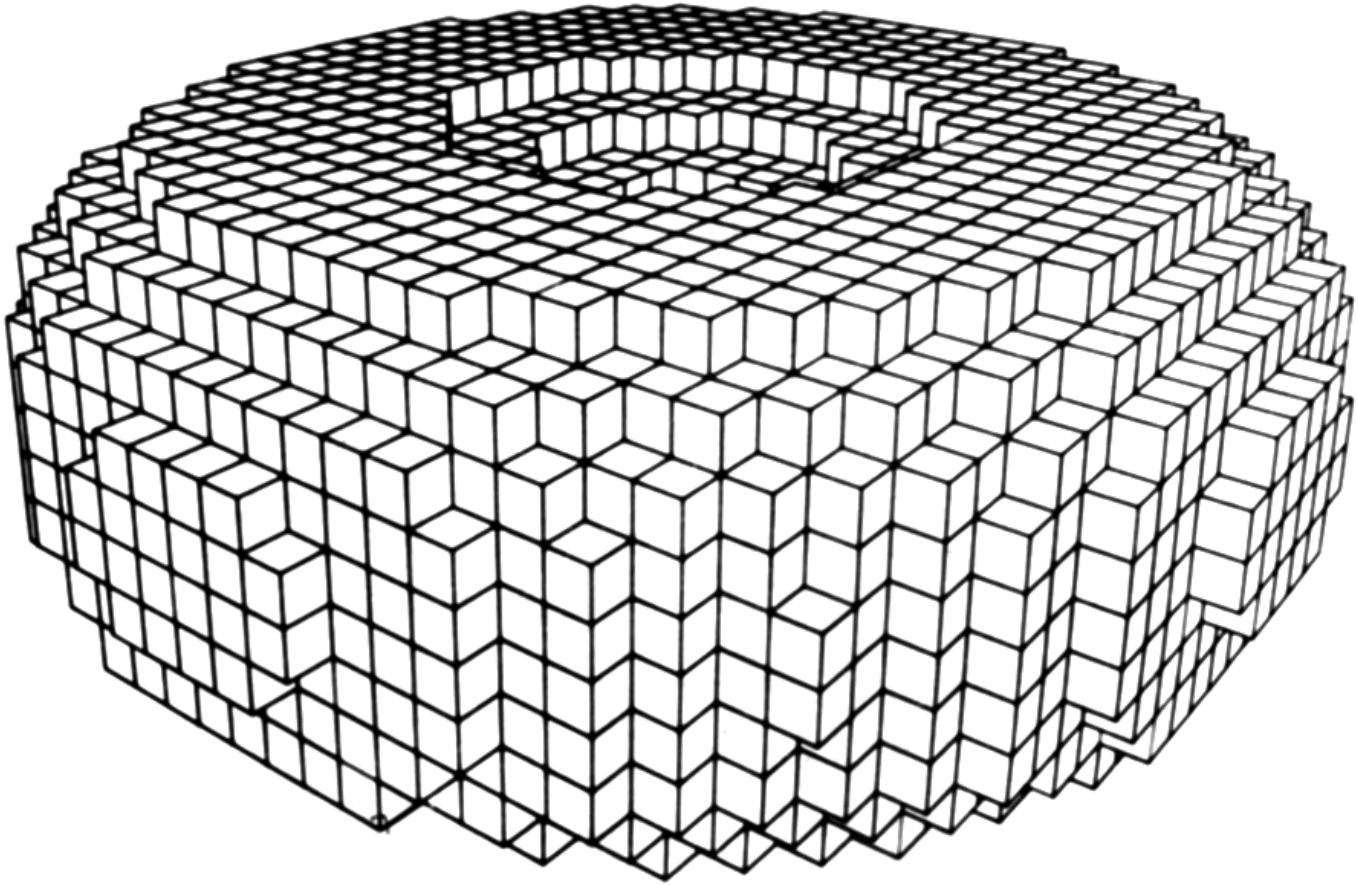
Typy rozdělení prostoru:

- buněčný model
- oktalový strom (*octree*)
- BSP strom

Buněčný model

- identické buňky v pevné pravidelné mřížce
- obvykle buňka (voxel) = krychle
- každá buňka má stav: obsazená / neobsazená
- přesné vyjádření jen objektů se stranami rovnoběžnými s mřížkou - aliasing
- reprezentace objemových dat v medicíně (CT, MRI)
- vykreslování: překreslování přivrácených stran buněk odzadu dopředu

Buněčný model - příklad



buněčný model toru

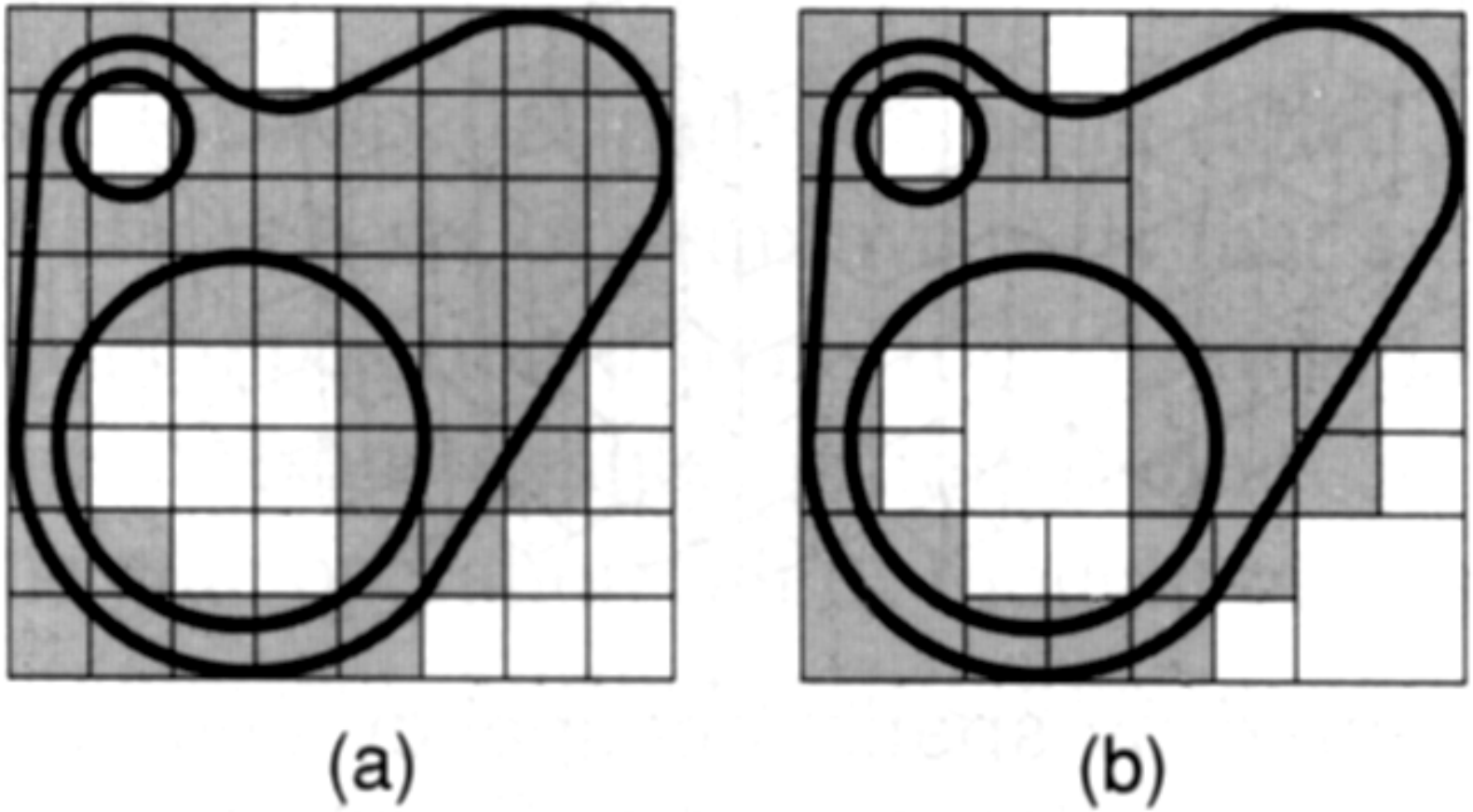
Oktalový strom (*octree*)

- hierarchická varianta buněčného modelu - adaptivní zjemnění podle tvaru tělesa

Princip:

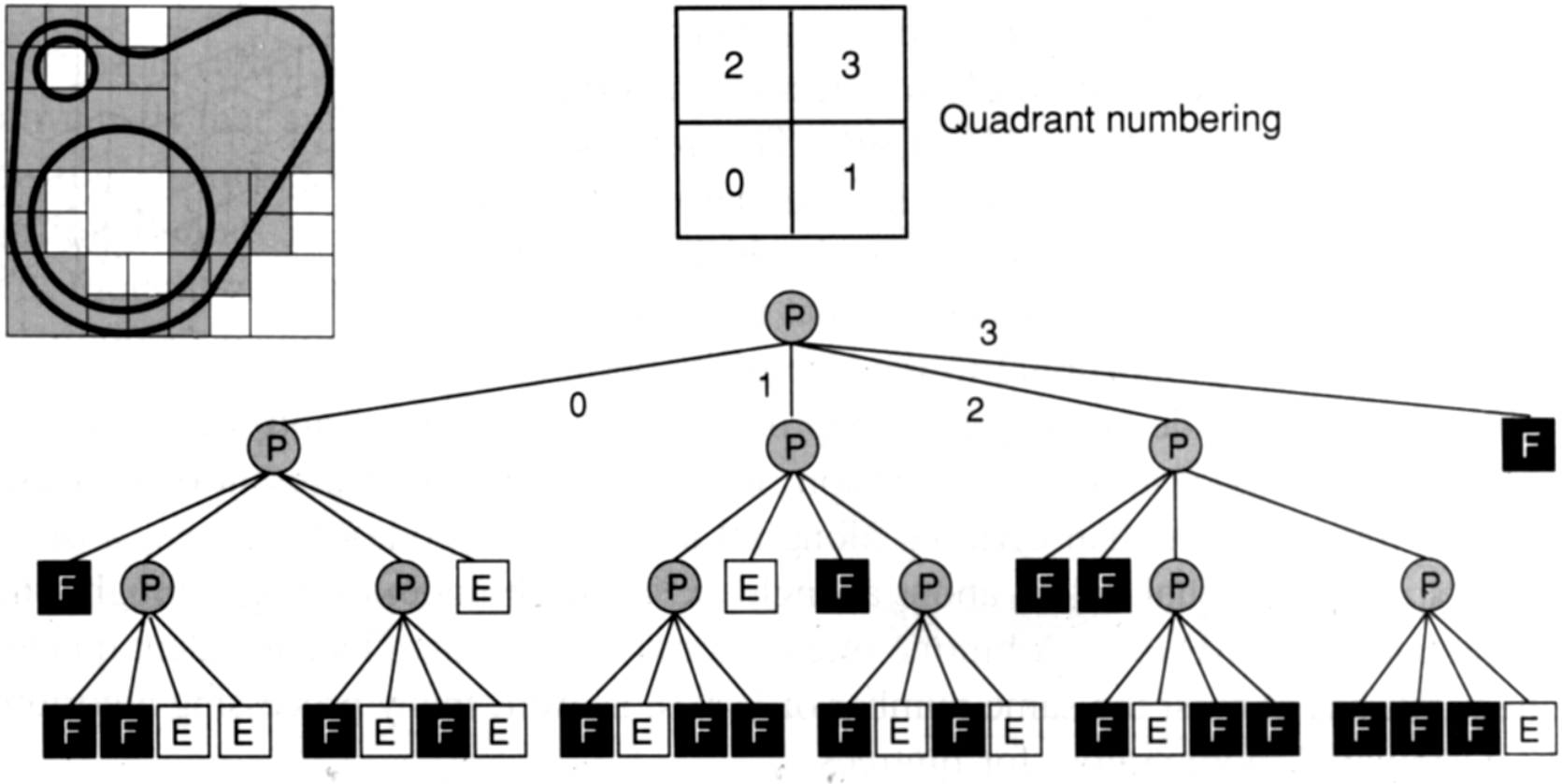
- konečný prostor (krychle) rozdělen v každém ze 3 rozměrů na polovinu $\Longrightarrow$ osm oktantů
- každý z oktantů může být
  - prázdný
  - plně obsazen objektem
  - částečně obsazen objektem
- částečně obsazený oktant se rekurzivně dělí stejným způsobem
- stejný princip v rovině: kvadrantový strom (*quadtree*)

Kvadrantový strom (*quadtree*) 1/2



(a) pravidelná mřížka, (b) quadtree reprezentace

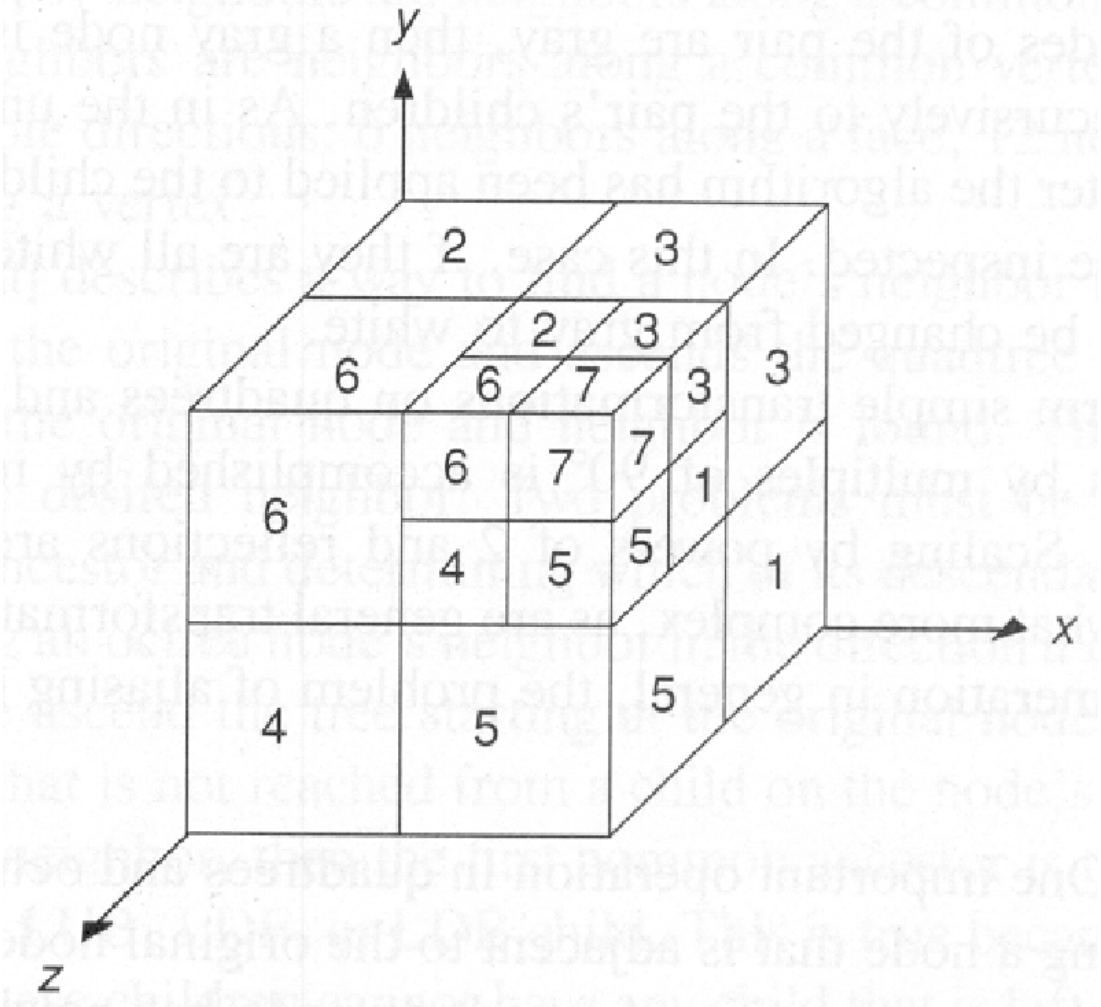
Kvadrantový strom (*quadtree*) 2/2



datová struktura quadtree pro objekt na předchozí stránce.

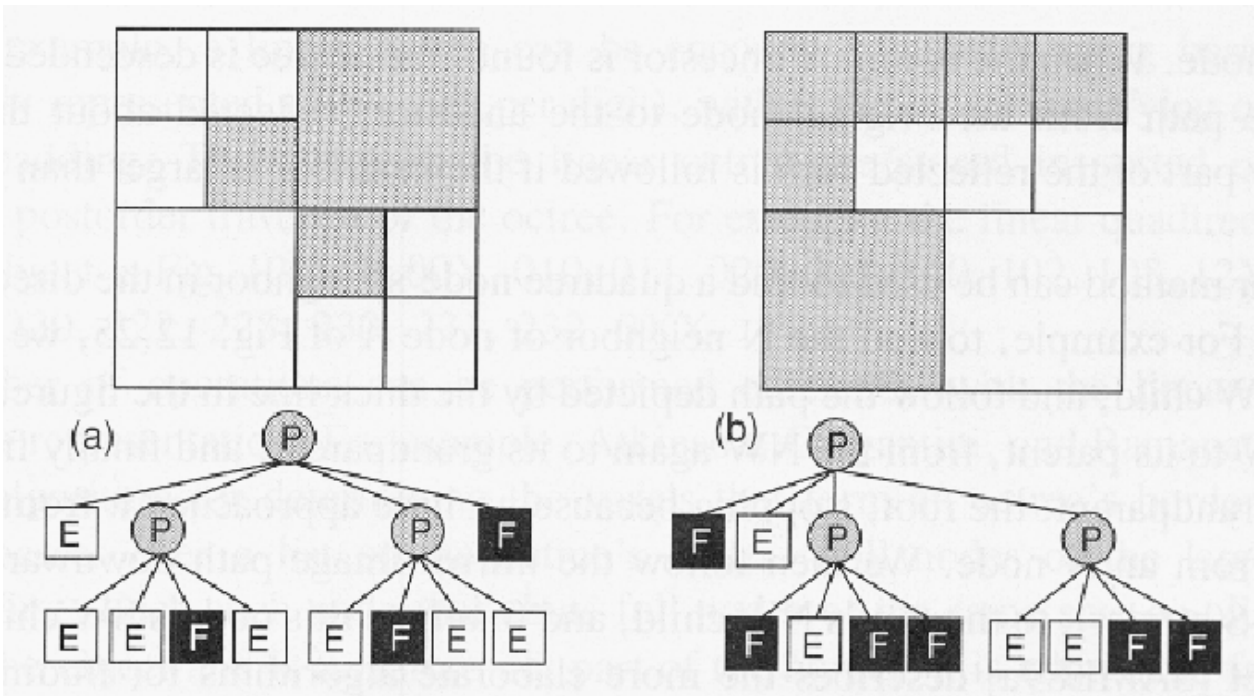
E = prázdný (empty), F = plný (full), P = částečně obsazený (partially occupied)

Enumerace oktantů v octree



Booleovské operace na quadtree a octree 1/4

- reprezentace uzavřená vůči booleovským operacím
- vhodná reprezentace pro snadné provedení sjednocení, průniku, rozdílu
- uvažujme objekty A,B a chceme $A \cup B$



Booleovské operace na quadtree a octree 2/4

- E = prázdný, F = plný, P = částečně obsazený
- zaveďme operaci sjednocení:
  $F \cup F = F$, $F \cup E = F$, $F \cup P = F$,
  $P \cup P = P$, $P \cup E = P$,
  $E \cup E = E$

Struktura quadtree

```
struct Qtree;
struct Qtree {
  enum {E,P,F} stat;
  Qtree * child[4];
};
```

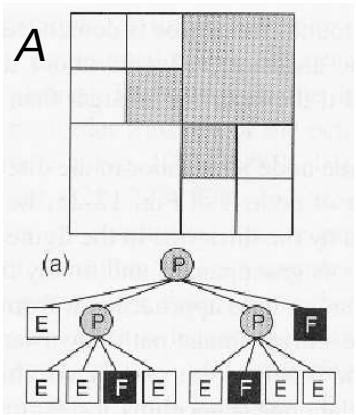
Booleovské operace na quadtree a octree 3/4

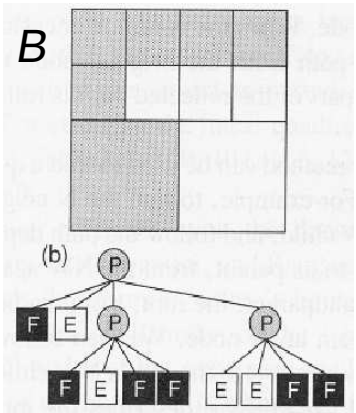
Algoritmus pro $A \cup B$

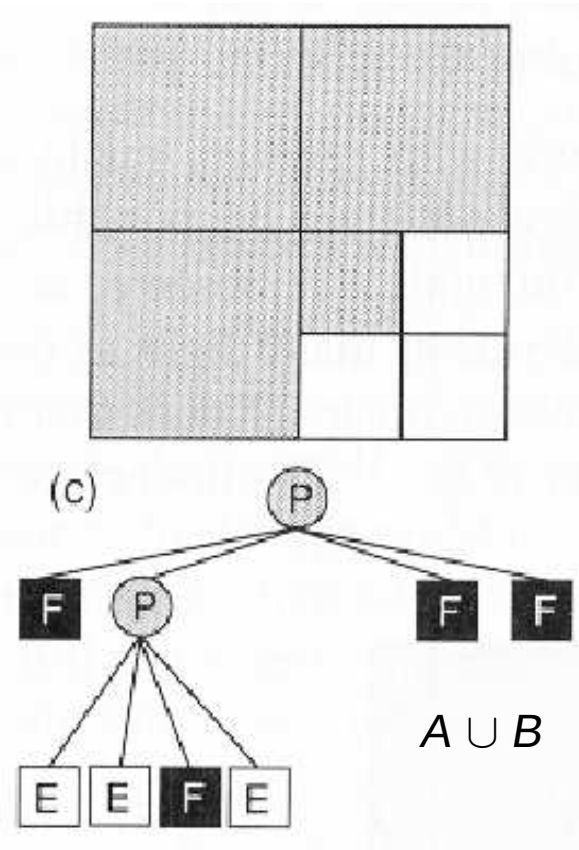
```
Qtree * Q_union(Qtree * A, Qtree * B) {
  bool full = true; Qtree * result = new Qtree;
  result->stat = A->stat ∪ B->stat;
  if(result->stat == P) {
    if(A->stat == P && B->stat == P) {
      for(int i=0;i<4;i++) { result->child[i] =
                  Q_union(A->child[i], B->child[i]);
        if(result->child[i].stat != F) full = false; }
      if(full) {
        for(int i=0;i<4;i++) delete result->child[i];
        result->stat = F;
       }
    } else if(A->stat == E) Q_copyTree(B, result);
           else Q_copyTree(A, result);
  }
  return result;
}
```

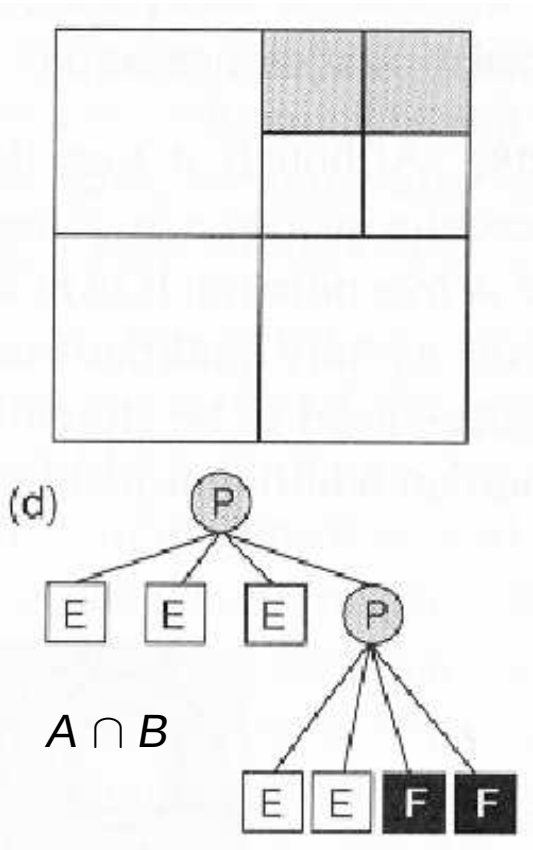
Booleovské operace na quadtree a octree 4/4

- průnik analogicky jako sjednocení - pouze prohození významu $E \leftrightarrow F$ a $\cup \leftrightarrow \cap$
- na octree vše úplně stejně (jen místo 4 potomků je jich 8)





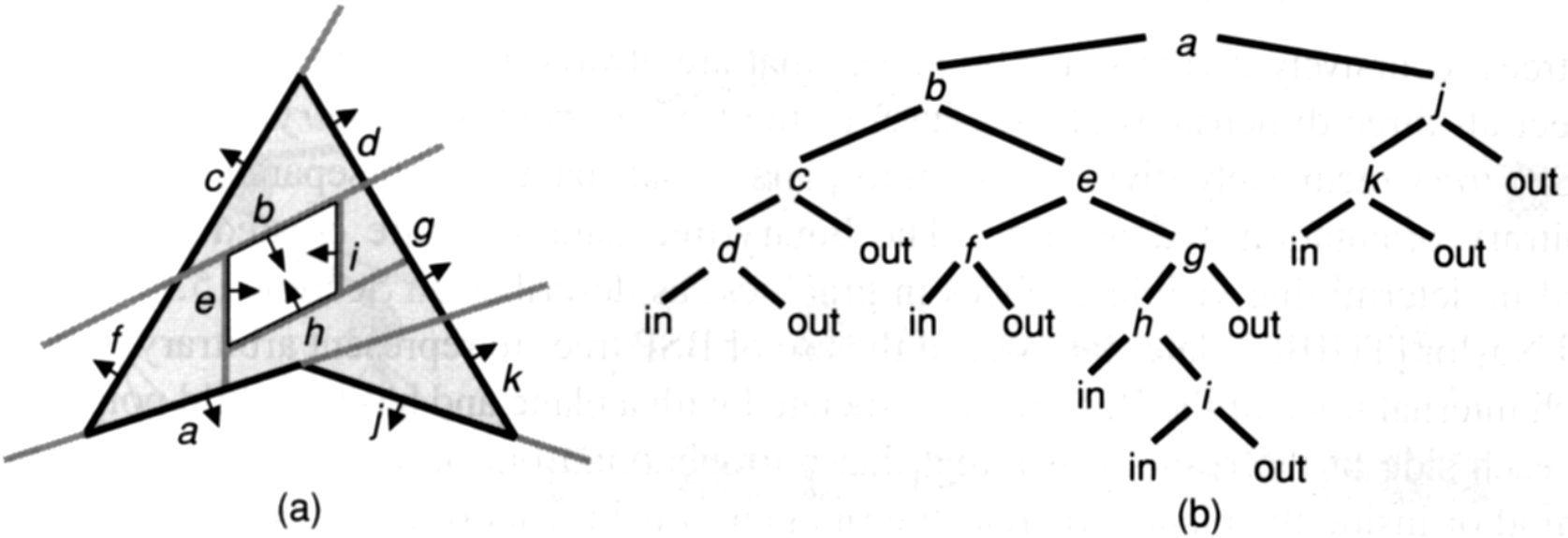




BSP stromy

- **BSP** = Binary Space Partitioning - binární rozdělení prostoru
- rekurzivní dělení prostoru na poloprostory oddělené libovolně umístěnou a orientovanou rovinou
- původní použití binárních stromů - viditelnost povrchů
  - později: reprezentace libovolných mnohostěnů
- uzel BSP stromu - svázán s rovinou, jejíž normála směřuje ven z objektu (úmluva)
- 2 potomci:
  - **levý** potomek = rozdělení poloprostoru **proti směru** normály k rovině, nebo NULL (lze implementovat jako list „*in*“ - identifikuje poloprostor, který leží uvnitř tělesa)
  - **pravý** potomek = rozdělení poloprostoru **ve směru** normály k rovině, nebo NULL (lze implementovat jako list „*out*“ - identifikuje poloprostor, který leží vně tělesa)

BSP Stromy - příklad



Konstruktivní geometrie CSG

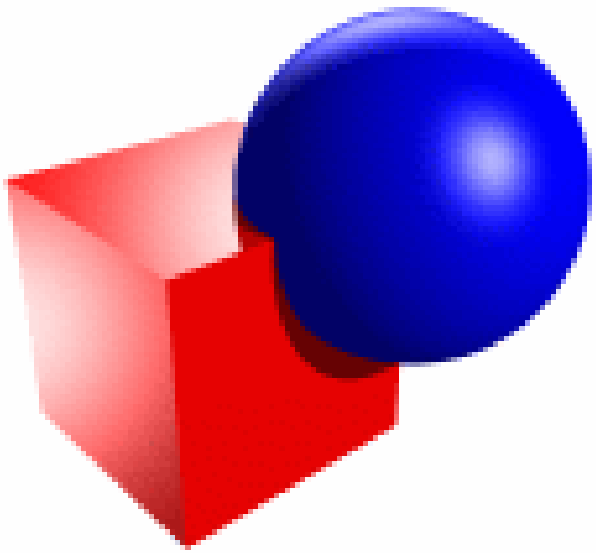
- CSG = Constructive Solid Geometry - konstruktivní geometrie pevných těles
- kombinace jednoduchých objektů (tzv. *primitiv* - koule, válec, krychle, ...) pomocí regularizovaných booleovských operací

Vyjádření objektu pomocí binárního stromu

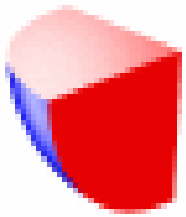
- uzel stromu obsahuje booleovskou operaci, potomci jsou jejími operandy
- listy jsou geometrická primitiva

CSG - příklady

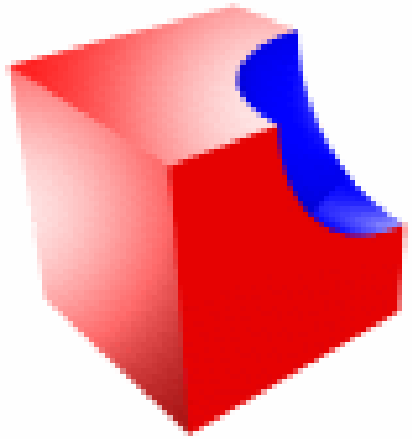
- A ... krychle, B ... koule



$A \cup B$

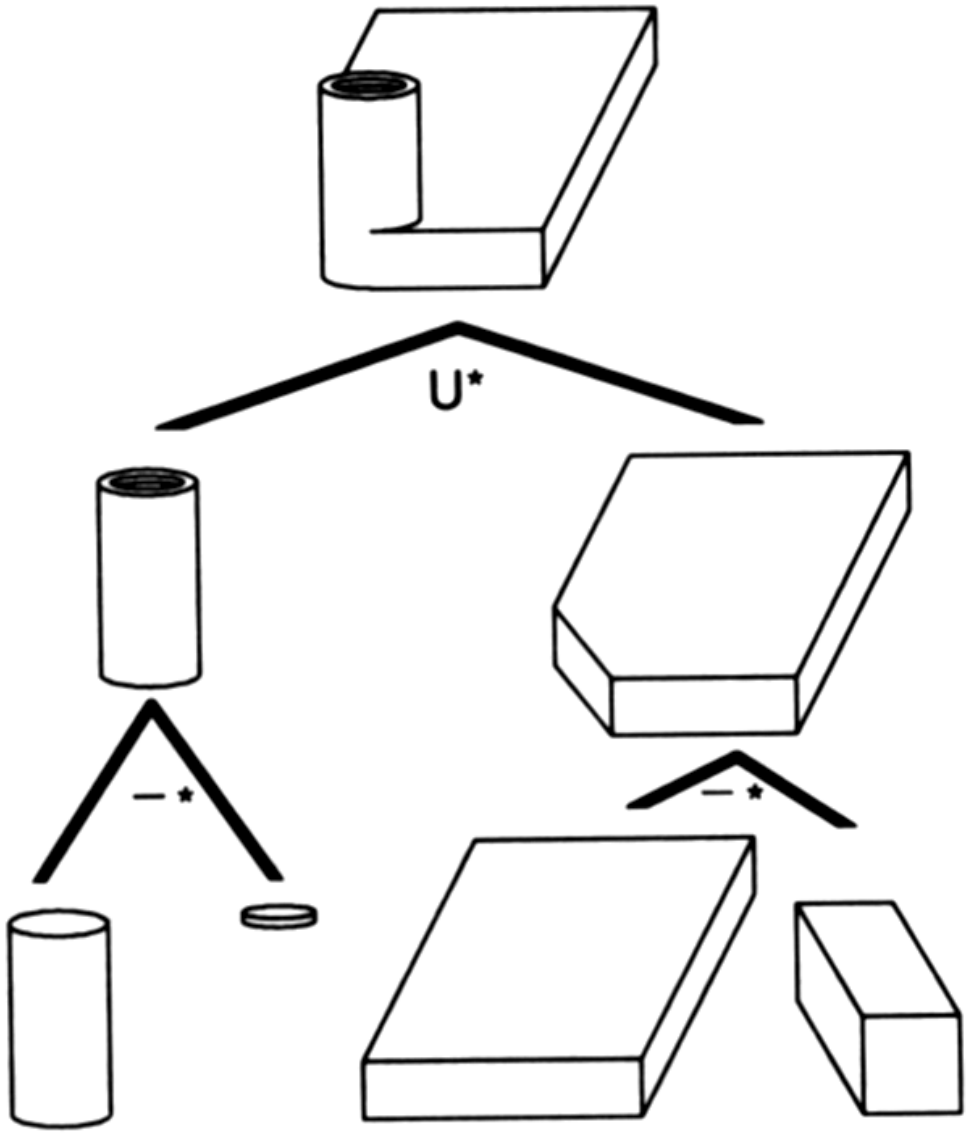


$A \cap B$



$A - B$

CSG - konstrukce stromu



CSG - možnosti

- CSG reprezentace není jednoznačná (stejný výsledek lze nakombinovat více způsoby)
- primitiva mohou být reprezentována v normovaném tvaru (jednotková koule v počátku, jednotková krychle atd.)
- k operacím mohou přibýt i maticové transformace (rotace, translace, škálování)
- transformace lze aplikovat na primitiva i kdekoliv výše ve stromu na složitější tělesa
- zobrazování CSG reprezentace - raytracing, raycasting

Obsah

Tahové reprezentace

- *sweep* representation, šablonování
- pohyb objektu $A_0 \in \mathbb{R}^3$ (nebo $A_0 \in \mathbb{R}^2$ - rovinný objekt)
  - tažení po určité trajektorii H_s dané funkcí $s : [0, T] \mapsto \mathbb{R}^3$
  - současná transformace pomocí operátoru $Q : [0, T] \mapsto \mathcal{L}$ závislého na „čase“

$\Longrightarrow$ vznik nového objektu A:

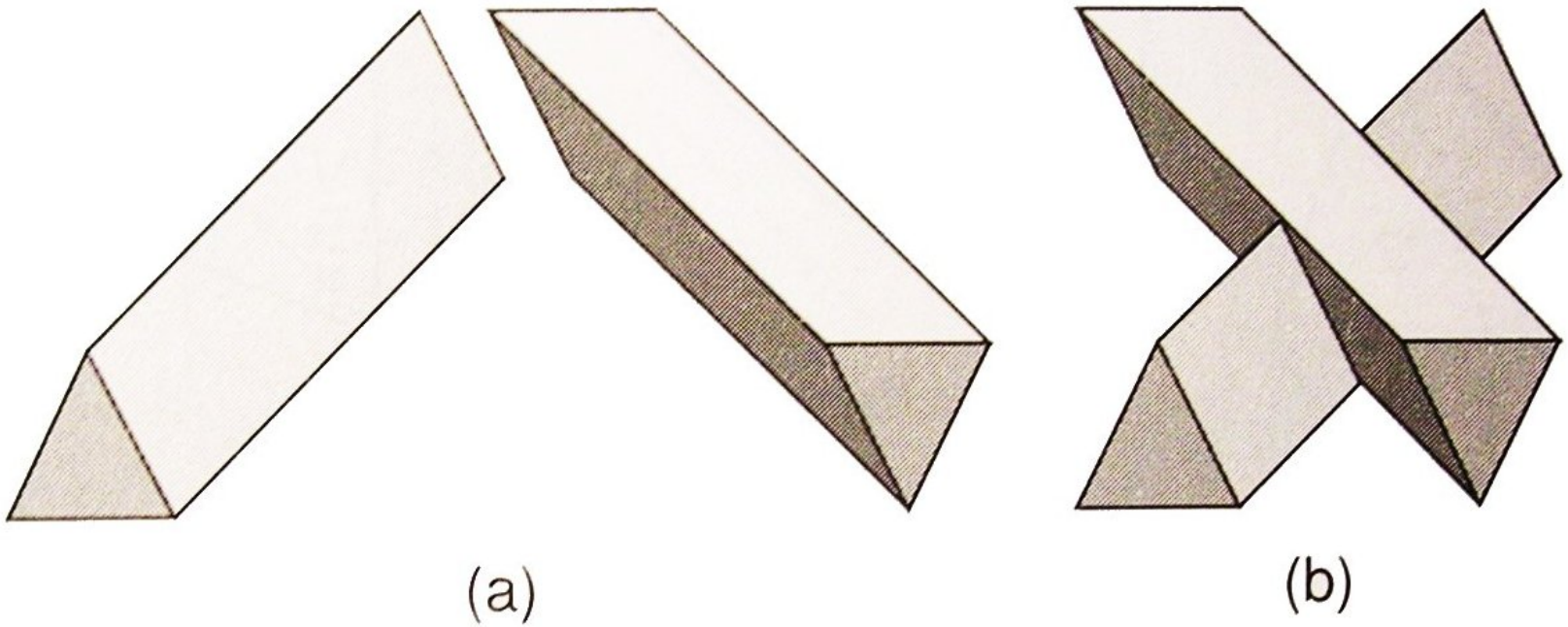
$$A = \left\{ Q(t)\, x + s(t) \middle| \, x \in A_0 \wedge t \in [0, T] \right\}$$

různé druhy objektů A_0 a trajektorií H_s:

- A_0 je 2D (rovinný) objekt, tažený po úsečce na něj kolmé - *translational sweep* nebo *extrusion*
- rotace 2D objektu kolem osy - *rotational sweep*
- obecný pohyb 2D objektu - tzv. zobecněný válec

Vlastnosti tahových reprezentací

- nejsou uzavřené vůči booleovským operacím



- jsou přirozeným postupem výroby mnoha objektů

Tahové reprezentace - příklady

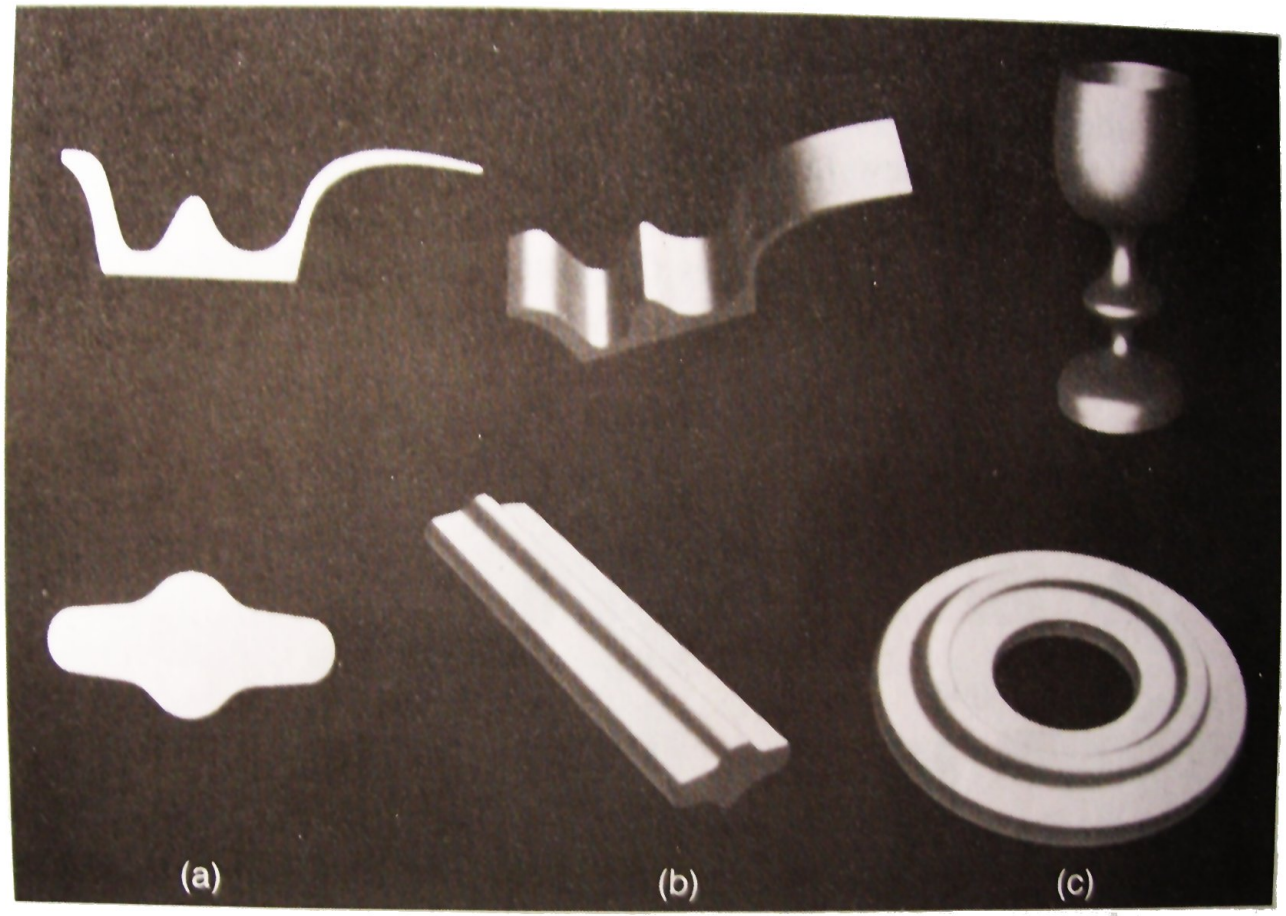
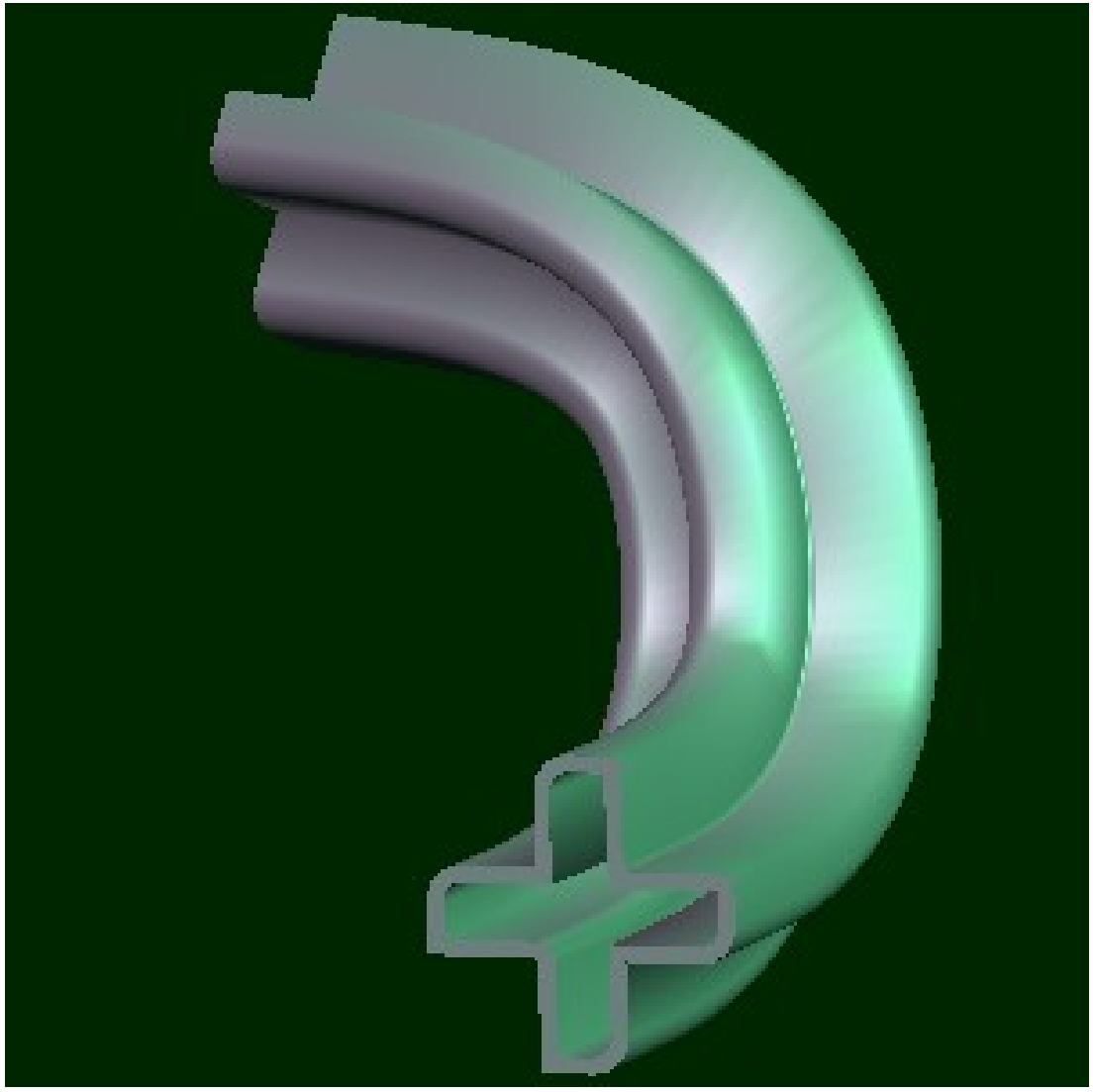


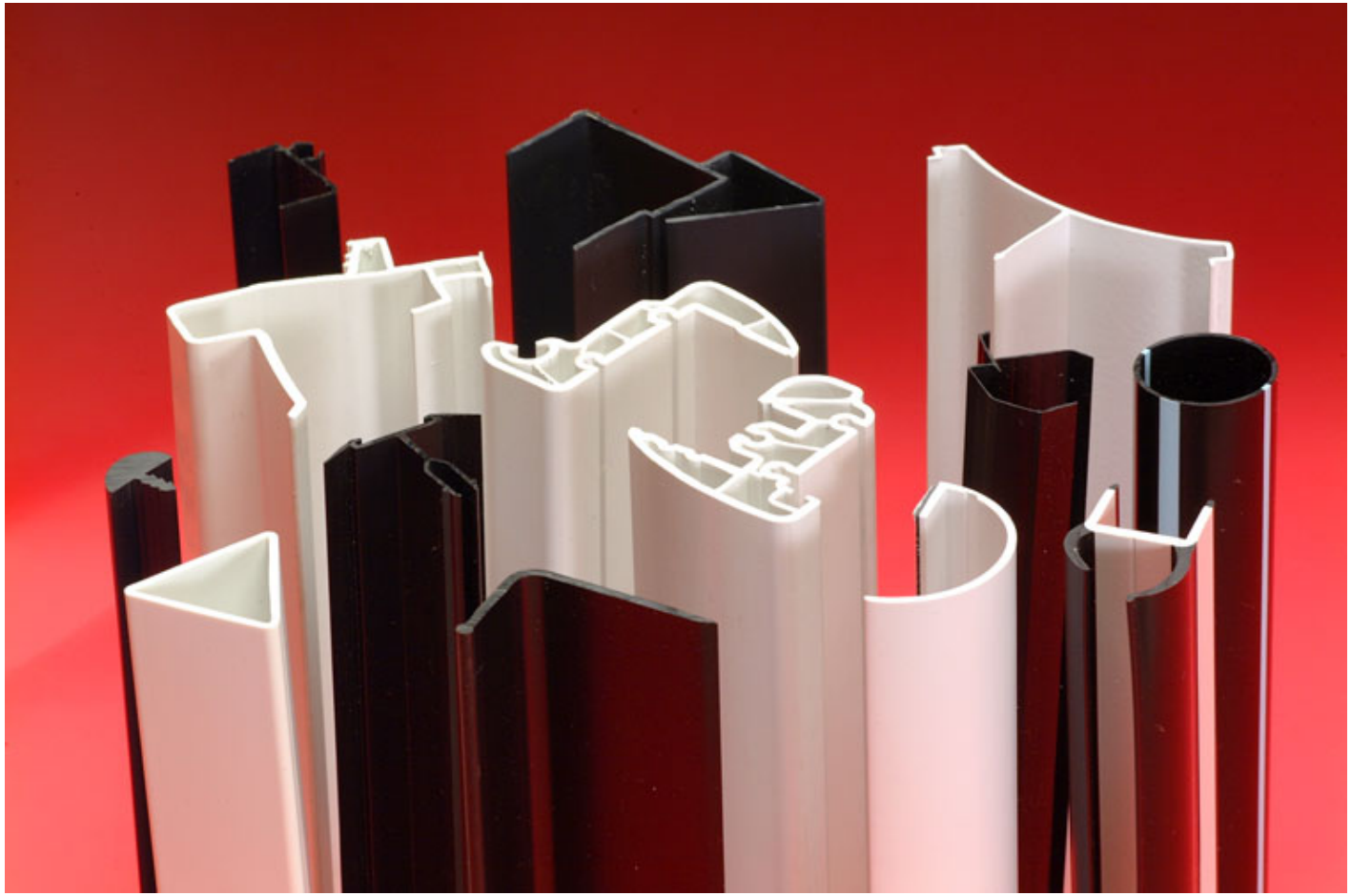
Fig. 12.8 Sweeps. (a) 2D areas are used to define (b) translational sweeps and (c) rotational sweeps. (Created using the Alpha_1 modeling system. Courtesy of the University of Utah.)

Tahové reprezentace - příklady



Tahové reprezentace - příklady

Tahové reprezentace - příklady

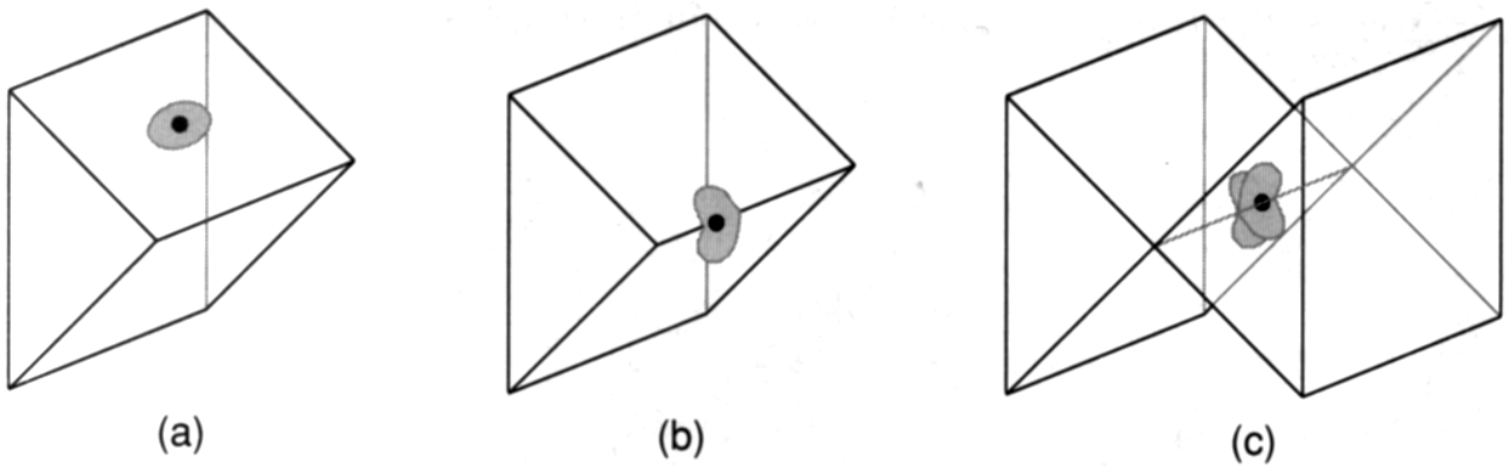


Plastové výrobky

Obsah

Hraniční reprezentace

- boundary representation, „*b-rep*“
- popis objektu pomocí jeho povrchu - vrcholy, hrany a stěny
- obvykle polygonální reprezentace, resp. navíc požadavek konvexnosti polygonů
- častý požadavek, aby objekty byly tzv. 2-variety (*manifold*): lokálně popsané homeomorfismem $f : \mathbb{R}^2 \mapsto \mathbb{R}^3$ (f prostá, f spojitá, f^{-1} spojitá), tj. okolí každého bodu je topologicky ekvivalentní s okolím v rovině



Mnohostěny

- *mnohostěn* - těleso ohraničené množinou polygonů, jejichž hrany jsou vždy společné sudému počtu stěn (pro 2-variety právě dvěma)
- jednoduchý mnohostěn - nemá díry $\Longrightarrow$ lze jej deformovat na kouli
- **Eulerův zákon** pro mnohostěny:

$$V - E + F = 2,$$

kde V je počet vrcholů (vertex), E počet hran (edge) a F počet stěn (face).

  - nutná, ale nikoliv postačující podmínka, aby dané těleso bylo mnohostěn

Další podmínky na mnohostěn

- každá hrana musí spojovat dva vrcholy a být společný dvěma stěnám
- ve vrcholech se musí stýkat alespoň tři hrany
- stěny se nesmí navzájem pronikat
- Eulerův zákon pro 2-variety s dírami

$$V - E + F - H = 2(C - G),$$

kde navíc H je počet děr ve stěnách, G počet děr procházejících skrz těleso a C je počet jednotlivých komponent objektu.

Generování vrstevnice - Marching cubes 1/2

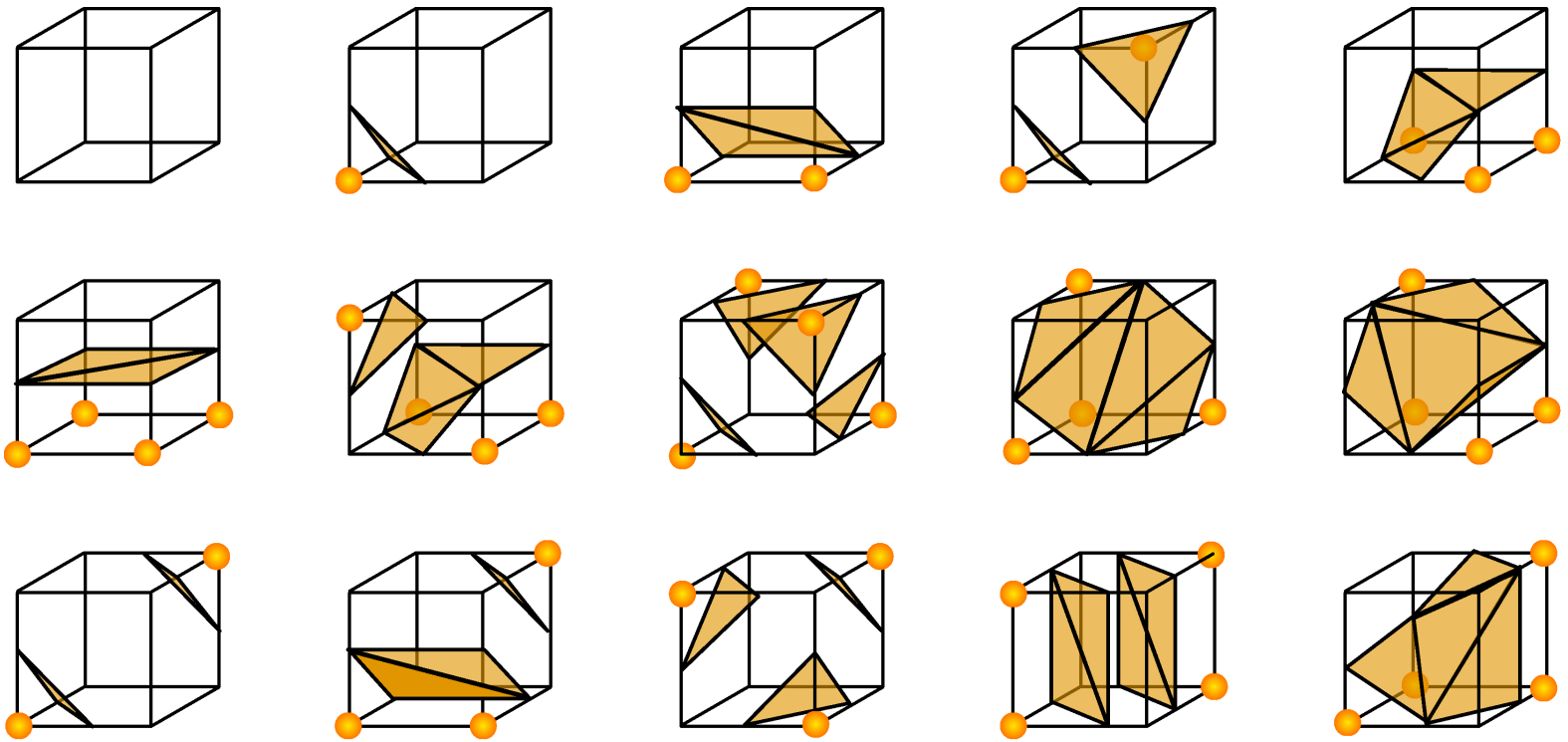
- skalární data definovaná v 3D objemu, resp. na 3D síti
- mnoho zdrojů dat: numerické simulace, měření
  - proudění (pole tlaku, teploty, složky rychlosti, hustoty, koncentrace, ...)
  - deformace pevných těles (tlak, napětí, ...)
  - medicína: MRI, CT, ...
- snaha rekonstruovat a zobrazit povrch

$$f(x, y, z) = C$$

- *marching cubes* - rozdělení prostoru na krychle, určení hodnoty f v rozích krychle
  - $f(x, y, z) > C$
  - $f(x, y, z) < C$
- krychlemi, kde se vyskytují oba případy, prochází vrstevnice

Generování vrstevnice - Marching cubes 2/2

- 15 různých případů (až na symetrie)

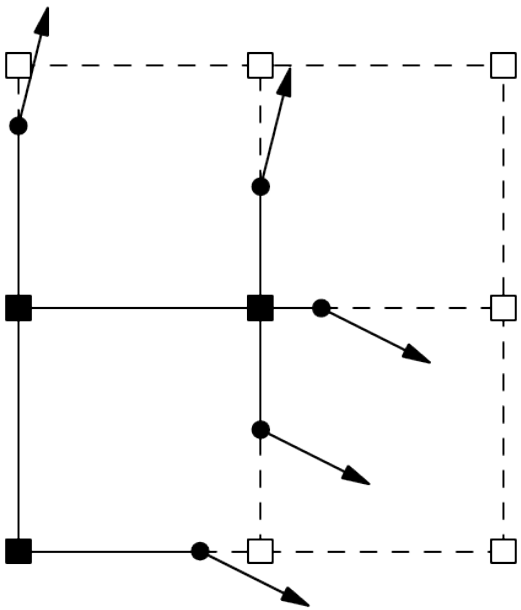


- výpočet normálových vektorů podle gradientu pole (pro osvětlovací model)

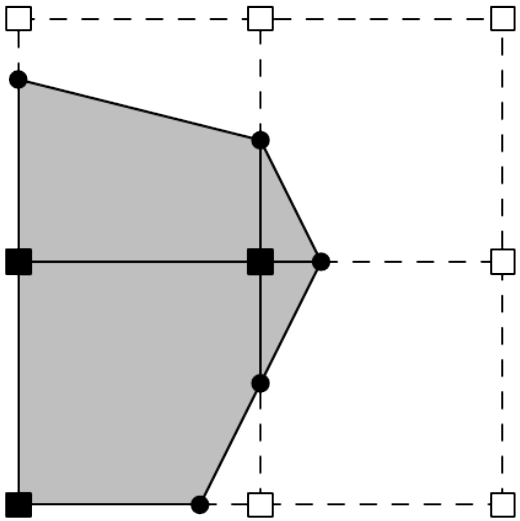
Generování vrstevnice - Dual Contouring

- rozdělení prostoru na krychličky (lze rovnoměrně i adaptivně)
- identifikace hran, kterými prochází izoplocha (stejně, jako v *marching cubes*)
  - tzv. *heterogenní* hrany
- výpočet průsečíků vrstevnice s heterogenními hranami
  - parametrický popis hrany + řešení rovnice $f(\boldsymbol{x}(t)) = 0$
- výpočet normálových vektorů k ploše v průsečících (výpočet ∇f)
- vygenerování **1 vrcholu polygonální sítě uvnitř krychle** pomocí dat ze všech heterogenních hran
  - minimalizace kvadratické funkce nebo iterativní pohyb
- každé heterogenní hraně přísluší 1 polygon s vrcholy ve všech krychlích, které ji sdílejí

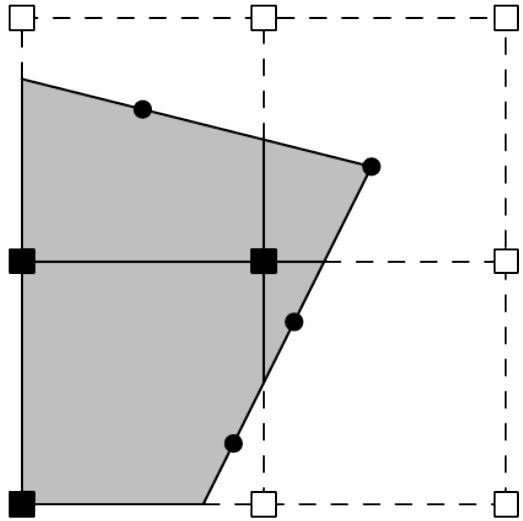
Marching Cubes vs. Dual Contouring



hermitovská data



Marching Cubes



Dual Contouring

CSG v implicitním popisu těles

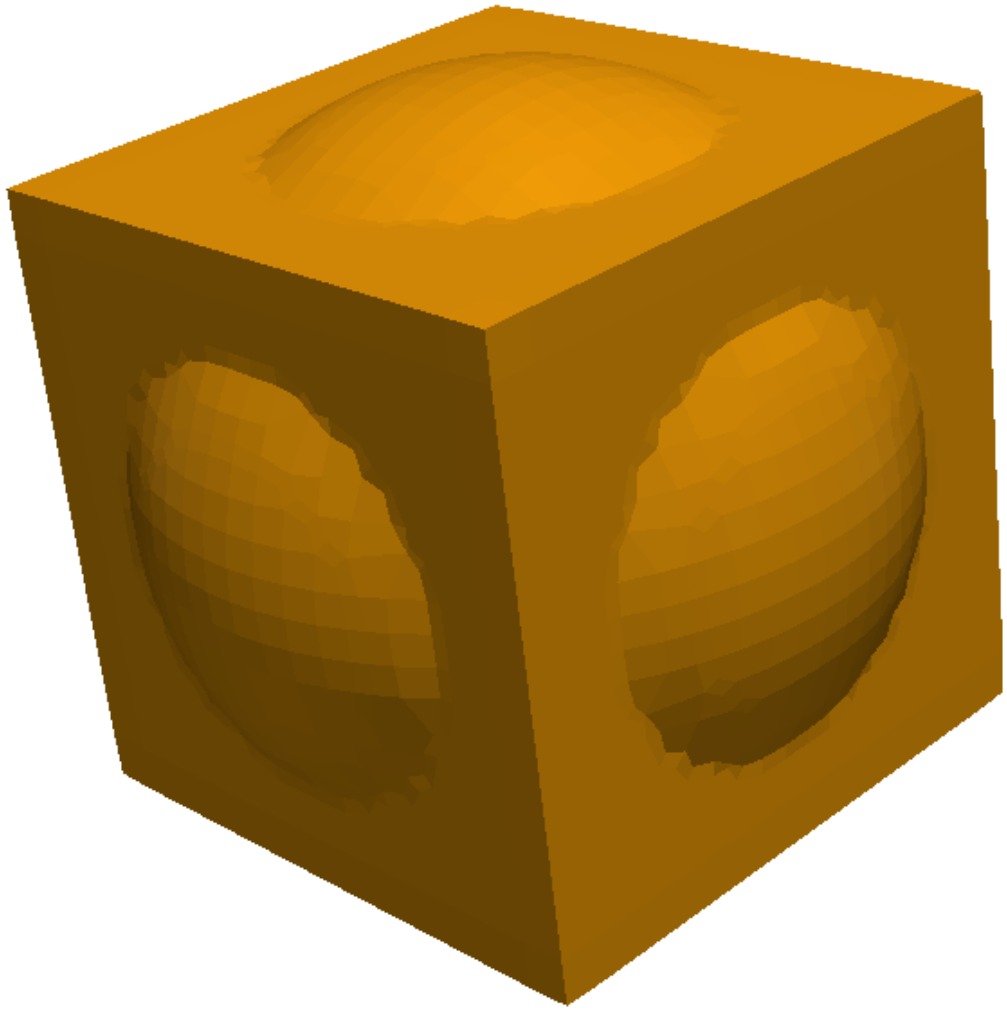
- dvě tělesa A, B popsaná pomocí nulových izoploch funkcí f, g jako

$$A = \left\{ \boldsymbol{x} \in \mathbb{R}^3 \middle| f(\boldsymbol{x}) \leq 0 \right\},\ B = \left\{ \boldsymbol{x} \in \mathbb{R}^3 \middle| g(\boldsymbol{x}) \leq 0 \right\}$$

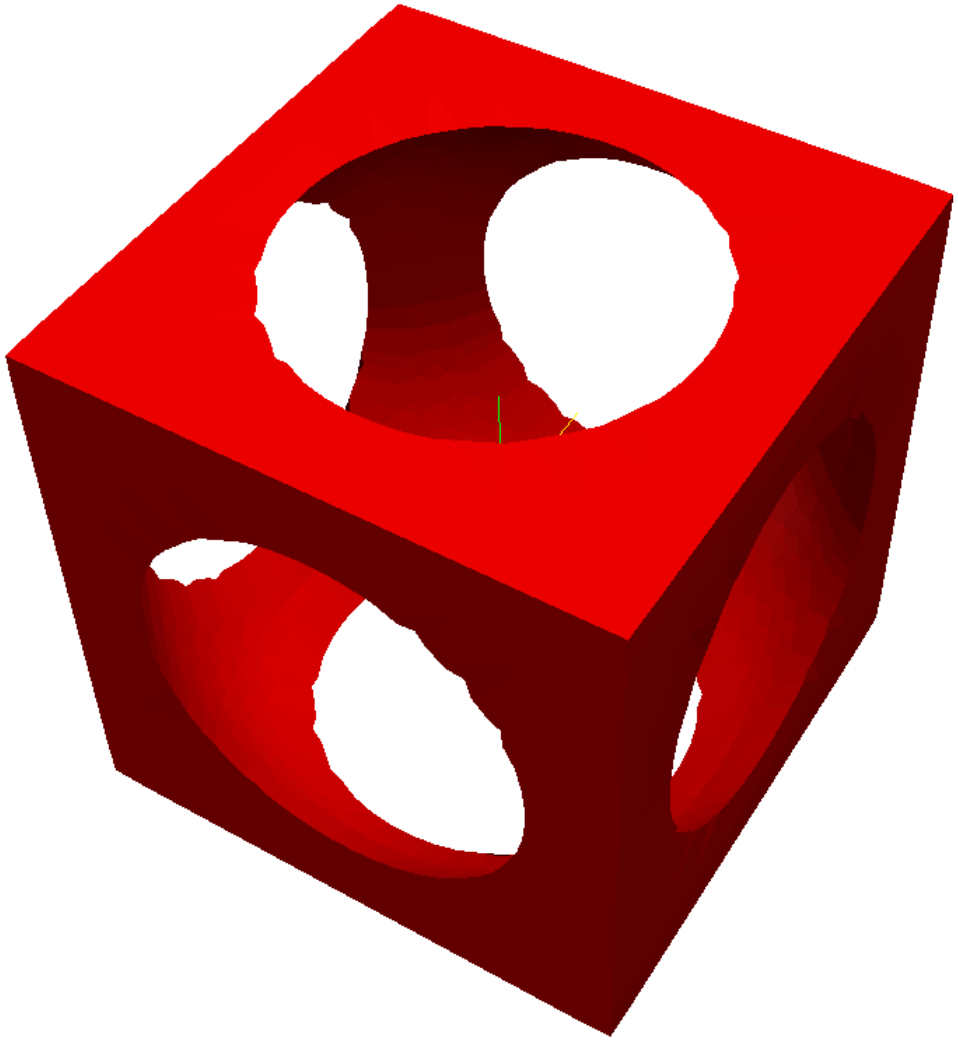
- **potom:**

- sjednocení $A \cup B = \left\{ \boldsymbol{x} \in \mathbb{R}^3 \middle| \min\{f(\boldsymbol{x}), g(\boldsymbol{x})\} \leq 0 \right\}$
- průnik $A \cap B = \left\{ \boldsymbol{x} \in \mathbb{R}^3 \middle| \max\{f(\boldsymbol{x}), g(\boldsymbol{x})\} \leq 0 \right\}$
- rozdíl $A \backslash B = \left\{ \boldsymbol{x} \in \mathbb{R}^3 \middle| \max\{f(\boldsymbol{x}), -g(\boldsymbol{x})\} \leq 0 \right\}$

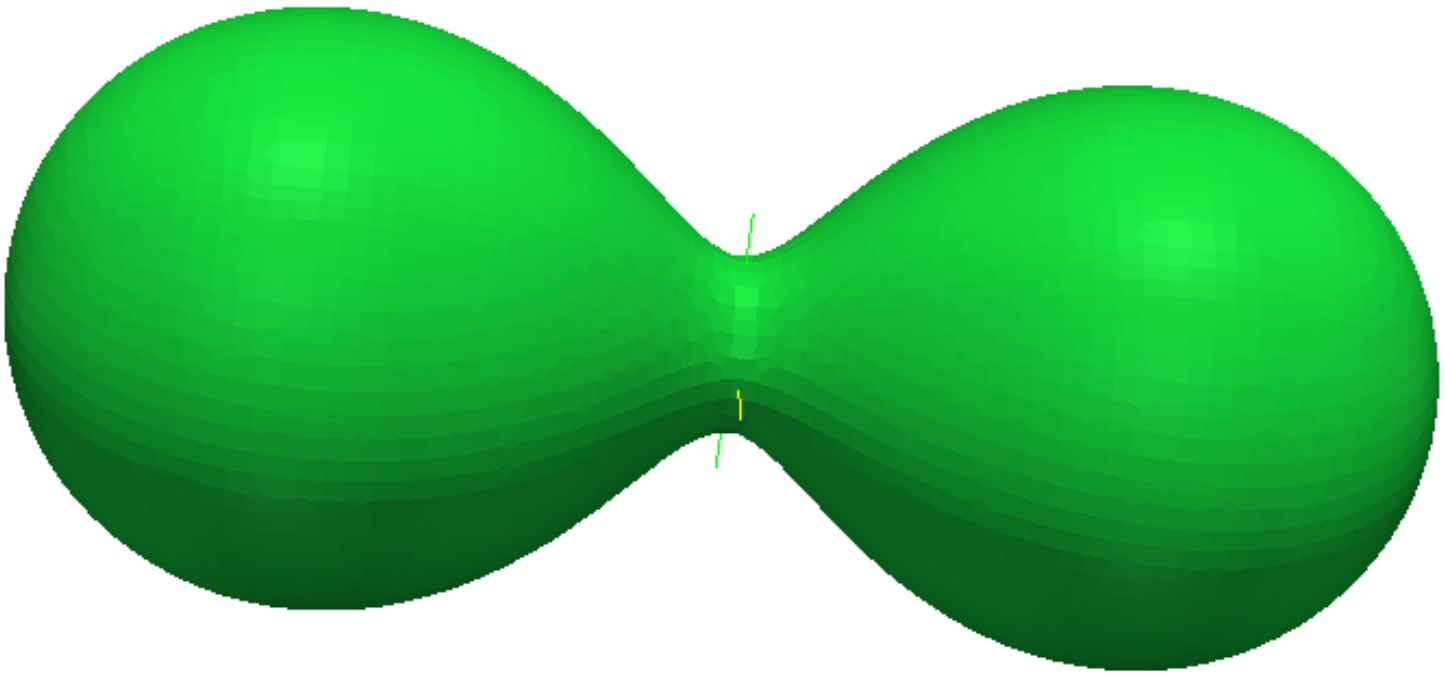
Dual Contouring - příklady



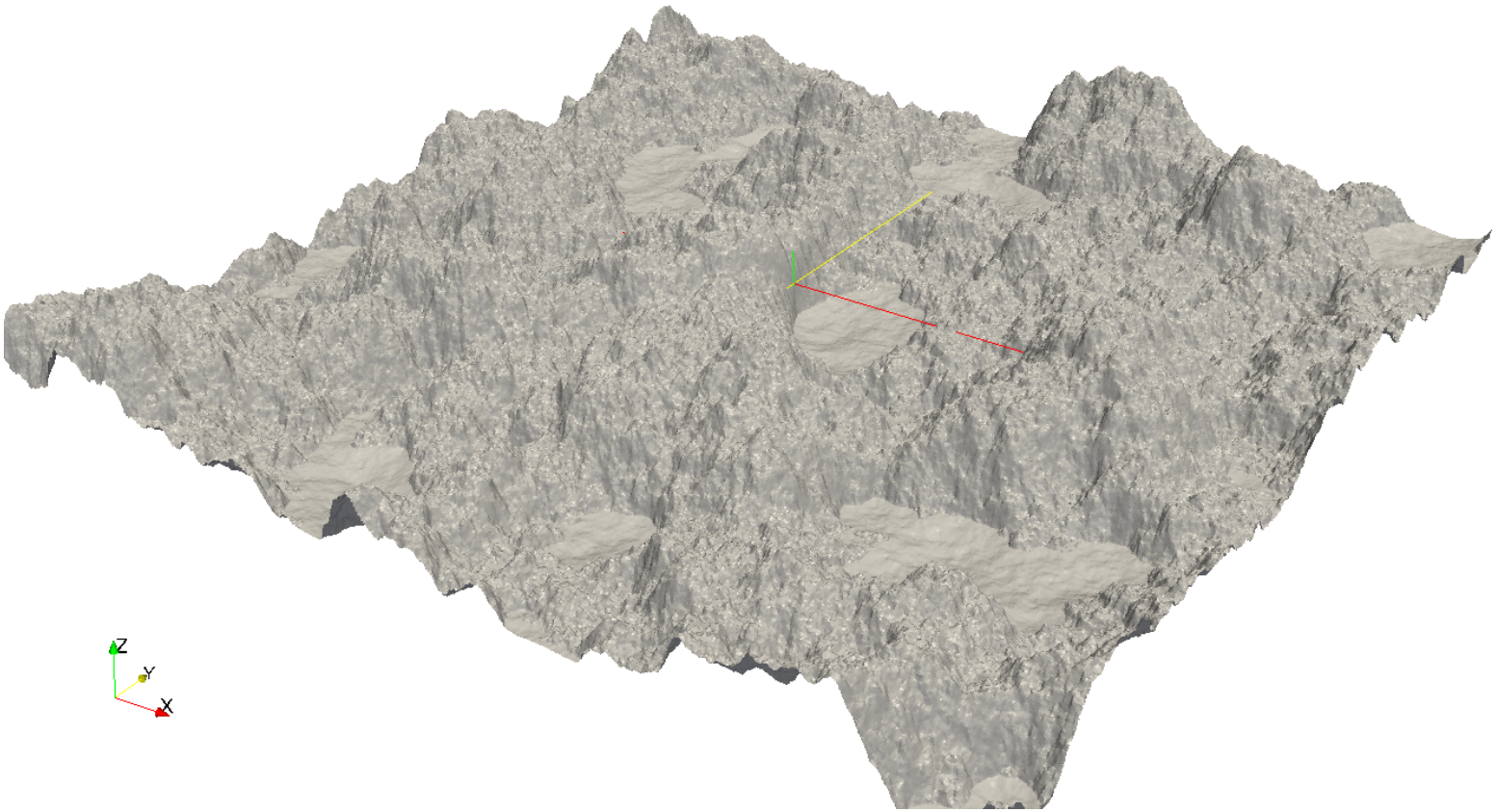
Dual Contouring - příklady



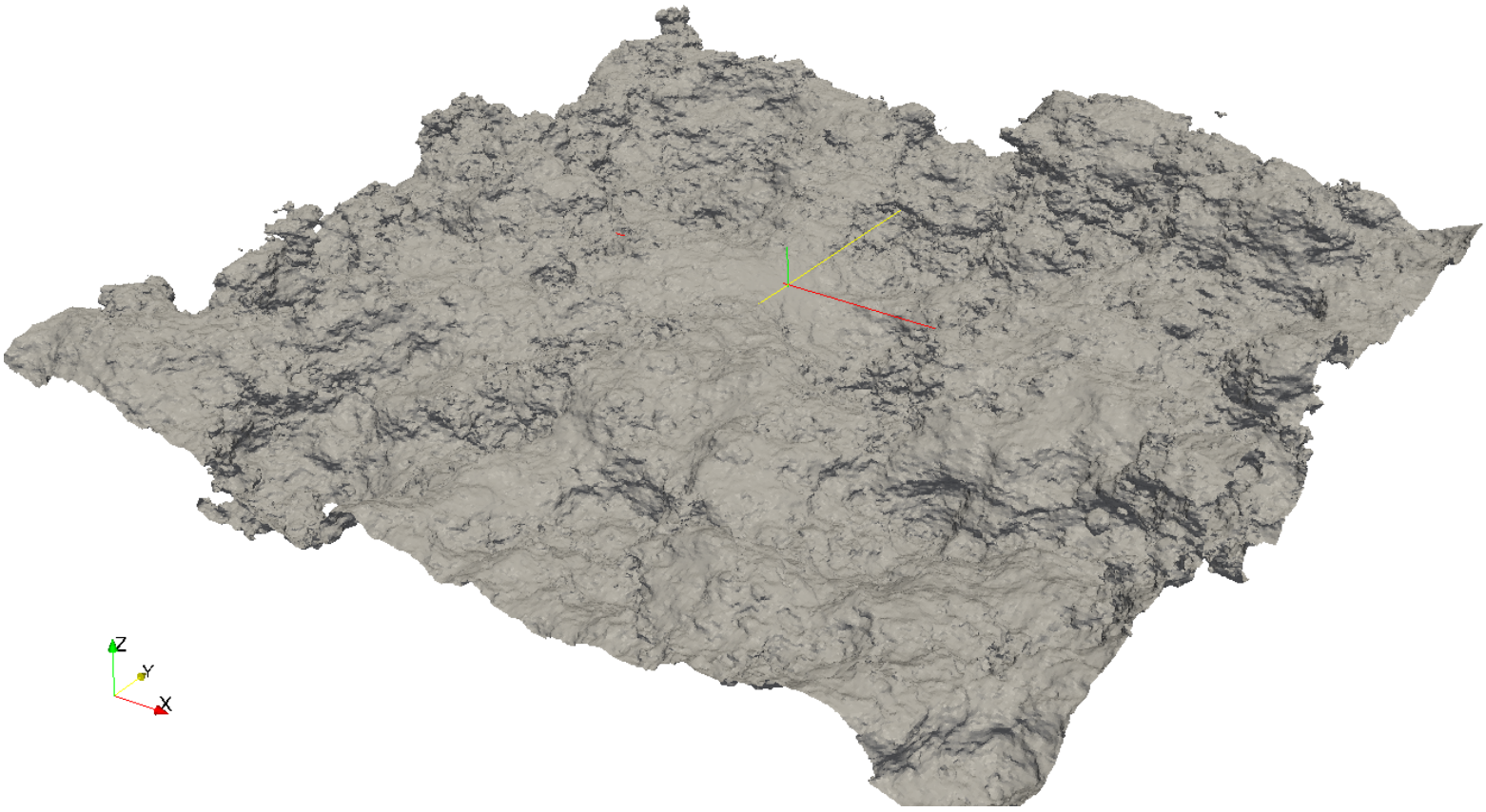
Dual Contouring - příklady



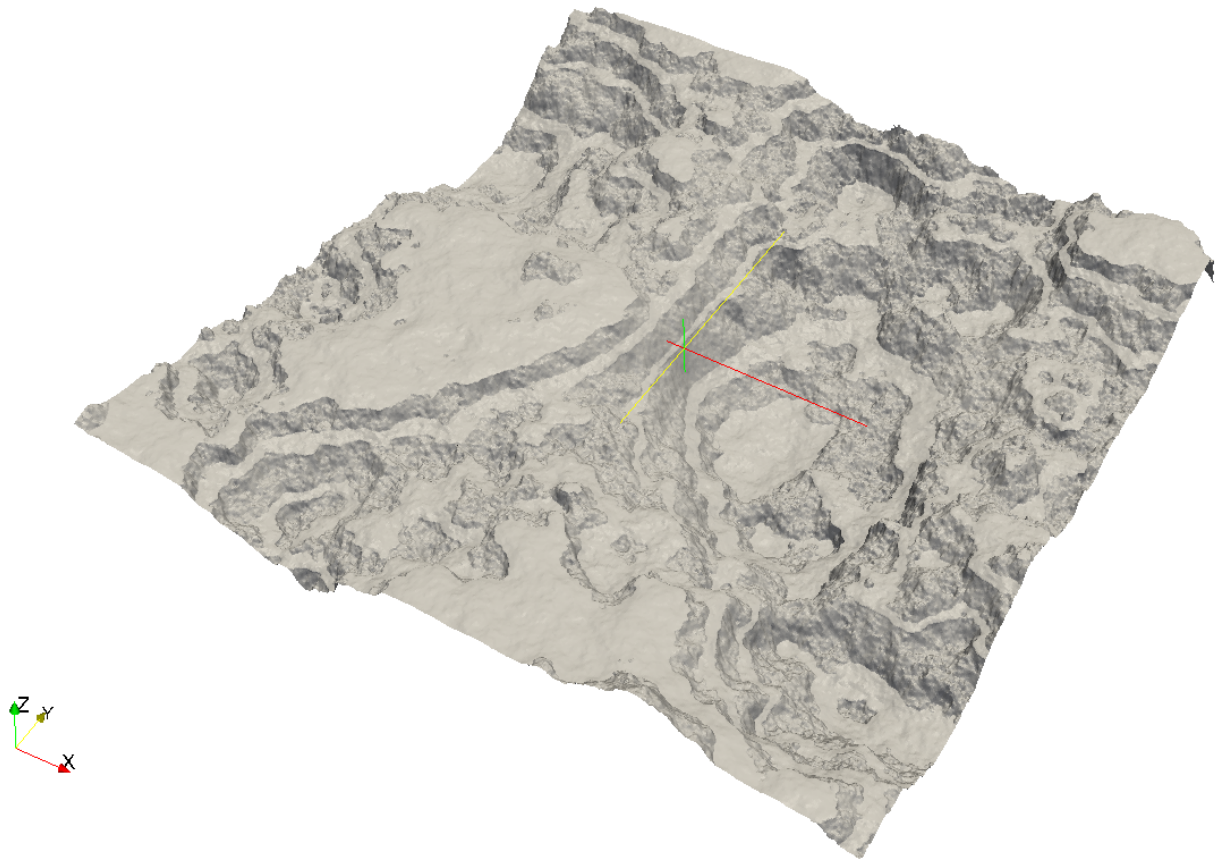
Dual Contouring - příklady



Dual Contouring - příklady



Dual Contouring - příklady



Obsah

Polygonální sítě

- množina vrcholů, hran a polygonů
  - hrana tvořená dvěma vrcholy
  - polygon tvořen (uzavřenou) sekvencí hran
  - hrana sdílena 2 polygony, vrchol sdílen alespoň třemi hranami
- vhodná reprezentace polygonální sítě v počítači
  - implicitně obsahuje pravidla výše
  - snadno umožňuje implementovat typické operace: najít hrany sdílející vrchol, najít polygony sdílející hranu nebo vrchol, najít hrany polygonu apod.

Reprezentace polygonálních sítí

5 způsobů uložení polygonální sítě v počítači

- explicitní
- ukazatele do seznamu vrcholů
- ukazatele do seznamu hran
- okřídlená hrana (winged edge)
- half edge

Explicitní reprezentace

- každý polygon reprezentován seznamem souřadnic vrcholů

$$P = (\boldsymbol{v}_1, \boldsymbol{v}_2, , \ldots, \boldsymbol{v}_n)$$

$$\boldsymbol{v}_i = [x_i, y_i, z_i]$$

- vrcholy v pořadí procházení kolem P
- hrany mezi $\boldsymbol{v}_i$ a $\boldsymbol{v}_{i+1}$ (kde $\boldsymbol{v}_{n+1} = \boldsymbol{v}_1$)
- mimořádně nešikovné:
  - plýtvání - souřadnice sdílených vrcholů uloženy vícekrát
  - není informace o sdílených hranách a vrcholech - nutné prohledávat všechny polygony + porovnávat `float` čísla ... nespolehlivé
  - při kreslení hran - každá hrana kreslena dvakrát

Ukazatele do seznamu vrcholů

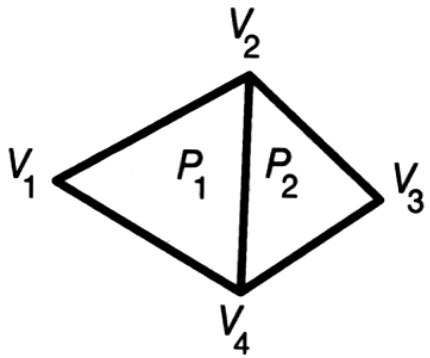
- všechny vrcholy uloženy v jednotném seznamu vrcholů

$$V = (\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_N)$$

- polygon definovaný seznamem indexů (resp. ukazatelů „$*$“) do seznamu vrcholů

$$P = (*\boldsymbol{v}_1, *\boldsymbol{v}_2, \ldots, *\boldsymbol{v}_n)$$

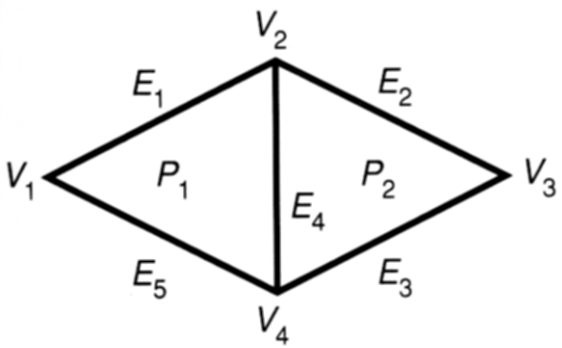
- výhoda - každý vrchol uložen jen jednou
- nevýhoda - stále žádná informace o společných hranách



$$V = (V_1, V_2, V_3, V_4) = ((x_1, y_1, z_1), \ldots, (x_4, y_4, z_4))$$
$$P_1 = (1, 2, 4)$$
$$P_2 = (4, 2, 3)$$

Ukazatele do seznamu hran

- seznam vrcholů V
- hrana $e = \{*v_1, *v_2, *P_1, *P_2\}$ definovaná **ukazateli** do V, navíc vybavená ukazateli na polygony, které ji sdílejí
- seznam hran $E = (e_1, e_2, \ldots, e_M)$
- polygon $P = (*e_1, *e_2, \ldots, *e_n)$ je n-tice ukazatelů do E
- stále není snadné určit hrany procházející z vrcholu (nutno prohledat všechny hrany)



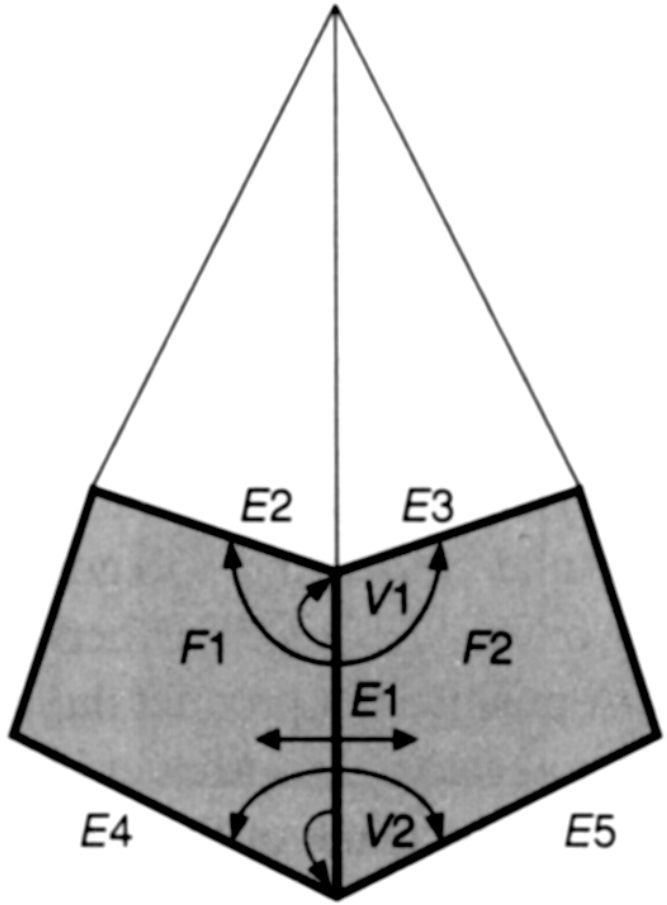
$$V = (V_1, V_2, V_3, V_4) = ((x_1, y_1, z_1), \ldots, (x_4, y_4, z_4))$$
$$E_1 = (V_1, V_2, P_1, \lambda)$$
$$E_2 = (V_2, V_3, P_2, \lambda)$$
$$E_3 = (V_3, V_4, P_2, \lambda)$$
$$E_4 = (V_4, V_2, P_1, P_2)$$
$$E_5 = (V_4, V_1, P_1, \lambda)$$
$$P_1 = (E_1, E_4, E_5)$$
$$P_2 = (E_2, E_3, E_4)$$

Okřídlená hrana 1/2

- hrana má ukazatele na:
  - své dva vrcholy
  - dva polygony, které ji sdílejí
  - čtyři další hrany, které z ní vycházejí
- vrchol má
  - souřadnice
  - ukazatel na jednu z hran, které z něj vycházejí
- polygon má
  - ukazatel na jednu ze svých hran (neumožňuje díry)

nebo

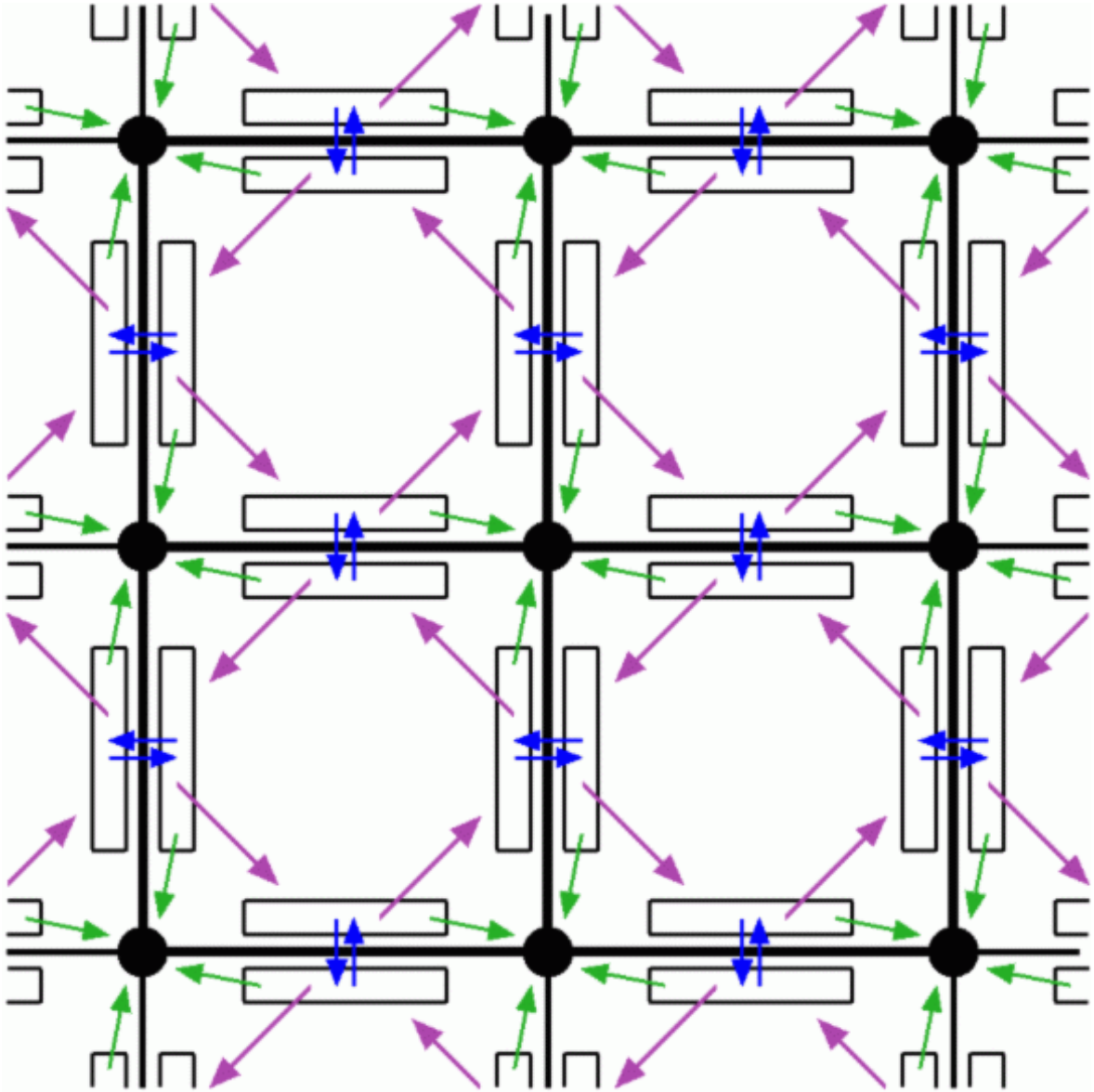
- seznam smyček hran - jedna vnější smyčka procházená po směru hod. ručiček a nula nebo více vnitřních smyček procházených proti směru hod. ručiček = děr



Okřídlená hrana 2/2

- geometrie zabírá málo místa oproti *topologii*
- snadné určení sousedství hran, vrcholů, polygonů, příslušnost hran polygonům atd.
- do reprezentace lze přidat další informace: normály ve vrcholech, barva + normála pro stěnu (pro stínování), ...
- nejčastěji používaná reprezentace, avšak stále ne vždy vhodná - založená na práci s hranami (dělené povrchy - práce se stěnami)

Half-edge



Zpracování polygonálních sítí

4 operace s polygonální sítí před vykreslením

1. *teselace* (*tessellation* = vydláždění) ... vyjádření 3D dat ve vhodné polygonální reprezentaci (trojúhelníky $\Longrightarrow$ *triangulace*, ale i konvexní polygony,...). Výslednou síť lze efektivně dále zpracovávat (hardwarová akcelerace).
2. *konsolidace* ... příprava pro zobrazování (generování normál, zajištění konzistentní orientace polygonů, označení ostrých hran, ...)
3. výroba pásů (*strip*), resp. vějířů (*fan*) trojúhelníků pro rychlý rendering (efektivní datová struktura, podpora HW akcelerace)
4. zjednodušení sítě (snížení počtu trojúhelníků - pro určité situace stále dostačující model)
   - uniformní
   - adaptivní

Triangulace 1/3

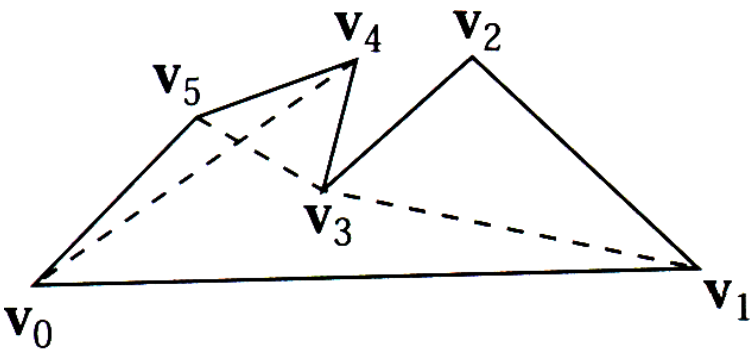
- **promítnutí polygonu do roviny** zjednoduší algoritmy triangulace
- zahození jedné ze souřadnic x, y, z $\Longrightarrow$ promítnutí do roviny yz, xz , xy
- jak vybrat nejvhodnější rovinu (vzhledem k robustnosti numerických výpočtů)
  - vybrat rovinu, kde má průmět polygonu největší plochu
  - zahodit souřadnici, odpovídající složce normálového vektoru ν polygonu, která má největší absolutní hodnotu
- oba uvedené testy nejsou vždy ekvivalentní - záleží na způsobu výpočtu ν
- průmět polygonu nemusí tvořit jednoduchou křivku ($\Longrightarrow$ průsečíky)

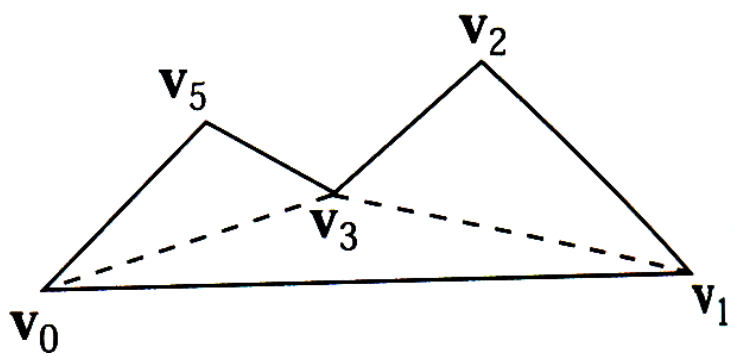
Triangulace 2/3

- triangulace polygonu o n vrcholech má $n-2$ trojúhelníků

ořezávání rohů (ear clipping) - nejjednodušší algoritmus, složitost $O\left(n^2\right)$

- polygon $P = (\boldsymbol{v}_0, \boldsymbol{v}_1, , \ldots, \boldsymbol{v}_{n-1})$; $\boldsymbol{v}_n = \boldsymbol{v}_0$
- procházíme vrcholy $\boldsymbol{v}_i, \boldsymbol{v}_{i+1}, \boldsymbol{v}_{i+2}$ (modulo n)
- jestliže úsečka $\boldsymbol{v}_i \boldsymbol{v}_{i+2}$ neprotíná žádnou hranu polygonu, tak vrchol $\boldsymbol{v}_{i+1}$ tvoří roh
- vrchol $\boldsymbol{v}_{i+1}$ odstraníme z polygonu, vytvoříme $\triangle \boldsymbol{v}_i \boldsymbol{v}_{i+1} \boldsymbol{v}_{i+2}$





Triangulace 3/3

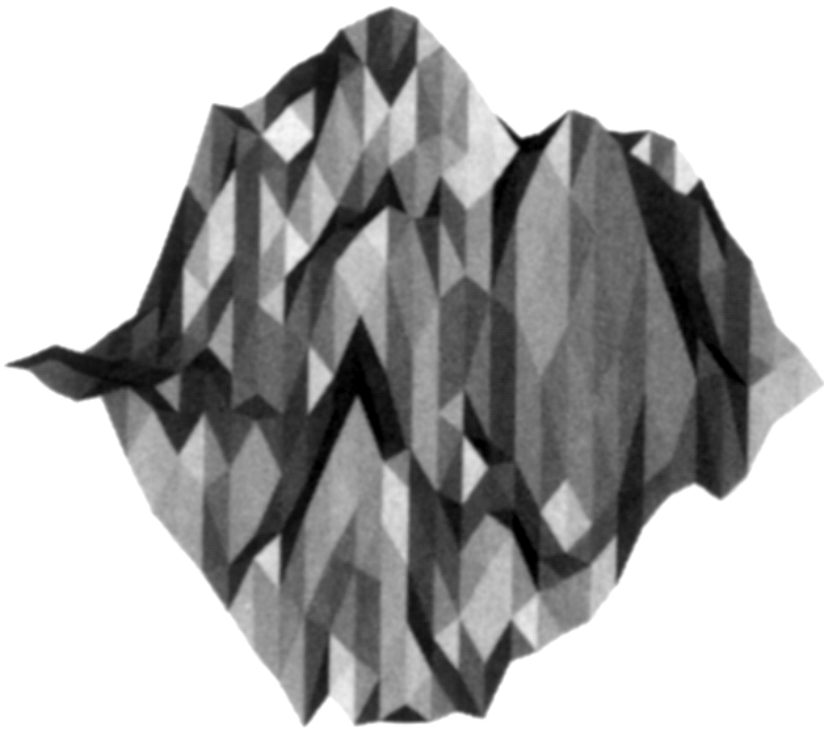
- komplexnější algoritmy - složitost $O(n \log n)$ nebo dokonce $O(n)$ (pouze průměrná složitost, existují protipříklady)
- někdy stačí rozdělit polygon na konvexní komponenty
- použitelný teselátor
  - ošetření patologických případů
  - ošetření problémů konečné přesnosti výpočtů (robustnost)
- *OpenGL Utility Library*

Triangulace čtyřúhelníků

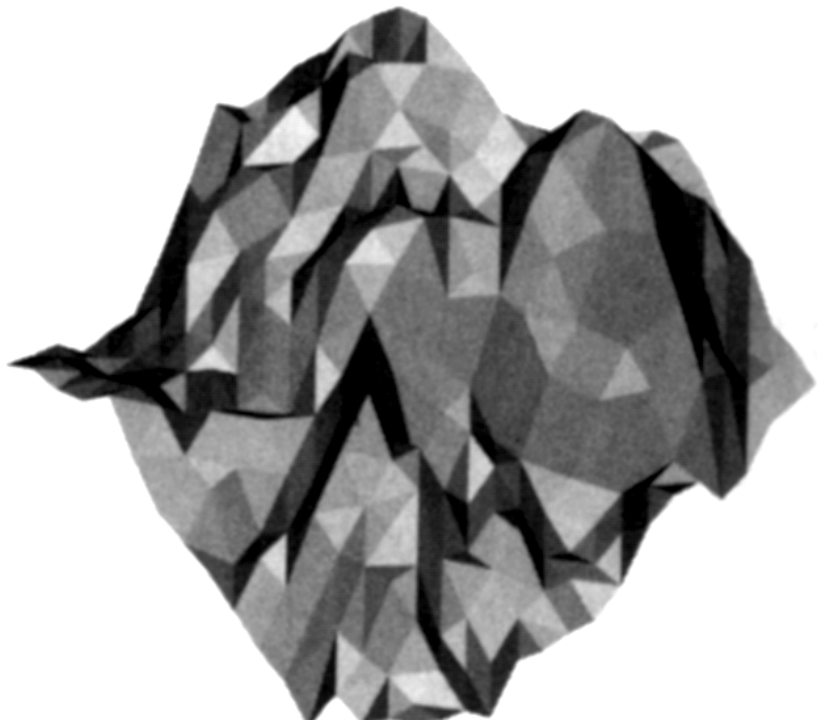
- vstupní data: např. dělení Catmull-Clark (generuje pouze čtyřúhelníky)
- nekonvexní čtyřúhelník - lze rozdělit jediným způsobem
- konvexní čtyřúhelník - volba diagonály ze 2 možností
  - bez dodatečných dat ve vrcholech $\Longrightarrow$ kratší diagonála
  - předpočítané osvětlení (viz radiozita - na konci semestru) $\Longrightarrow$ diagonála s menšími rozdíly mezi barvami

  - terén $\Longrightarrow$ výběr diagonály s menším/větším rozdílem výšek, úhlů normál $\triangle$, ...

Rozdíl v triangulaci čtyřúhelníků - terén



pravidelná volba diagonály



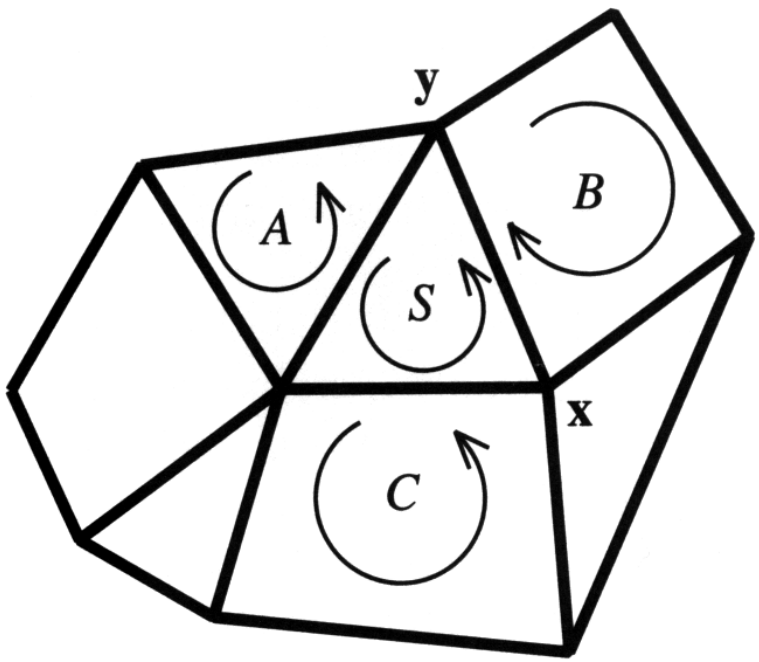
diagonála přes vrcholy s menším rozdílem výšek

Konsolidace - orientace polygonů 1/2

- vstup - množina libovolných polygonů bez orientace a bez normál
- konsolidace
  - vytvoření datové struktury: pro každou hranu koncové vrcholy + polygon, jemuž náleží
  - lexikografické uspořádání vrcholů v hraně, odstranění hran s nulovou nebo zanedbatelnou délkou
  - nalezení identických hran (snadné díky uspořádání) $\Longrightarrow$ styčné hrany polygonů + krajní hrany
  - orientace polygonů (chceme u všech stejnou orientaci - např. proti směru hodinových ručiček)

Konsolidace - orientace polygonů 2/2

1. sjednocení orientací všech polygonů ve skupině
   1. libovolný polygon P
   2. nastavení shodné orientace u sousedů P (= abychom společnou hranu procházeli v **opačném** směru než u P)
   3. rekurzivní opakování předchozího kroku pro všechny nezkontrolované sousedy již zkontrolovaných polygonů
2. v případě potřeby se orientace u všech polygonů otočí



Konsolidace - generování normál ve vrcholech

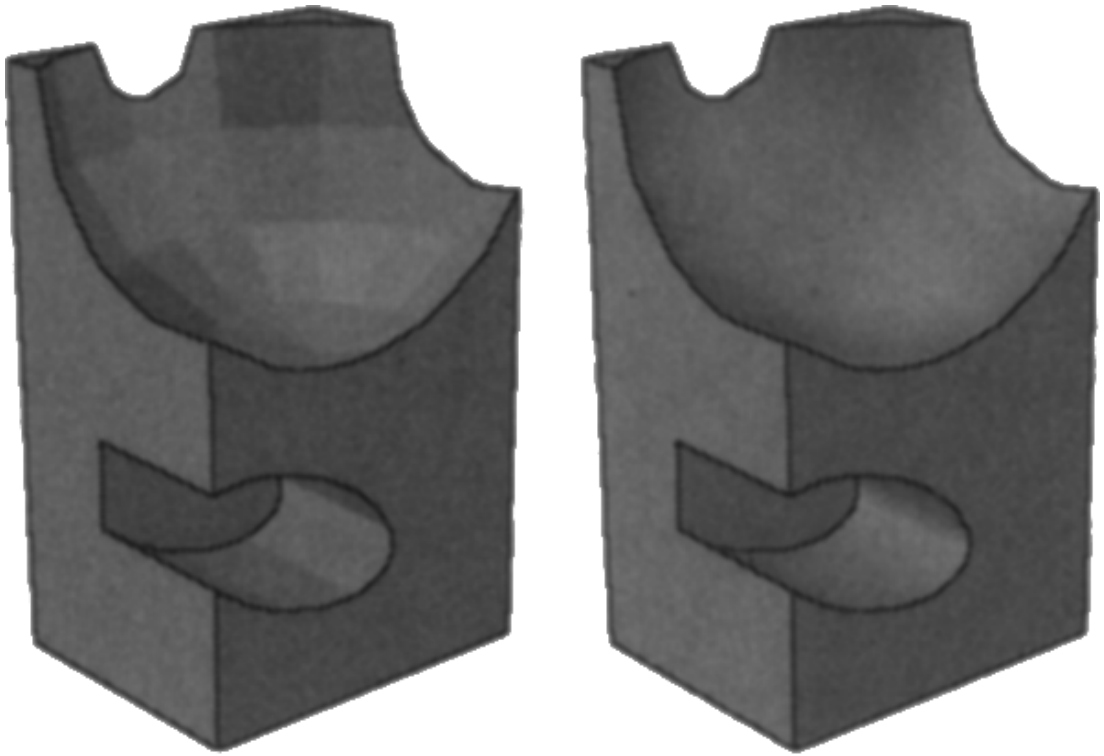
- normály se definují ve vrcholech, kde má být povrch ve skutečnosti hladký (jsou potřebné pro stínování - viz extra přednáška)
- způsoby, jak určit hladkost
  - vyhlazovací skupiny polygonů
  - mezní úhel (vyhlazujeme, pokud úhel mezi polygony (resp. jejich normálami) je $>$ (resp. $<$ pro normály) než daný práh)
- výpočet normály ve vrcholu
  - průměr normál přilehlých stěn (nevhodné)
  - vážený průměr podle délky hran polygonů (čím delší, tím menší vliv)

$$\boldsymbol{\nu} = \sum_{i=1}^{n} \frac{\boldsymbol{e}_i \times \boldsymbol{e}_{i+1}}{\|\boldsymbol{e}_i\|^2 \left\|\boldsymbol{e}_{i+1}\right\|^2}$$

pro n polygonů orient. proti směru hod. ručiček, $\boldsymbol{e}_i$ ... hrany vycházející z vrcholu, s indexy modulo n

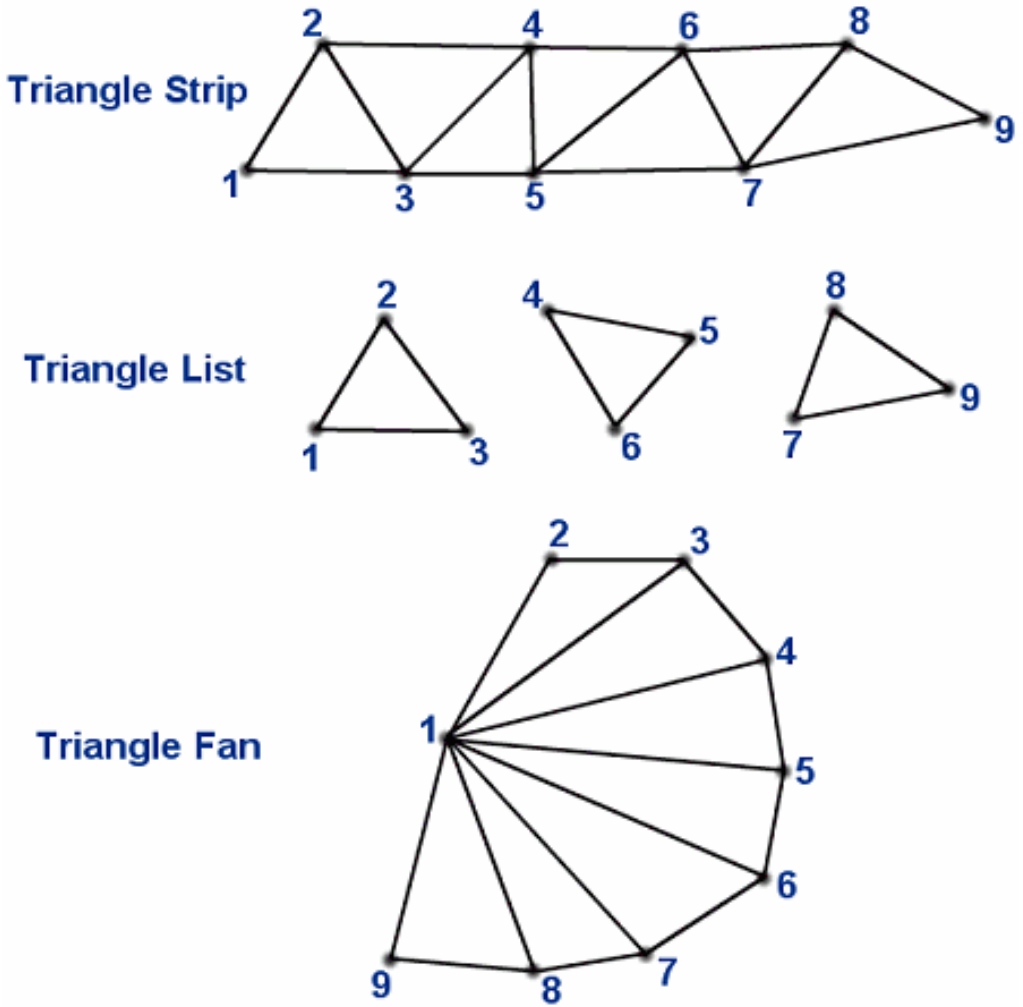
Konsolidace - označení ostrých hran

- ostrá hrana (*crease*) - hranice mezi polygony patřícími do různých vyhlazovacích skupin



Tvorba pásů a vějířů trojúhelníků

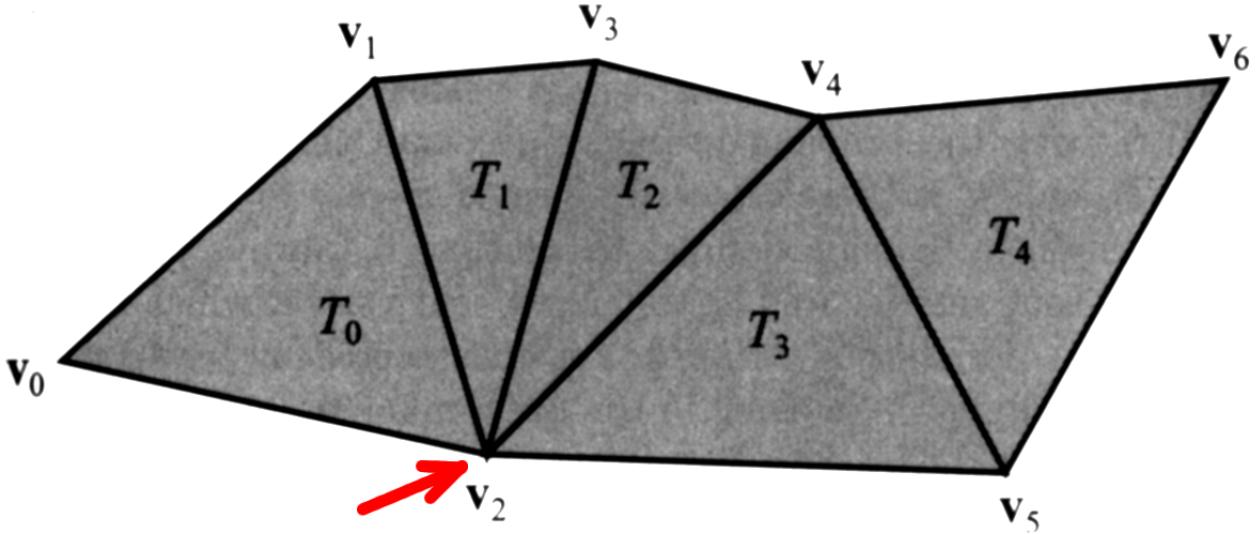
- snadná datová reprezentace, rychle se vykresluje
- definice jako seznam vrcholů $(\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_n)$



Prohození trojúhelníků (*swap*)

- za cenu přidání 1 vrcholu umožňuje generovat delší pásy (stále výhodnější než začít nový pás)

$$(\boldsymbol{v}_0, \boldsymbol{v}_1, \boldsymbol{v}_2, \boldsymbol{v}_3, \boldsymbol{v}_4, \ldots, \boldsymbol{v}_n) \mapsto \left(\boldsymbol{v}_0, \boldsymbol{v}_1, \boldsymbol{v}_2, \boldsymbol{v}_3, \underline{\boldsymbol{v}_2}, \boldsymbol{v}_4, \ldots, \boldsymbol{v}_n\right)$$



z vrcholu $\boldsymbol{v}_2$ nyní vychází o jednu hranu více

Literatura

- J. D. Foley, A. van Dam, S. K. Feiner, J. F. Hughes: *Computer Graphics: Principles and Practice*, Addison Wesley, 1997.
- Žára, Beneš, Sochor, Felkel: *Moderní počítačová grafika*. Computer Press, 2005
- B.G. Baumgart: *Winged-edge polyhedron representation*. Technical report STAN-CS-320, Stanford University, 1972.
- Martti Maentylae: *An introduction to solid modelling*. Computer Science Press, 1988. (→half-edge)