
Reprezentace 3D scény

© 1995-2001 Josef Pelikán
KSVI MFF UK Praha

e-mail: Josef.Pelikan@mff.cuni.cz

WWW: <http://cgg.ms.mff.cuni.cz/~pepca/>

Metody reprezentace 3D scén

◆ **objemové reprezentace**

- přímé informace o vnitřních objemech těles
- snadný test “**bod×těleso**” (leží daný bod uvnitř tělesa?), **zobrazování** může být obtížnější
- používají se též jako **pomocné datové struktury** pro rychlé vyhledávání

◆ **povrchové reprezentace**

- přímé informace o povrchu těles (hrany, stěny)
- obtížnější test “**bod×těleso**” (tělesa vůbec nemusí mít vnitřní objem), poměrně snadné **zobrazování**

Objemové reprezentace

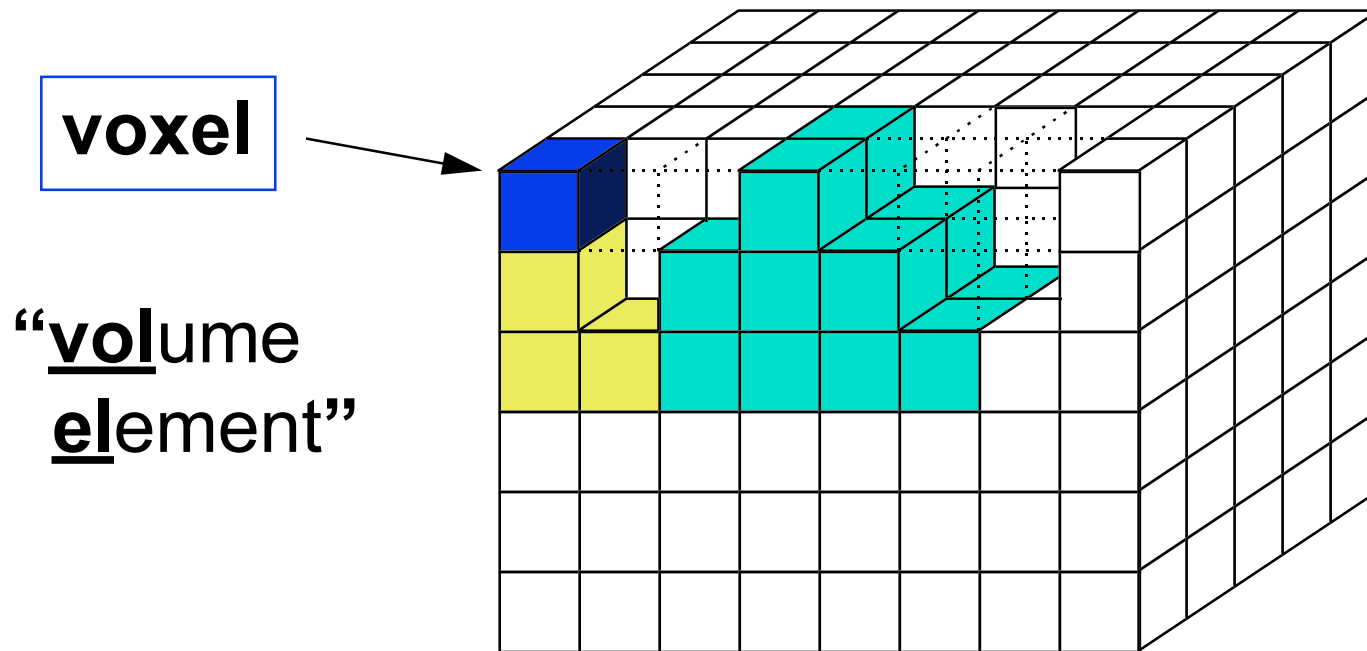
✓ výčtové reprezentace

- přímé vyčíslení obsazeného prostoru (diskrétní reprezentace - omezená přesnost)
- používají se hlavně jako pomocné datové struktury pro rychlé vyhledávání
- **buněčný model, oktantový strom**

✓ CSG reprezentace

- velice silná a přesná metoda (elementární tělesa, geometrické transformace, množinové operace)
- obtížnější **zobrazování** (vrhání paprsku)

Buněčný model



pole $k \times l \times m$ voxelů

jednobitová varianta: 0 - nic, 1 - těleso

vícebitová varianta: 0 - nic, $n > 0$ - těleso číslo n

Zobrazování buněčného modelu

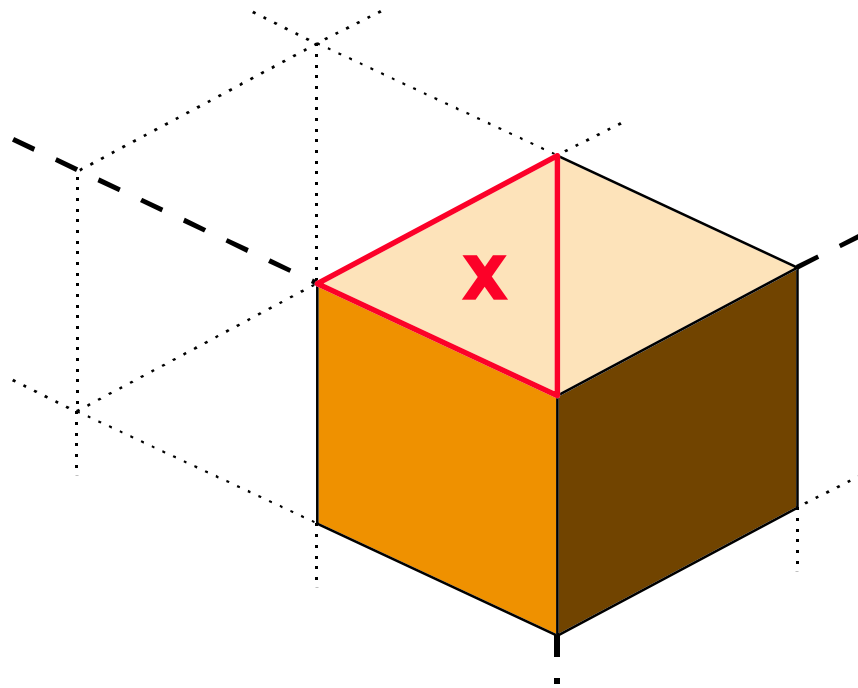
➔ kreslení odzadu-dopředu

- pouze přivrácené stěny voxelů
- pouze stěny na povrchu těles (stěny mezi **0** a **>0**)
- vícenásobné překreslování

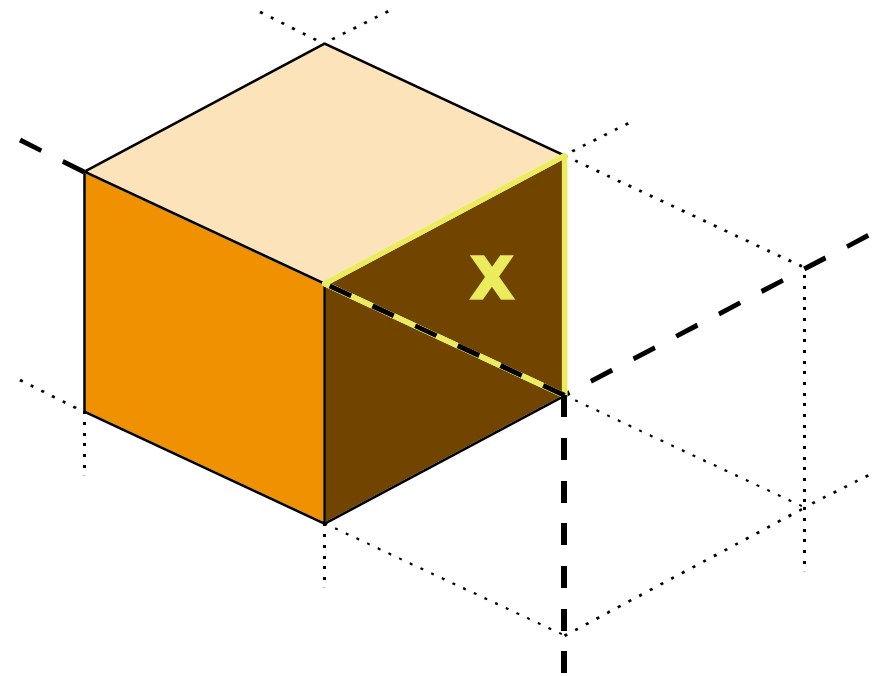
➔ speciální promítání

- velmi efektivní algoritmus bez zbytečného překreslování
- “**Ant-attack**” na ZX-Spectru (128×128×8 voxelů)

Speciální promítání

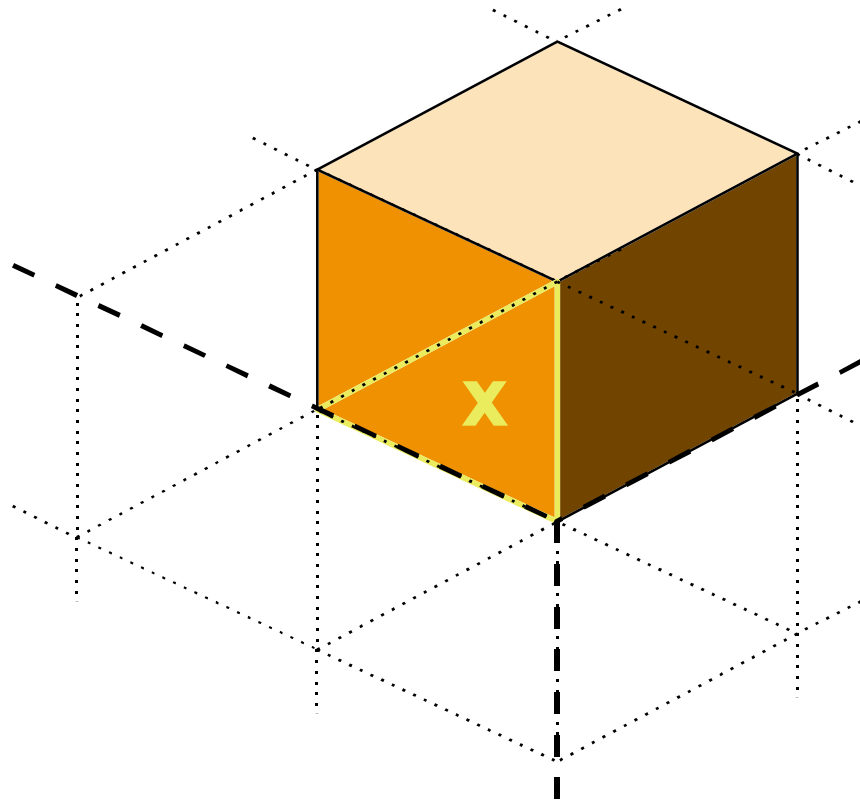


1. horní stěna
[0,0,0]

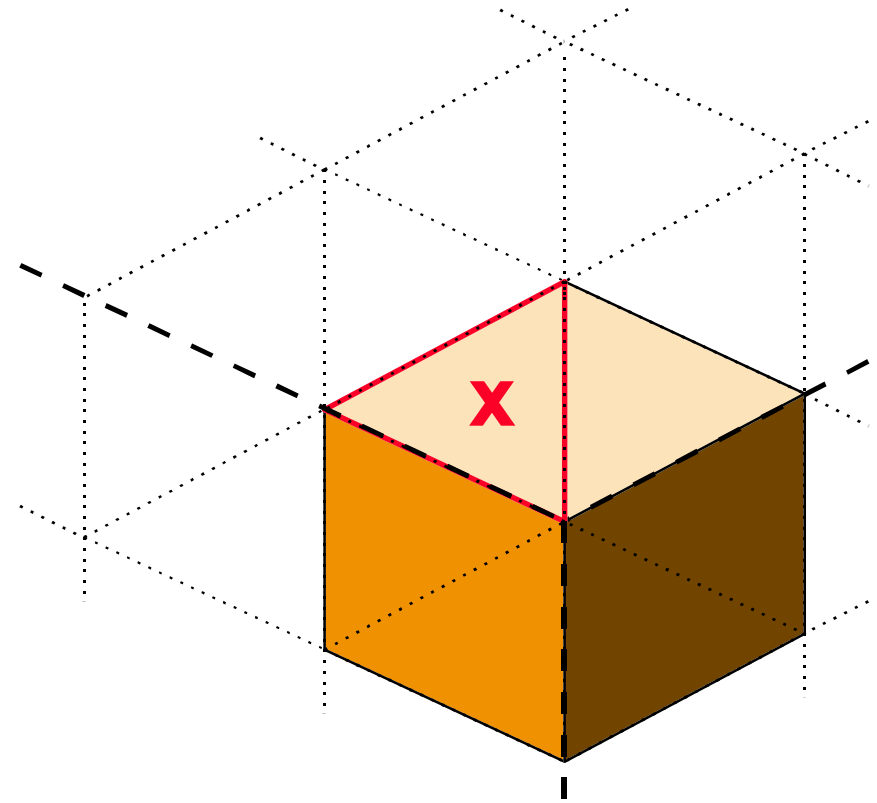


2. pravá stěna
[0,1,0]

Speciální promítání

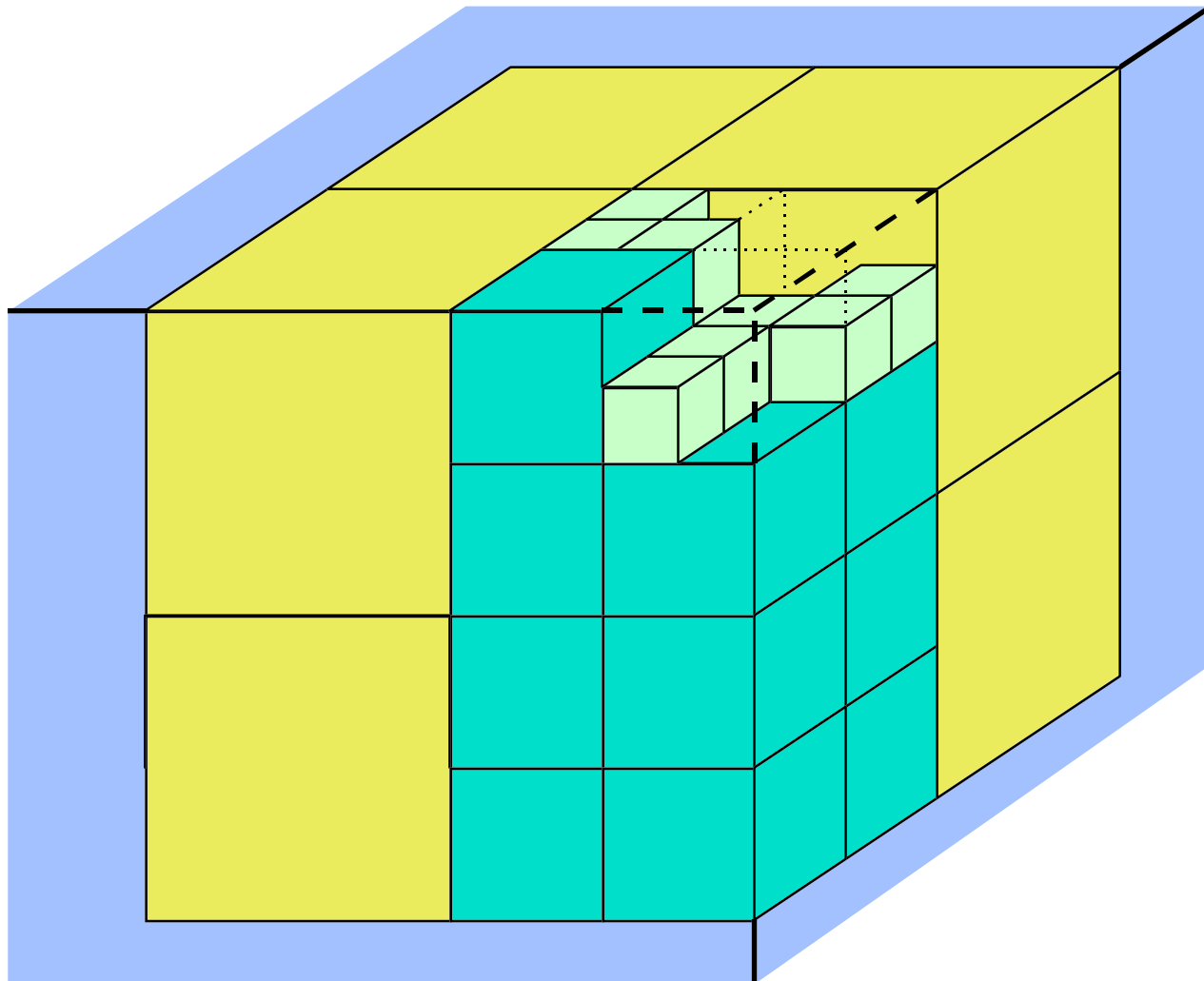


3. levá stěna
[1,1,0]



4. horní stěna
[1,1,1]

Oktantový strom (“octree”)



Oktantový strom

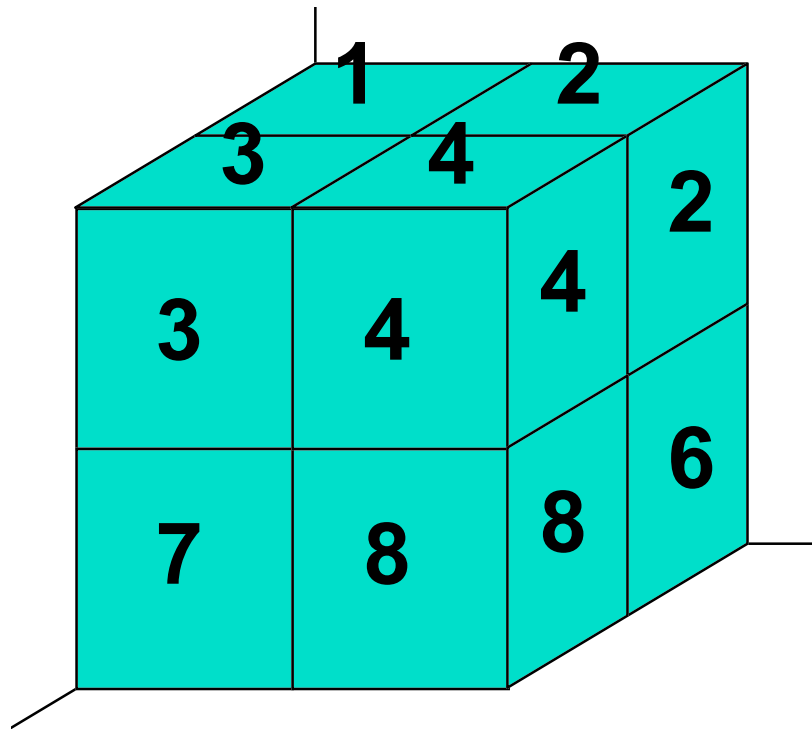
◆ 3D analogie kvadrantového stromu

- je-li vnitřek krychle nehomogenní, rozdělí se na osm částí (dělí se až do úrovně voxelu)
- úspora paměti proti buněčnému modelu

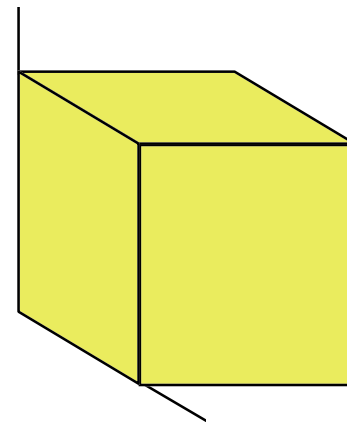
➔ kreslení odzadu-dopředu

- pouze přivrácené stěny krychlí
- pouze stěny na povrchu těles (stěny mezi **0** a **>0**)
- několikanásobné překreslování některých pixelů

Kreslení odzadu-dopředu



pořadí:
5-6-1-2-7-8-3-4



pořadí:
6-5-2-1-8-7-4-3

CSG (Constructive Solid Geometry)

◆ elementární geometrická tělesa

- snadno definovatelná a vyčíslitelná
- kvádr, poloprostor, hranol, koule, válec, kužel,...

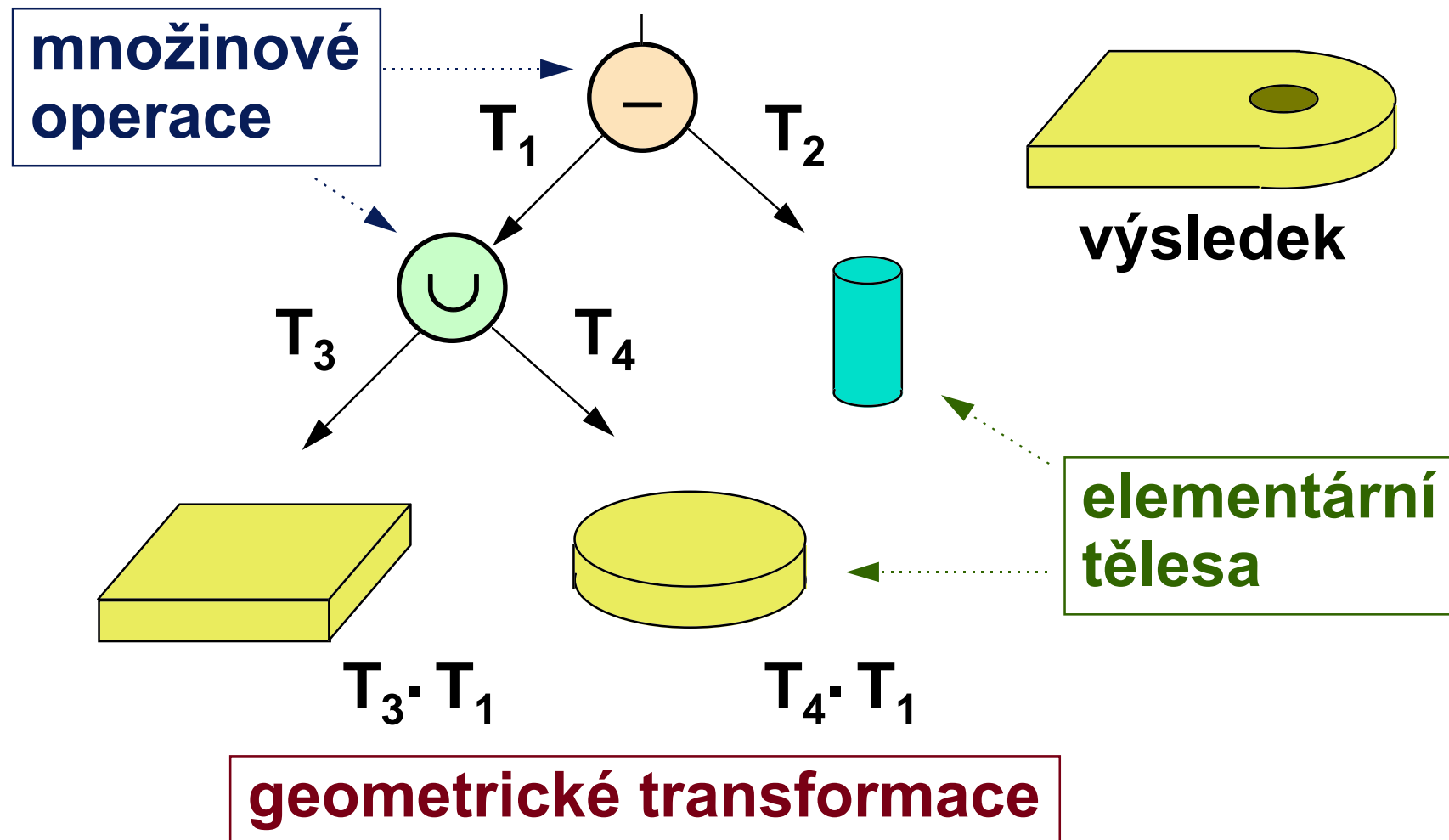
◆ množinové operace

- kompozice složitějších těles z elementárních
- sjednocení, průnik, rozdíl, ..

◆ geometrické transformace

- modifikace elementárních i složitějších těles
- (homogenní) maticové transformace

CSG strom



Transformace v CSG stromu

- ◆ **význam (sémantika) transformace T_i**
 - T_i mohou být uloženy v každé hraně CSG stromu
 - převod souřadnic ze soustavy podtělesa (podstromu, elementárního tělesa) do soustavy nadtělesa
 - “podtěleso transformuji pomocí T_i před tím, než ho přidám do nadtělesa”
- ➔ **snadná transformace libovolného podstromu**
 - změním pouze jednu matici
- ➔ **inverzní transformace T_i^{-1}**
 - pro vyčíslovací algoritmy (test bod×CSG, zobrazení)

Transformace v CSG stromu

→ uložení transformací jen v listech

- kumulované součiny (např. $T_3 \cdot T_2 \cdot T_1$ nebo inverzní $T_1^{-1} \cdot T_2^{-1} \cdot T_3^{-1}$)
- urychlení vyčíslovacích algoritmů (pro editaci je výhodnější distribuované uložení transformací)

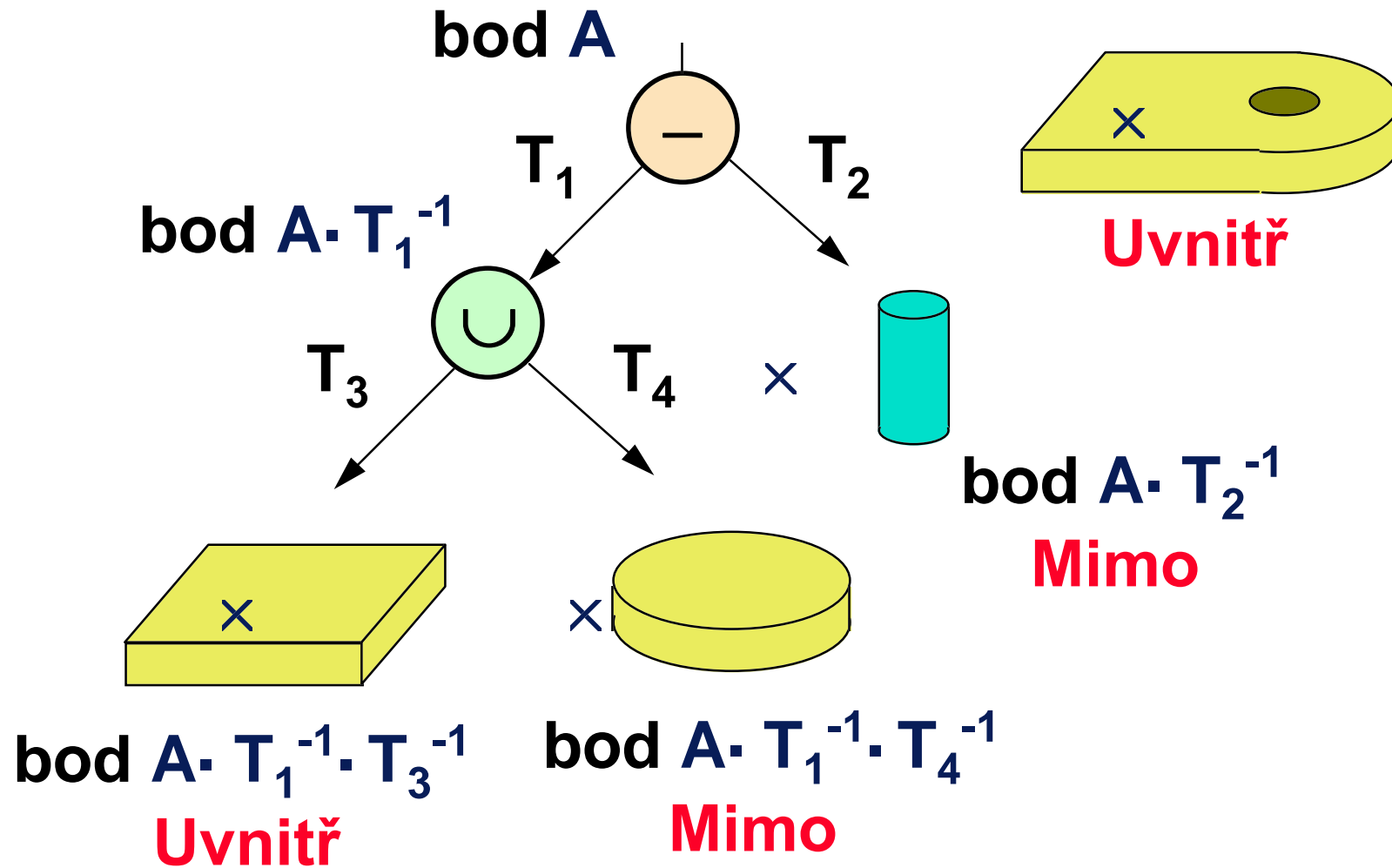
→ úsporné uložení elementárních těles

- tělesa jsou uložena v **normovaném tvaru**, všechny změny se provádí geometrickými transformacemi
- krychle (jednotková, vrchol v počátku), koule (jednotková, střed v počátku), válec (vodorovná podstava - jednotkový kruh, svislá osa, výška 1), ...

Test “bod×CSG strom”

- ◆ **leží daný bod A uvnitř tělesa?**
 - někdy chceme zjistit i podtělesa obsahující bod A
- ➔ **testy “bod×elementární těleso” jsou snadné!**
(především pro normované tvary těles)
- ➔ **průchod CSG stromem**
 - souřadnice bodu A se převádějí do souřadných soustav elementárních těles (inverzní transformace)
 - místo množinových operací se provádějí jejich **booleovské ekvivalenty** (\vee místo \cup , \wedge místo \cap , ...)

Test “bod×CSG strom”



Zobrazování CSG reprezentace

- ➔ **převedení do povrchové reprezentace**
 - pro každý druh **elementárního tělesa**: rutina převádějící těleso na **mnohostěn**
 - **množinové operace nad mnohostěny** (omezená přesnost - výsledek nemusí být správně ani v topologickém smyslu)
- ➔ **vrhání paprsku (“Ray-casting”)**
 - přesné zobrazování v **rastrovém prostředí** (pixelová přesnost)
 - výpočetně **náročnější metoda**

Povrchové reprezentace

✓ “drátový model”

- pseudo-povrchová reprezentace
- pouze **vrcholy** a **hrany** těles (nelze použít pro výpočet viditelnosti)

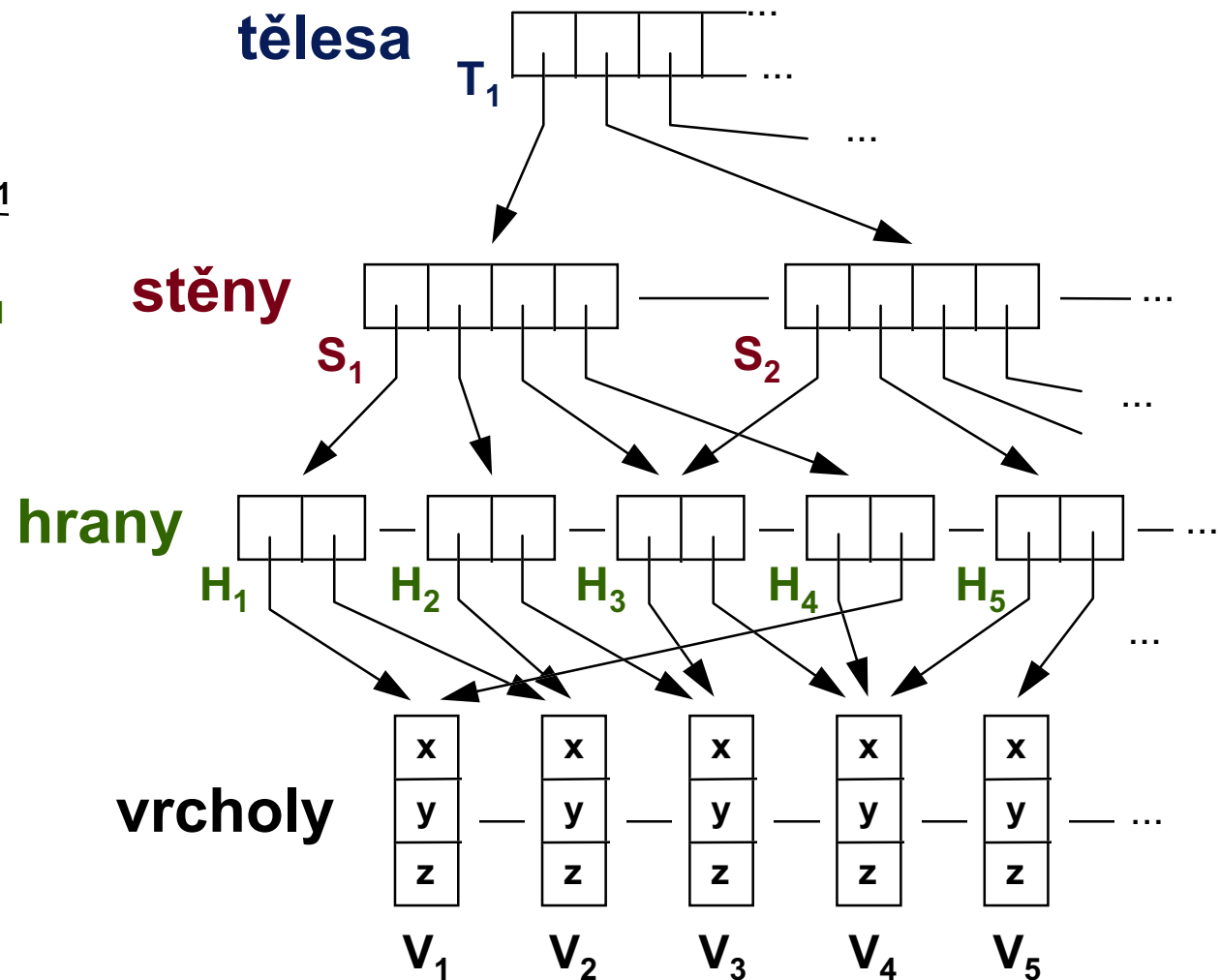
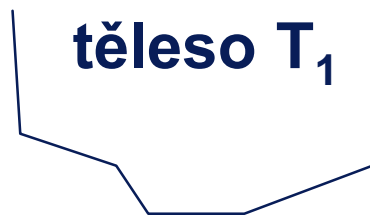
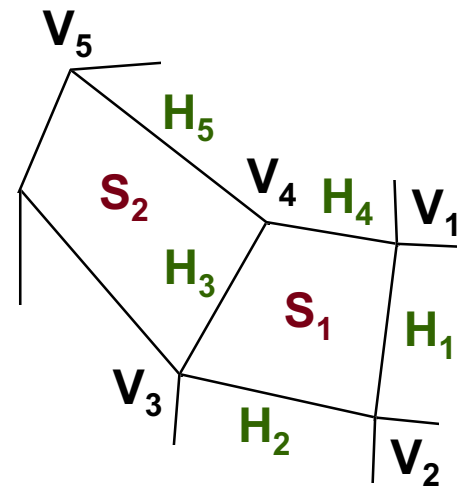
✓ VHS(T) reprezentace

- kompletní topologická informace: seznamy **vrcholů**, **hran**, **stěn** (a **těles**)

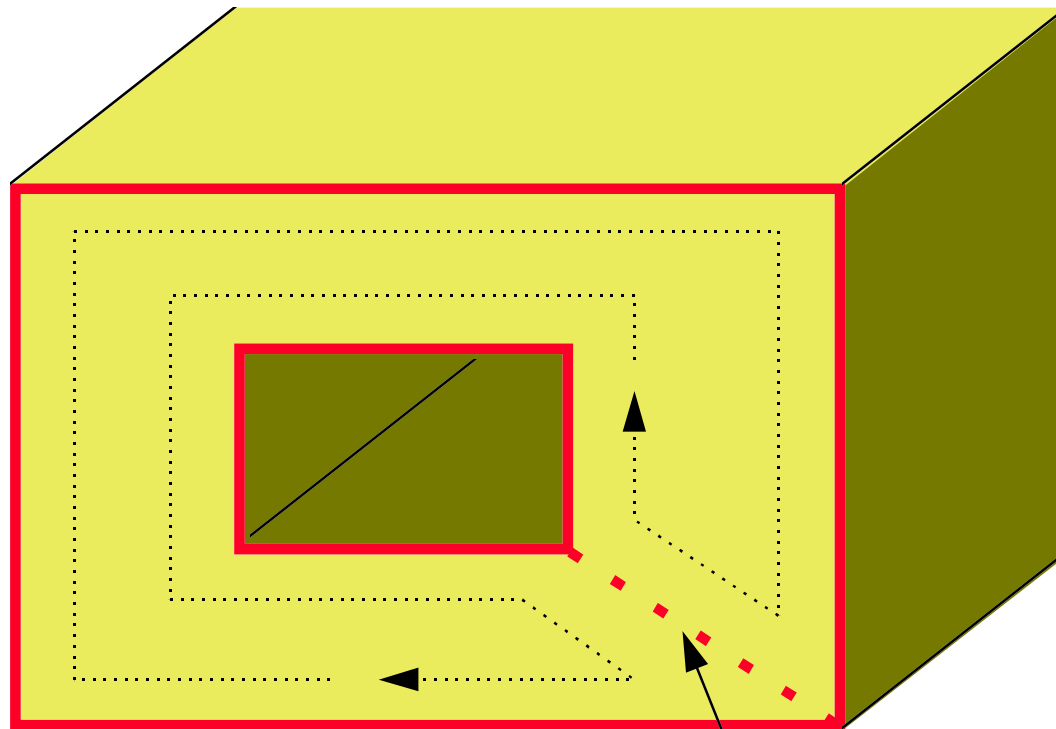
✓ “okřídlená hrana” (“winged-edge”)

- redundantní informace pro **rychlé vyhledávání** sousedních objektů (hrany incidentní s vrcholem, ..)

Povrchová reprezentace VHST



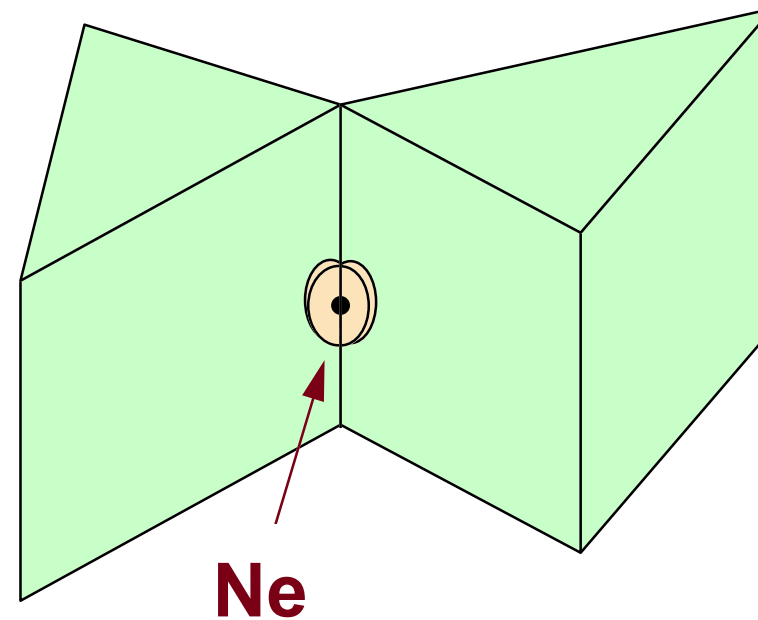
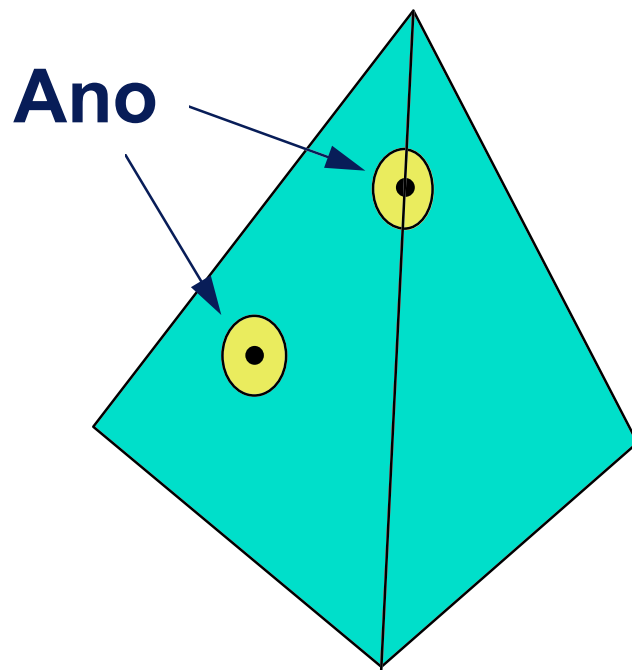
“Děravá” stěna



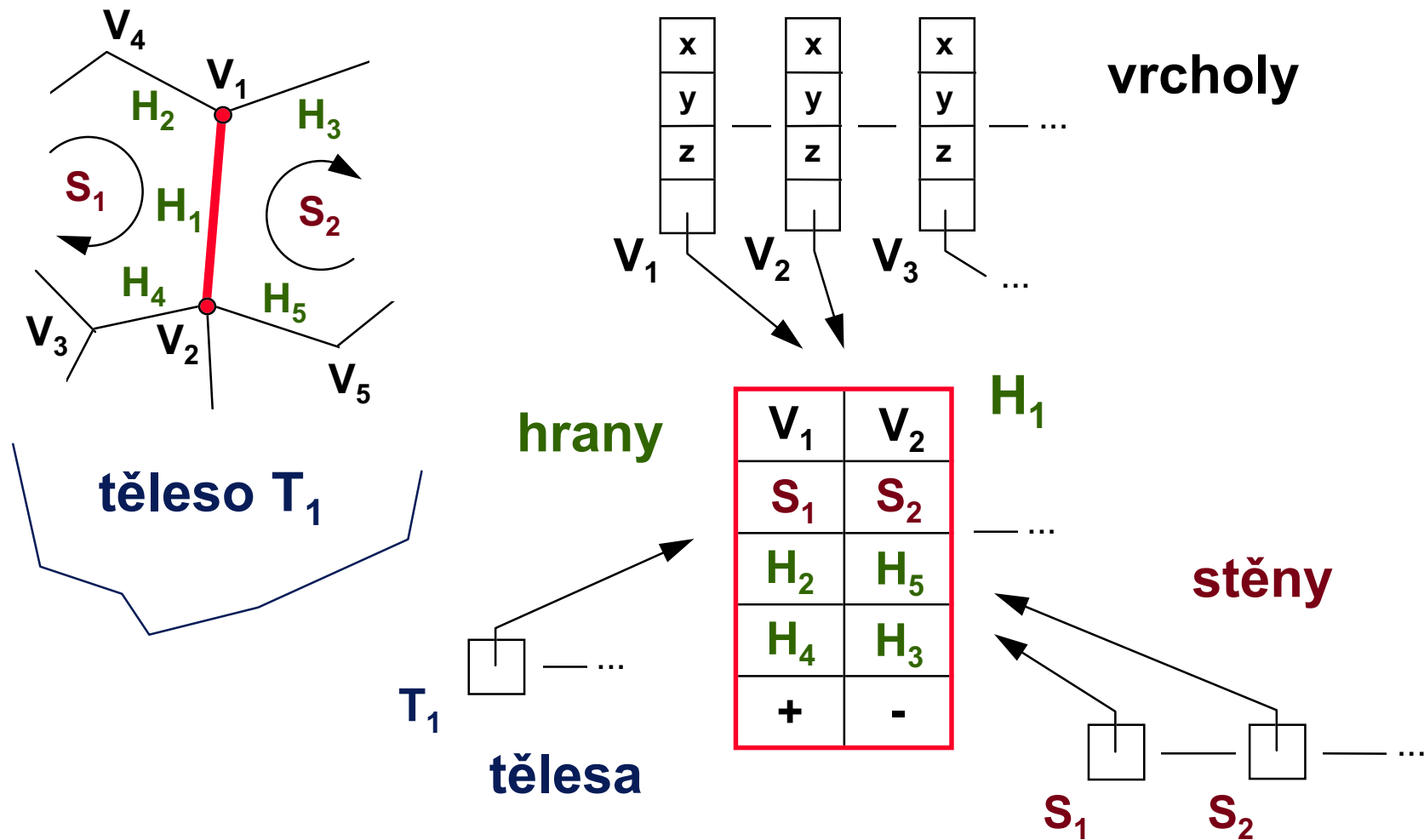
**umělá hrana
(nevykresluje se)**

“2-manifold” (manifold)

Def: Pro každý povrchový bod existuje okolí, které je topologicky ekvivalentní s rovinou



Okřídlená hrana (“winged-edge”)



Další informace v databázi

◆ **vrchol:**

- (normálový vektor pro spojitě stínování)

◆ **hrana:**

- příznak umělé hrany: pro reprezentaci děravých stěn nebo aproximaci křivých ploch sítí polygonů

◆ **stěna:**

- barva
- normálový vektor (stínování, přivrácená/odvrácená)

◆ **těleso:**

- barva

Eulerovy zákony

➔ pro **jednoduchý polyedr** (bez děr) platí vzorec

$$\mathbf{V - H + S = 2}$$

V - počet vrcholů, **H** - počet hran, **S** - počet stěn

➔ **zobecněný vzorec** (dovoluje díry):

$$\mathbf{V - H + S - D = 2 \cdot (T - G)}$$

D - počet děr ve stěnách, **T** - počet těles, **G** - počet děr procházejících celým tělesem (Genus)

Eulerovy operátory

➔ konstrukce 2-manifoldsu po krocích

- v každém kroku je zajištěna platnost Eulerových vzorců (těleso je topologicky korektní)
- ke každému operátoru existuje inverzní (snadná implementace příkazu “undo”)

➔ příklady Eulerových operátorů:

Msfevv(P_1, P_2): “make solid, face, edge, vertex, vert.”,

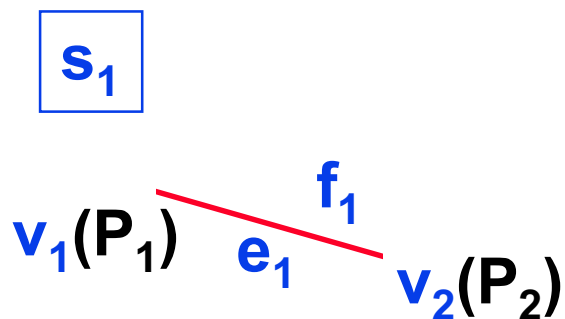
Mev(f_1, v_1, P_2): “make edge, vertex”,

Mef(f_1, v_1, v_2): “make edge, face”,

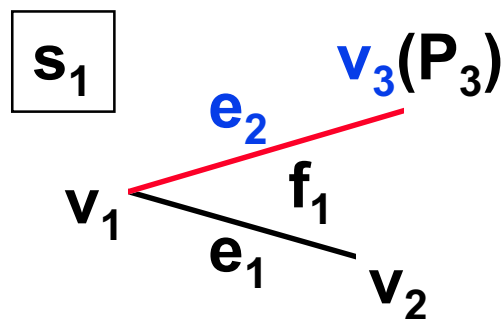
Kef(e): “kill edge, face”, ...

Konstrukce čtyřstěnu

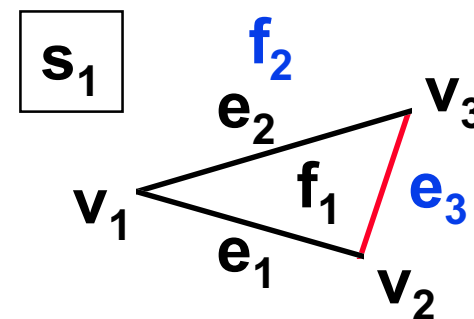
1. Msfevv(P_1, P_2)



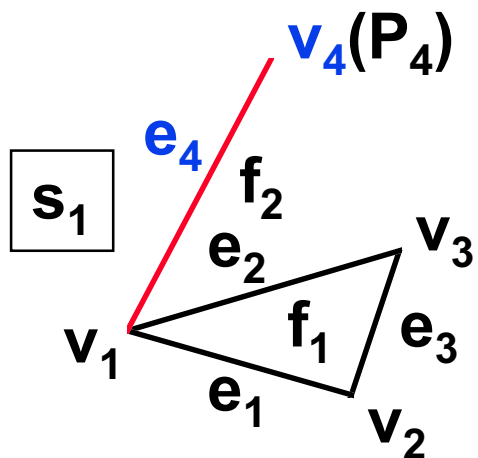
2. Mev(f_1, v_1, P_3)



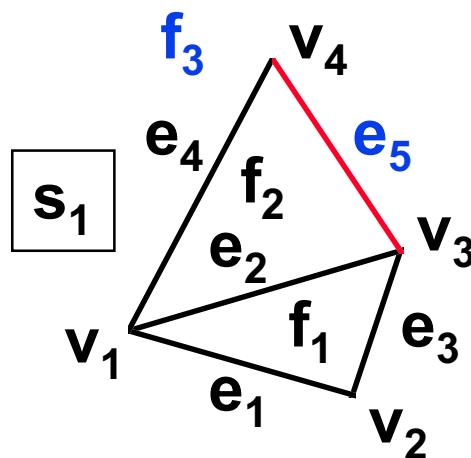
3. Mef(f_1, v_2, v_3)



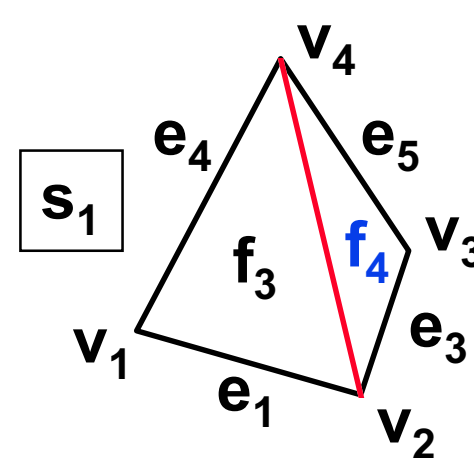
4. Mev(f_2, v_1, P_4)



5. Mef(f_2, v_3, v_4)



6. Mef(f_3, v_2, v_4)



Konec

Další informace:

- **J. Foley, A. van Dam, S. Feiner, J. Hughes:** *Computer Graphics, Principles and Practice*, 534-562, 712-714
- **Jiří Žára a kol.:** *Počítačová grafika*, principy a algoritmy, 234-238
- ➔ **LAN na Malé Straně:**
 - **barbora\usr:\vyuka\pelikan\6**