

# Vizualizace objemových dat

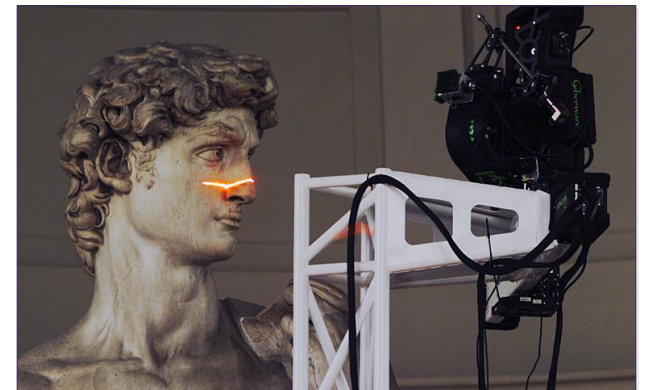
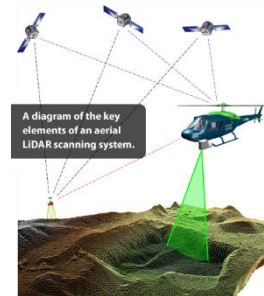
JOSEF KOHOUT  
BESOFT@KIV.ZCU.CZ

# Objemová data

- ▶ Množina vzorků  $(x, y, z, v)$ 
  - ▶  $(x, y, z)$  = bod v  $E^3$ , ve kterém je hodnota  $v$
  - ▶  $v$  = skalární, vektorová, příp. tenzorová hodnota

# Objemová data

- ▶ Pocházejí z fyzikálních měření a simulací
- ▶ Několik málo příkladů:
  - ▶ LIDAR – skenování zemského povrchu
  - ▶ 3D skenování historických památek (resp. továren v USA)
  - ▶ Lékařská vyšetření počítačovou tomografií (CT, MRI, ...)
  - ▶ Ultrazvuková vyšetření (např. "sono" jater)



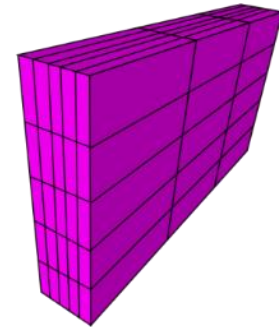
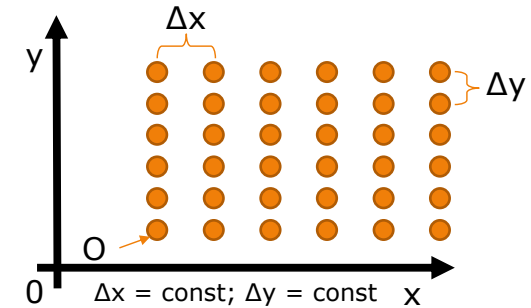
# Objemová data

- ▶ Měření směru proudění vzduchu v aerodynamickém tunelu
- ▶ Měření teploty vzduchu v různých místech
- ▶ Simulace proudění krve v cévách
- ▶ Simulace zemětřesení
- ▶ Simulace zatížení kostí během pohybu
- ▶ Simulace vyhladovění nádorových buněk
- ▶ Simulace štěpení proteinů



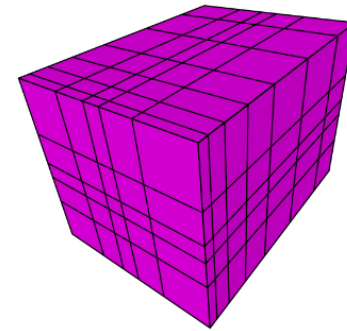
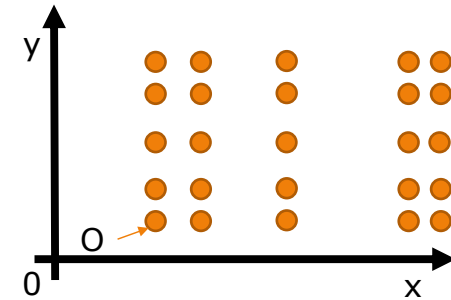
# Objemová data

- ▶ Body  $(x, y, z)$  mohou být rozmístěny:
  - ▶ Nestrukturovaně (roztrošeně)
  - ▶ Strukturovaně v mřížkách
- ▶ Pravidelná mřížka (regular grid)
  - ▶ Postačuje uložit:
    - ▶ Souřadnice prvního bodu (O)
      - ▶ Často lze ignorovat, pokud absolutní poloha není nedůležitá
    - ▶ Vzdálenost mezi body na jednotlivých osách ( $\Delta x, \Delta y, \Delta z$ )
  - ▶ Polohu ostatních bodů lze vypočítat

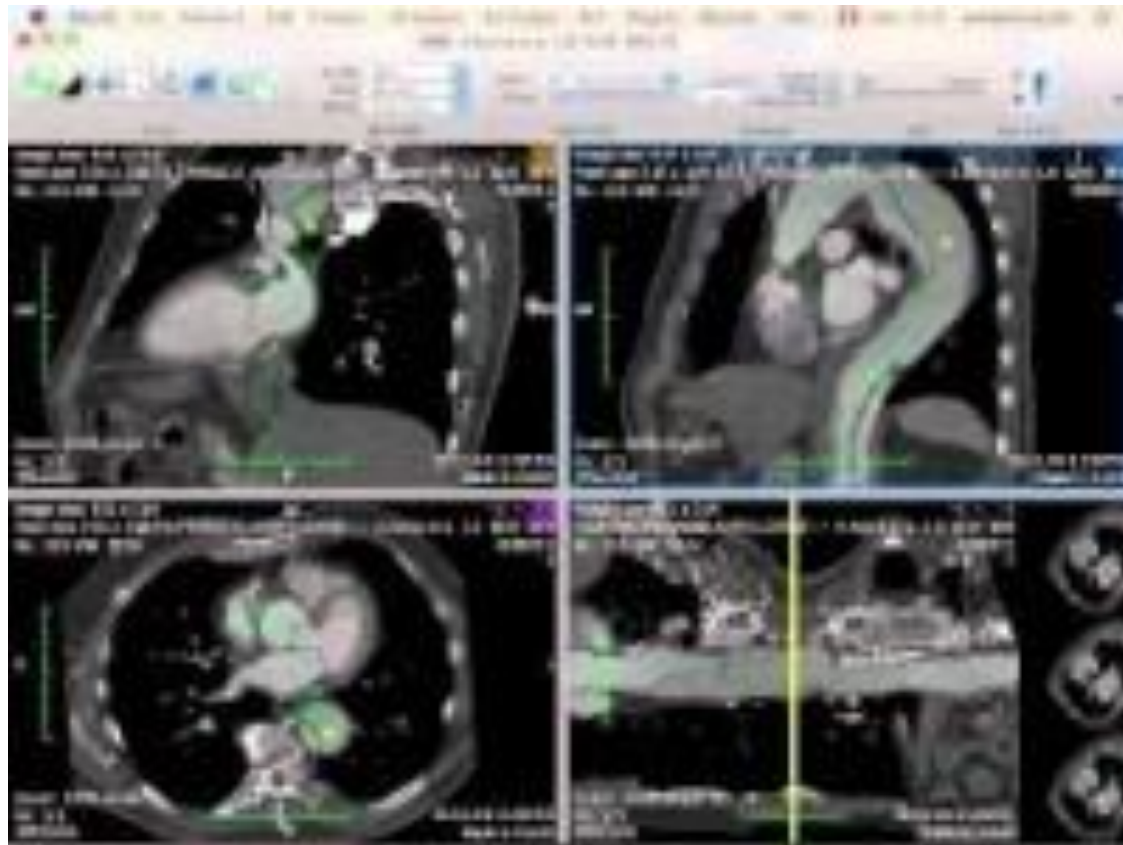


# Objemová data

- ▶ Pravoúhlá mřížka (rectilinear grid)
  - ▶ Postačuje uložit:
    - ▶ Souřadnice na jednotlivých osách
      - ▶ Např. pro objemová data ve 3D se jedná o 3 pole
  - ▶ Polohu ostatních bodů lze vypočítat



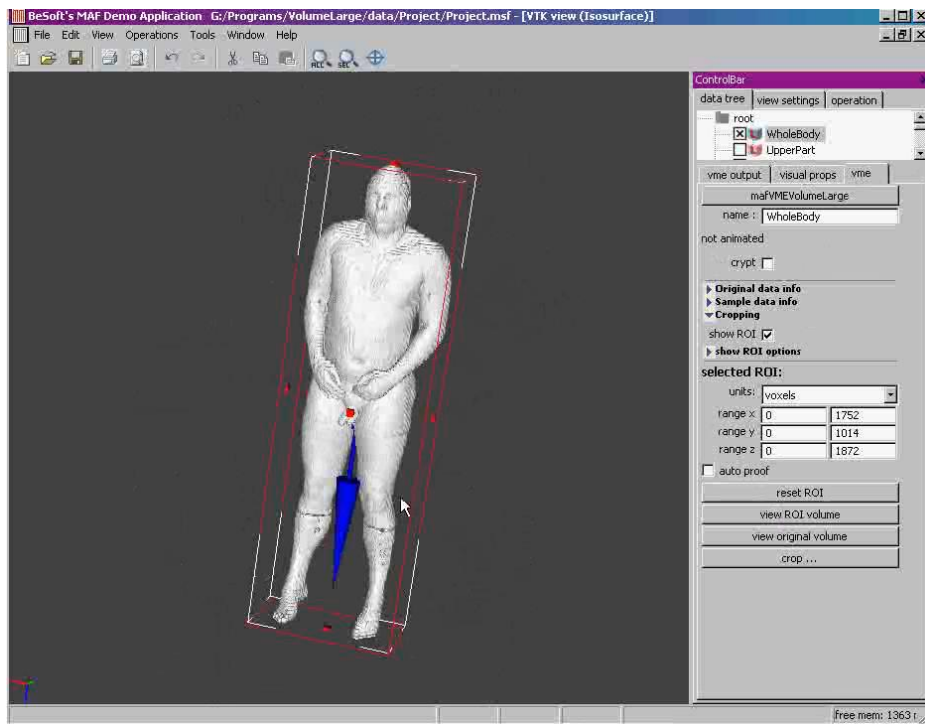
# Ukázky



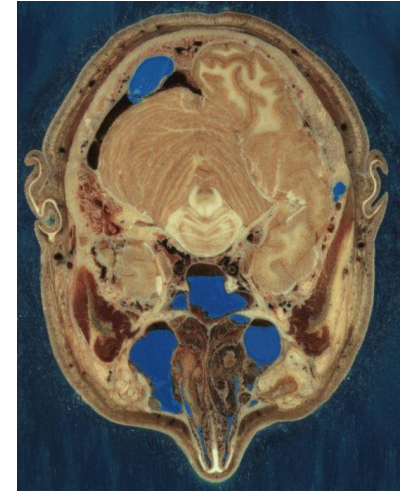
OsiriX



# Ukázky

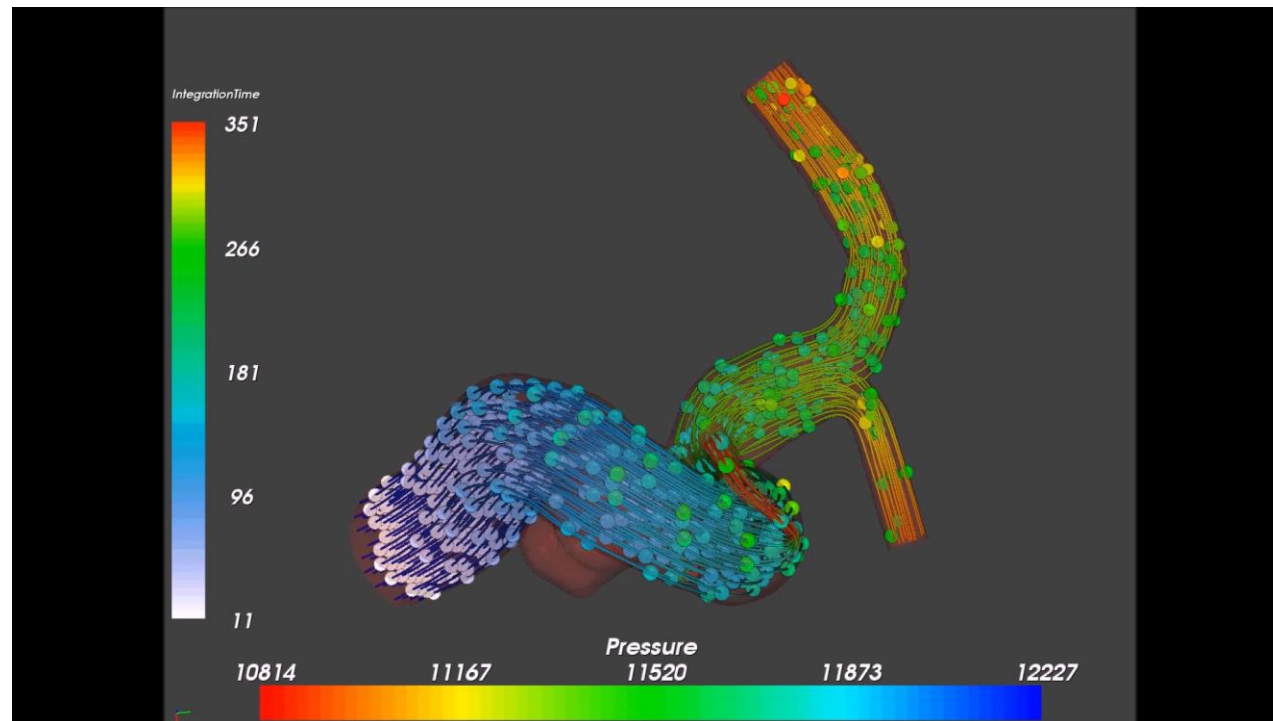


Agrawal, A., Kohout, J., Clapworthy, G.J. *et al.* Enabling the interactive display of large medical volume datasets by multiresolution bricking. *J Supercomput* **51**, 3–19 (2010). <https://doi.org/10.1007/s11227-009-0289-2>





# Ukázky

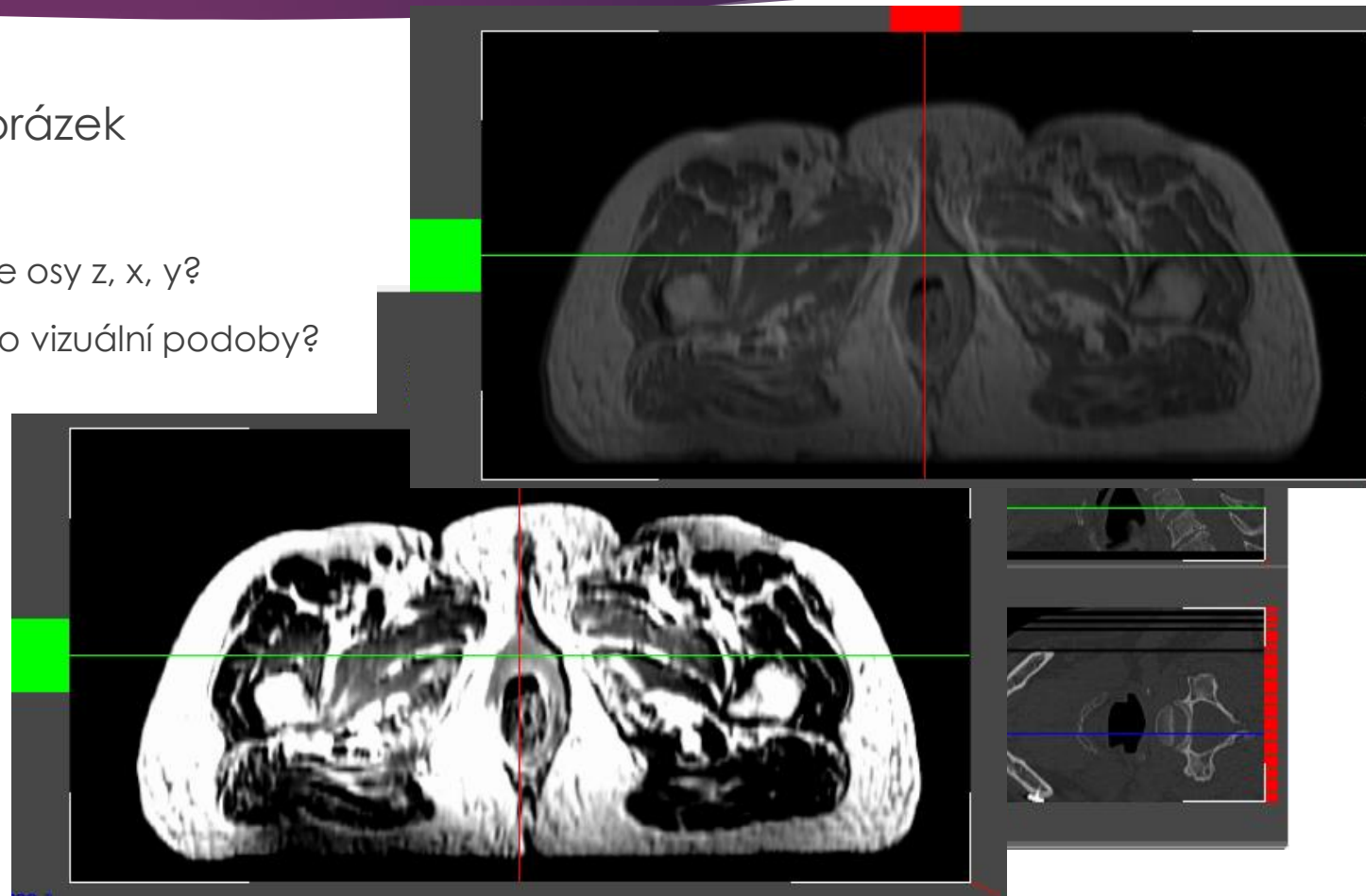


# Vizualizace skalárních polí

- ▶ Základní vizualizační přístupy:
  - ▶ Multiplanar reconstruction (MPR)
    - ▶ Vizualizace po jednotlivých „řezech“
  - ▶ Direct Volume Rendering
    - ▶ Přímé metody vizualizace
  - ▶ Vizualizace povrchů
    - ▶ Nepřímé metody vizualizace

# Multiplanar reconstruction

- ▶ Lze snadno zobrazit jen jeden obrázek
  - ▶ Diskuze:
    - ▶ Jak vytvořit obraz ze směru podle osy z, x, y?
    - ▶ Jak převést hodnoty v datech do vizuální podoby?
      - ▶ Pozor: důležité věci musí být jasně viditelné
    - ▶ Co je důležité?



# Převod hodnot na barvu

- ▶ Pro převod hodnot z dat na barvu (RGB), resp. průhlednost (A) slouží tzv. přenosová funkce (transfer function)
  - ▶ Volba funkce aplikačně závislá



# Přenosová funkce: 1D

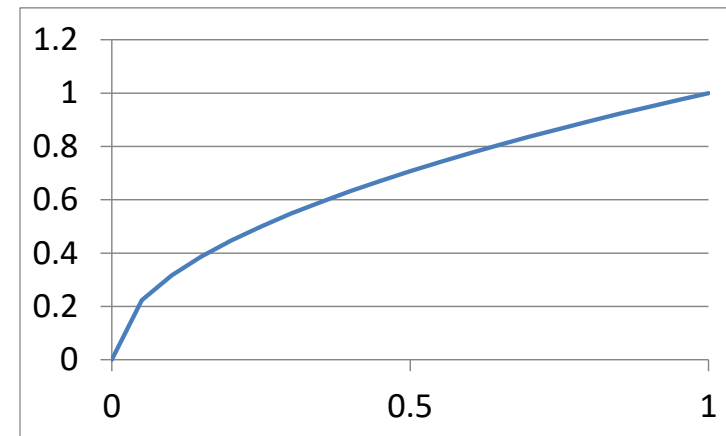
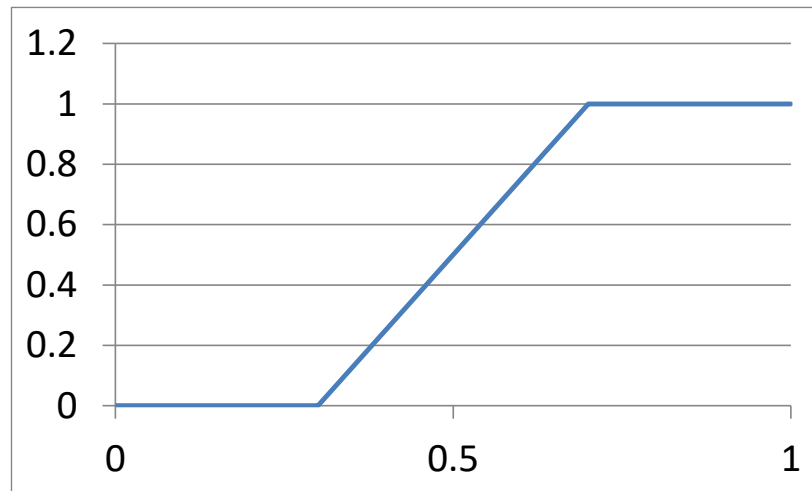
- ▶ Typicky se jedná o složenou funkci
- ▶  $f_{RGBA}(h(x, y, z)) = (f_R, f_G, f_B, f_A)$ 
  - ▶  $f_R(h(x, y, z)) = \varphi_R(\psi(h(x, y, z)))$ ,
  - ▶  $f_G(h(x, y, z)) = \varphi_G(\psi(h(x, y, z)))$ ,
  - ▶  $f_B(h(x, y, z)) = \varphi_B(\psi(h(x, y, z)))$

# Přenosová funkce: 1D

- ▶ Přenosová funkce  $\psi$  řídí výsledný kontrast
  - ▶  $\psi(h) > h \rightarrow$  zesvětlení místa
  - ▶  $\psi(h) < h \rightarrow$  ztmavení místa
  - ▶ Slouží k potlačení šumu v nějaké oblasti
    - ▶ Např. namísto černo-černého šumu budu mít jen černou
    - ▶ Vede k zvýšení kontrastu (mám méně různých hodnot)

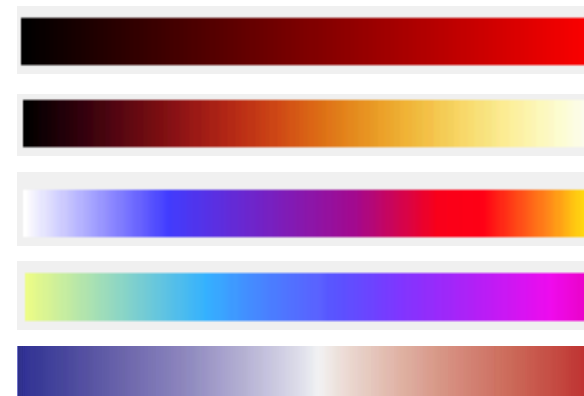
# Přenosová funkce : 1D

- ▶ Nejčastější tvar přenosové funkce  $\psi$  je "S" nebo "log"



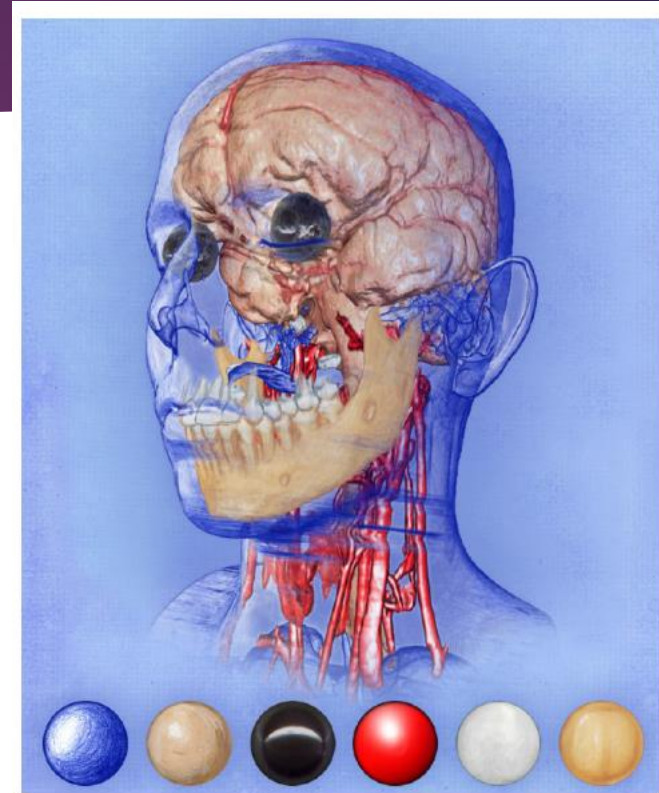
# Přenosová funkce : 1D

- ▶ Přenosové funkce  $\varphi_{R,G,B}$  slouží k obarvení
  - ▶ Volba barevné škály je typicky aplikačně závislá
    - ▶ Duhové škály nejsou pro typicky vhodné
      - ▶ **Diskuze:** proč?
  - ▶ Často se jedná po částech lomenou funkci
    - ▶ Pro klíčové hodnoty se stanoví RGB barva ostatní interpolací
    - ▶ Lineární interpolace nebo interpolace nejbližším sousedem



# Přenosová funkce : 2D

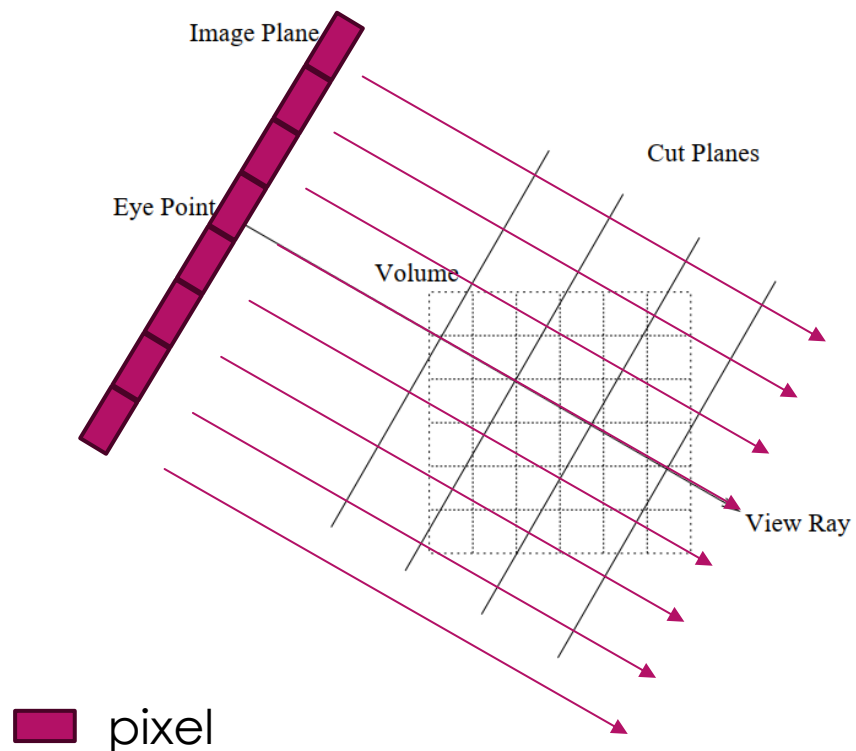
- ▶ Přidána dodatečná dimenze
- ▶ Nejčastěji:
  - ▶  $f_{RGBA}(h(x, y, z), \nabla(x, y, z))$ 
    - ▶ umožňuje lepší percepci hloubky
  - ▶  $f_{RGBA}(h(x, y, z), id(x, y, z))$ 
    - ▶ id = identifikátor segmentu
    - ▶ vyžaduje segmentovaná data



**Figure 11:** Segmented volume rendered with a MD style TF based on data value and object membership. The base of the image depicts the lit spheres for the different styles. Image courtesy of Bruckner et al. [BG07].

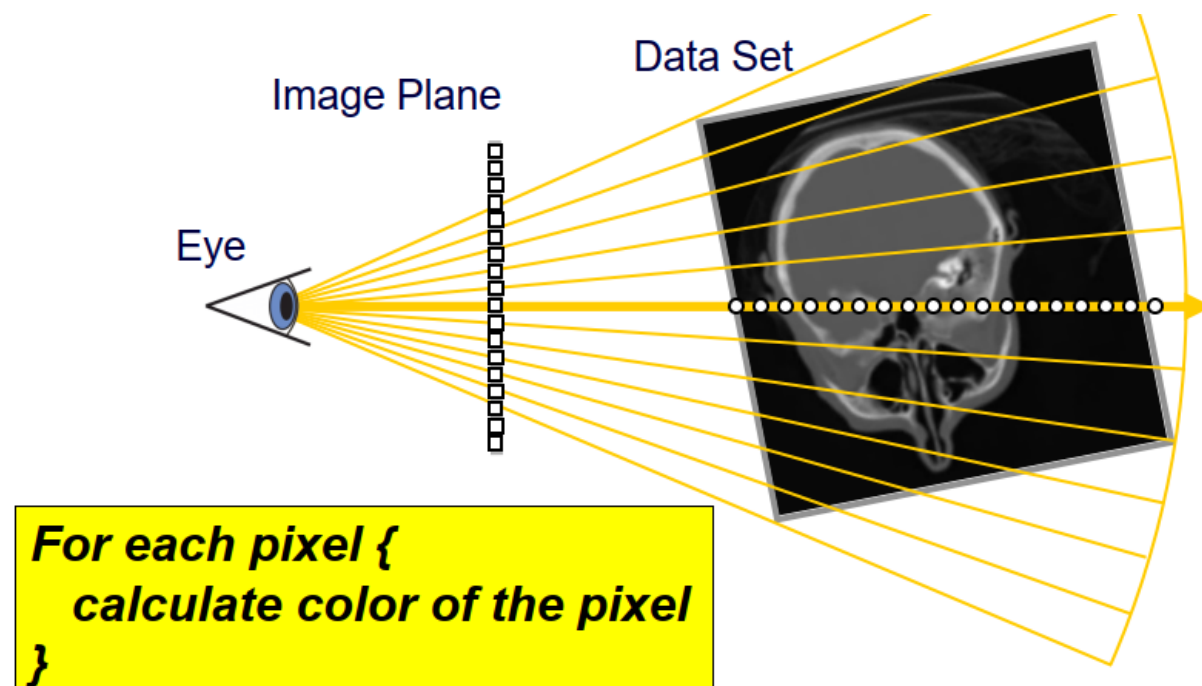
# Multiplanar reconstruction

- ▶ Nejsložitější je obecný řez
- ▶ **Diskuze:**
  - ▶ Jaké problémy přináší?
  - ▶ Jak řešit?



# Přímé metody vizualizace

- ▶ Výsledná hodnota pixelu dána funkcí hodnot voxelů protnutých paprskem poslaným do scény skrz pixel



# Přímé metody vizualizace

- ▶ Různé funkce pro dosažení různého vzhledu
- ▶ Nejjednodušší = Maximum Intensity Projection (MIP)
  - ▶  $I = \max \mu_i$ 
    - ▶  $\mu_i$  = hodnota i-tého protnutého voxelu
  - ▶ Vhodné např. pro CT
    - ▶ Kostí mají nejvyšší hodnoty

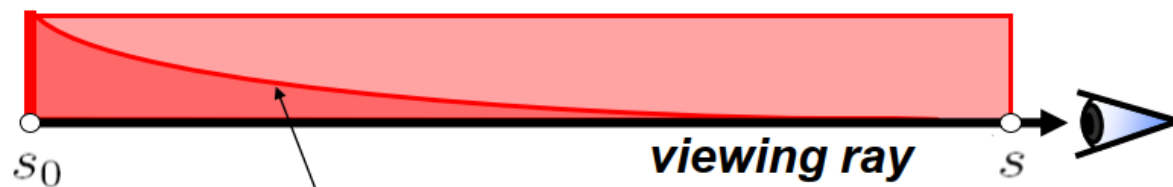
# Přímé metody vizualizace

- ▶ Digitally Reconstructed Radiography (DRR)

- ▶  $I = I_0 \cdot e^{-\sum \mu_i \cdot x_i}$

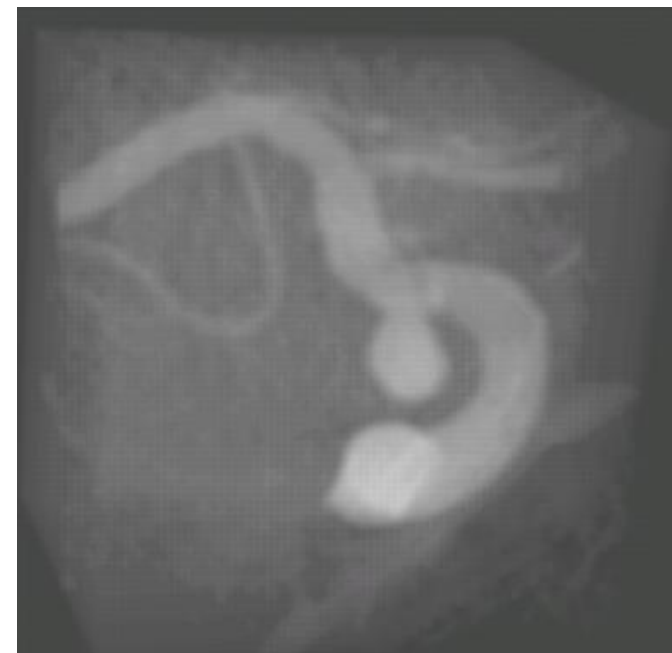
- ▶  $x_i$  = délka úseku paprsku v i-tém voxelu

- ▶ Výpočetně náročné → aproximace



$$I_0 = I(S_0)$$

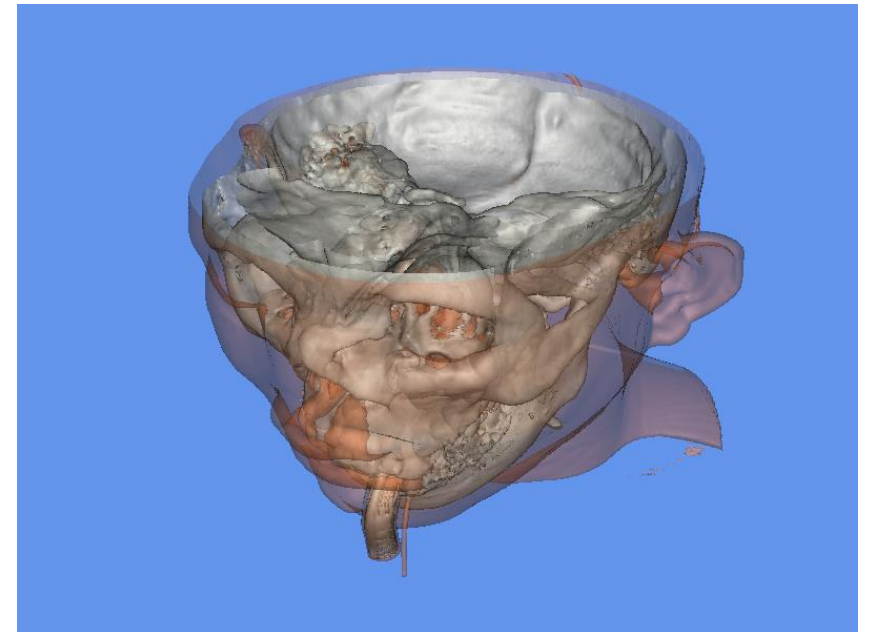
útlum podél paprsku z důvodu absorpce záření materiálem



# Přímé metody vizualizace

- ▶ Sofistikovanější metody hledají povrch podél paprsku
  - ▶ Nutno specifikovat iso-hodnotu, tj. skalární hodnotu v datech určující, kudy povrch prochází
  - ▶ **Diskuze:**
    - ▶ Jak volit iso-hodnotu?
    - ▶ Lze automatizovat?
    - ▶ Jedna hodnota nebo více?

vtkOpenGLGPUVolumeRayCastMapper  
(ISO hodnoty: 500, 1150, průhlednost: 0.3, 0.6)



# Přímé metody vizualizace

- ▶ V případě CT jsou hodnoty v datech typicky normalizovány na škálu dle Hounsfielda
  - ▶ Pro MRI nic takového neexistuje!!

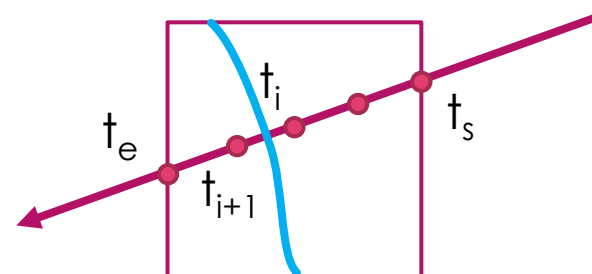
Tkáň	Hounsfield units (HU)
Air	-1000
Fat	-100
Water	0
Blood	30-45
Muscle	40
White matter	20-30
Gray matter	37-45
CSF (Cerebrospinal fluid)	15
Bone	> 150

# Přímé metody vizualizace

- ▶ Otsuova metoda
  - ▶ izohodnota volena z histogramu hodnot skalárního pole
  - ▶ izohodnota rozdělí histogram na dvě části
  - ▶ volba izohodnoty tak aby:
    - ▶ minimalizovala rozptyly uvnitř oblastí
    - ▶ maximalizovala rozptyl mezi skupinami
- ▶ Globální izohodnota nemusí být dostatečná → segmentace
  - ▶ Obecnější než izoplochy (neříká, jak oddělit)
  - ▶ Vhodné, pokud jsou vstupní data problematická (šum)

# Přímé metody vizualizace

- ▶ Problém: povrch neprochází pouze středem voxelu
- ▶ Buňka = kvádr s vrcholy ve středech sousedních voxelů, tj. ve vrcholech má hodnoty z dat
- ▶ Je třeba identifikovat „buňky“, skrz které prochází hledaný povrch
  - ▶ **Diskuze:** jak?
- ▶ Hadwiger et al., 2005:
  1. Postupuj od  $t_s$  k  $t_e$ , dokud  $h(t_i) < h_{iso}$ 
    - ▶  $h(t_i)$  dáno typicky tri-lineární interpolací
  2. Spočti polohu povrchu na úseku  $t_i, t_{i+1}$ 
    - ▶ lineární interpolace
    - ▶ výsledkem bod  $\mathbf{x}$



# Přímé metody vizualizace

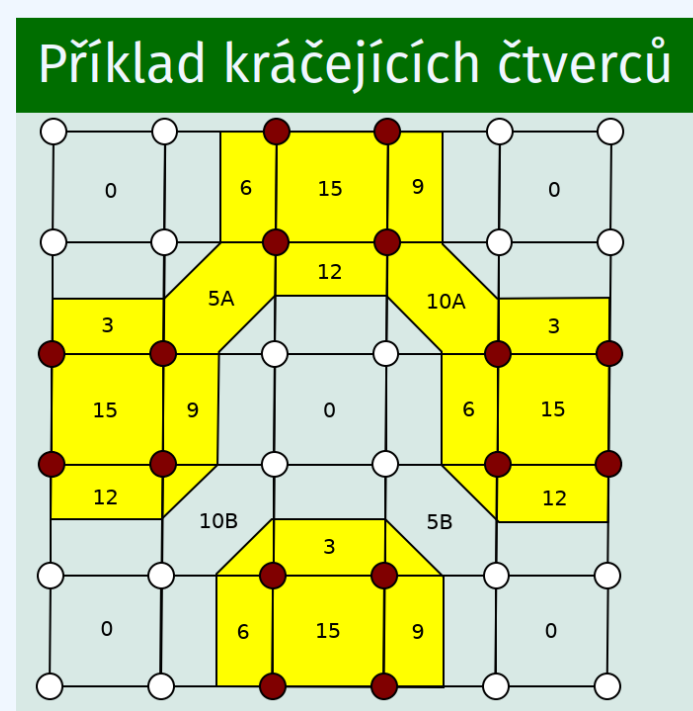
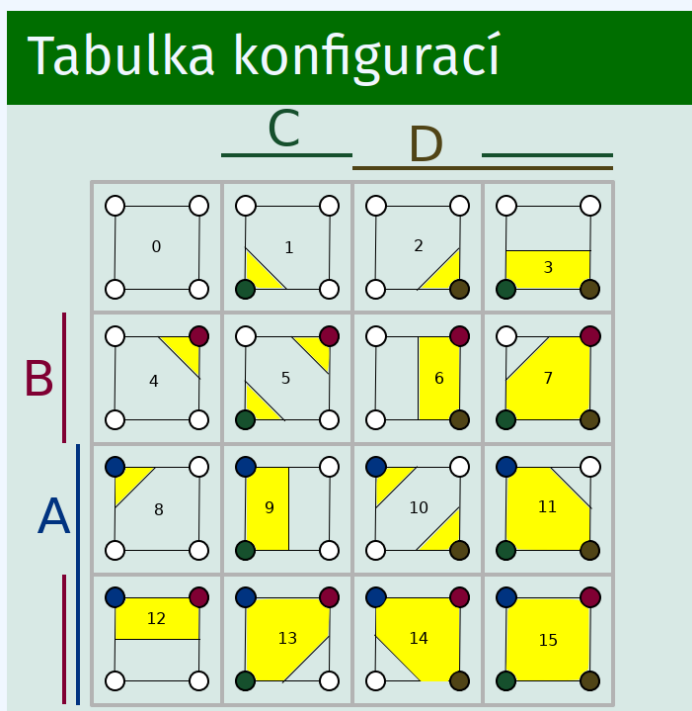
3. Spočti normálu v bodě  $\mathbf{x}$ 
  - ▶ nejjednodušší: tri-lineární interpolace gradientů skalárního pole ve vrcholech buněk
    - ▶ **Diskuze**: jak určit gradienty?
  - ▶ Hadwiger et al., 2005: tri-kubická interpolace
4. Použij např. Blinn-Phong osvětlovací model pro určení výsledné hodnoty pixelu
  - ▶ Poznámka: pro urychlení se typicky využívá GPU

# Nepřímé metody vizualizace

- ▶ Nejrozumnější metody extrakce iso-ploch
  - ▶ V praxi se používá nejčastěji Marching-Cubes
- ▶ Převádí objemová data na povrchová
- ▶ Povrchová data zobrazena standardním přístupem

# Marching squares

- Postupuje buňku po buňce a ohodnocuje, zda hodnota ve vrcholech je nad nebo pod zvolenou izohodnotou



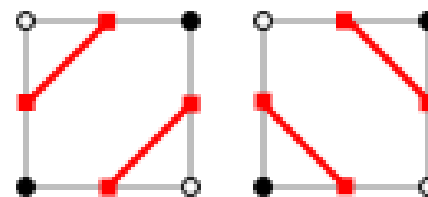
Zjednodušený  
příklad bez lineární  
interpolace  
(izočáry prochází  
středem hran buněk)

# Marching squares

- Problém: Nejednoznačnost dvou konfigurací (0101/5 a 1010/10)

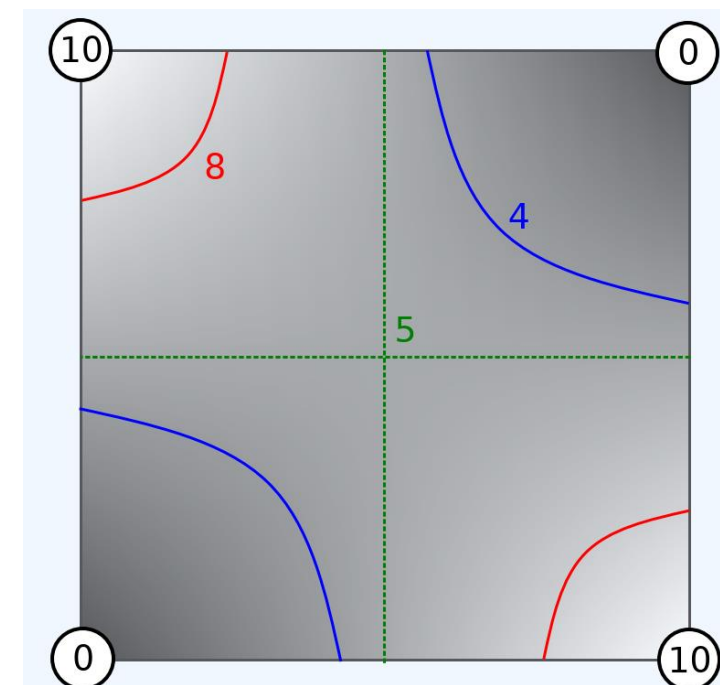
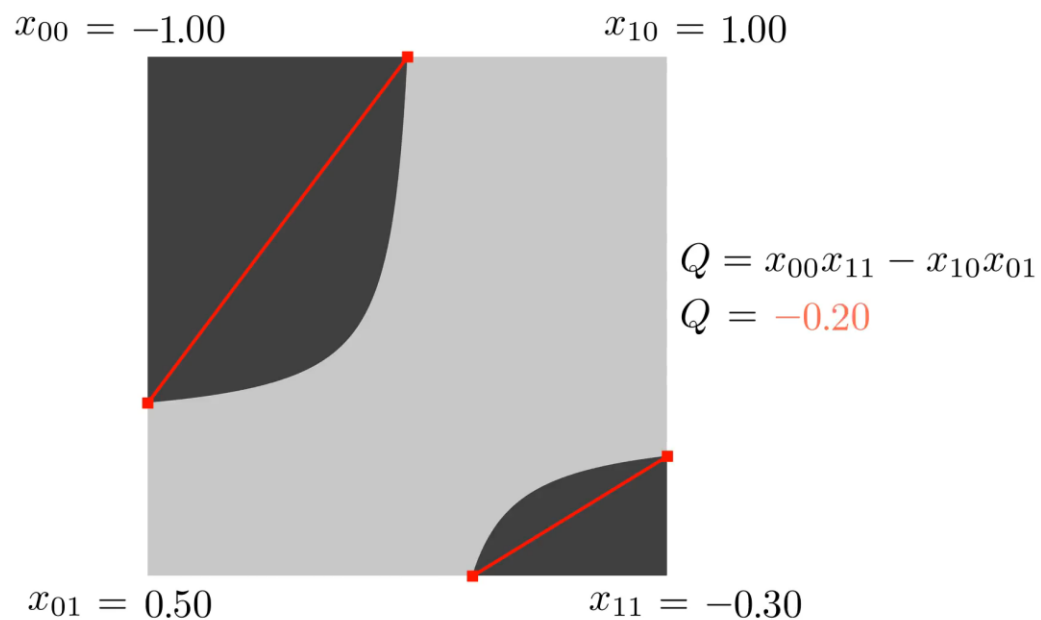
Tabulka konfigurací

		C		D	
		0	1	2	3
B		4	5	6	7
	A	8	9	10	11
		12	13	14	15



# Marching squares

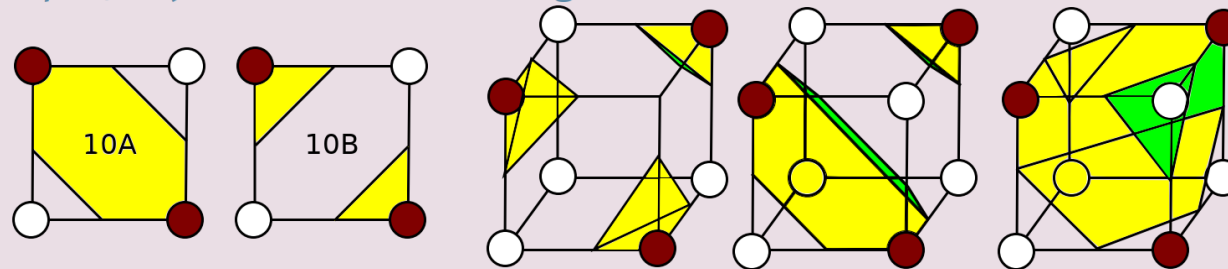
- ▶ Řešení: asymptotický rozhodovač (Nielson & Hamann)
  - ▶ Založeno na faktu, že známe hodnoty v buňkách, nejen znaménko
  - ▶ → Rozhodování pomocí bilineárního interpolantu



# Marching cubes

- ▶ Analogie k Marching squares

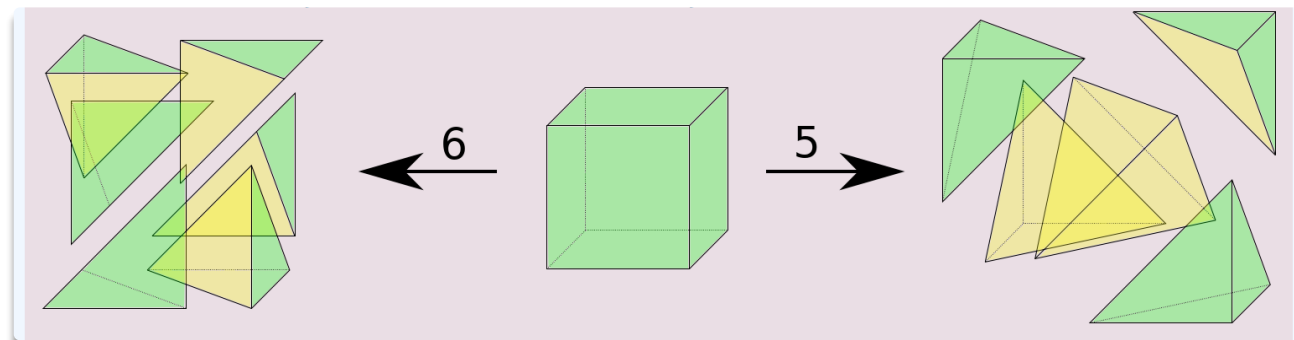
Opět, nejednoznačnost konfigurací



Řešení trilineárním asymptotickým rozhodovačem

# Marching tetrahedra

- ▶ Rozdělení kostek na čtyřstěny
- ▶ Čtyřstěny vyhodnoceny nezávisle
- ▶ Jednoznačné konfigurace
- ▶ Diskuse:
  - ▶ Nevýhody oproti MC?

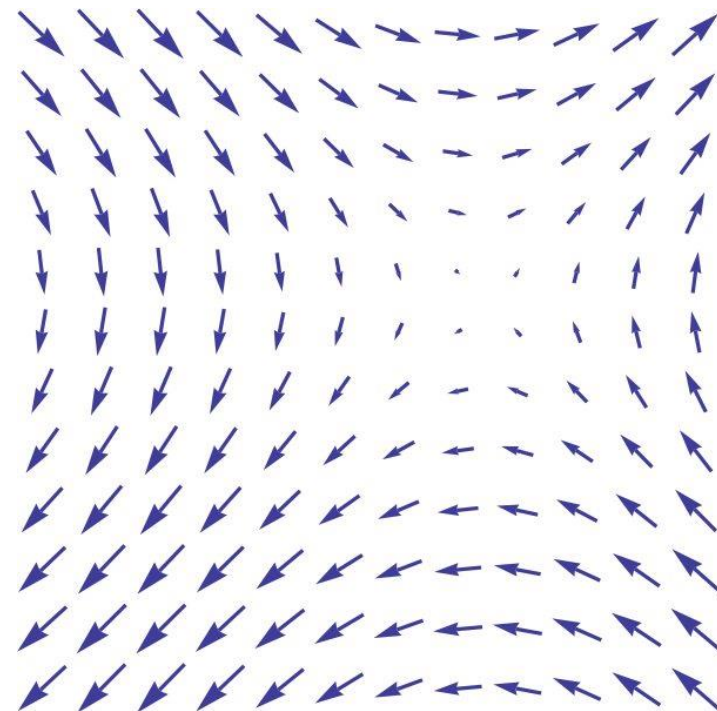
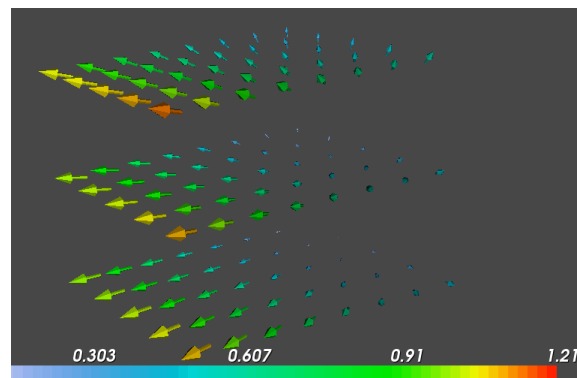


# Vizualizace vektorových polí

- ▶ Vektorová pole reprezentována typicky množinou bodů s přiřazeným vektorem
- ▶ Často jsou výsledkem nejrůznějších simulací
  - ▶ např. rychlost proudění krve v cévách
  - ▶ distribuce zatížení kosti
- ▶ Vizualizace různými způsoby, nejčastěji:
  - ▶ glyfy
  - ▶ streamlines a streaklines (proudnice)

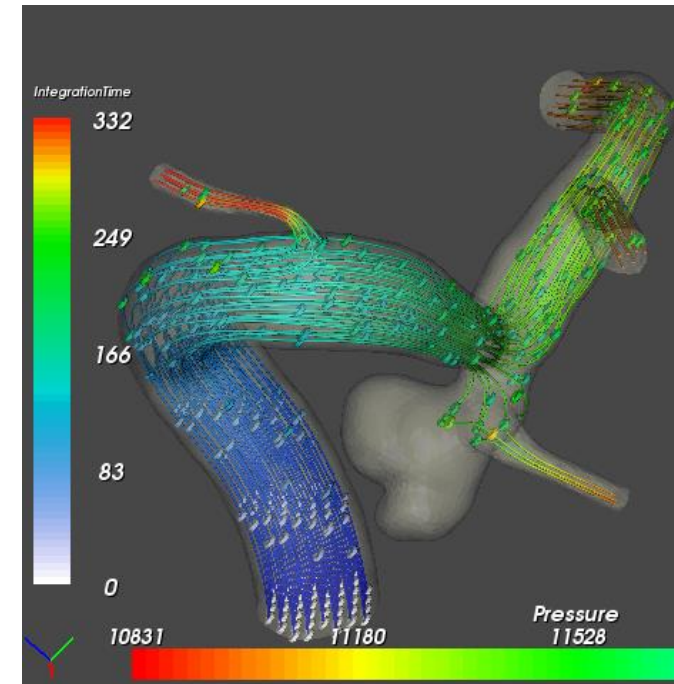
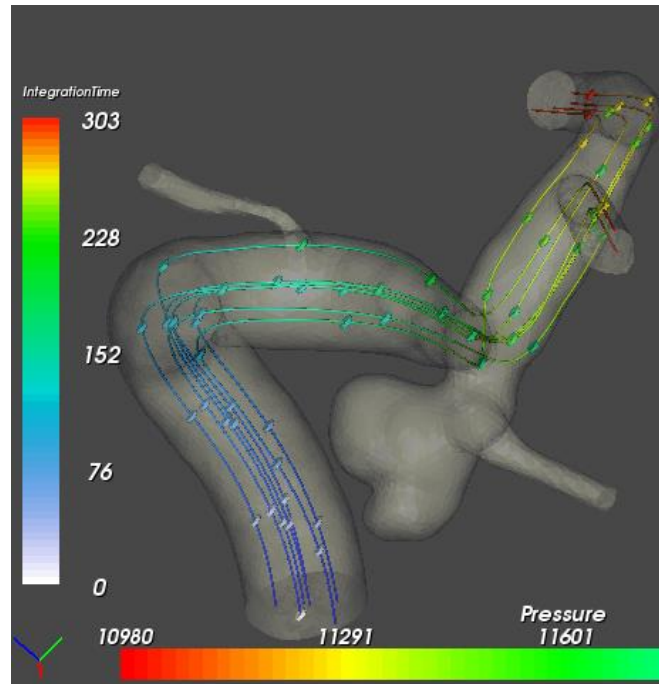
# Vektorová pole: Glyfy

- ▶ Vektorová pole reprezentována typicky množinou bodů s přiřazeným vektorem (např. rychlost)
- ▶ Vizualizace tzv. glyfy
  - ▶ V bodech zobrazen objekt, jehož tvar závisí na velikosti a směru vektoru v daném bodě
  - ▶ Nejjednodušší případ = šipka
    - ▶ velikost šipky je dána nelineární funkcí
      - ▶ Vhodné zejména pro pole s velkou variabilitou velikostí



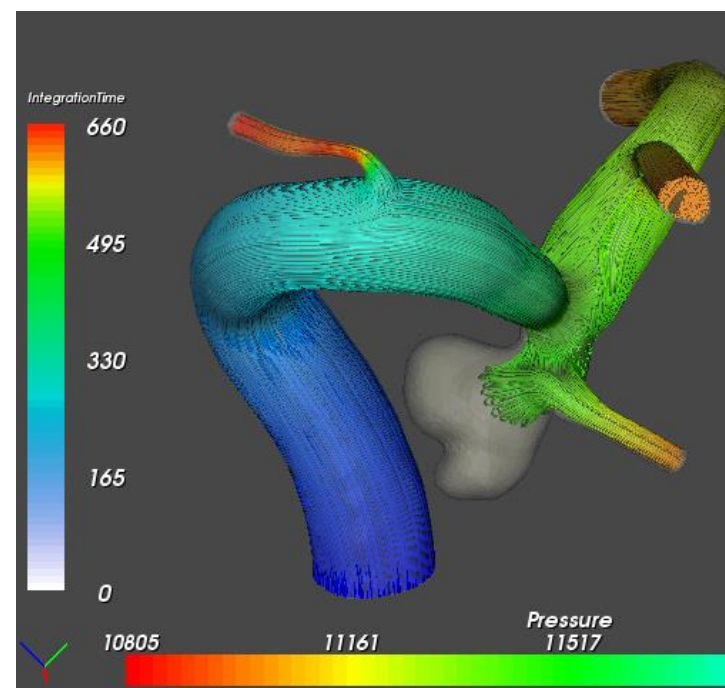
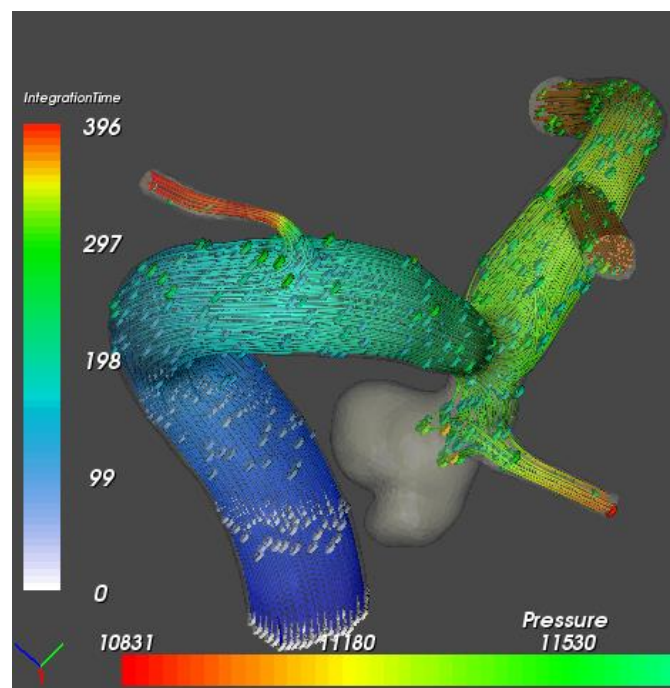
# Vektorová pole: Streamlines

- ▶ Křivky trajektorií částic ve vektorovém poli
  - ▶ Pole nesmí být časově proměnlivé
- ▶ Nutno specifikovat startovní body
  - ▶ Málo bodů = nedostatečně vystihuje chování



# Vektorová pole: Streamlines

- ▶ Mnoho bodů = vysoké časové i paměťové nároky, navíc může být nepřehledné pro uživatele



# Vektorová pole: Streamlines

- ▶ Ruční specifikace startovacích bodů nepraktická
- ▶ Zajímavé chování je často v okolí kritických bodů
- ▶ V kritických bodech je velikost vektoru = 0
- ▶ Opět: kritické body se mohou nacházet i známe
- ▶ **Diskuze:**
  - ▶ Jak detekovat?

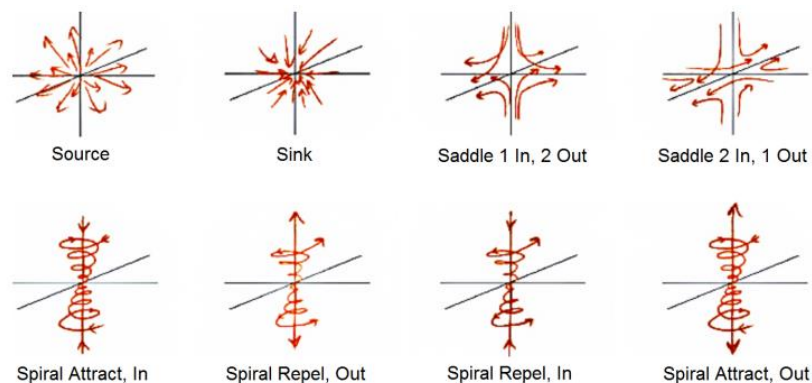


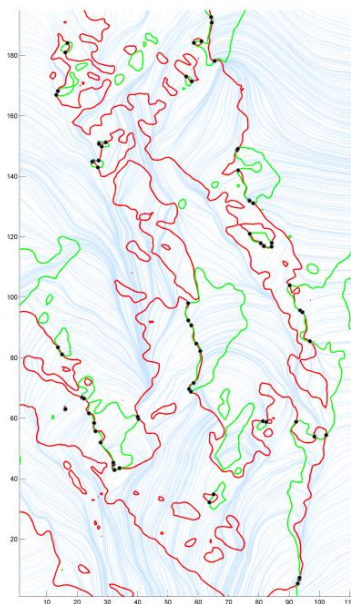
Figure 2.6: Classification of 3D first order critical points.

Šmolík, 2019

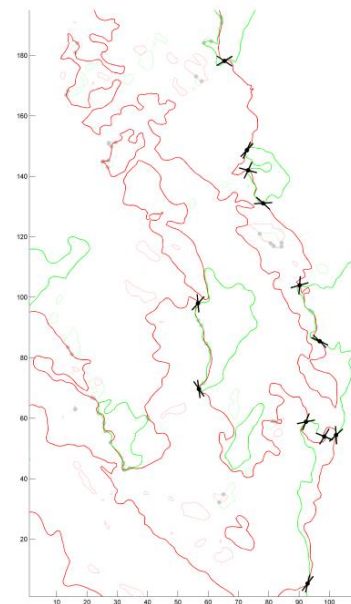
# Vektorová pole: Streamlines

- Kritické body může být vhodné dále redukovat

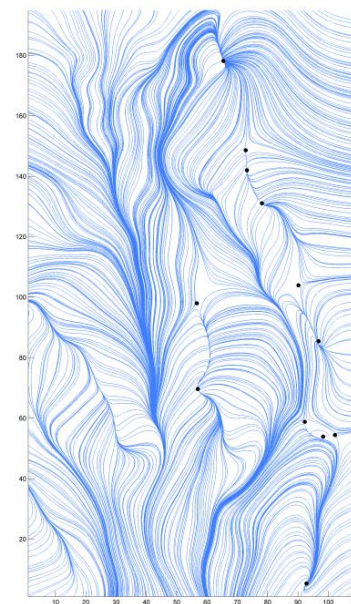
Michal Smolik, Vaclav Skala, and Zuzana Majdisova. Vector field radial basis function approximation. *Advances in Engineering Software*, 123(1):117–129, 2018



(a) All critical points.



(b) Important critical points.



(c) Vector field approximation.

# Vektorová pole: Streamlines

- ▶ Trajektorie určena rovnicí:
- ▶  $s(t) = s_0 + \int_{0 \leq u \leq t} v(s(u)) \cdot du$ 
  - ▶  $s_0$  = startovací bod, tj. místo částice v čase  $t = 0$
  - ▶  $v(x)$  = vektor rychlosti v místě  $x$
- ▶ Nemá analytické řešení, protože  $s$  na obou stranách

# Vektorová pole: Streamlines

- ▶ Řeší se numerickou integrací
  - ▶ Euler:  $s_{i+1} = s_i + v(s_i) \cdot dt$ 
    - ▶  $dt$  musí být dostatečně malé
      - ▶ Co to je?
  - ▶ Runge-Kutta 4. stupně
    - ▶ výpočetně náročnější, přesnější i pro větší  $dt$
    - ▶  $s_{i+1} = s_i + \frac{1}{6} \cdot (a + 2b + 2c + d)$ ,
    - ▶  $a = v(s_i) \cdot dt$ ,  $b = v(s_i + a/2) \cdot dt$ ,  $c = v(s_i + b/2) \cdot dt$ ,  $d = v(s_i + c) \cdot dt$

# Vektorová pole: Částice

- ▶ Zobrazení objektů dynamicky se pohybujících ve vektorovém poli
  - ▶ v případě statického vektorového pole se pohybují po streamlines

