

# Textury a šumové funkce

© 1998-2016 Josef Pelikán  
CGG MFF UK Praha

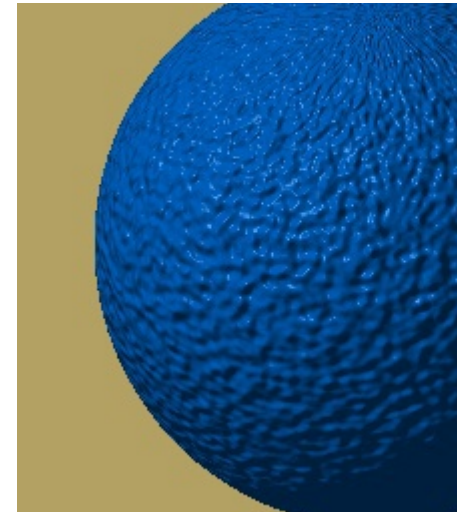
pepca@cgg.mff.cuni.cz

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



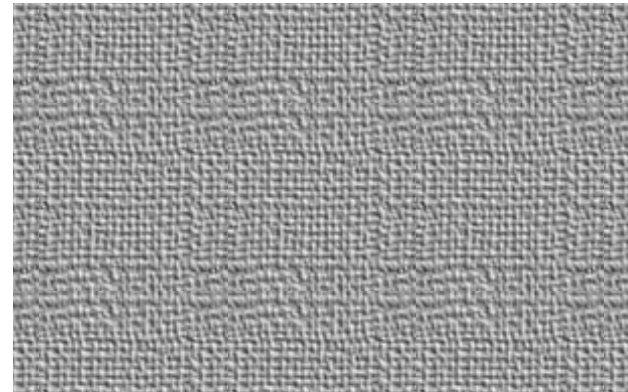
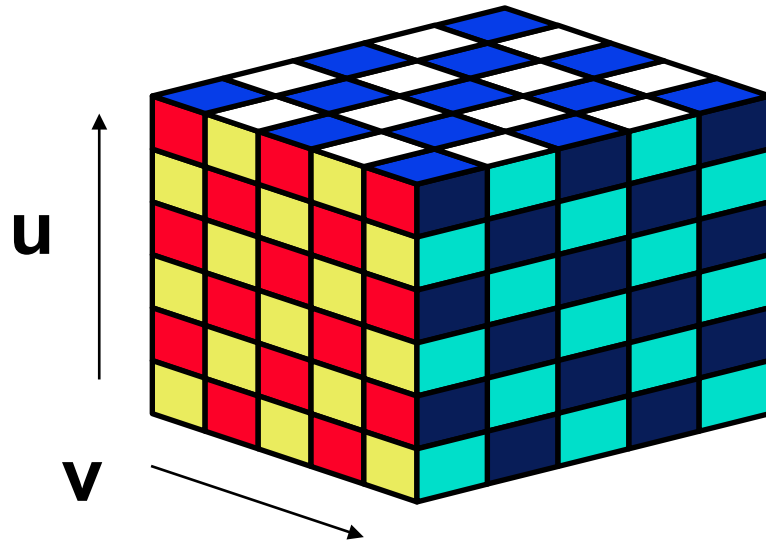
# Účinek textury

- ♦ modulace **barvy** na povrchu předmětů
- ♦ změna parametrů **světelného modelu**
  - např. odrazivost ( $k_D$  a  $k_S$ ), ...
- ♦ modifikace **normálového vektoru**
  - „bump-texture”
  - nahrazení složité (mikro)geometrie
- ➔ napodobení komplikovaných přírodních jevů
  - náhodné textury (užití syntézy šumu)
  - fraktální textury (deterministické i stochastické)





# 2D textura



- ♦ pokrývá **povrch** tělesa (jako tapeta)
- ➡ **mapování textury**:  $[x,y,z] \rightarrow [u,v]$ 
  - inverzní mapovací funkce
- ➡ **vlastní textura**:  $[u,v] \rightarrow \text{barva}$  (normála, ..)



# 3D textura

- ♦ zachycuje změny veličin **uvnitř tělesa**
- ♦ napodobuje **vnitřní strukturu materiálu** (dřevo, mramor, ...)
- ➔ není třeba počítat **inverzní mapování**
- ➔ **3D textura**:  $[x,y,z] \rightarrow \text{barva}$  (odrazivost, atd.)
- ▲ pro napodobení přírodních jevů se využívají tzv. **šumové funkce**
  - pseudonáhodné spojitě „vrásnění“

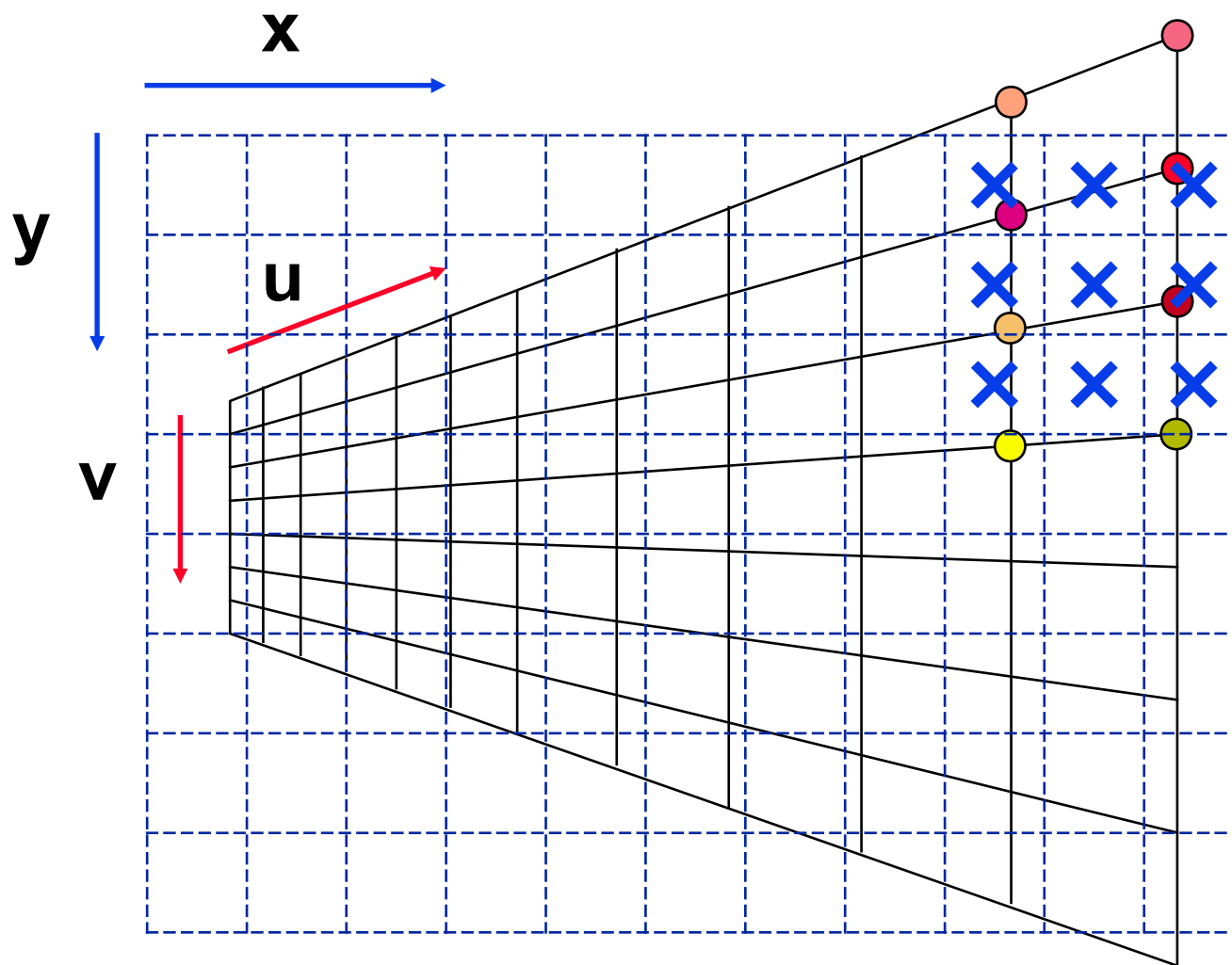




# Implementace textury

- ♦ předem připravené **pole dat** (tabulka, bitmapa)
  - především u 2D textur
  - skutečná (přírodní) data, obrázky, etikety
  - pro větší kvalitu (spojitost) - různé typy interpolace
- ♦ textury definované **algoritmem** (předpisem)
  - jednoduché geometrické tvary
  - fraktály, stochastické funkce (šum, turbulence)
- ♦ **smíšené metody** (předem spočítaná tabulka)
  - náročné modelovací metody (reakční difuze, ..)

# Textura definovaná tabulkou

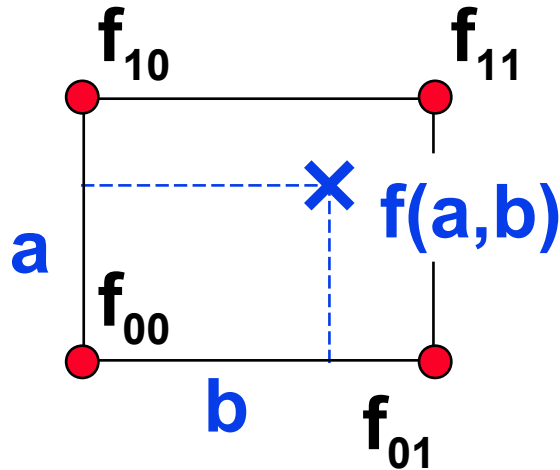




# Typy interpolace

- ➔ **bez interpolace** (zaokrouhlení)
  - nejjednodušší a nejrychlejší metoda
  - pokud se rozlišení obrázku blíží rozlišení textury, vznikají výrazné a nepříjemné artefakty (Doom)
- ➔ **bilineární** interpolace
  - zajišťuje **spojitost** obrazové funkce ( $C^0$ )
- ➔ **polynomiální** interpolace (např. spline funkce)
  - spojitost **vyšších řádů** (u bikubické až  $C^2$ )
  - výpočetně náročná (kombinace 9-16 hodnot ve 2D)

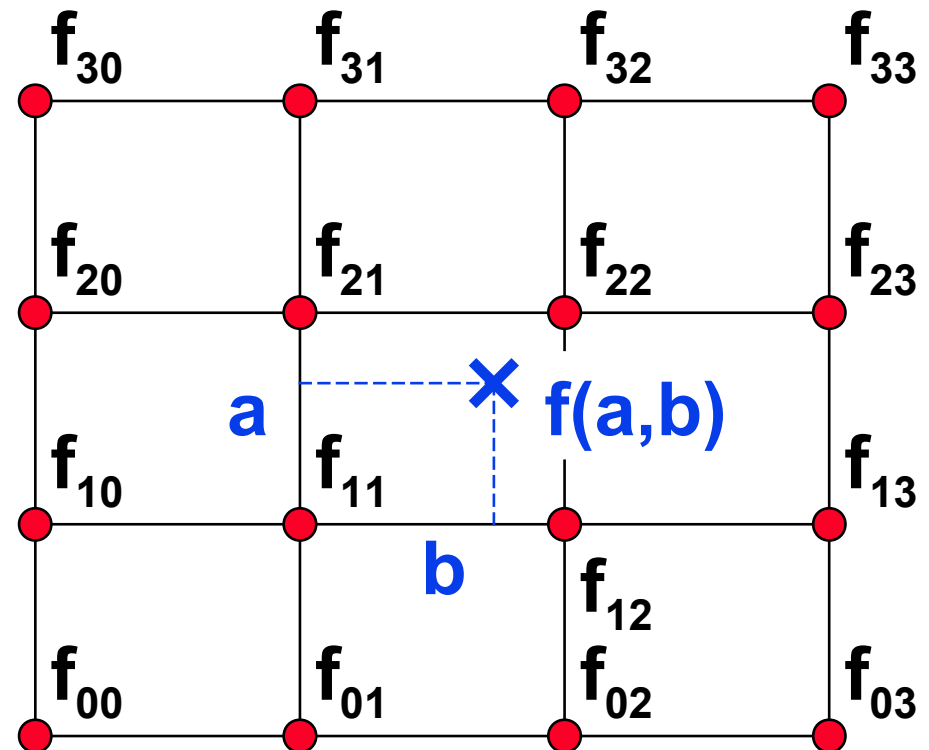
# Bilineární a bikubická interpolace



$$f(a,b) = a \cdot [b \cdot f_{11} + (1-b) \cdot f_{10}] + (1-a) \cdot [b \cdot f_{01} + (1-b) \cdot f_{00}]$$

$$f(a,b) = \sum_{i,j=0}^3 C_i(a) C_j(b) f_{ij}$$

$C_i(t)$  .. kubické polynomy







# Kubická B-spline interpolace

$$f(\mathbf{a}, \mathbf{b}) = \sum_{i=0}^3 \sum_{j=0}^3 C_i(\mathbf{a}) C_j(\mathbf{b}) f_{ij}$$

**B-spline**

váhové funkce:

$$\sum_{i=0}^3 C_i(\mathbf{t}) = 1$$

$$0 \leq C_i(\mathbf{t}) \leq 1 \quad \text{pro} \quad 0 \leq t \leq 1$$

$$C_0(t) = \frac{1}{6}(1-t)^3$$

$$C_1(t) = \frac{1}{6}(3t^3 - 6t^2 + 4)$$

$$C_2(t) = \frac{1}{6}(-3t^3 + 3t^2 + 3t + 1)$$

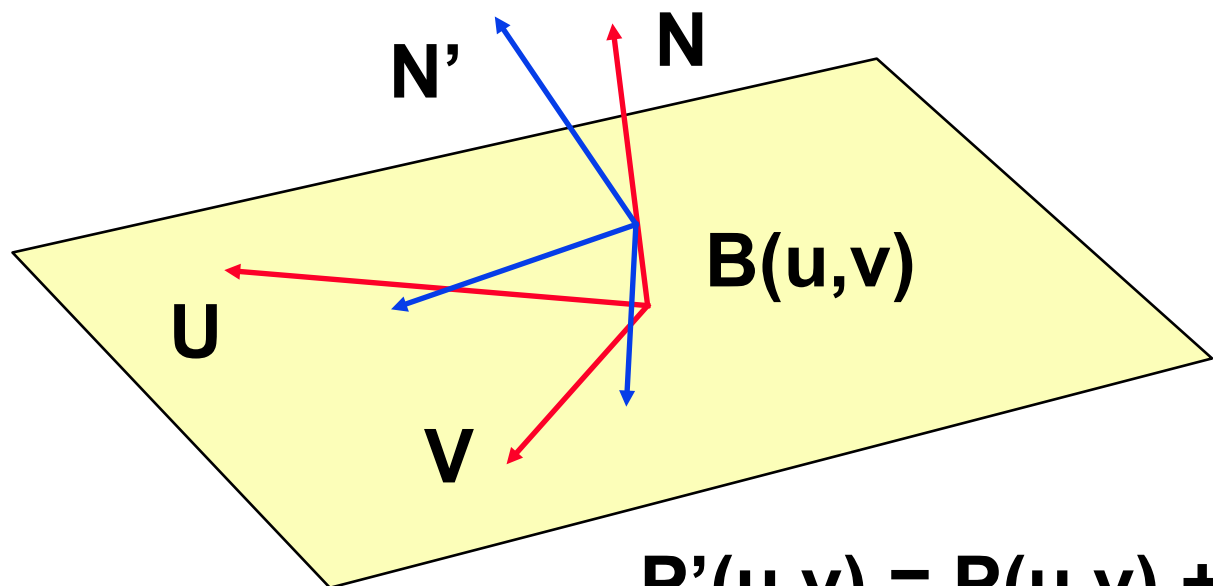
$$C_3(t) = \frac{1}{6}t^3$$

# Algoritmické a smíšené textury



- ➔ jednoduché **geometrické tvary**, vzorky
  - šachovnice, pravidelné pruhy, ..
- ➔ napodobení **přírodních tvarů**
  - často se využívají **pseudonáhodné algoritmy** (syntéza spojité šumové funkce)
  - fraktály, turbulence (mraky, špína, ..)
  - reakční difuze (napodobení kůže a srsti zvířat)
  - prostorové náhodné textury (dřevo, mramor, ..)

# Modulace normály („bump map“)

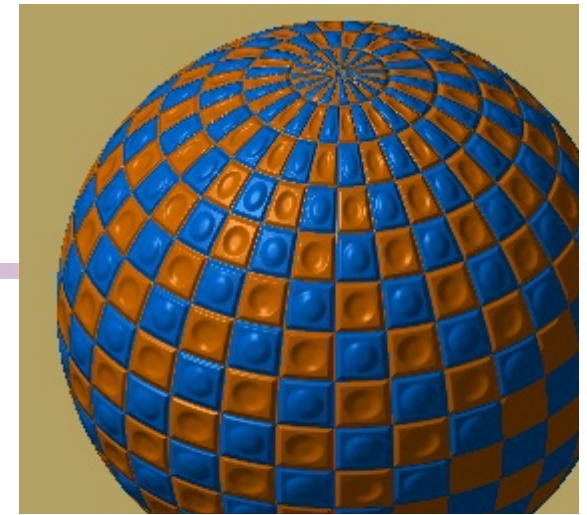


$$\underline{N} = U \times V$$

$$\underline{P'(u,v) = P(u,v) + B(u,v) \cdot N / |N|}$$

- ➔ napodobení **nerovností** na povrchu tělesa
- ➔ **B(u,v)** je funkce lokálního posunutí povrchu:  
+ ven z tělesa, – dovnitř tělesa

# Modulace normály



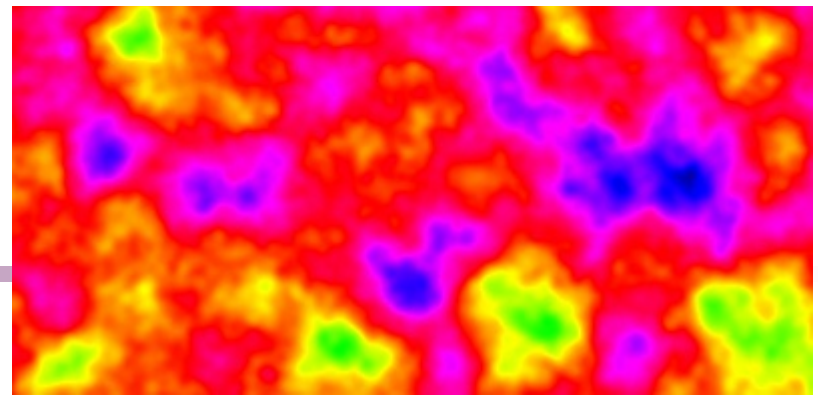
Původní normála:  $\mathbf{N} = \mathbf{U} \times \mathbf{V}$

posunutý bod:  $\mathbf{P}'(\mathbf{u}, \mathbf{v})$  =  $\mathbf{P}(\mathbf{u}, \mathbf{v}) + \frac{\mathbf{B}(\mathbf{u}, \mathbf{v}) \cdot \mathbf{N}}{|\mathbf{N}|}$

aproximace modifikované normály:

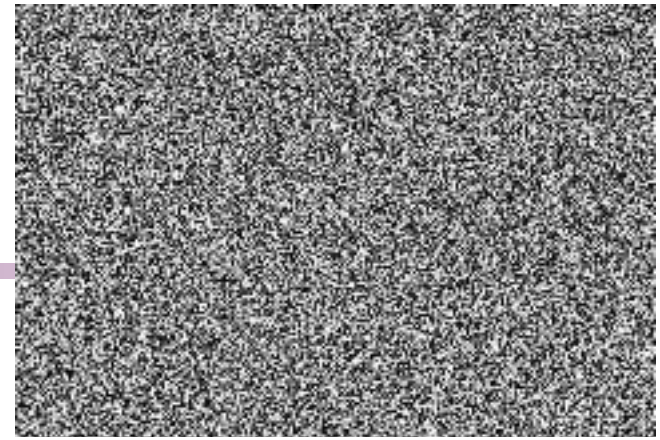
$$\mathbf{N}' = \mathbf{N} + \frac{\frac{\partial \mathbf{B}}{\partial \mathbf{u}}(\mathbf{u}, \mathbf{v}) \cdot (\mathbf{N} \times \mathbf{V}) - \frac{\partial \mathbf{B}}{\partial \mathbf{v}}(\mathbf{u}, \mathbf{v}) \cdot (\mathbf{N} \times \mathbf{U})}{|\mathbf{N}|}$$

# Syntéza šumu



- ➔ cílem je **subjektivně nahodilý** vzhled (tvar)
  - napodobení komplikovaných přírodních jevů
  - výsledek působení chaotického systému, náhodné difuze, systému s částečnou zpětnou vazbou, ..
- ➔ výpočet hodnoty šumové funkce v konkrétním bodě musí být **deterministický** (opakovatelný)
  - paralelní implementace, opakovaný dotaz
- ➔ požadovaná **spektrální charakteristika** šumu
  - nekorelovaný (bílý) šum, spojitý šum

# Bílý šum



- ♦ šum s **neomezeným spektrem**
  - hodnoty v různých bodech mají nulovou korelaci
- ➔ příklad implementace 2D bílého šumu:

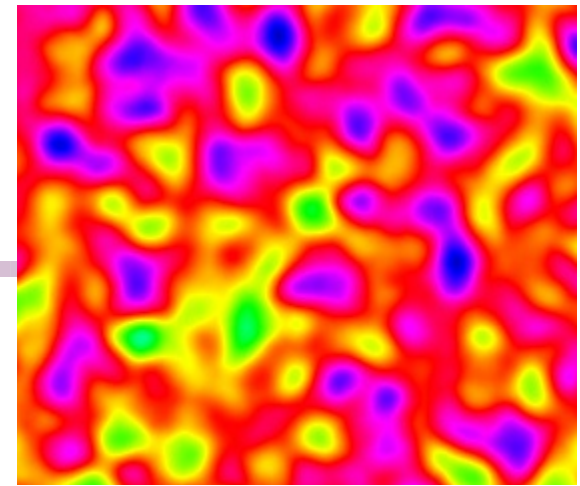
```
double RandomTab[RANDOMTABLEN];           // random values
int Indx[ILEN], Indy[ILEN];                // random permutations

double white_noise_2D ( double x, double y )
{ int i = HASH( Indx[LOWBITS(x)], Indy[LOWBITS(y)] );
  return( RandomTab[ i % RANDOMTABLEN ] );
}
```

využívá **k** nejnižších bitů mantisy (**LOWBITS**)

**RandomTab**, **Indx**, **Indy** jsou předem spočítané tabulky

# Spojité šum



- ♦ **spojitá funkce** s omezeným spektrem
  - stacionární, izotropní (invarian. na posunutí, otočení)
  - krátká perioda funkce může být na závadu
- ➔ **Fourierovská syntéza**
  - přesně mohu ovlivňovat frekvenční charakteristiku
- ➔ **interpolace** náhodných hodnot v mřížce
  - klasická interpolace (např. pomocí B-spline funkcí)
  - Hermitovská interpolace - gradienty (Perlin)
  - stochastická mřížka - řídká konvoluce (Lewis)



# Interpolace v pravidelné mřížce

- ① předem vygeneruji **síť pseudonáhodných čísel** (vektorů, směrníc, ..)
  - požadované rozdělení pravděpodobnosti
  - 1D, 2D nebo 3D topologie
  - ve vícerozměrných případech mohu ušetřit paměť pomocí hašovací funkce **HASH (x, y, z)** - viz výše
- ② **interpolace** v neuzlových bodech
  - separabilní metody (složky počítám postupně)
  - nejčastěji kvadratické nebo kubické polynomy
  - **2D**: 4 až 16 uzlů, **3D**: 8 až 64 uzlů

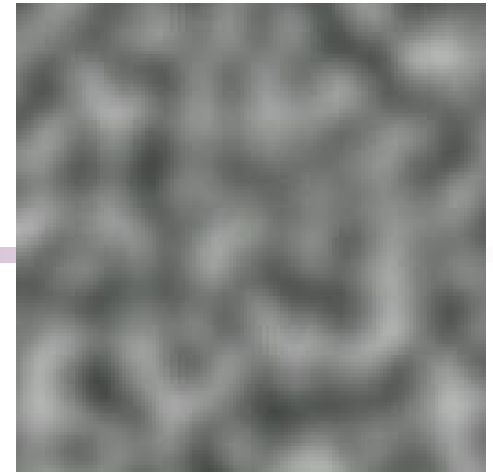




# Metoda Kena Perlina (3D šum)

- ♦ spektrum je omezené na jednu oktávu ( $f \div 2f$ )
- ♦ efektivní implementace
- ① předem vygeneruji **síť pseudonáhodných gradientních vektorů**  $[a,b,c,d]_{ijk}$ 
  - $[a,b,c]_{ijk}$  je pseudonáhodný **jednotkový směr** (vybírám a normalizuji pouze náhodné vektory kratší než 1)
  - $d_{ijk}$  je hodnota funkce v uzlovém bodě  $[x_i, y_j, z_k]$
  - přepočítám  $d'_{ijk} = d_{ijk} - a_{ijk} \cdot x_i - b_{ijk} \cdot y_j - c_{ijk} \cdot z_k$

# Metoda Kena Perlina



② hodnoty v **uzlech**:

$$K_{ijk}(\mathbf{x}, \mathbf{y}, \mathbf{z}) = d'_{ijk} + a_{ijk} \cdot x + b_{ijk} \cdot y + c_{ijk} \cdot z$$

③ **interpolační** kubické spline polynomy:

$$w(t) = 2|t|^3 - 3t^2 + 1 \quad \text{pro } |t| < 1$$

$$w(t) = 0 \quad \text{jinde}$$

– nosič má poloměr 1  $\Rightarrow$  potřebuji pouze  $2^D$  uzlů

$$a(\mathbf{x}, \mathbf{y}, \mathbf{z}) = \sum_{i=\lfloor \mathbf{x} \rfloor}^{\lfloor \mathbf{x} \rfloor + 1} w(\mathbf{x} - \mathbf{i}) \sum_{j=\lfloor \mathbf{y} \rfloor}^{\lfloor \mathbf{y} \rfloor + 1} w(\mathbf{y} - \mathbf{j}) \sum_{k=\lfloor \mathbf{z} \rfloor}^{\lfloor \mathbf{z} \rfloor + 1} w(\mathbf{z} - \mathbf{k}) \cdot a_{ijk}$$



# Metoda řídké konvoluce (Lewis)

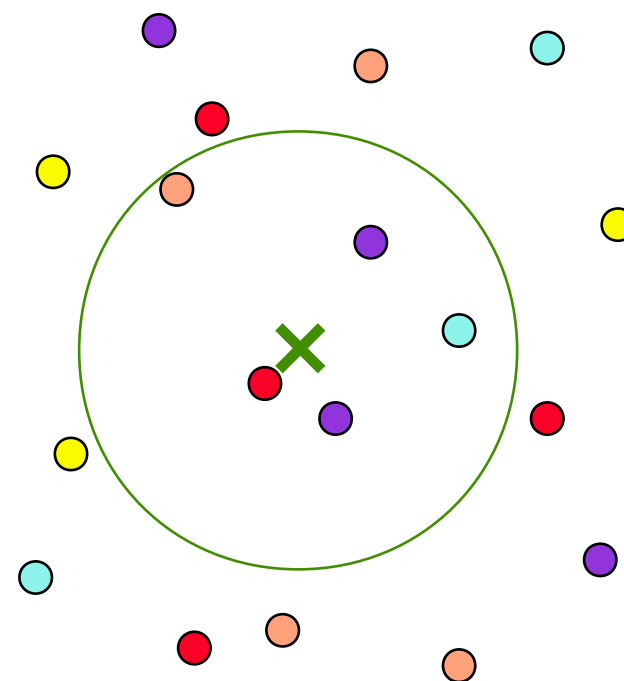
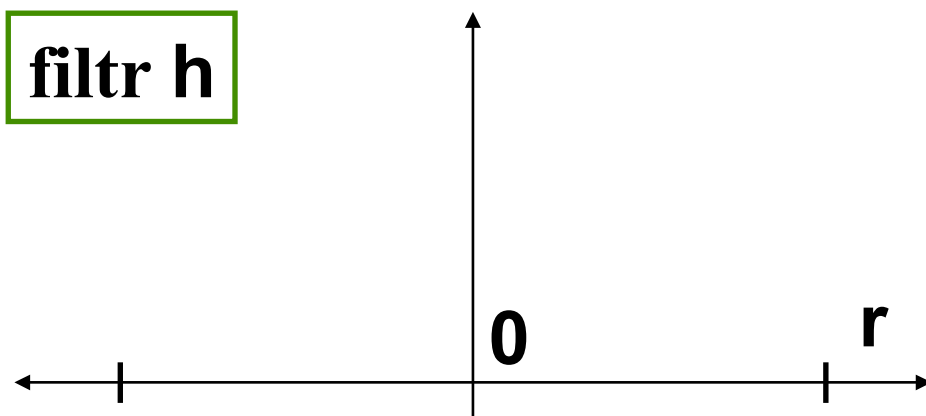
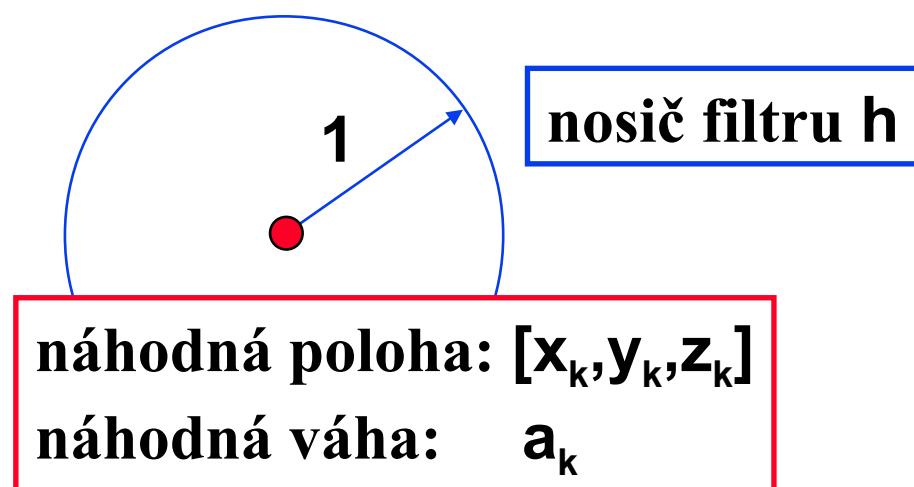
- ♦ možnost řízení spektrální charakteristiky
- ♦ efektivní implementace
- ➔ **konvoluce 3D filtru  $h(\mathbf{x}, \mathbf{y}, \mathbf{z})$  s Poissonovým šumem  $\gamma$ :**

$$\eta(\mathbf{x}, \mathbf{y}, \mathbf{z}) = \int_{\mathbb{R}^3} \gamma(\mathbf{u}, \mathbf{v}, \mathbf{w}) \cdot \underline{h(\mathbf{x} - \mathbf{u}, \mathbf{y} - \mathbf{v}, \mathbf{z} - \mathbf{w})} \, d\mathbf{u} \, d\mathbf{v} \, d\mathbf{w}$$

$$\gamma(\mathbf{x}, \mathbf{y}, \mathbf{z}) = \sum_{\mathbf{k}} \underline{\mathbf{a}_{\mathbf{k}}} \cdot \delta(\mathbf{x} - \underline{\mathbf{x}_{\mathbf{k}}}, \mathbf{y} - \underline{\mathbf{y}_{\mathbf{k}}}, \mathbf{z} - \underline{\mathbf{z}_{\mathbf{k}}})$$



# Konvoluce Poissonova šumu





# Metoda řídké konvoluce

Díky diskrétnímu charakteru Poissonova šumu:

$$n(\mathbf{x}, \mathbf{y}, \mathbf{z}) = \sum_k \underline{a_k} \cdot h(\mathbf{x} - \underline{\mathbf{x}_k}, \mathbf{y} - \underline{\mathbf{y}_k}, \mathbf{z} - \underline{\mathbf{z}_k})$$

- ➔ **hustotou impulsů  $[\mathbf{x}_k, \mathbf{y}_k, \mathbf{z}_k]$  mohou řídit kvalitu šumu**
  - při **10** a více impulsech na plochu filtru **h** je kvalita téměř nerozeznatelná od interpolačního šumu
  - při stejné kvalitě je řídká konvoluce efektivnější



# Efektivní implementace

- prostor rozdělím na krychlové **buňky** se stranou délky **r** (poloměr filtru **h** - obvykle **1**)
- v každé buňce generuji impulsy **nezávisle** pomocí **pseudonáhodného generátoru** s počáteční hodnotou semínka **Seed<sub>ijk</sub>**
  - hodnoty **Seed<sub>ijk</sub>** nageneruji předem nějakým jiným (nezávislým) generátorem
  - nebo mohu pro úsporu paměti (a proti opakování vzorku) použít hašovací funkci **HASH(x, y, z)**



# Efektivní implementace

- při výpočtu hodnoty **Noise(x,y,z)** stačí projít pouze **několik sousedních buněk**
  - v 2D případě je to **4 ÷ 9** buněk
  - v 3D případě je to **8 ÷ 27** buněk
- pro **izotropní šum** (symetrický filtr **h**) mohu předem spočítat hodnoty **h(r<sup>2</sup>)** do tabulky
  - při výpočtu konvoluce nemusím odmocňovat



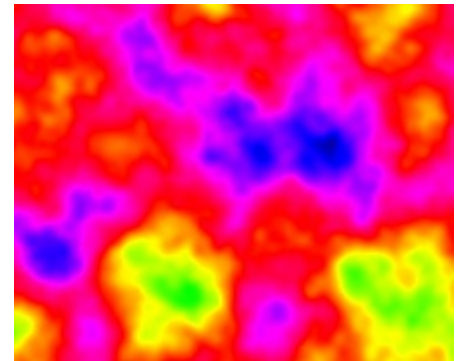
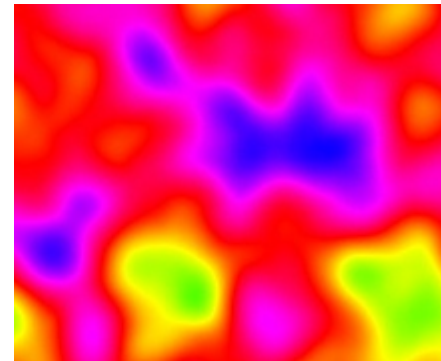
# Kombinace šumových funkcí

- složení několika frekvencí  $\mathbf{f}_i$  s amplitudami  $\mathbf{a}_i$  posunutých o vektory  $[\mathbf{x}_i, \mathbf{y}_i, \mathbf{z}_i]$ :

$$\sum_i \mathbf{a}_i \cdot \text{Noise}[\mathbf{f}_i \cdot (\mathbf{x} + \mathbf{x}_i), \mathbf{f}_i \cdot (\mathbf{y} + \mathbf{y}_i), \mathbf{f}_i \cdot (\mathbf{z} + \mathbf{z}_i)]$$

- napodobení **turbulence**:

$$\mathbf{f}_i = \mathbf{F}^i, \mathbf{a}_i = \mathbf{A}^{-i}$$



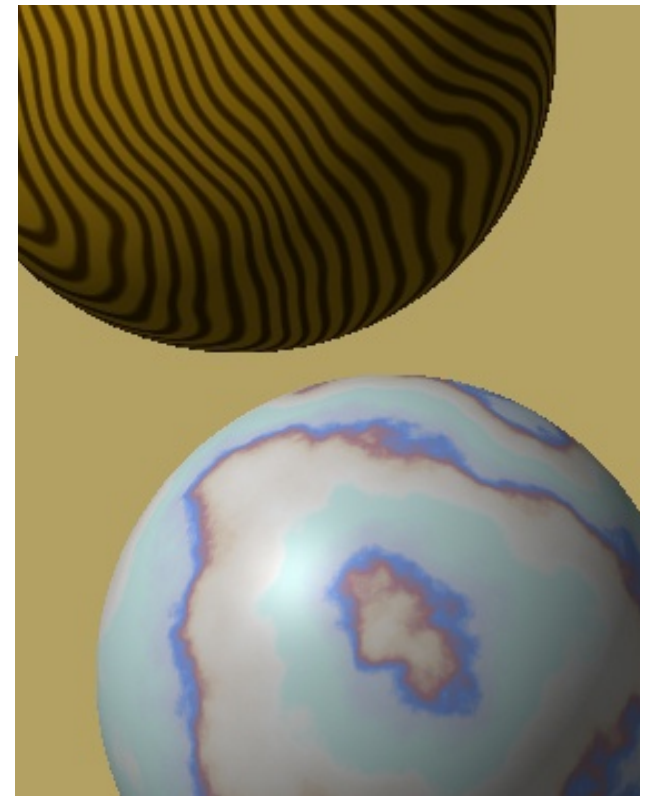
$$\sum_i \frac{1}{\mathbf{A}^i} \cdot \text{Noise}[\mathbf{F}^i \cdot (\mathbf{x} + \mathbf{x}_i), \mathbf{F}^i \cdot (\mathbf{y} + \mathbf{y}_i), \mathbf{F}^i \cdot (\mathbf{z} + \mathbf{z}_i)]$$



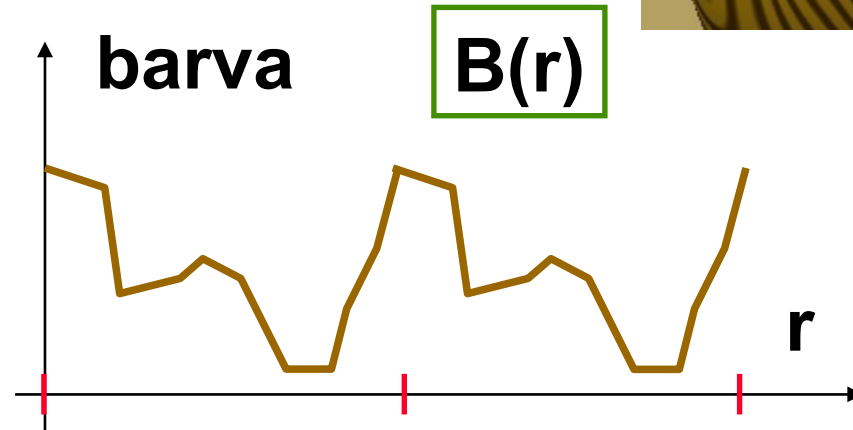
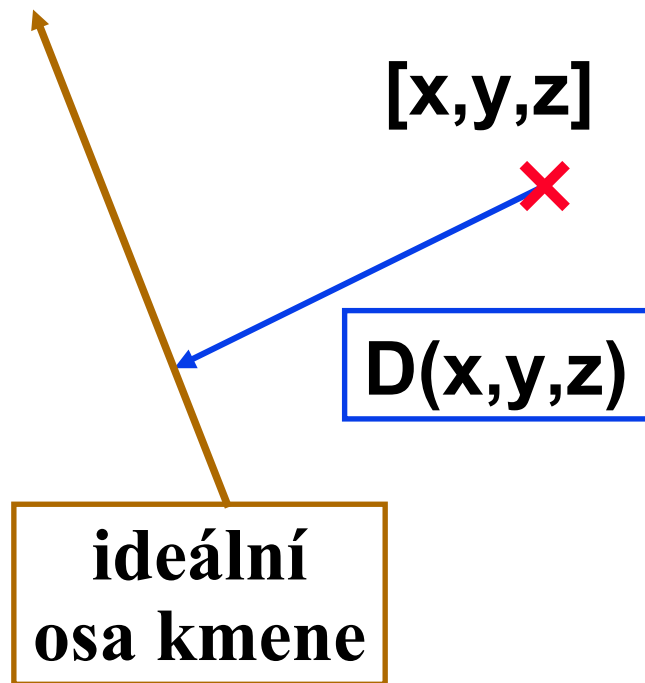
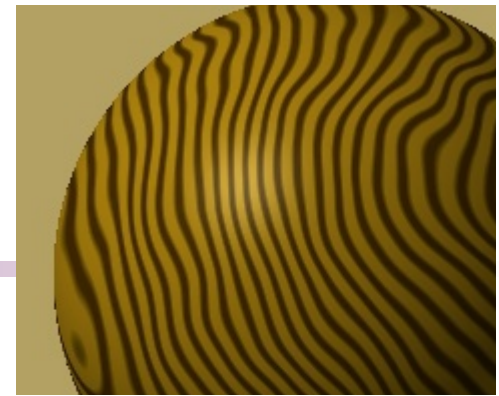


# Aplikace šumových funkcí

- náhodné **modifikace normály** („bump map”)
  - iluze nepravidelně zvrásněného povrchu těles
  - „pomerančová kůra”
- **turbulence**
  - mlha, mraky, použití v modelování
- **3D texture**
  - vnitřní struktura materiálu
  - dřevo, mramor, ..



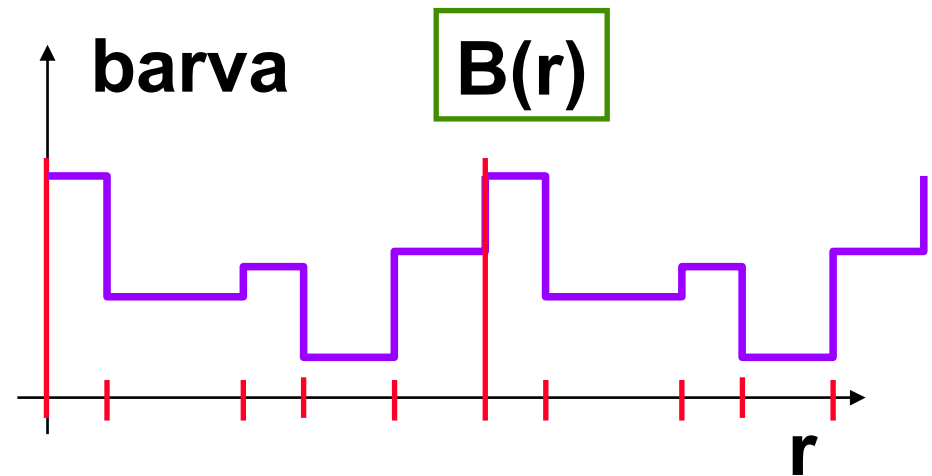
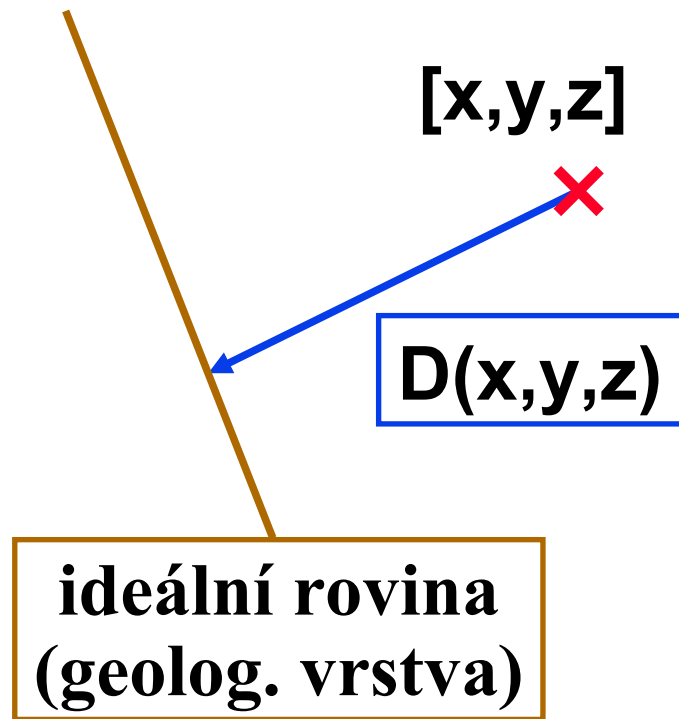
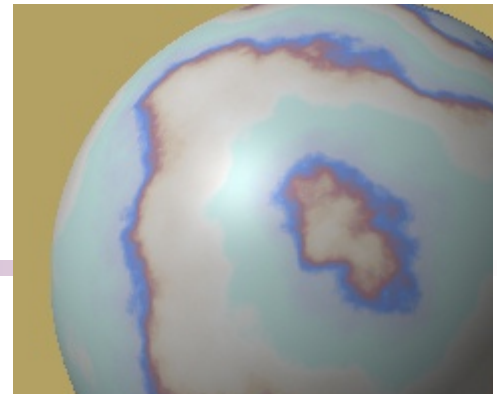
# Simulace struktury dřeva



$$\underline{B[D(x, y, z) + \text{Noise}(x, y, z)]}$$

$$\underline{B[D(x, y, z) \cdot (1 + \text{Noise}_1(x, y, z)) + \text{Noise}_2(x, y, z)]}$$

# Simulace struktury mramoru



$$\underline{B[D(x, y, z) + \text{Turb}(x, y, z)]}$$



# Literatura

- **K. Perlin: *An Image Synthesizer***, Computer Graphics, Vol. 19, #3, July 1985, 287-296
- **K. Perlin, E. M. Hoffert: *Hypertexture***, Computer Graphics, Vol. 23, #3, July 1989, 253-262
- **J. P. Lewis: *Algorithms for Solid Noise Synthesis***, Computer Graphics, Vol. 23, #3, July 1989, 263-270
- **J. Foley, A. van Dam, S. Feiner, J. Hughes: *Computer Graphics, Principles and Practice***, 741-745, 1015-1018, 1043-1047