

Rekurzivní sledování paprsku

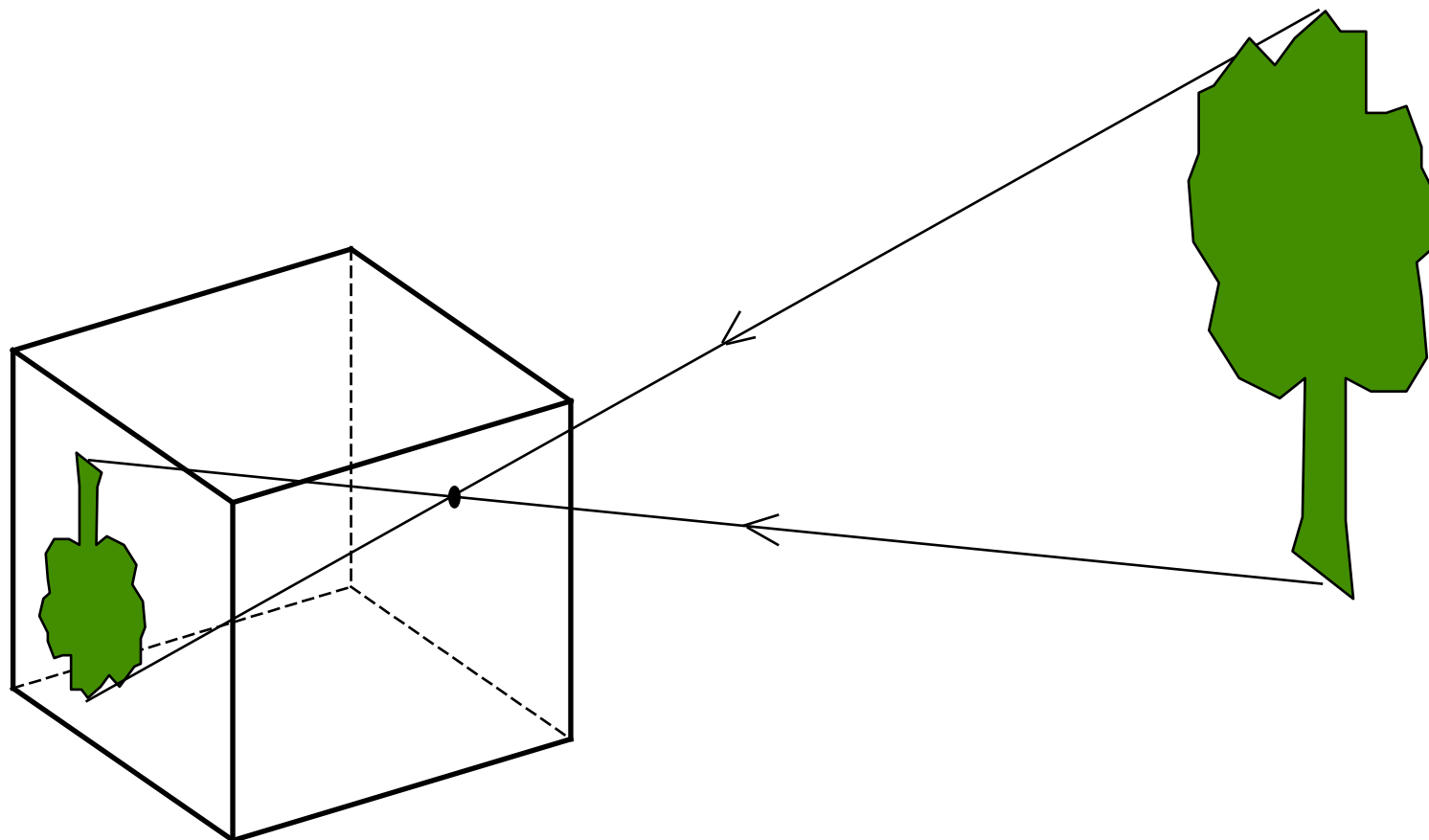
© 1996-2019 Josef Pelikán
CGG MFF UK Praha

pepca@cgg.mff.cuni.cz

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

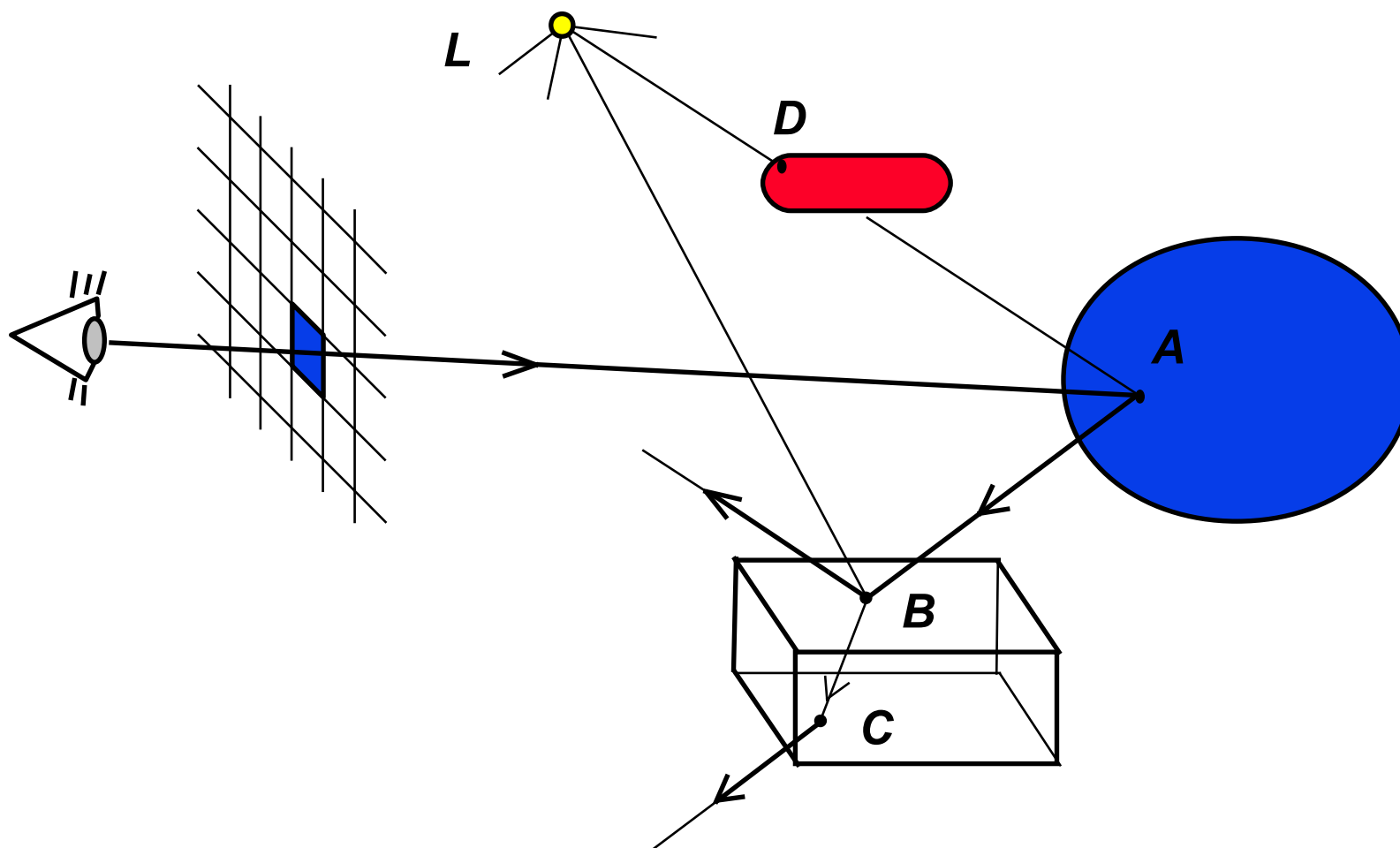


Model dírkové kamery



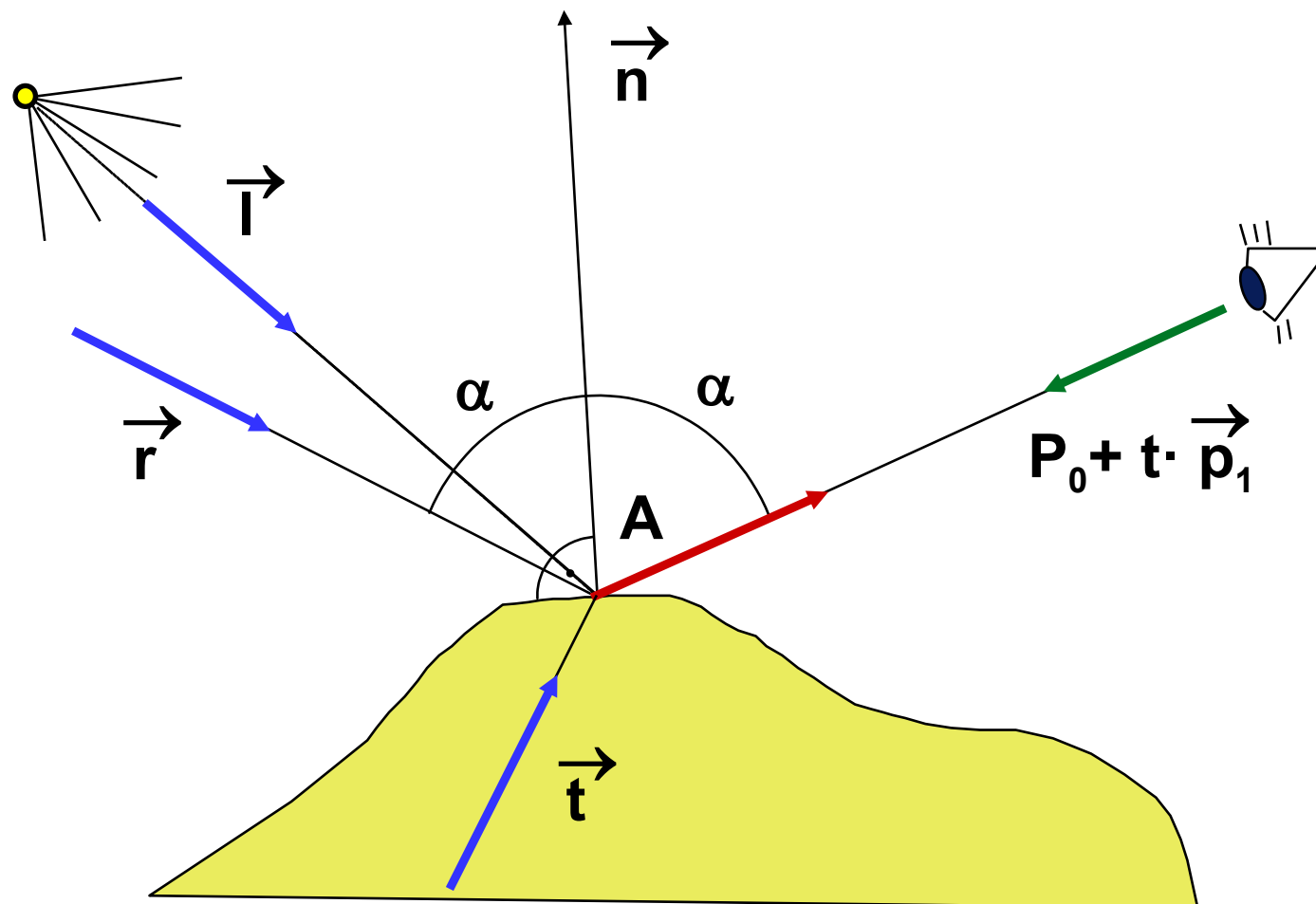


Zpětné sledování paprsku





Skládání světla





Rekurzivní implementace

```
function Sleduj ( P0, P1 : Point3D; hloubka : integer ) : RGB;
    { P0..počátek paprsku, P1..směr, hloubka..počet odražení }
var A, R, T : Point3D;           { pomocné body a vektory }
    B : RGB;                       { výsledná barva }
begin
    A := Prusecik (Scena, P0, P1); { průsečík paprsku se scénou }
    if A==0 then Sleduj := Pozadi { paprsek na nic nenarazil }
    else
        begin                       { paprsek narazil na těleso }
            B := 0;
            for i := 1 to N do       { příspěvky od světelných zdrojů }
                if Prusecik (Scena, A, L[i]-A) == 0 then B := B + kL * Svetlo (A, L[i]);
            hloubka := hloubka + 1;
            if hloubka <= maxhloubka then { konec rekurze }
                begin
                    if "A je odrazivé" then
                        begin
                            "spočítej R"           { odražený paprsek }
                            B := B + kR * Sleduj (A, R, hloubka);
                        end;
                end;
    end;
```

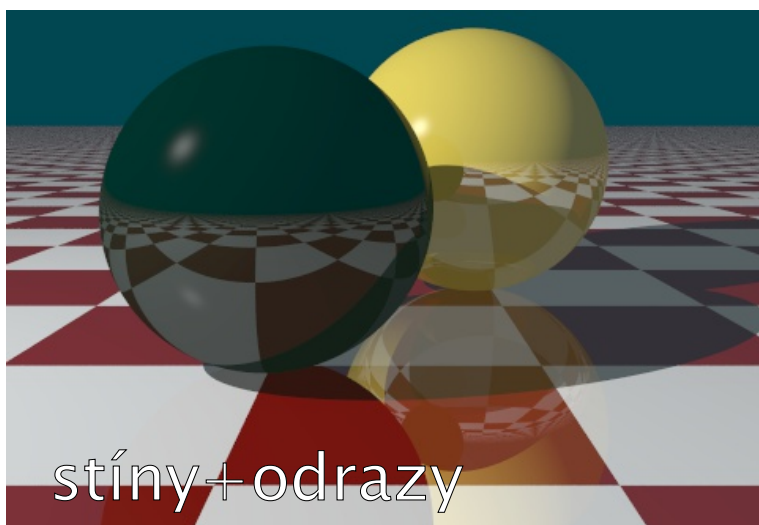
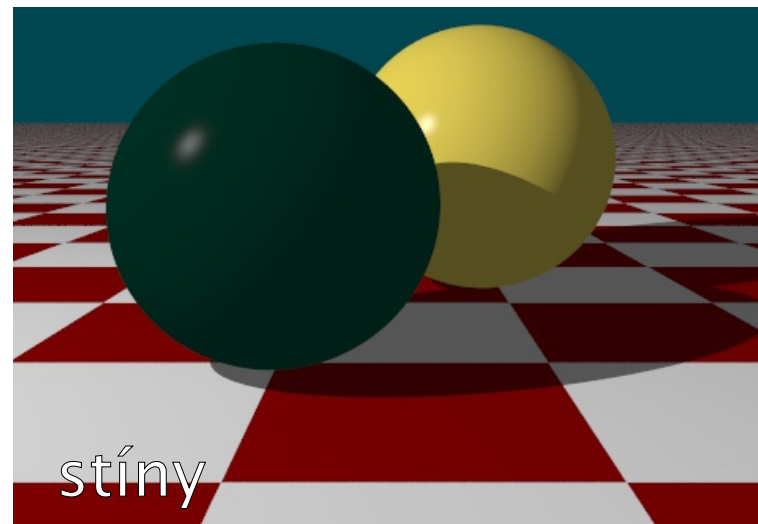
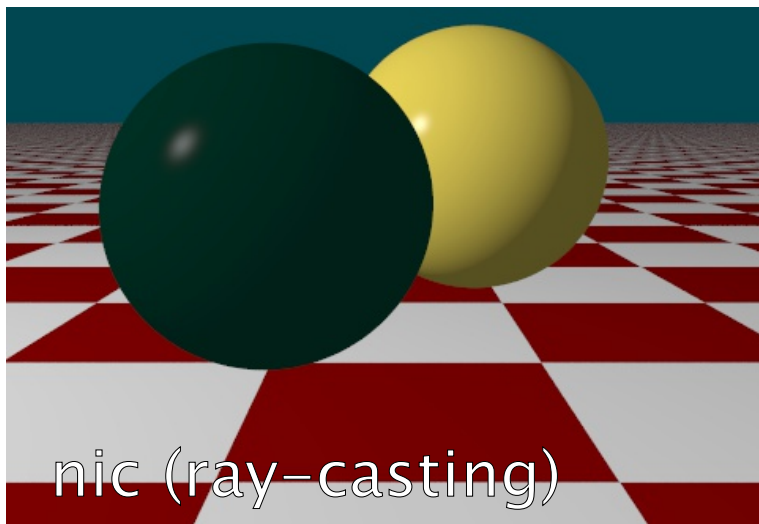


Rekurzivní implementace

```
if "A je průhledné" then  
  begin  
    "spočítej T"          { lomený paprsek }  
    B := B + kT * Sleduj (A, T, hloubka) ;  
  end ;  
end ;  
Sleduj := B;           { nastrádaná návratová hodnota }  
end ;
```



Jednotlivé složky





Řízení hloubky rekurze

- 1 **statické** - omezení konstantou (nehodí se pro scény obsahující zrcadla i méně odrazivé lesklé povrchy)
- 2 **dynamické** - podle „významu“ paprsku
 - ➔ „význam“ je procentuální podíl právě sledovaného paprsku na výsledné barvě pixelu (pro primární paprsky: 100%)
 - ➔ omezení „významu“ konstantou (např. 2-10%)
- 3 **kombinované** - omezení hloubky i „významu“ paprsku

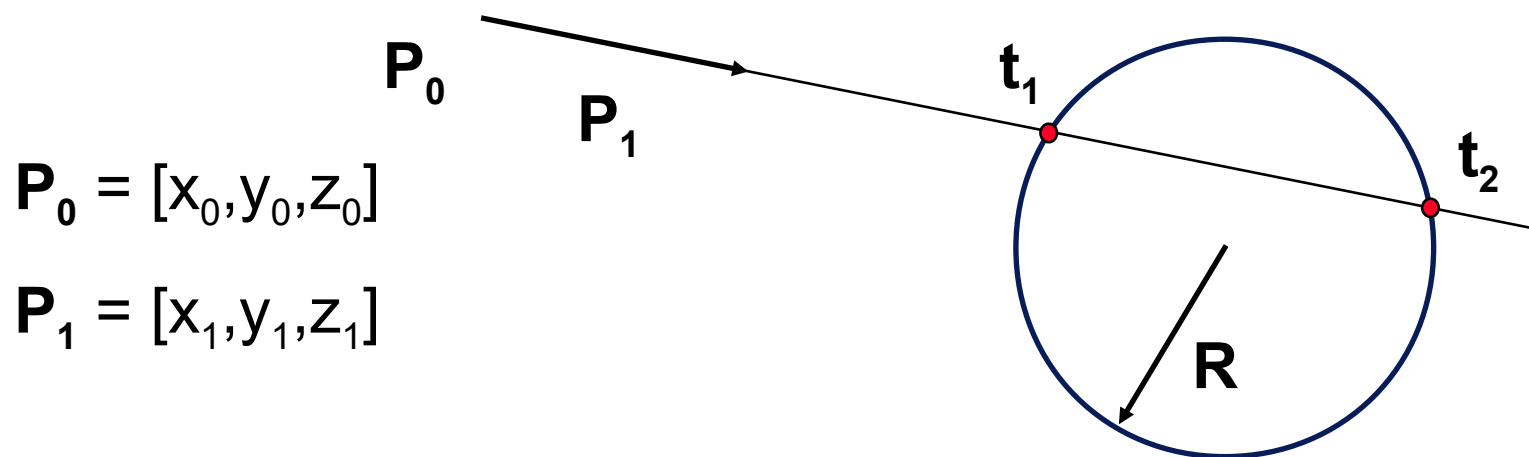


Výpočet průsečíku

- 1 souřadnice průsečíku (nebo „nekonečno“)
 - 2 číslo tělesa (plochy)
 - 3 normálový vektor povrchu tělesa
- ➔ **časově nejnáročnější operace (80-90% času)**
– urychlovací metody
- ◆ **analytický výpočet** (koule, válec, kvádr, ..)
 - ◆ **numerický výpočet** (aproximační plochy, rotační tělesa, implicitní povrchy, ..)



Průsečík paprsku s koulí



→ paprsek: $P_0 + t P_1, t > 0$ (1)

→ koule (střed v počátku): $x^2 + y^2 + z^2 - R^2 = 0$ (2)

→ po dosazení (1) do (2) vyjde **kvadratická rovnice (t)**:

$$t^2 (x_1^2 + y_1^2 + z_1^2) + 2t (x_0 x_1 + y_0 y_1 + z_0 z_1) + x_0^2 + y_0^2 + z_0^2 - R^2 = 0$$

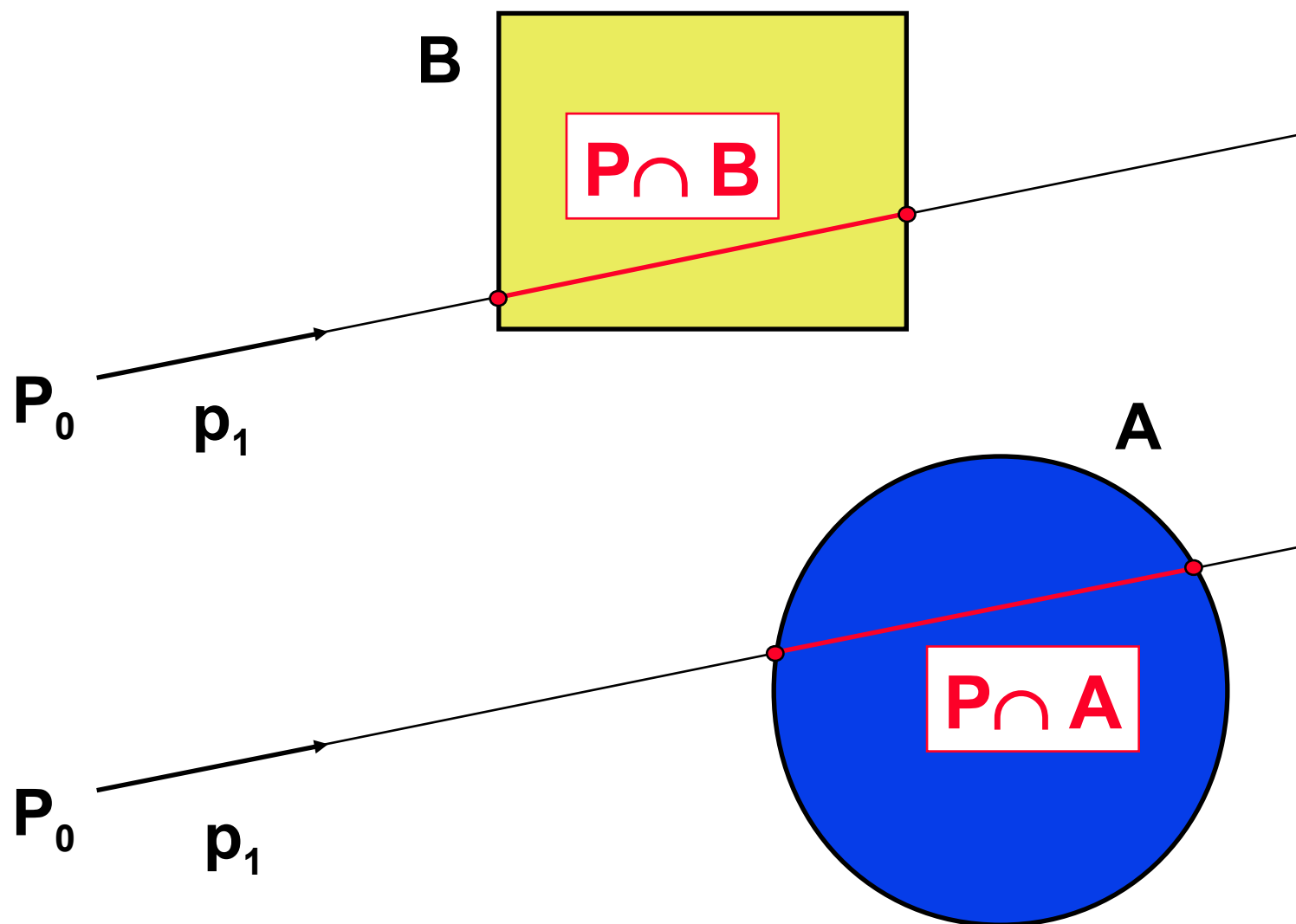


Průsečík s CSG scénou

- ◆ pro **elementární tělesa** umím průsečíky spočítat
 - začátek a konec průniku paprsku s tělesem pro konvexní tělesa
- ◆ **množinové operace** provádím na polopřímce paprsku:
 - díky distributivitě: $P \cap (A - B) = (P \cap A) - (P \cap B)$
 - obecný průnik paprsku se scénou je množina intervalů

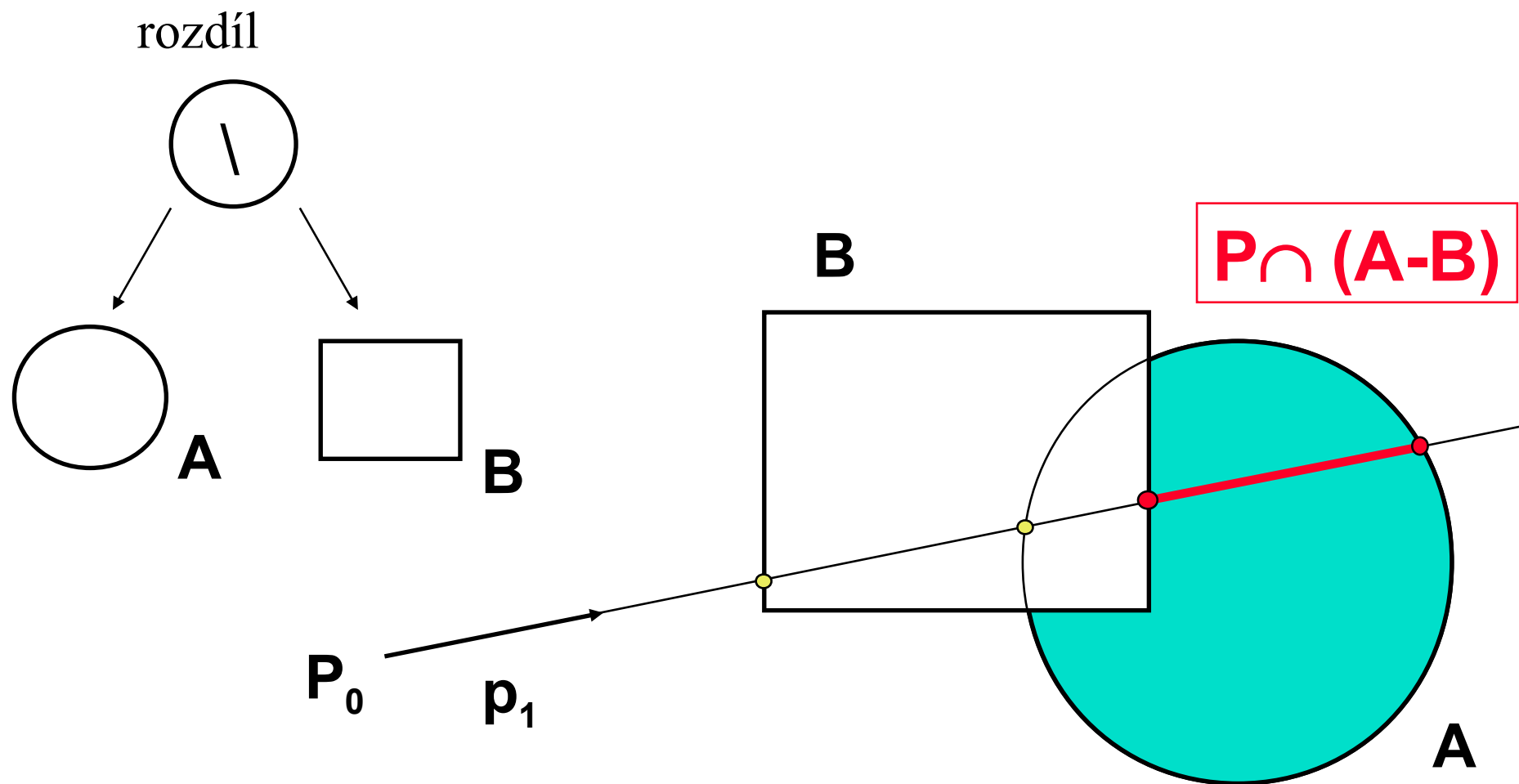


Průsečíky $P \cap A$, $P \cap B$



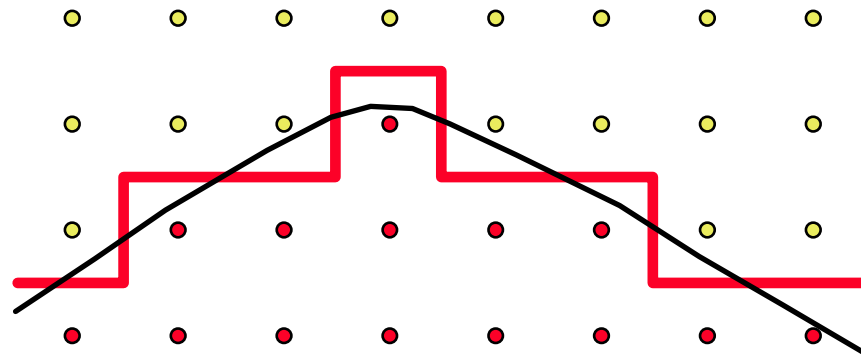


Průsečík $P \cap (A-B)$



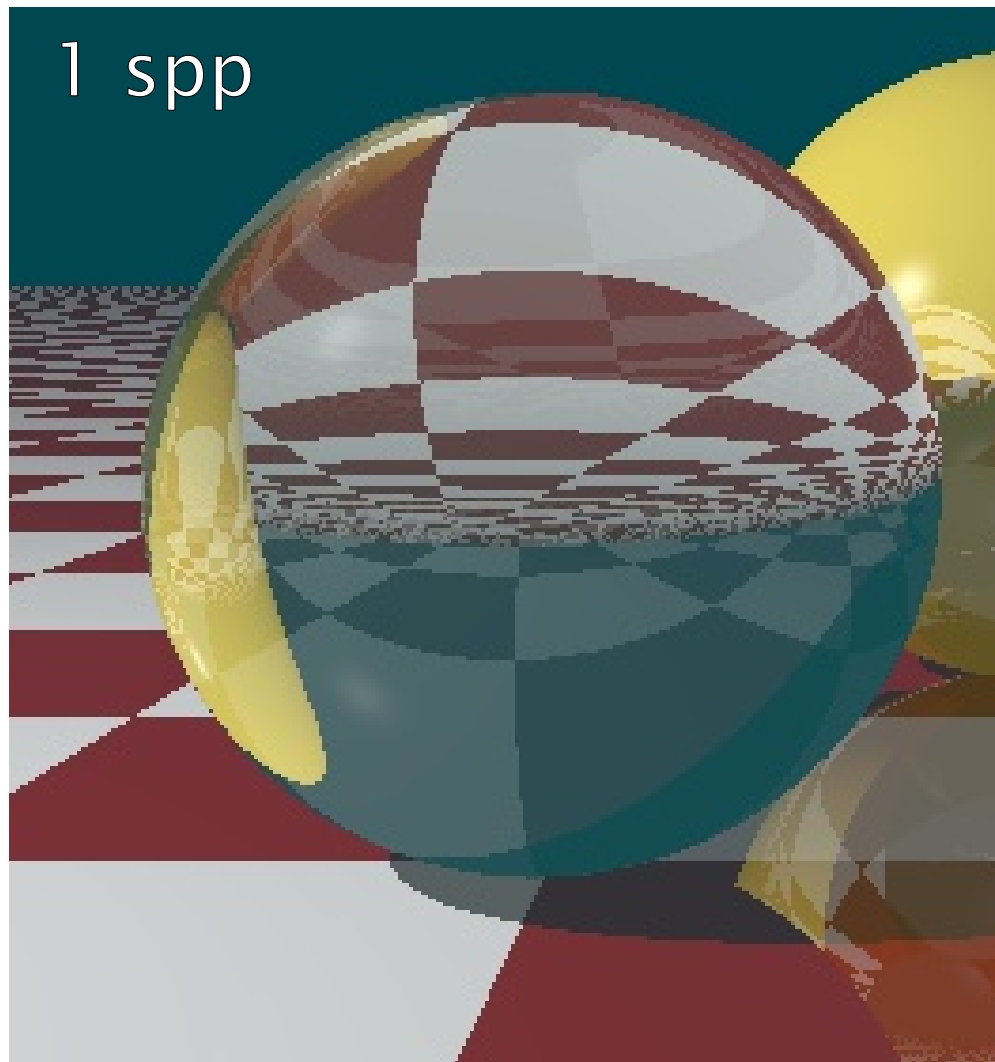


Vyhlazování (anti-aliasing)



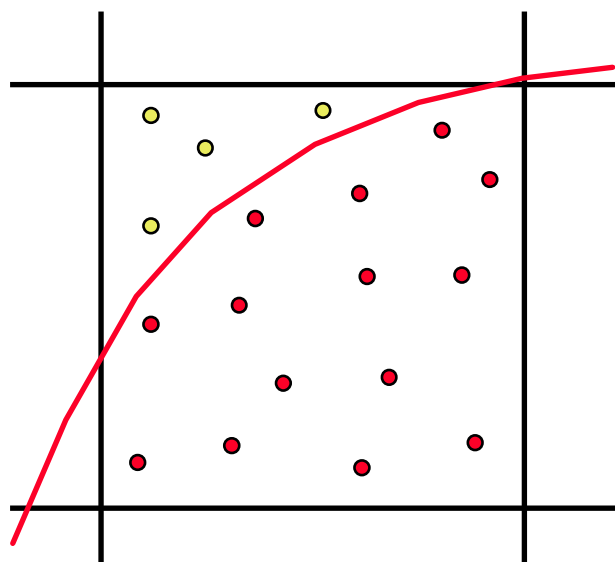
- ◆ pouze jeden paprsek na jeden pixel - vzniká tzv. „alias”
 - zubaté okraje
 - interference
- ◆ **zvětšením rozlišení** se problémem nevyřeší

Ukázka vyhlazování (super-sampling)

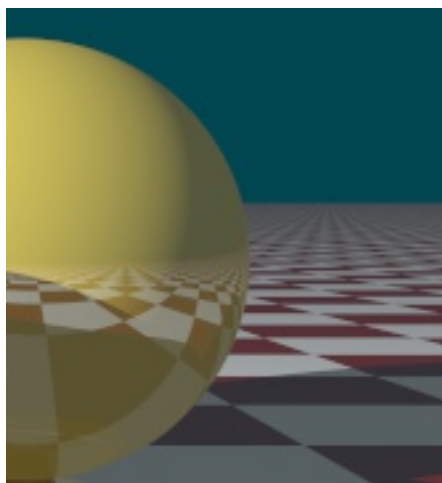




Vícenásobné vzorkování



- ◆ pošleme **více paprsků** jedním pixelem
- ◆ výslednou barvu spočítáme jako **aritmetický průměr**
- ◆ přechody budou **jemnější** (bez zubů)
- ◆ paprsky by měly pokrývat plochu pixelu **rovnoměrně**, ale ne úplně pravidelně!



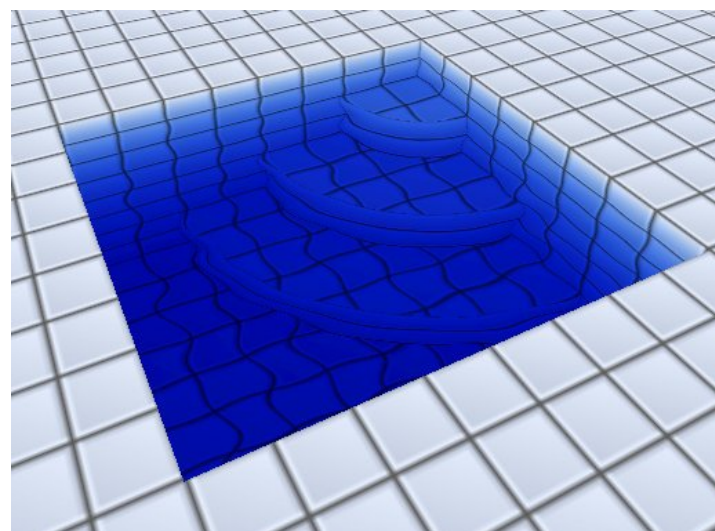
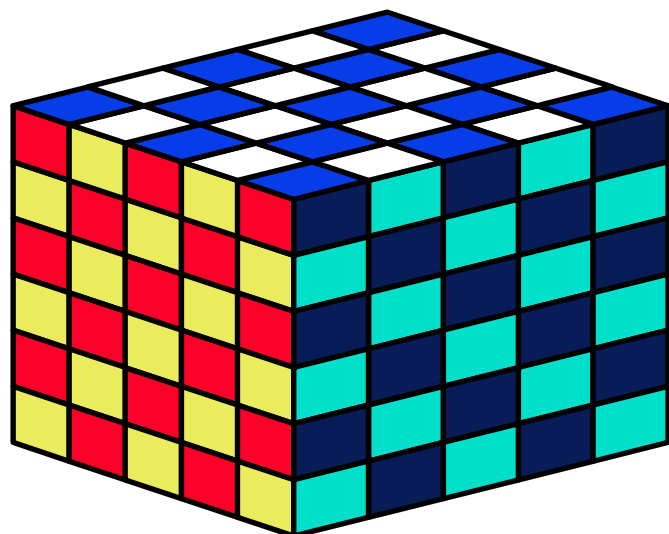


Texture

- ◆ změna **barvy** na povrchu předmětů
- ◆ mohou ovlivňovat též **odrazivost** (k_D a k_S), **normálový vektor**, ...
- ➔ realistické napodobení **fyzikálních vlastností materiálu** (barevný vzorek, mikro- i makro-struktura povrchu)
 - dřevo, kůra pomeranče, leštěný kov, ..
- ➔ nahrazení složité **geometrie** (vlny na vodě, ..)



2D textura



- ◆ pokrývá **povrch** tělesa (jako tapeta)
- ➔ **mapování textury**: $[x, y, z] \rightarrow [u, v]$
- ➔ **vlastní textura**: $[u, v] \rightarrow$ **barva** (normála, ..)

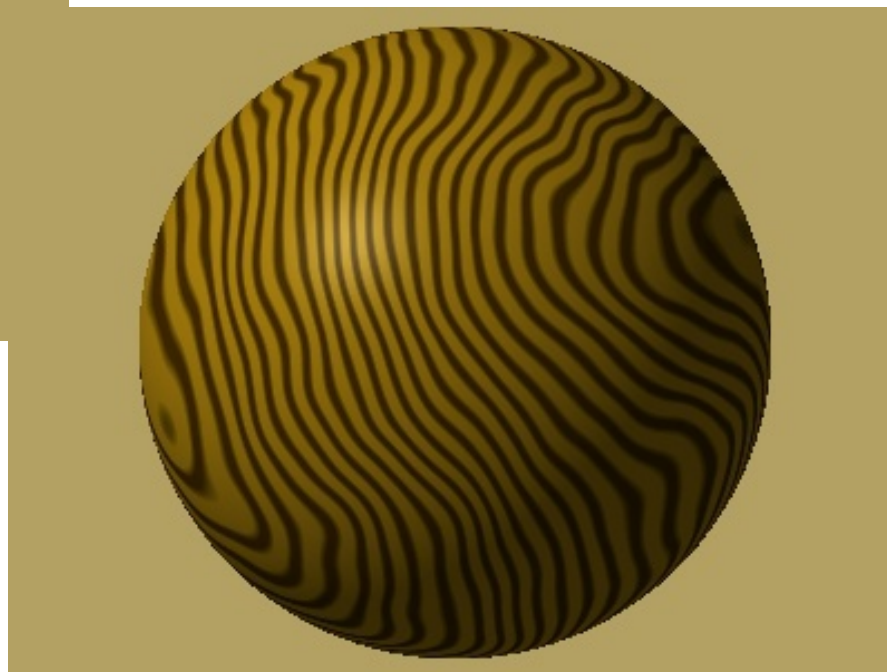
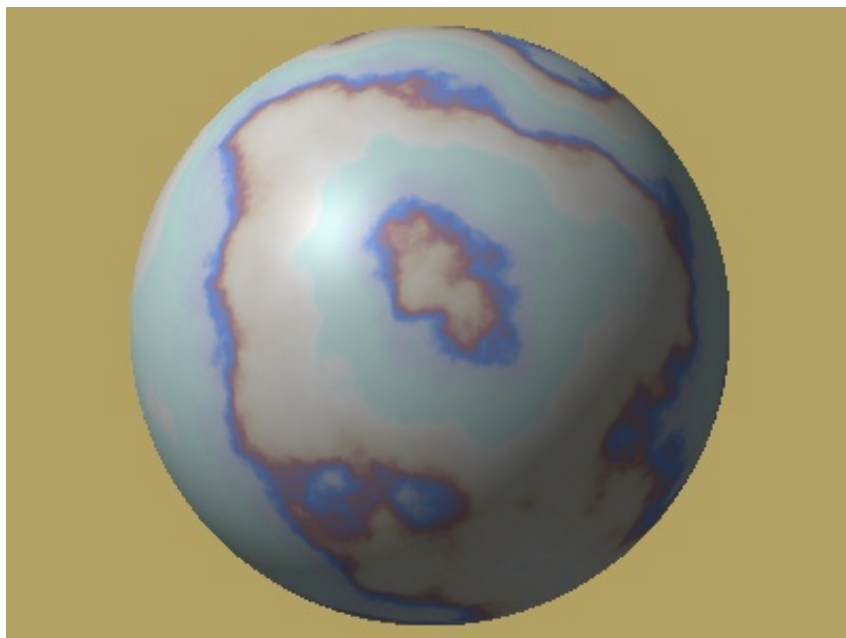


3D texture

- ◆ zachycují změny veličin **uvnitř tělesa**
- ◆ napodobují **vnitřní strukturu materiálu** (dřevo, mramor, ...)
- ➔ není třeba **mapování**
- ➔ **3D textura: $[x, y, z] \rightarrow$ barva** (odrazivost, apod.)
- ▲ často se využívají **3D šumové funkce** (napodobení náhodného vrásnění)



Příklady 3D textur





Konec

Další informace:

- **A. Glassner: *An Introduction to Ray Tracing*, Academic Press, London 1989, 1-31**
- **Jiří Žára a kol.: *Počítačová grafika*, principy a algoritmy, 374-378**