matfyz

Computer Graphics Charles University

Urychlovací metody pro Ray-tracing

© 1996-2016 Josef Pelikán

CGG MFF UK Praha

pepca@cgg.mff.cuni.cz

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

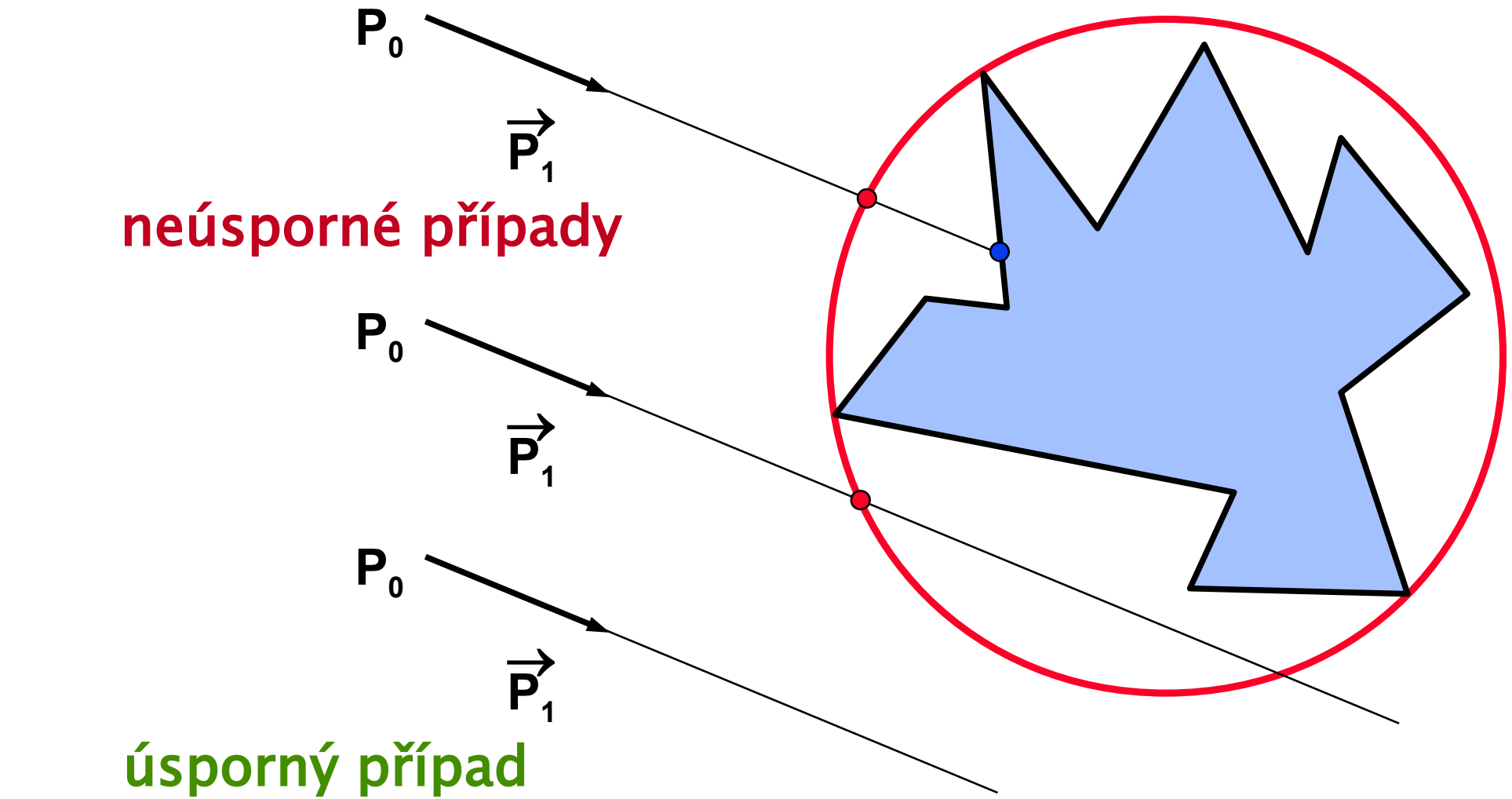
Průsečík paprsku s 3D scénou

- spotřebuje **většinu strojového času** (až 95% podle Whitteda, 1980)
- scéna je složena z **elementárních těles**
  - koule, kvádr, válec, kužel, jehlan, polygon, ..
  - elementární tělesa v CSG
  - počet elementárních těles .. **N**
- klasický algoritmus testuje **každý paprsek** (do hloubky rekurze **H**) s **každým element. tělesem**
  - **O(N)** testů pro jeden paprsek

Klasifikace urychlovacích metod

1. urychlení výpočtu **„paprsek × scéna“**
   - urychlení testu **„paprsek × těleso”**
     - » obalová tělesa, efektivní algoritmy výpočtu průsečíků
   - menší počet testů **„paprsek × těleso”**
     - » hierarchie obalových těles, dělení prostoru (prostorové adresáře), směrové techniky (+2D adresáře)
2. menší **počet testovaných paprsků**
   - » dynamické řízení rekurze, adaptivní vyhlazování
3. **zobecněné paprsky** (dávající více informace)
   - » polygonální svazek paprsků, kužel, ..

Obalové těleso



Obalové těleso

1. výpočet průsečíku je **jednodušší** než u původního tělesa
   - koule, kvádr v obecné nebo osově rovnoběžné poloze, průnik pásů, ..
2. obal by měl **co nejtěsněji obklopovat** původní těleso (pro maximální urychlení)

- **efektivita** obalového tělesa záleží na vhodném kompromisu mezi 1 a 2
  - celková asymptotická složitost zůstává **O(N)**

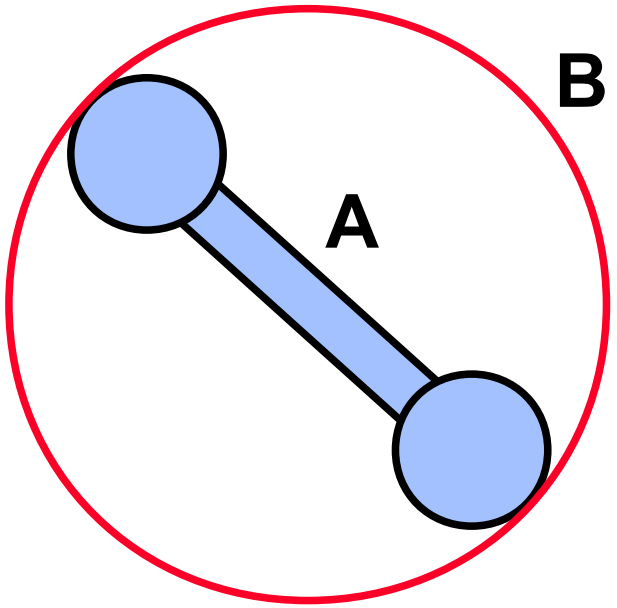
Efektivita obalového tělesa

Očekávaný **průměrný čas výpočtu** průsečíku paprsku s tělesem:

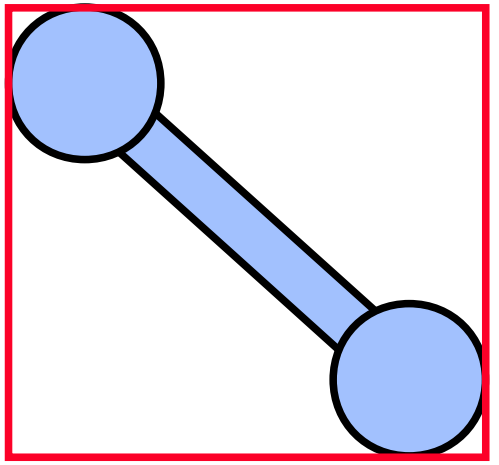
$$B + p \cdot I \quad < \quad I$$

- **I** .. čas výpočtu průsečíku s **původním tělesem**
- **B** .. čas výpočtu průsečíku s **obalovým tělesem**
- **p** .. **pravděpodobnost zásahu** obalového tělesa paprskem (kolik % paprsků protne obalové těleso)

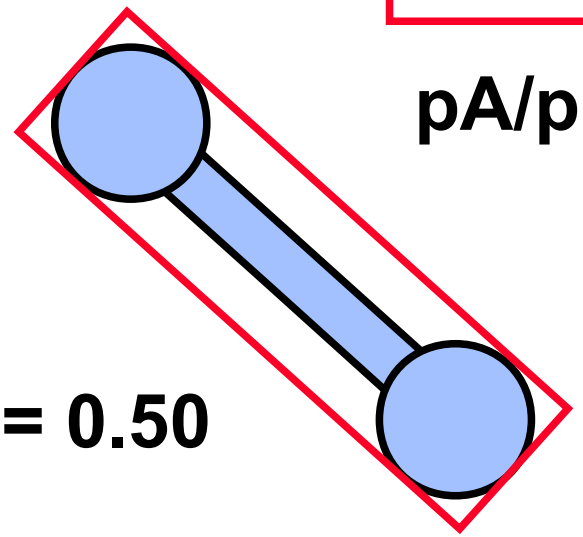
Různě efektivní obaly



pA/pB = 0.20



pA/pB = 0.23



pA/pB = 0.50

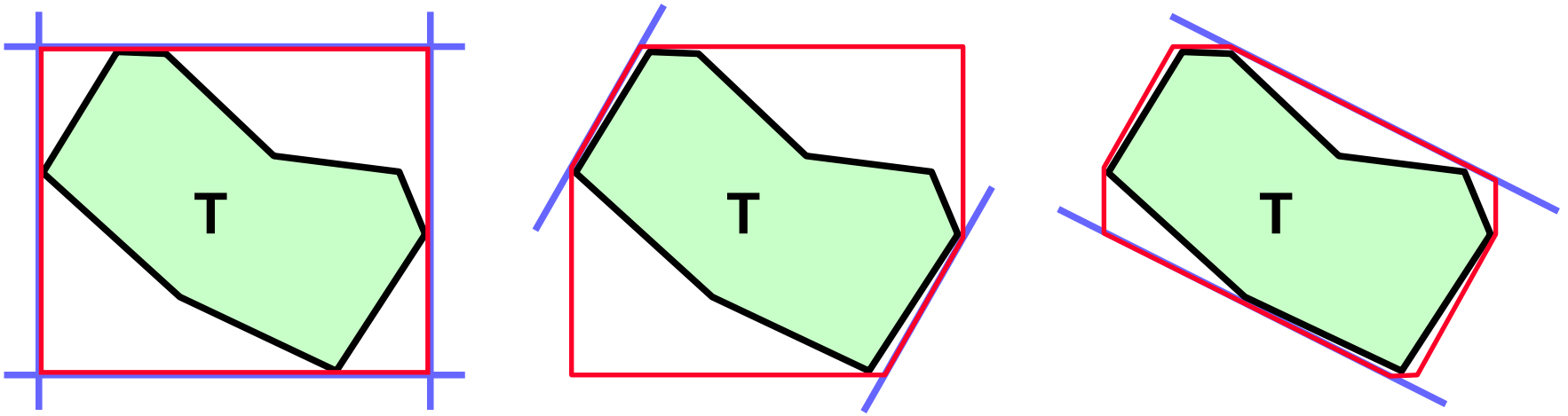
Kombinovaná obalová tělesa

- **lepší aproximace tvaru** původního tělesa
- **sjednocení a průniky** jednodušších obalových těles:

Aproximace konvexního obalu

- výhodné obalové těleso pro **konvexní objekty**
- průnik několika **pásů** („k-dops”)
  - pás je omezen dvěma rovnoběžnými rovinami
  - nutnost efektivního výpočtu konstant **d** a **D**:

$$d = \min_{[x,y,z] \in T} \{ ax + by + cz \}, \quad D = \max_{[x,y,z] \in T} \{ ax + by + cz \}$$

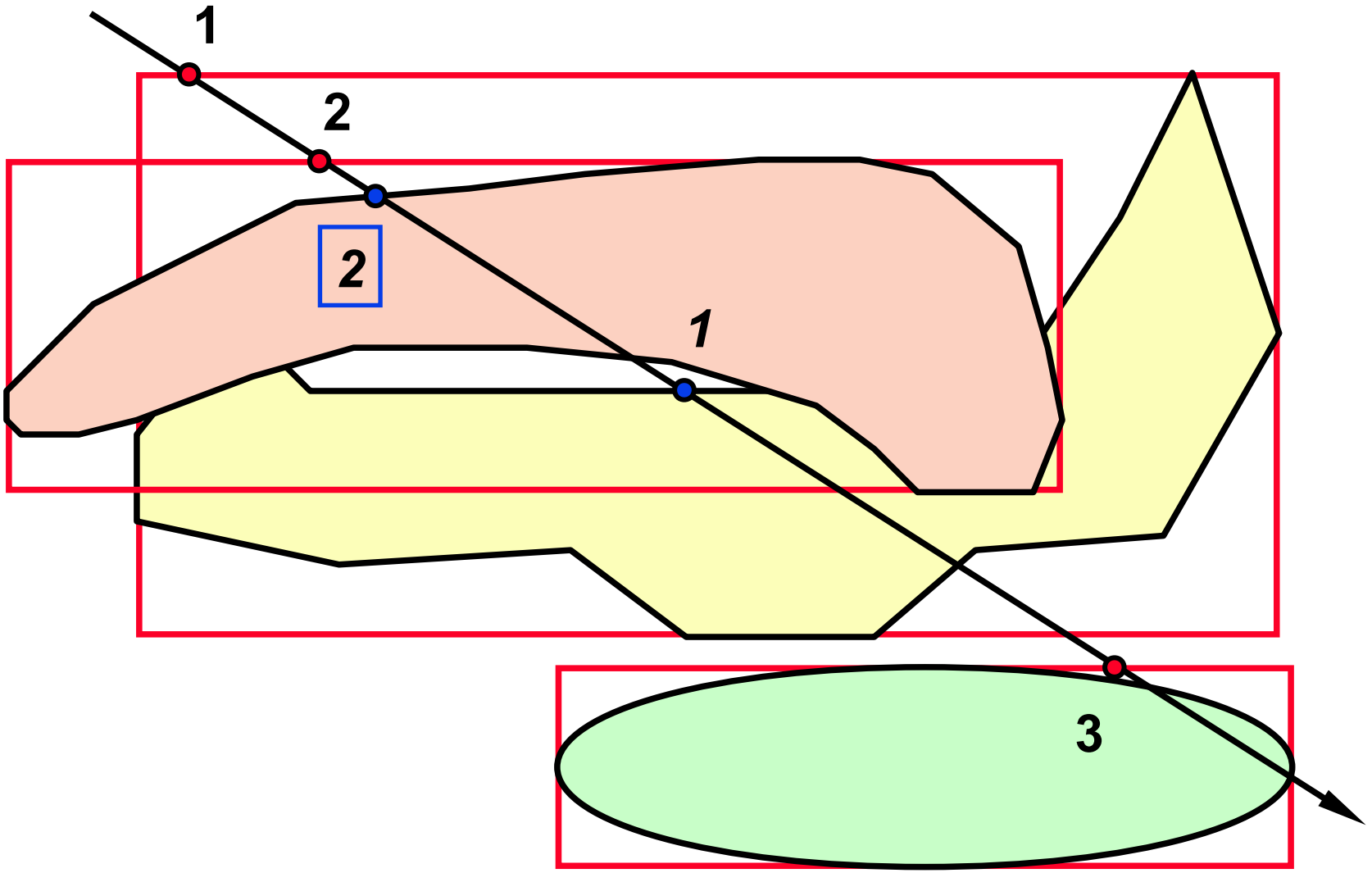


Efektivní implementace

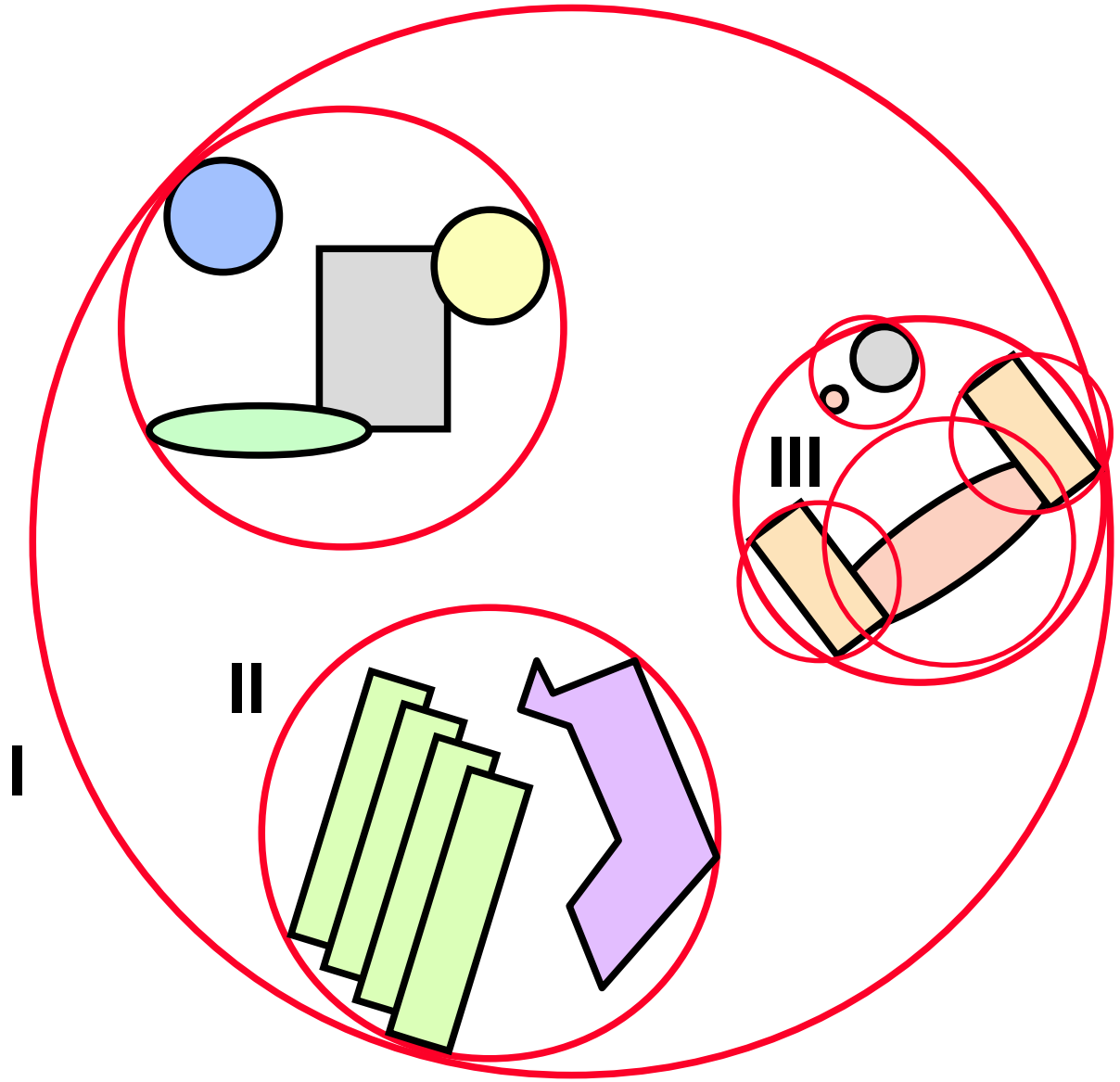
1. výpočítám průsečíky paprsku se **všemi obalovými tělesy**
2. protnutá obalová tělesa seřadím podle **vzdálenosti od počátku paprsku**
3. objekty testuji v tomto pořadí (od nejbližších ke vzdálenějším)

➡ skončím, jestliže jsem **našel průsečík** a otestoval všechny objekty sahající před něj

Efektivní implementace



Hierarchie obalových těles (BVH)



Hierarchie obalových těles

- v ideálním případě **snižuje asymptotickou složitost** na **O(log N)**
- vyplatí se zejména u **dobře strukturovaných scén**
  - množství dobře oddělených malých objektů
  - přirozená implementace v CSG reprezentaci (prořezávání CSG stromu)
- možnost **automatické konstrukce** hierarchie
  - inkrementální algoritmus
- v orientaci „AABB“ je to přesně **R-tree** (Guttman)
  - viz databázové vyhledávací datové struktury

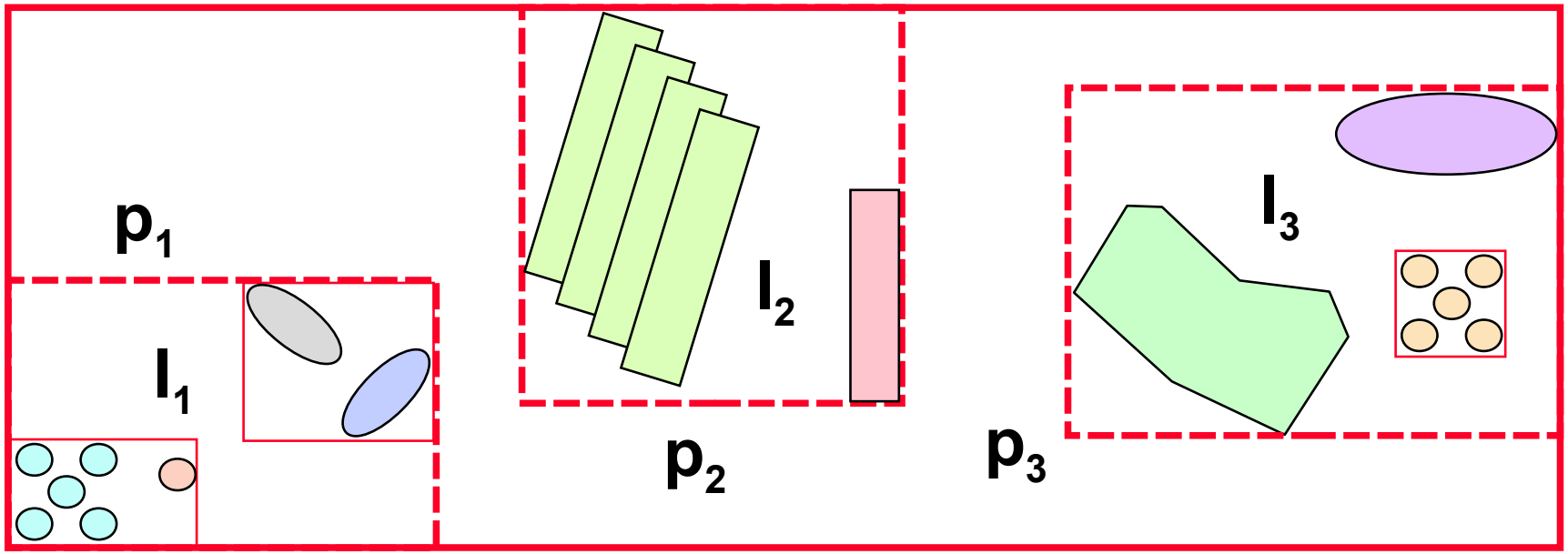
Efektivita hierarchie

$$K \cdot B + \sum_{i=1}^{K} p_i \, l_i \;\; ? \;\; < \;\; \sum_{i=1}^{K} l_i$$

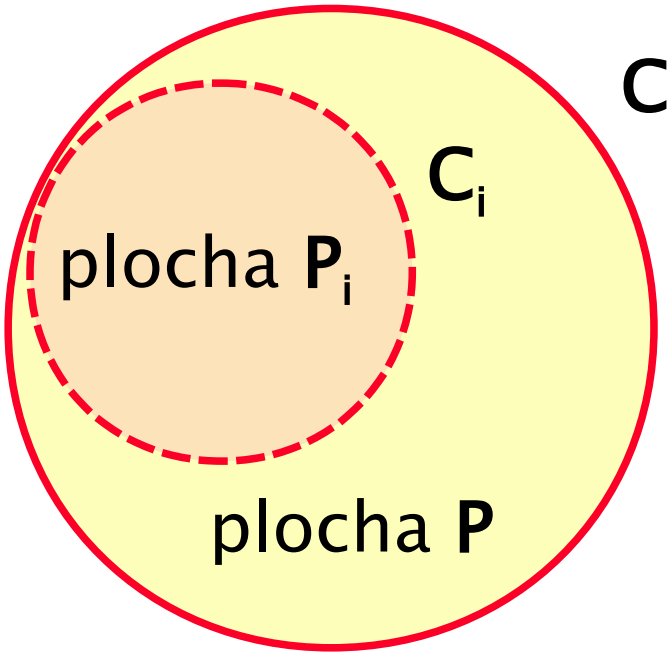
B .. čas výpočtu průsečíku s obalovým tělesem

p_i .. pravděpodobnost zásahu i-tého obalového tělesa

l_i .. čas výpočtu pro objekty uzavřené v i-tém obalovém tělese



Efektivita hierarchie



P(d), $\mathbf{P_i(d)}$ .. plocha průmětu tělesa ze směru **d**

S, $\mathbf{S_i}$ .. povrch tělesa

Pro jeden směr pohledu **d**:

$$p_i = Pr(\text{zásah } C_i \mid \text{zásah } C) = \frac{P_i(d)}{P(d)}$$

Pro všechny směry a **konvexní tělesa**:

$$p_i = \frac{\int P_i(d)\ dd}{\int P(d)\ dd} = \frac{S_i}{S}$$

Inkrementální konstrukce

1. vytvořím **prázdnou hierarchii** (kořen stromu)
2. vezmu nový objekt a **přidám ho do kořene**
   - opravím obalové těleso kořene
3. vyberu **nejvýhodnější možnost** (v rámci obal.t.):
   - objekt bude samostatný (bez vlastního obalu)
   - objekt bude mít sám nové obalové podtěleso
   - objekt přidám do existujícího obalového podtělesa

➡ záleží na **pořadí přidávání** objektů !
   - setřídění podle 3D polohy a náhodné zamíchání

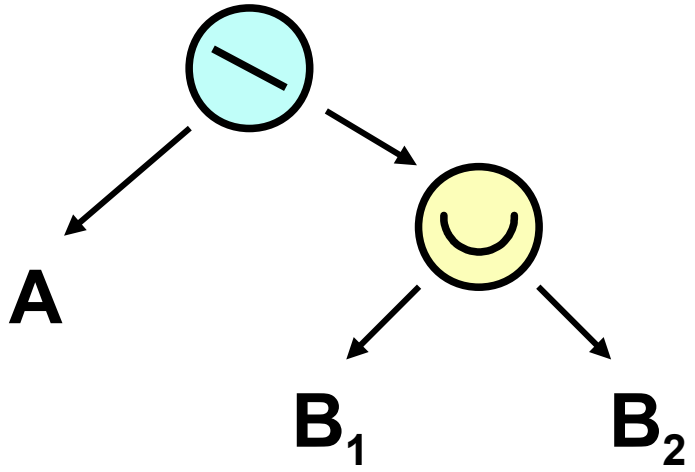
Hierarchické obalové systémy

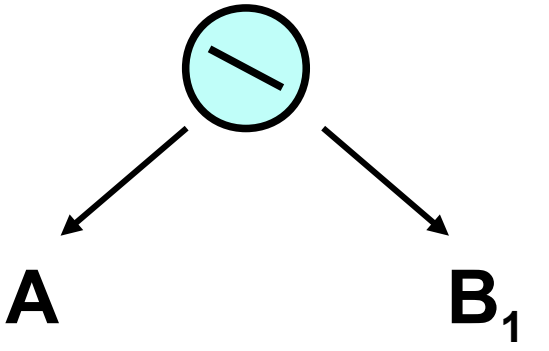
- „**Sphere tree**” (Palmer, Grimsdale, 1995)
  – jednoduchý test i transformace, horší aproximace
- „**AABB tree**”, „**R-tree**“ (Held, Klosowski, Mitchell, '95)
  – jednoduchý test, složitější transformace
- „**OBB tree**” (Gottschalk, Lin, Manocha, 1996)
  – jednoduchá transformace, složitější test, slušná aproximace
- „**K-dop tree**” (Klosowski, Held, Mitchell, 1998)
  – složitější transformace a test, výborná aproximace

„Prořezávání” CSG stromu

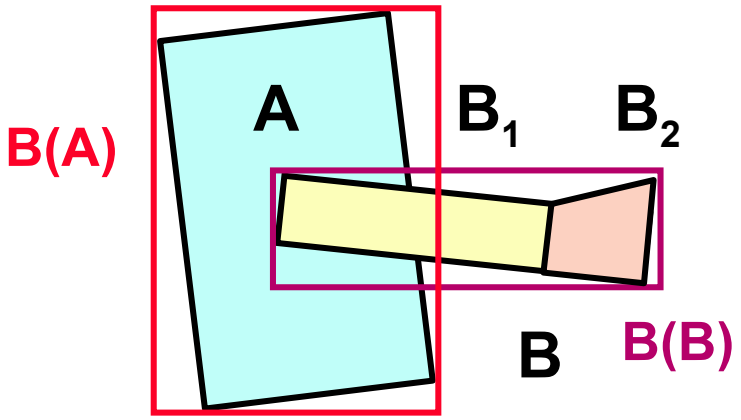
- efektivní především pro **subtraktivní množinové operace** (průnik, rozdíl)
- primární obalová tělesa jsou přiřazena (omezeným) **elementárním tělesům**
  - velikost se většinou určuje analyticky
- obalová tělesa se pomocí množinových operací **propagují směrem ke kořeni**
- u argumentů **subtraktivních operací** se mohou obalová tělesa **zmenšovat**

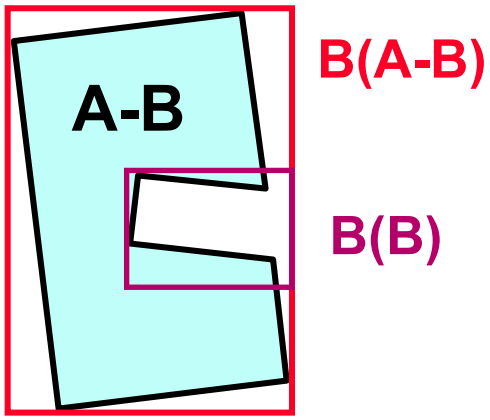
„Prořezávání” CSG stromu





$B(A-B) \cap B(B_2) = 0$

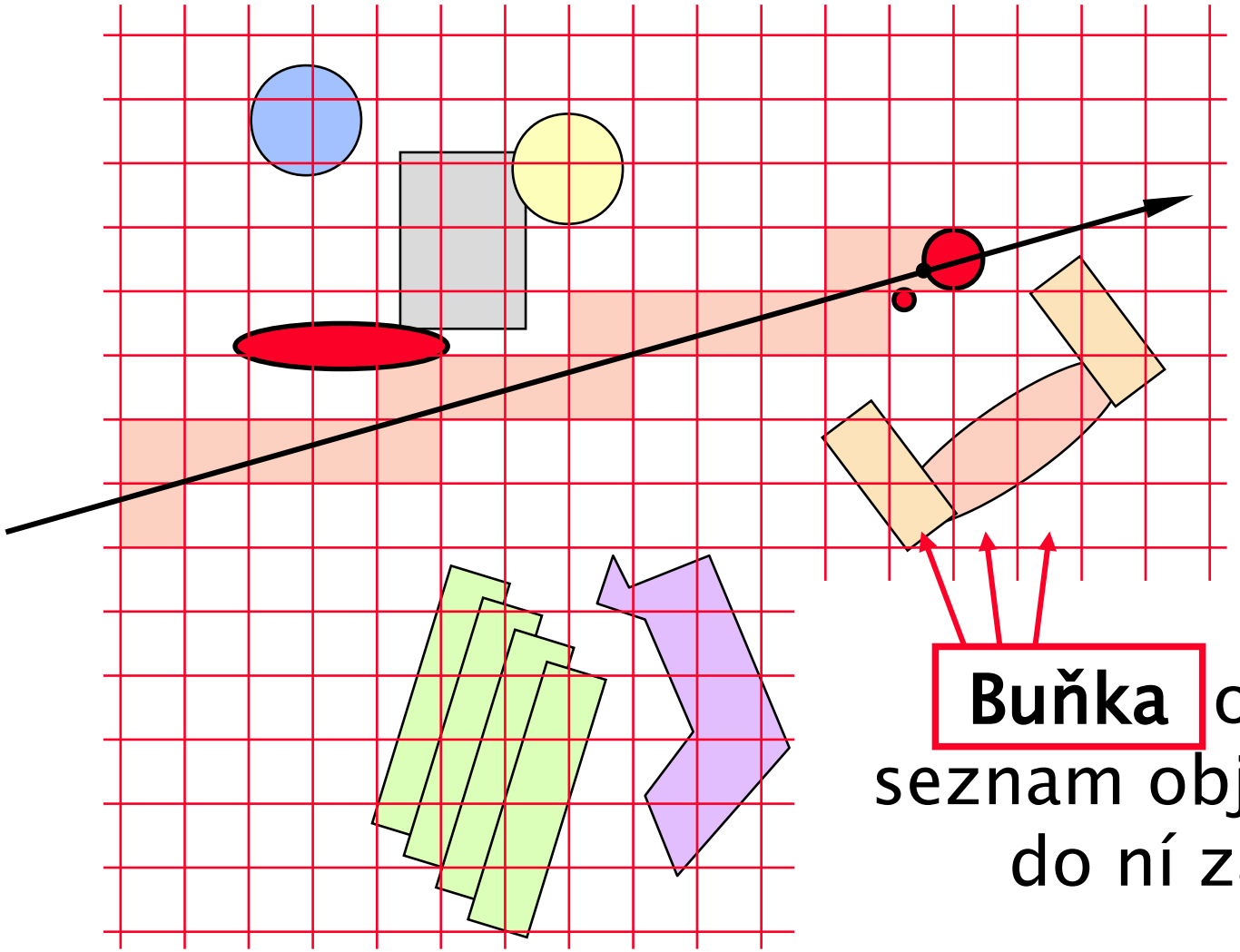




Dělení prostoru (prostorové adresáře)

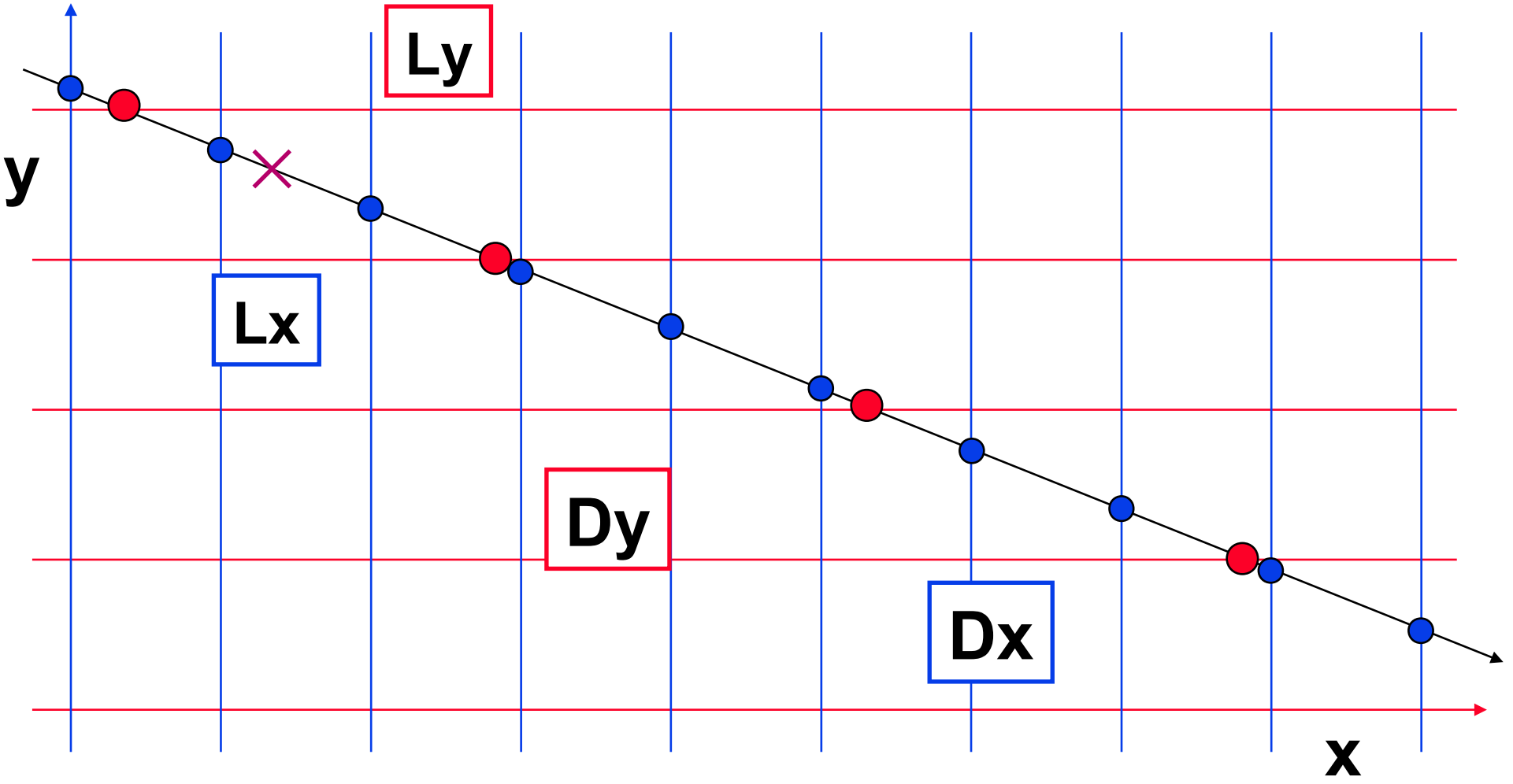
- **uniformní dělení** (stejně velké buňky)
  - \+ jednoduchý průchod
  - – mnoho kroků výpočtu
  - – velký objem dat
- **neuniformní dělení** (většinou adaptivní)
  - \+ méně kroků výpočtu
  - \+ menší objem dat
  - – složitější implementace datové struktury i algoritmu procházení

Uniformní dělení prostoru



Buňka obsahuje seznam objektů, které do ní zasahují

Průchod sítí buněk (3D DDA)



Průchod sítí buněk (3D DDA)

- **paprsek:** $P_0 + t \cdot \vec{P_1}$ pro $t > 0$

1. pro daný směr P_1 se předem spočítají **konstanty** **Dx**, **Dy**, **Dz**:
   - vzdálenost mezi sousedními průsečíky paprsku se sítí rovnoběžných rovin (kolmých na osy **x**, **y**, **z**)

1. pro bod P_0 se určí **počáteční buňka [i, j, k]** a hodnoty proměnných **t**, **Lx**, **Ly**, **Lz**:
   - parametr polopřímky **t**, vzdálenosti k nejbližším průsečíkům paprsku se stěnami

Průchod sítí buněk (3D DDA)

2. **zpracování buňky** [**i**, **j**, **k**] (výpočet průsečíků)

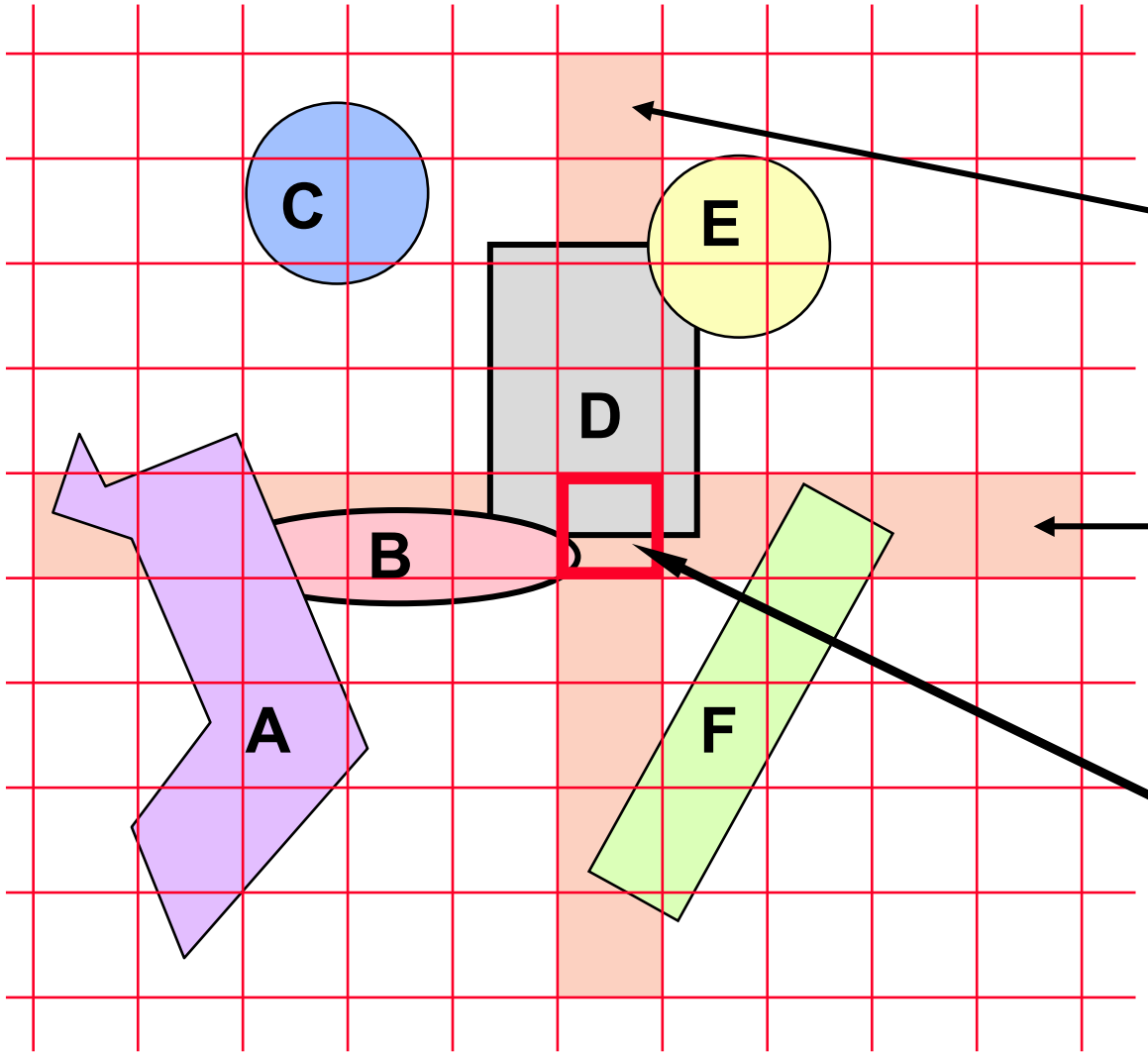
3. postup do **sousední buňky**:
   - **D = min {Lx,Ly,Lz};** /* předpoklad: **D = Lx** */
   - **Lx = Dx; Ly = Ly - D; Lz = Lz - D;**
   - **i = i ± 1;** /* podle znaménka $\mathbf{P_{1x}}$ */

4. **koncové podmínky**:
   - našel jsem nejbližší průsečík paprsku se scénou, a ten leží v aktuální buňce
   - nová buňka leží mimo oblast scény

Separace dat podle dimenzí

- redukce velkých **paměťových nároků** $O(K^3)$
  - kompromis mezi efektivitou a úsporou paměti
- počet testovaných těles se může mírně **zvětšit**
  - některá tělesa se mohou testovat zbytečně, ale žádné důležité těleso nemůže být vynecháno
  - velká protáhlá tělesa nejsou výhodná
- seznam relevantních těles se spočítá jako **průnik** příslušných řádkových a sloupcových seznamů
  - efektivní implementace: bitové vektory, uspoř. seznamy

Separace dat podle dimenzí

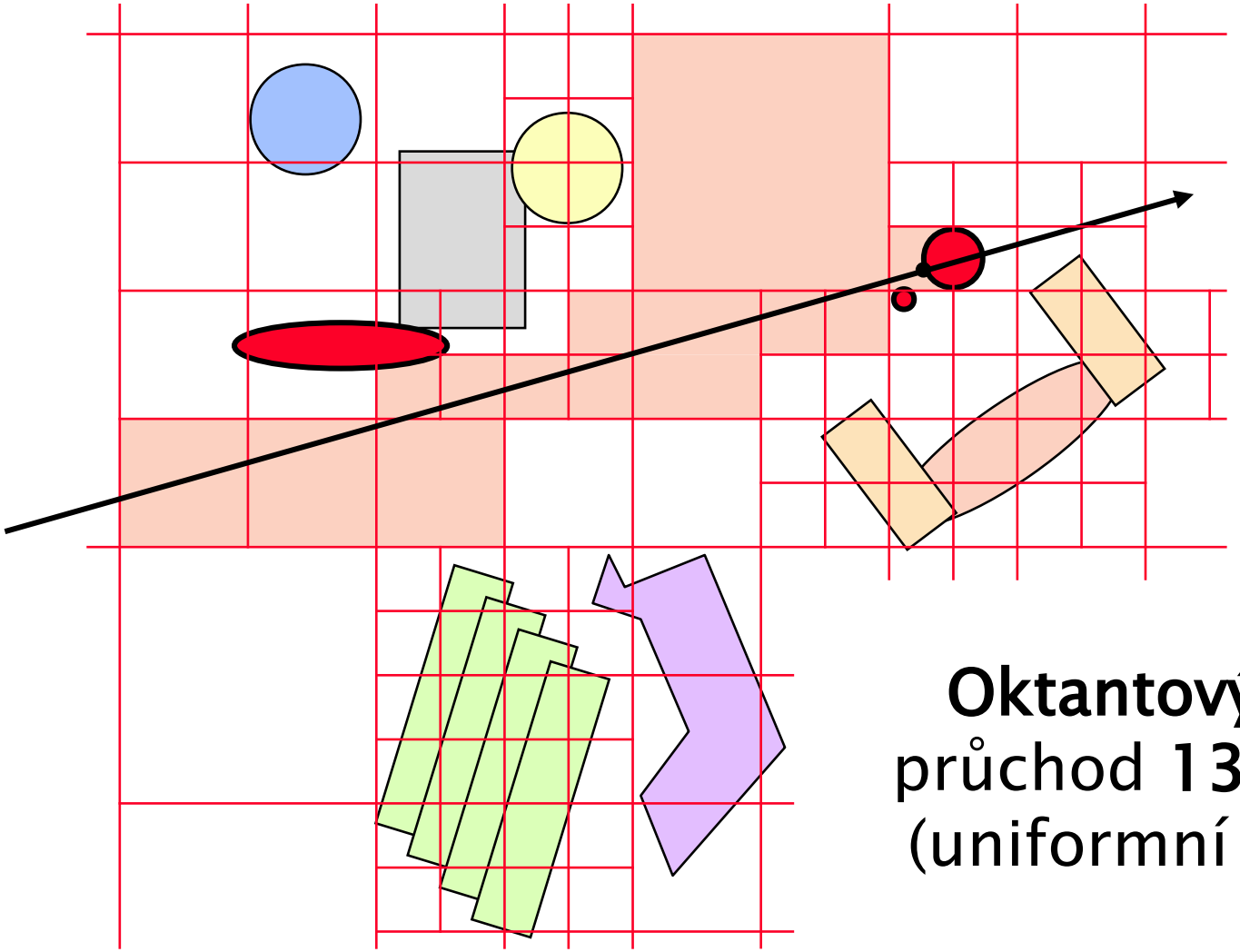


sloupec:
[B, D, E, F]

řádek:
[A, B, D, F]

průnik:
[B, D, F]

Neuniformní dělení prostoru

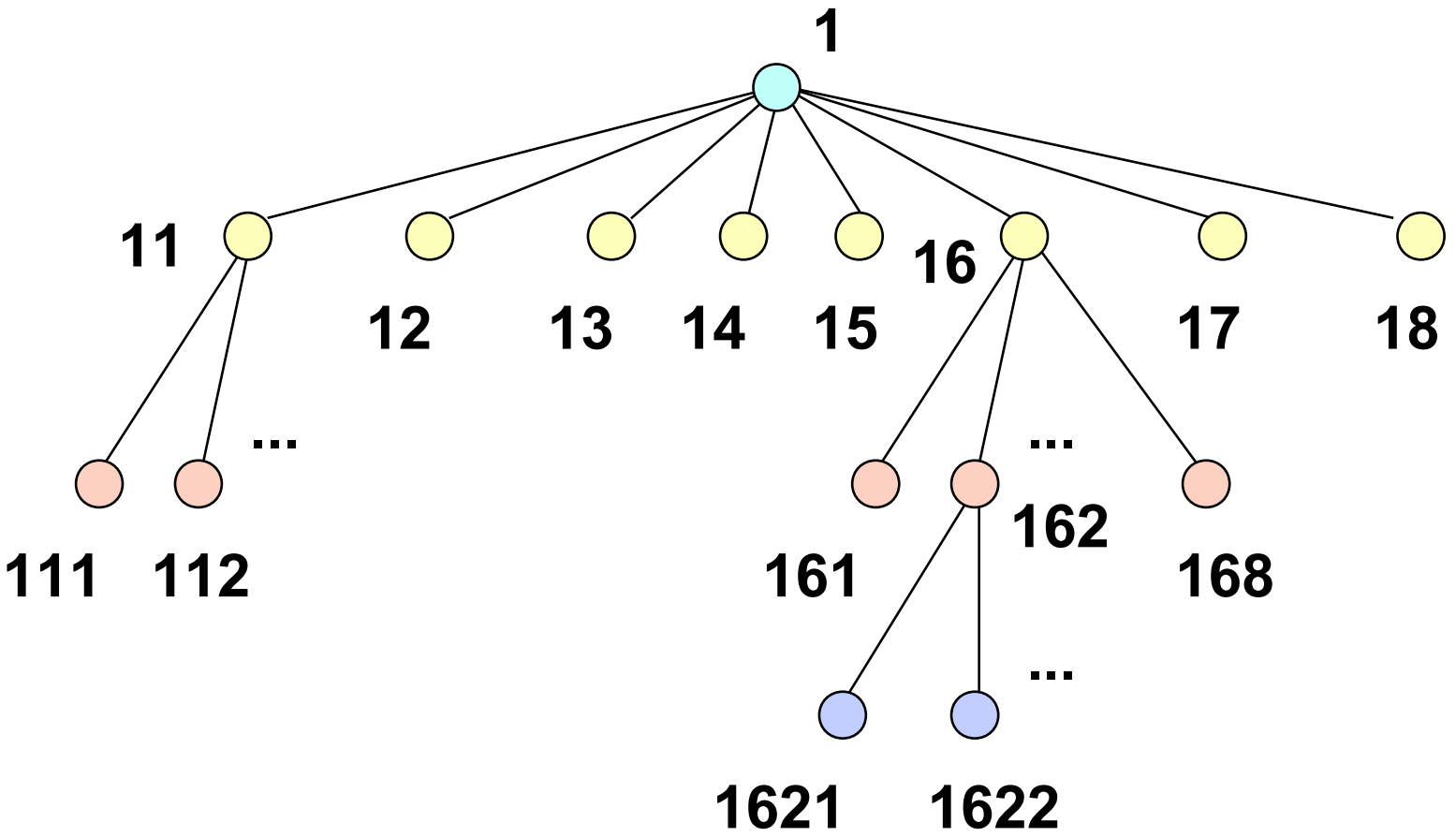


Oktantový strom:
průchod **13 buňkami**
(uniformní pole: **17**)

Geometrie adaptivního dělení

- **oktantový strom** s půlením stran
  - reprezentace pomocí ukazatelů, implicitní reprezentace nebo hašovací tabulkou (Glassner)
- **KD-strom** (Bentley) [dříve „osově orientovaný BSP“]
  - buňky se dělí v polovině, cyklicky se střídají směry dělení
  - buňky se dělí adaptivně, i směry dělení jsou adaptivní
- [obecný **BSP strom**]
  - dělicí roviny mají libovolnou orientaci

Oktantový strom podle Glassnera



Oktantový strom podle Glassnera

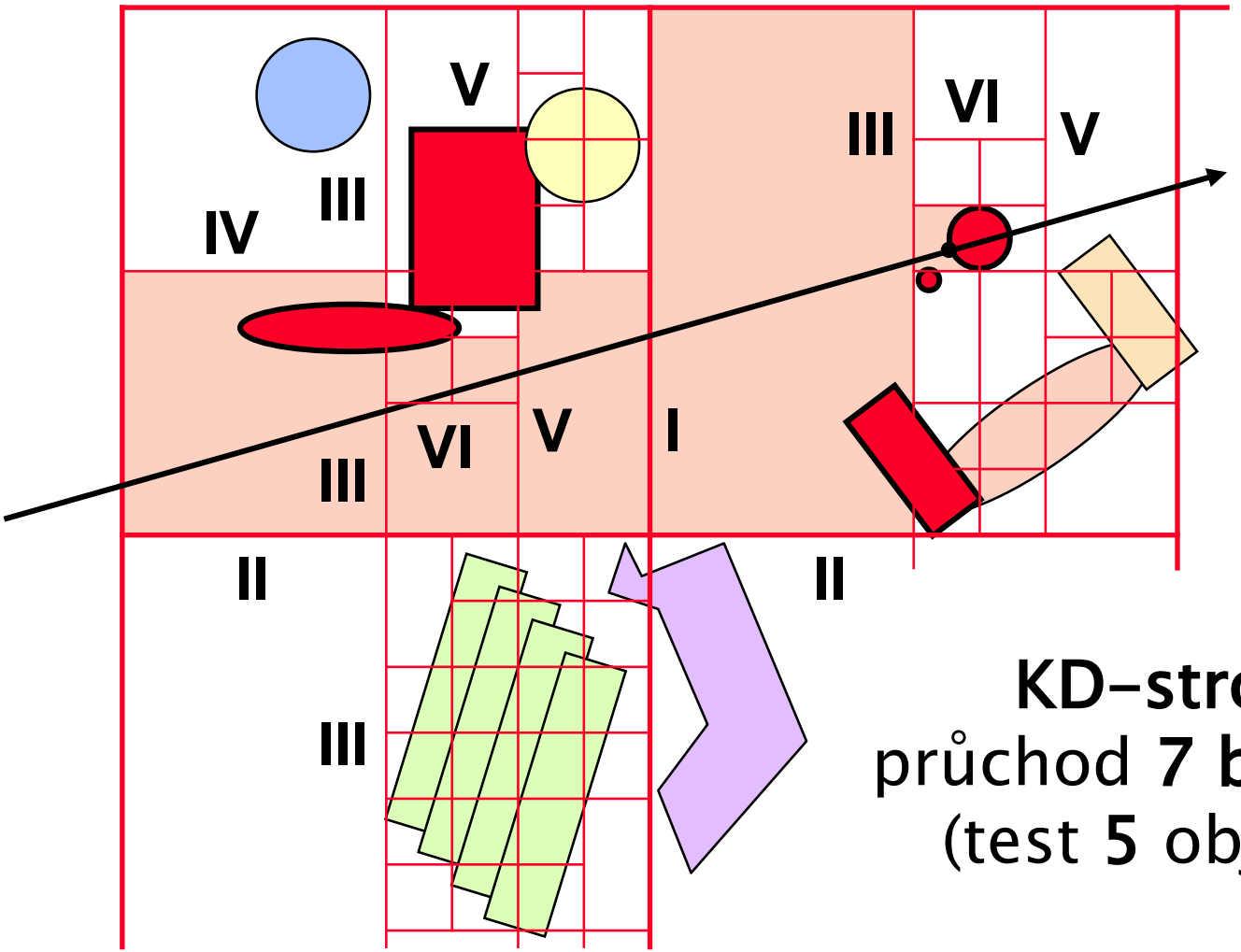
- jednotlivé buňky jsou **hierarchicky očíslovány**
  - kořen .. **1**
  - jeho potomci .. **11** až **18**, .. atd.
  - každý voxel má přiřazen kód bez ohledu na to, zda je listem aktuálního oktantového stromu nebo ne

- skutečné **listy stromu** se ukládají do řídké **hašovací tabulky**
  - příklad hašovací fce: **Kód mod VelikostTabulky**

Průchod stromem (Glassner)

- bod ležící na paprsku .. **[x, y, z]**
  - umím pro něj najít kód voxelu .. **[1 – 8]^k**
- při konstrukci kódu hledám v hašovací tabulce všechny **prefixy**
  - nalezený prefix je listem obsahujícím bod **[x, y, z]**
- po zpracování buňky stromu pokračuji **posunutím bodu [x, y, z]** ve směru paprsku (**P_1**)
  - pokračuji opět lokalizací nového bodu, ...

KD-strom



KD-strom:
průchod **7 buňkami**
(test **5** objektů)

Kritéria adaptivního dělení

1. omezení **počtu těles** i **hloubky dělení**
   - rozděl buňku, zasahuje-li do ní více než **M těles** (např. **M = 1 .. 5**)
   - **maximální úroveň dělení** je **K** (např. **K = 3 .. 5**)

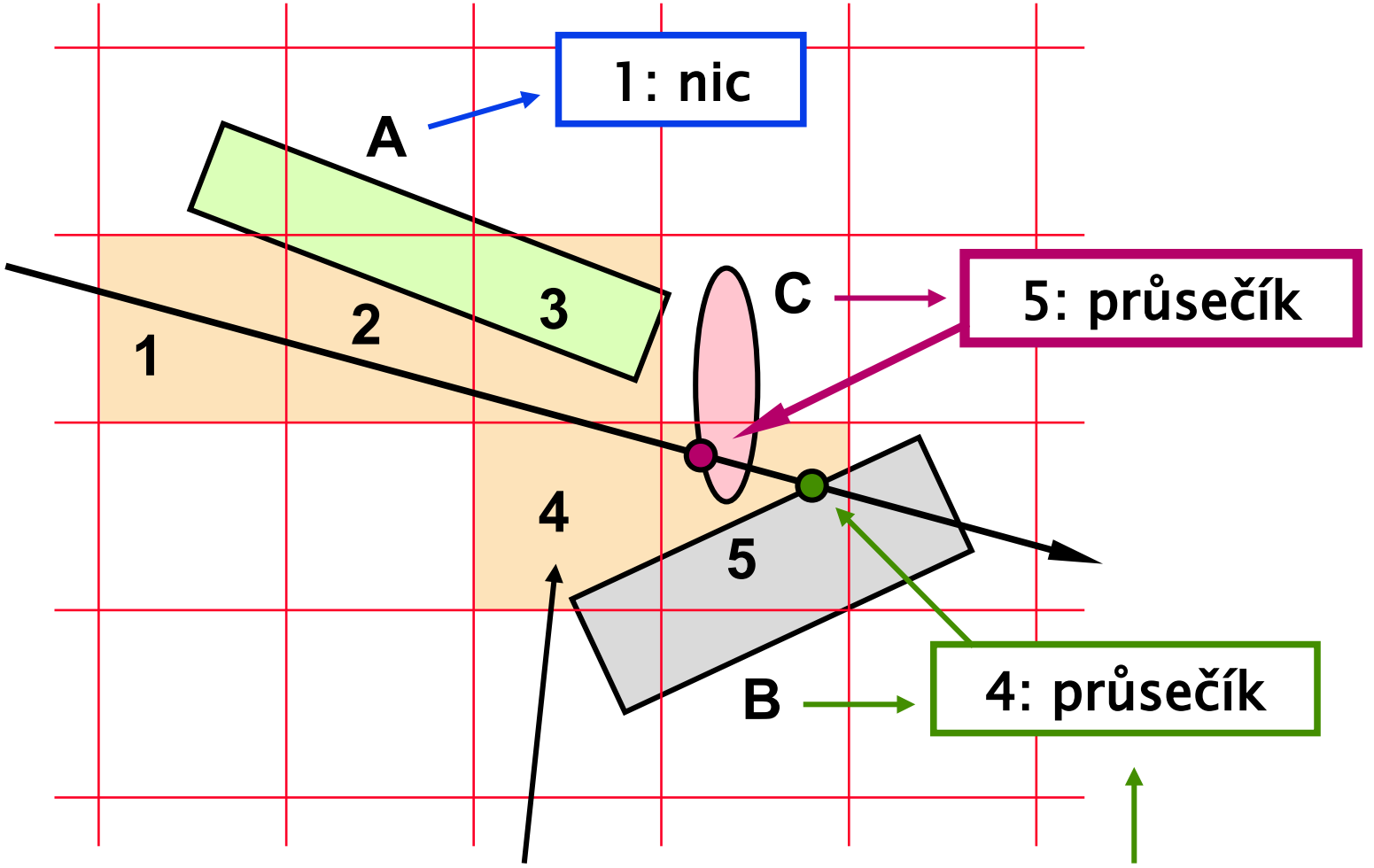
2. omezení **počtu těles** a **spotřeby paměti**

   místo omezení úrovně dělení:
   - dělení se ukončí při zaplnění **vyhrazeného úseku** paměti
   - při dělení je nutné postupovat **do šířky** (fronta kandidátů na dělení)

Průchod adaptivními strukturami

- posunuji se po paprsku a hledám sousední buňku vždy až **od kořene** (viz Glassnerova metoda)
- **přípravná fáze**: průchod stromem a rozdělení paprsku na intervaly
  - intervaly parametru **t** přiřazené jednotlivým buňkám, kterými budu procházet
- **pomocné údaje** v dat. strukturách (à la „finger tree”)
  - ukazatele na sousední buňky (na stejné úrovni ve stromu)
- rekurzivní **průchod do hloubky** s haldou
  - seznam nejnadějnějších sektorů v haldě

Schránka („mailbox”)

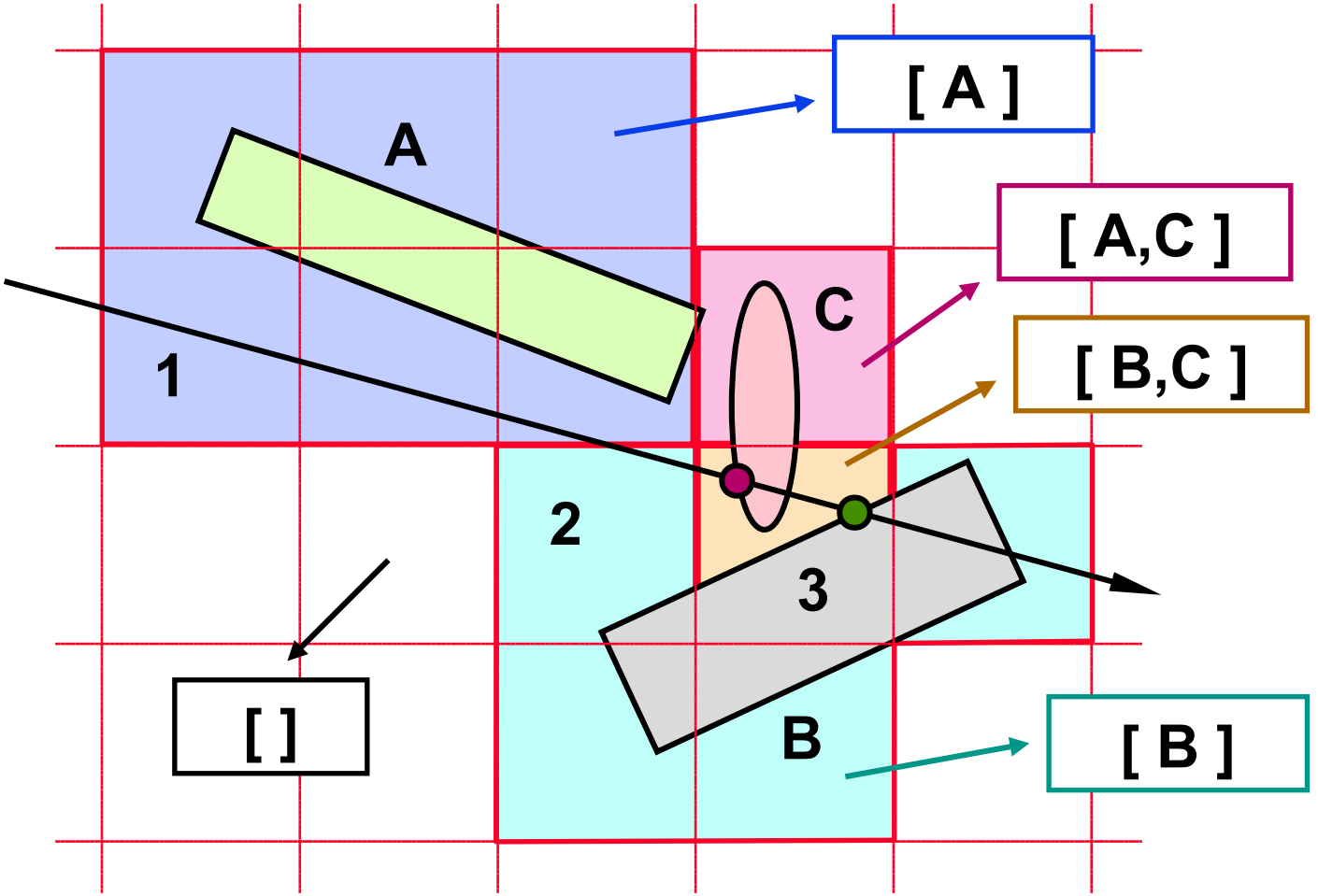


Průsečík musí ležet v aktuální buňce (jinak ho odložím)

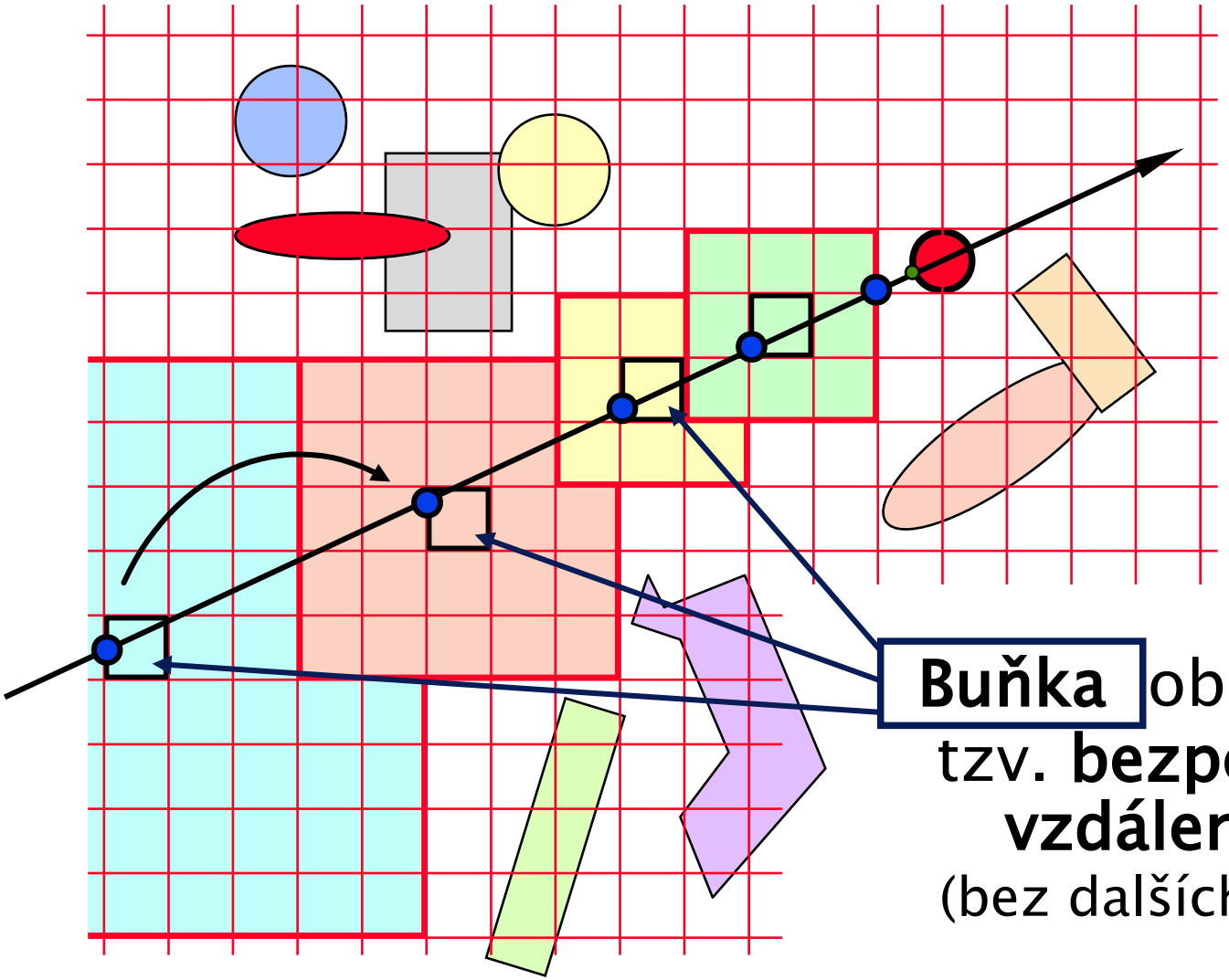
Abstraktní dělení prostoru

- **není třeba testovat** (ani procházet!) seznamy, které jsem již testoval
- seznam musím procházet až v takové buňce, do které zasahuje **jiná** (větší) **množina těles**
- buňky mohou **sdílet shodné seznamy těles**
  - otestované seznamy **označuji** zvláštním příznakem
  - procházím pouze **neoznačené** seznamy
  - na úrovni těles používám techniku **schránek**

Abstraktní dělení prostoru



Makrobuňky (M. Šrámek)

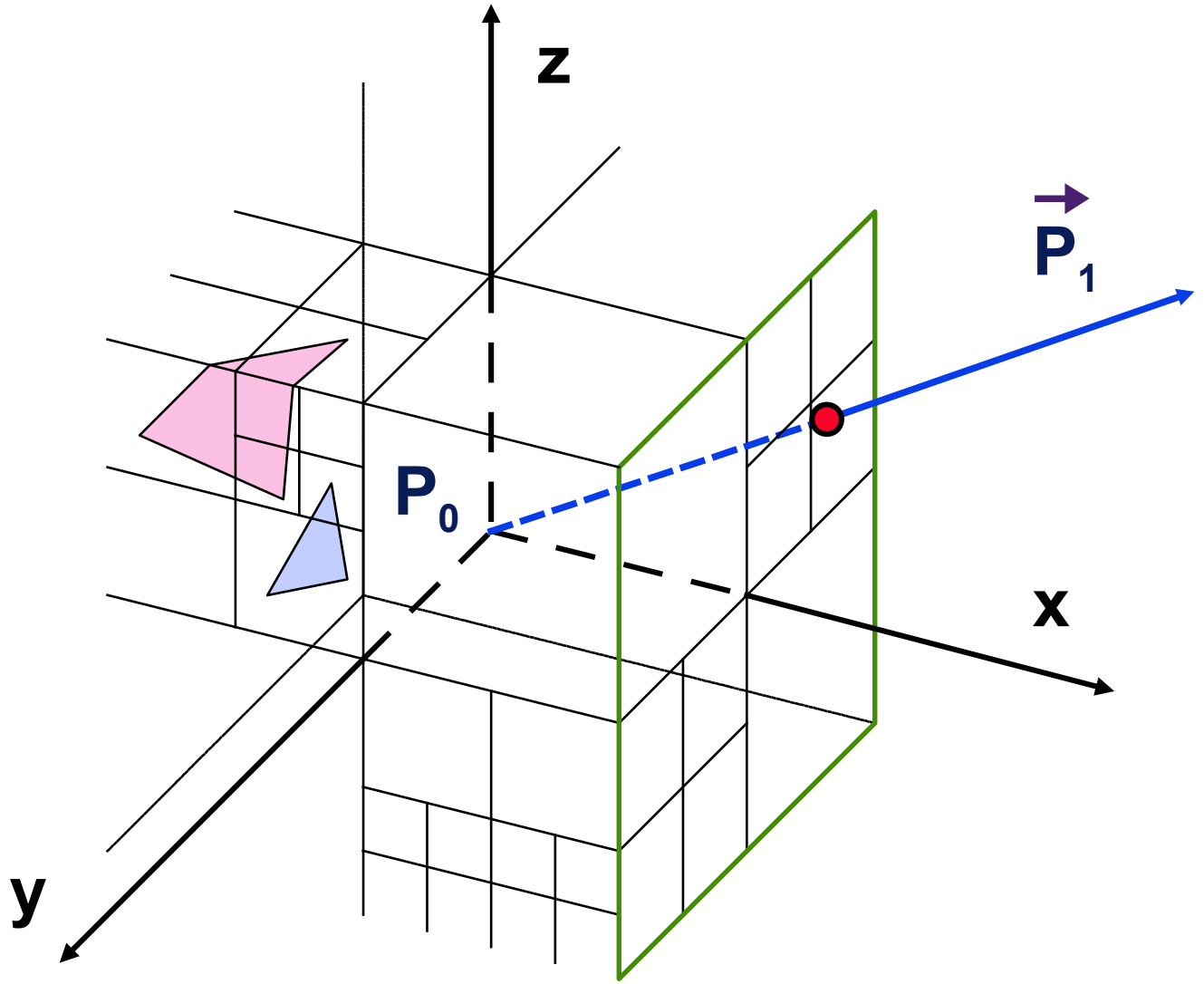


Buňka obsahuje tzv. **bezpečnou vzdálenost** (bez dalších těles)

Směrové urychlovací techniky

- metody využívající **směrové krychle**:
- **světelný buffer**
  – urychluje stínovací paprsky k bodovým zdrojům
- **koherence paprsků**
  – urychluje všechny sekundární paprsky
- 5D **klasifikace paprsků**
- **adresář v průmětně** (předvýpočet viditelnosti)
  – urychluje pouze primární paprsky

Směrová krychle (adaptivní síť)



Směrová krychle

- **orientována** rovnoběžně s osami **x**, **y**, **z**
- jednotlivé stěny jsou rozděleny na **buňky**
  - uniformní nebo adaptivní dělení
  - každá buňka obsahuje seznam relevantních objektů (mohou být navíc setříděny vzestupně podle vzdálenosti od středu krychle)
- ➡ při uniformním dělení lze pro urychlení využít **HW výpočtu viditelnosti** (z-buffer)

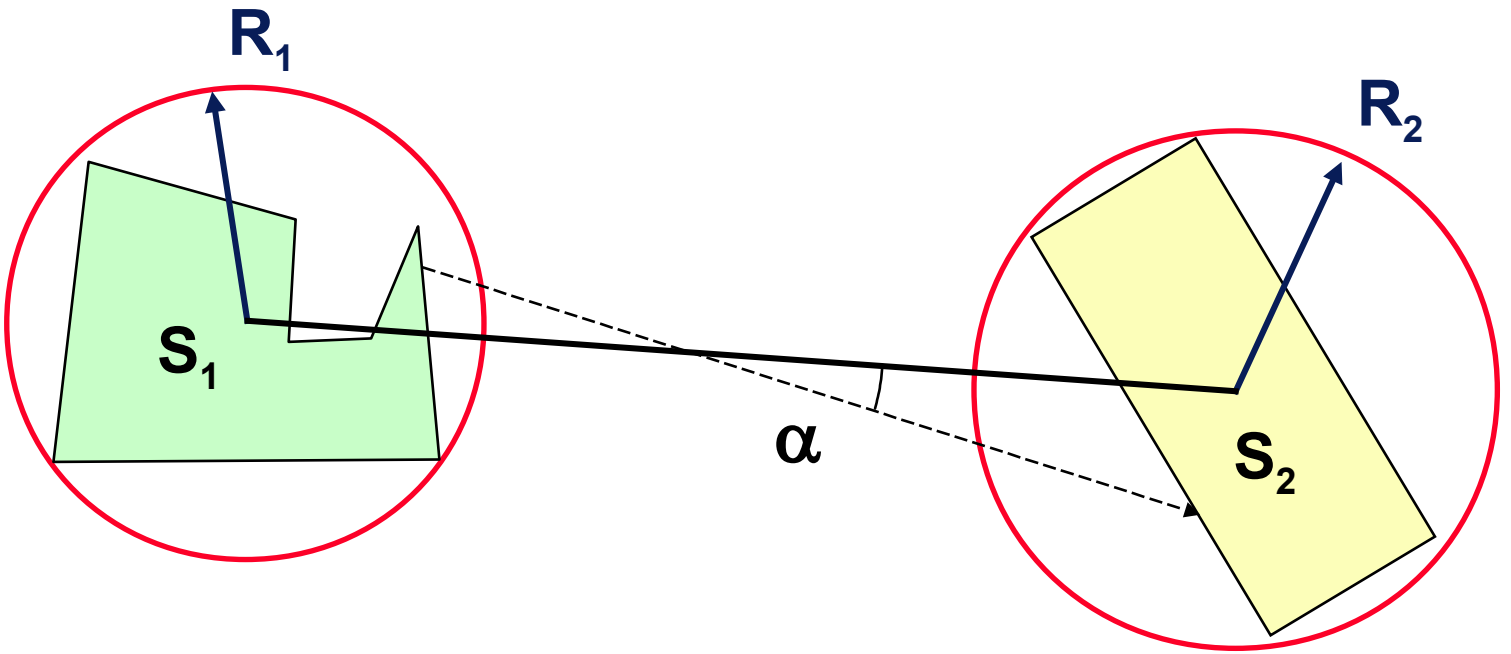
Světelný buffer

- urychluje stínovací paprsky k **bodovým světelným zdrojům**

- do každého zdroje umístím **směrovou krychli**
  - spočítám potenciální viditelnost jednotlivých těles z místa světelného zdroje
  - některé buňky mohou být zcela zakryty 1 tělesem

- při **výpočtu stínovacího paprsku** beru v úvahu jen tělesa zaznamenaná v buňce směrové krychle pro příslušný směr

Koherence paprsků



$$\cos\alpha \geq \sqrt{1 - \frac{R_1 + R_2}{\|S_1 - S_2\|}}$$

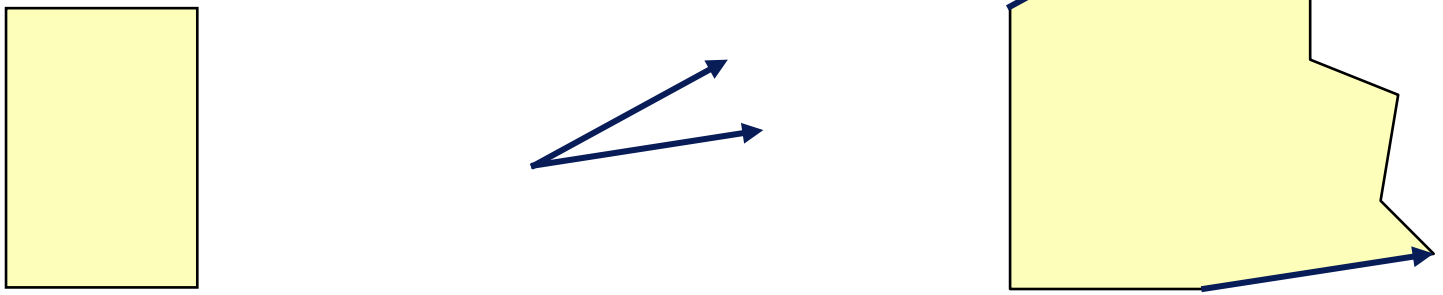
Urychlovací algoritmus

- urychluje všechny **sekundární paprsky**
  - odražené, zalomené, stínovací
- předpokládám obalová tělesa tvaru **koule**
- směrové krychle umístím do **středu** každého **obalového tělesa**
  - v každé buňce krychle spočítám seznam zasahujících objektů a světelných zdrojů (s využitím koherenční nerovnosti)
  - seznamy mohou být setříděné podle vzdálenosti

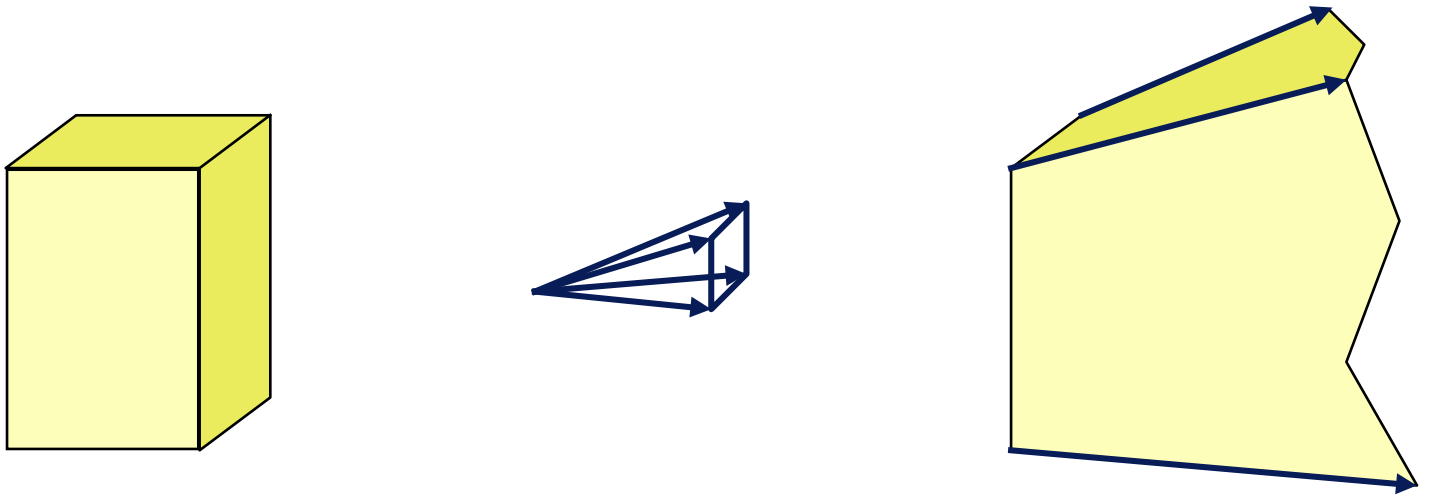
5D klasifikace paprsků

- paprsky ve scéně mají **5 stupňů volnosti**:
  - **počátek P_0 - [x, y, z]**
  - **směr [φ, θ]**
- **5D hyperkrychle** se rozdělí na **buňky**
  - každá buňka obsahuje seznam objektů, které mohou být paprskem z daného svazku („beam”) zasaženy
  - adaptivní dělení (slučování sousedních buněk se stejnými nebo podobnými seznamy)
- **6D varianta**: urychlení výpočtu animační sekvence

Klasifikace paprsků



počátek (2–3D) + směr (1D, 2D) = svazek



Adresář v průmětně

- urychluje **primární paprsky**
- průmětna se (adaptivně) rozdělí na **buňky**
  - v každé buňce zjistím potenciální viditelnost jednotlivých těles scény (spolu s pořadím)
  - některé buňky mohou být zcela zakryty jedním tělesem (obtížně se testuje – „vepsaná tělesa”)
- robustní varianta **algoritmu viditelnosti**
  - může pro většinu pixelů bezpečně určit zasažené těleso

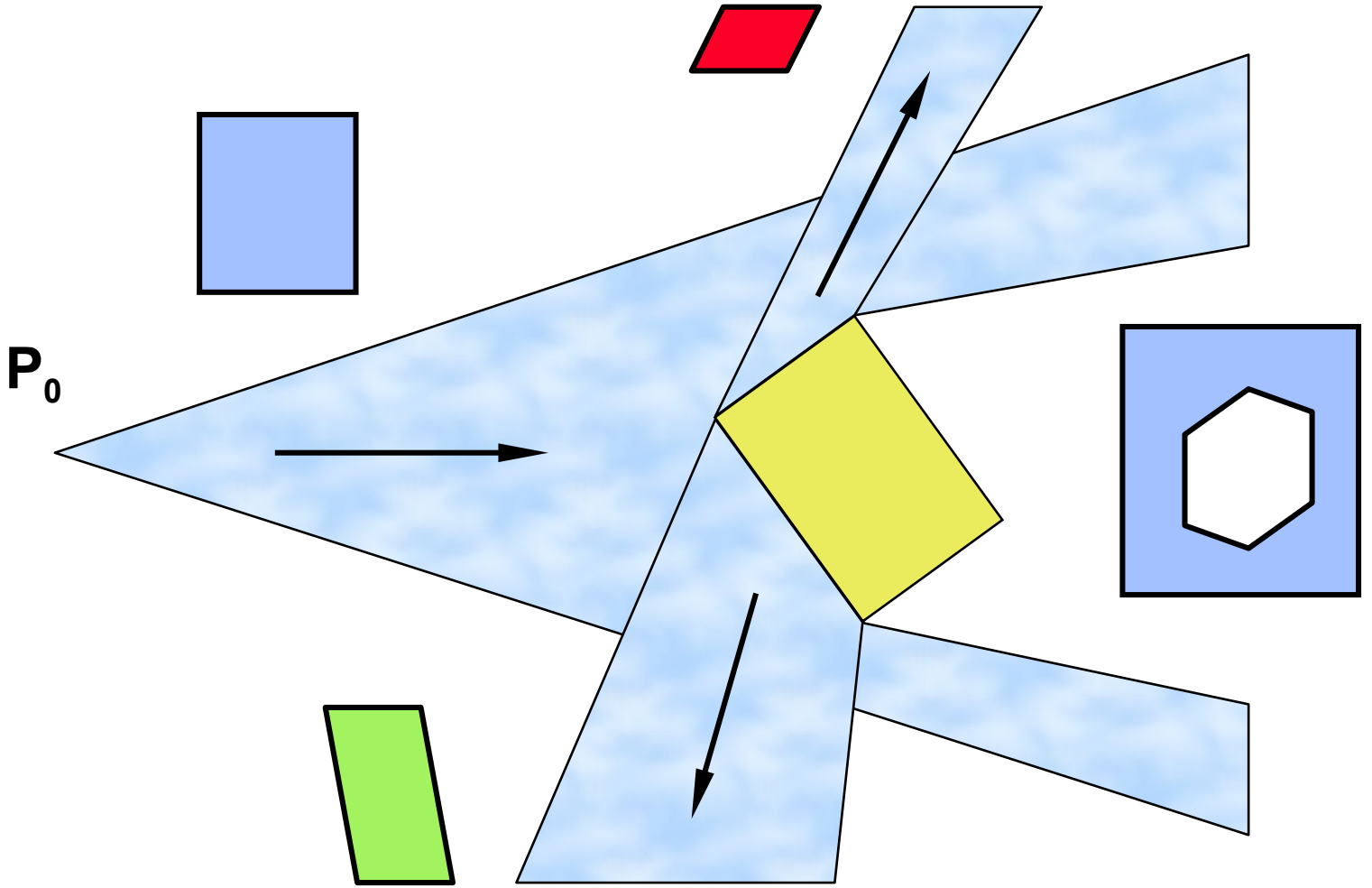
Zobecněné paprsky

- spočítám najednou více informace než **f(x,y)**
  - pro vyhlazování (odhad integrální střední hodnoty) nebo měkké stíny (podíl zastínění)
  - vždy musím obětovat obecnost scény

➡ různé tvary **zobecněných paprsků**

  - rotační nebo eliptický kužel, pravidelný jehlan
  - jehlan s polygonálním průřezem (scéna složená pouze z polygonů)

Polygonální scéna



Literatura

- **A. Glassner**: *An Introduction to Ray Tracing*, Academic Press, London 1989, 201-262
- **A. Watt, M. Watt**: *Advanced Animation and Rendering Techniques*, Addison-Wesley, Wokingham 1992, 233-248
- **V. Havran:** *Heuristic Ray Shooting Algorithms*, PhD práce, FEL ČVUT Praha, 2001
- **P. Konečný:** *Obalová tělesa v počítačové grafice*, diplomová práce, Masarykova univerzita, Brno 1998

Literatura II

- **J. Klosowski, M. Held, J. Mitchell, H. Sowizral, K. Zikan:** *Efficient collision detection using bounding volume hierarchies of k-dops*, IEEE Transactions on VaCG, 21–36, January-March 1998
- **H. Samet:** *Foundations of Multidimensional and Metric Data Structures*, Morgan Kaufmann, 2006
- **H. Samet:** *The Design and Analysis of Spatial Data Structures*, Addison-Wesley, 1990