Broad-Phase Collision Detection Using Semi-Adjusting BSP-trees

autori metódy: Rodrigo G. Luque, Joao L. D. Comba, Carla M. D. S. Freitas

prezentuje: Petra Horňáková

Úvod do problematiky

- n objektov - n^2 možností kolidujúcich dvojíc
- 2 fázy:

  1. Broad phase - aproximuje - výstup = potenciálne kolidujúce dvojice

  2. Narrow phase - presné prieniky

Úvod do problematiky

- Broad phase - musí byť rýchla

  - bounding volumes (AABB, OBB, k-DOP...)

  - BSP-tree

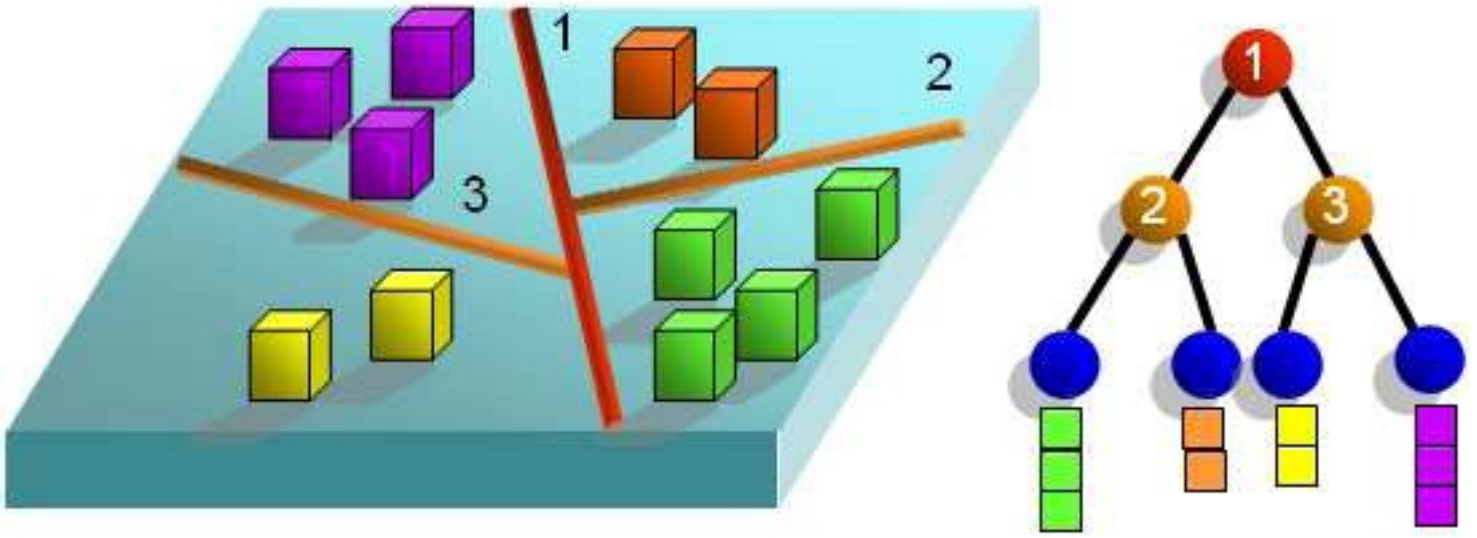
- BSP-tree - navrhnuté pre statické scény
- Semi-Adjusting BSP-tree - používajú lokálne aktualizácie - nie je nutné prerábať celý strom

Príbuzné práce

- Mirtich (1997) - vymaže objekt z pôvodnej pozície a vloži ho na novú
- Thatcher (2000) - dovoluje prekryvať sa susedným uzlom
- Torres (1990) - čiastočné rebalansovanie
- Agarwal (1998), Comba (2000) - BSP založené na predvídaní trajektórie objektov

BSP stromy

- nadrovina ľubovolným smerom
- nadroviny v uzloch
- objekty v listoch



Ako určovať partície BSP stromu

- kritéria kvality nadroviny zohladňujú:

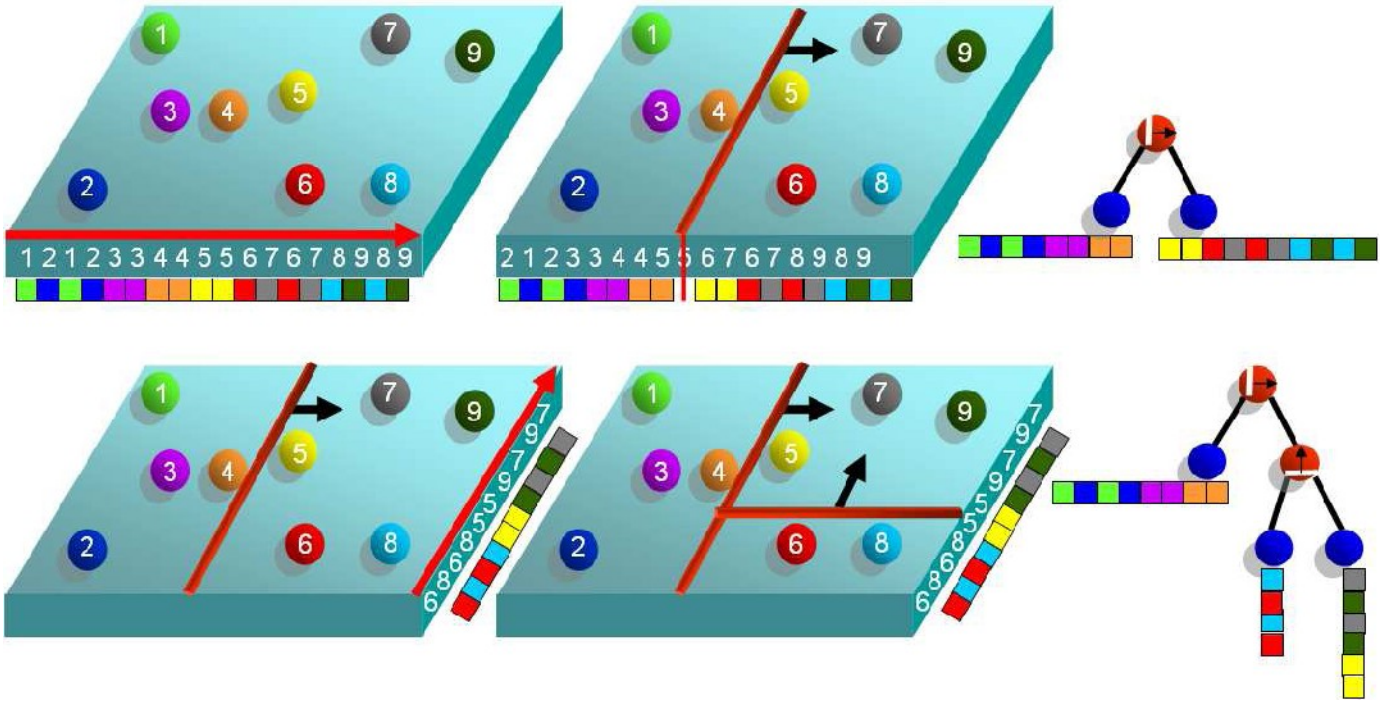
  1. population(p) - počet objektov

  2. balance(p) - pomer objektov v jednotlivých polrovinách

  3. redundancy(p) - počet objektov, ktoré obsahujú obidve polroviny

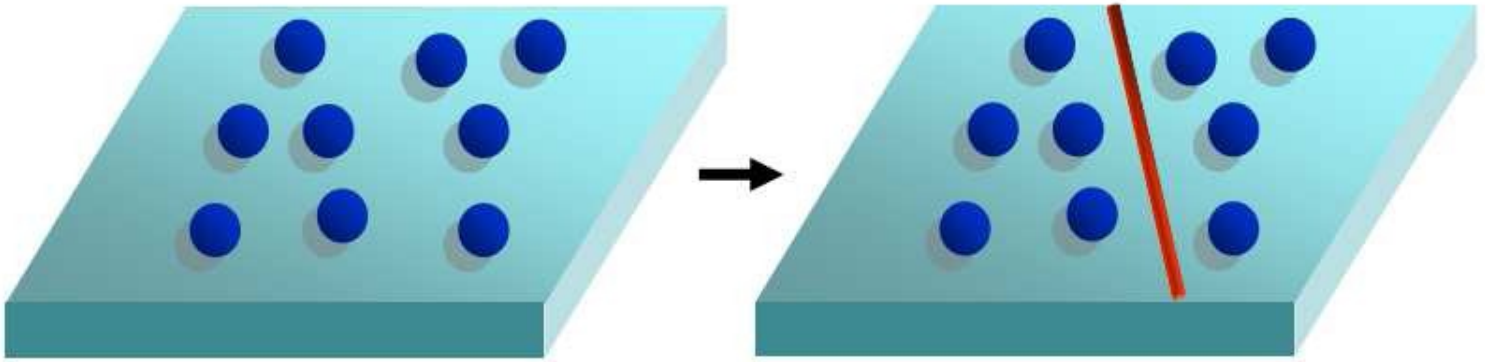
- vyberá sa nadrovina, ktorá maximalizuje balance a minimalizuje redundancy

Projekcia na nadrovinu



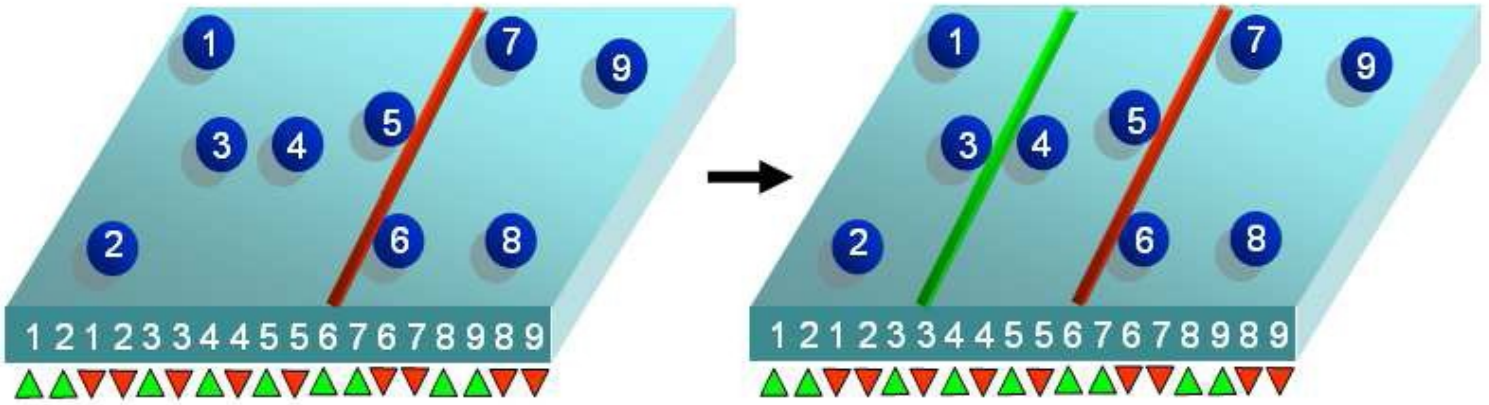
Semi-Adjusting BSP-tree Operators

- **Split** - rozdelí uzol s populáciou väčšou ako určená hranica



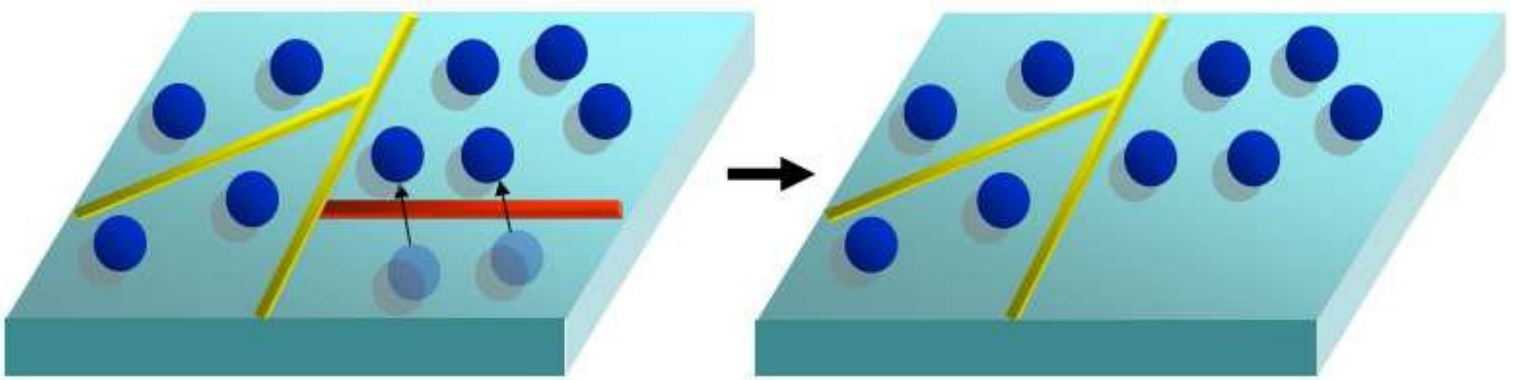
Semi-Adjusting BSP-tree Operators

- **Shift-split** - namiesto pôvodnej deliacej nadroviny použije inú - rovnobežnú s pôvodnou



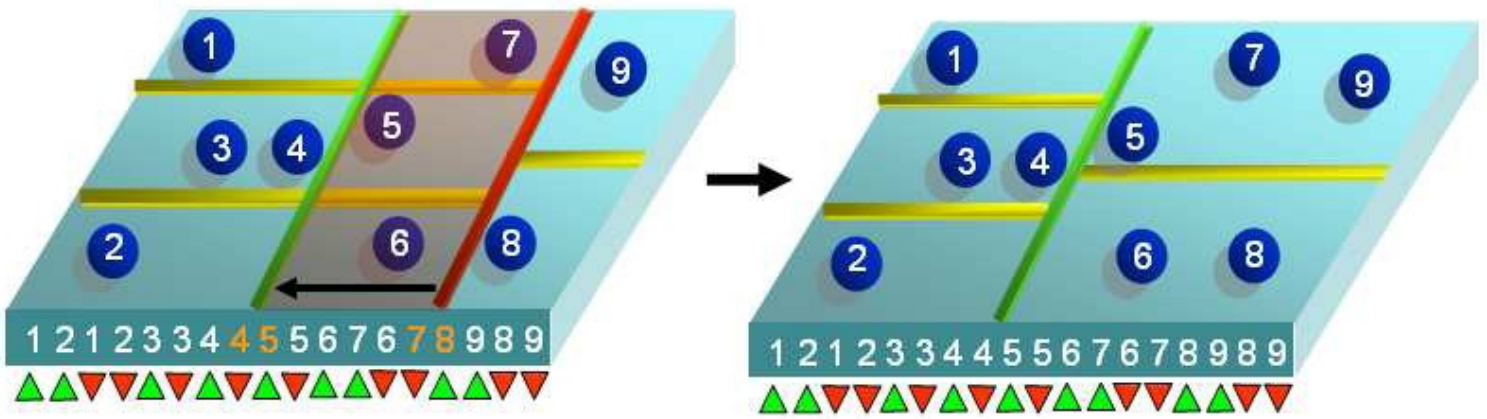
Semi-Adjusting BSP-tree Operators

- **Merge** - zlúči uzly s populáciou menšou ako určená hranica



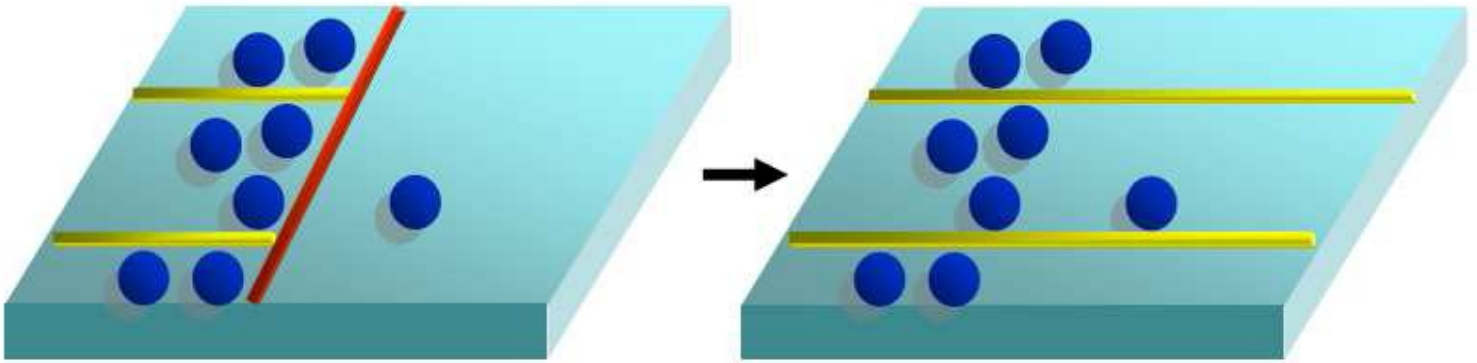
Semi-Adjusting BSP-tree Operators

- **Balance** - upravuje podstromy, ktoré zostanú nevyvážené



Semi-Adjusting BSP-tree Operators

- **Swap** - vymaže nadrovinu



Semi-Adjusting BSP-tree Algorithm

- 2 časti:

  1. aktualizácia pozícií jednotlivých objektov

  2. aktualizácia štruktúry BSP-stromu

- aktualizácia stromu - drahá
- zle usporiadaný strom - nárast počtu kolidujúcich párov
- rozumný kompromis - aktualizácia stromu v určitých intervaloch

Aktualizácia pozícií objektov

- v každom kroku
- zhora nadol
- objekt sa aktualizuje podľa jeho nového bounding boxu
- najlepšie výsledky s AABB

Aktualizácia štruktúry stromu

- dôležitá je aplikácia operácií
- balance ako prvý
- shift-split vždy pred/namiesto split
- balance vždy pred/namiesto swap
- merge - používa sa iba ak je naňho čas

Aktualizácia štruktúry stromu

Algorithm 1 Scheduling Semi-Adjusting Operators

1: **if** not pending deferred events **then**
2: start traversal at the root *n* of the BSP
3: **if** *n* is unbalanced **then**
4: evaluate balance operator
5: **if** balancing is valid perform balance operator
6: **else** schedule swap operator
7: stop traversal
8: **end if**
9: check for shift-split, split and merge, and schedule events.
10: **if** any event was scheduled stop traversal.
11: **else** repeat steps 3-10 for each subtree of *n*.
12: **end if**
13: calculate number of events to perform depending on the size of the scheduled events li
14: process as much as possible shift-split events.
15: process as much as possible split events.
16: process as much as possible swap events.
17: process as much as possible merge events.
18: defer remaining events

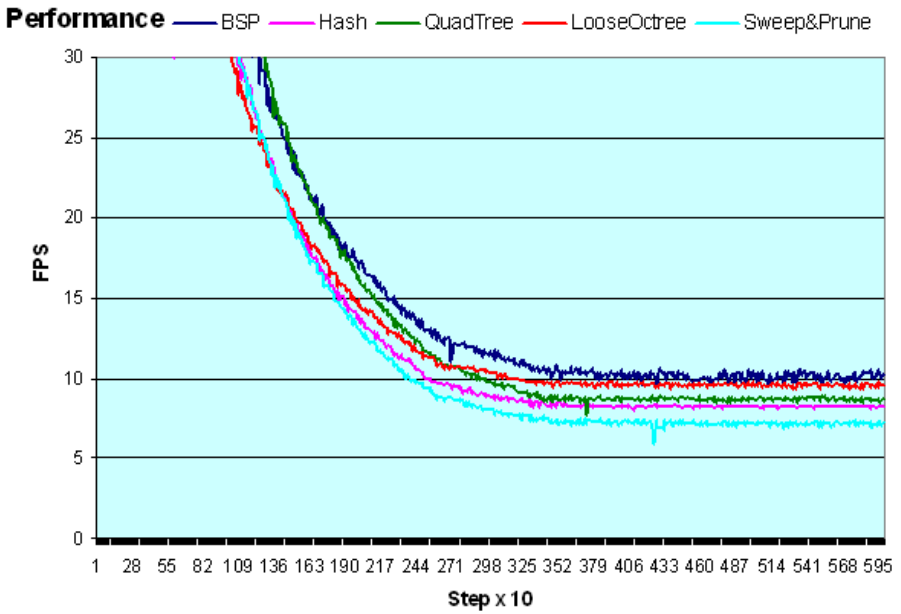
Parametre algoritmu

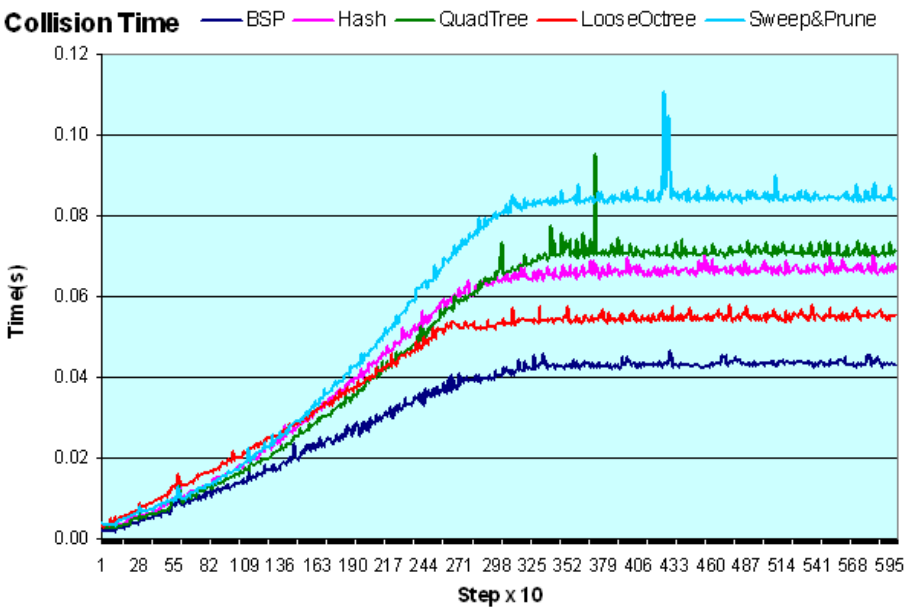
- počet kandidátov na deliacu nadrovinu pri každej split operácií $2*\sqrt{n}$
- uzol p je nevyvážený ak balance(p)<0,5
- maximalná redundancia 2,5% nad počet objektov
- hranica t pre split a merge - 16-128
- aktualizácia stromu - v každom 10-tom kroku

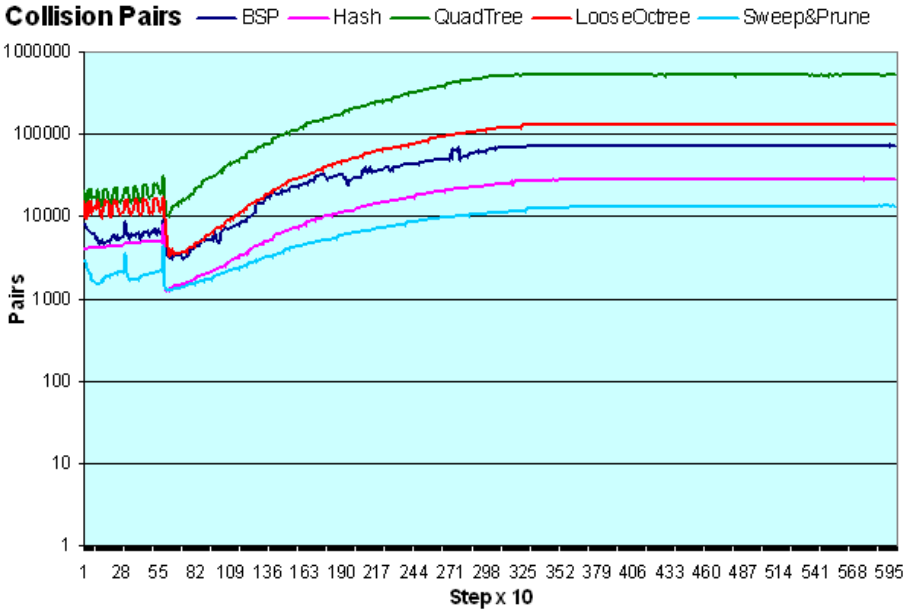
Testovanie

- 4 rôzne simulácie s 4250 pohybujúcimi sa objektami
- 4 min - 6000 krokov
- porovnávané s 4 algoritmami:
  - quadtree
  - spatial hash
  - loose octrees
  - sweep-and-prune

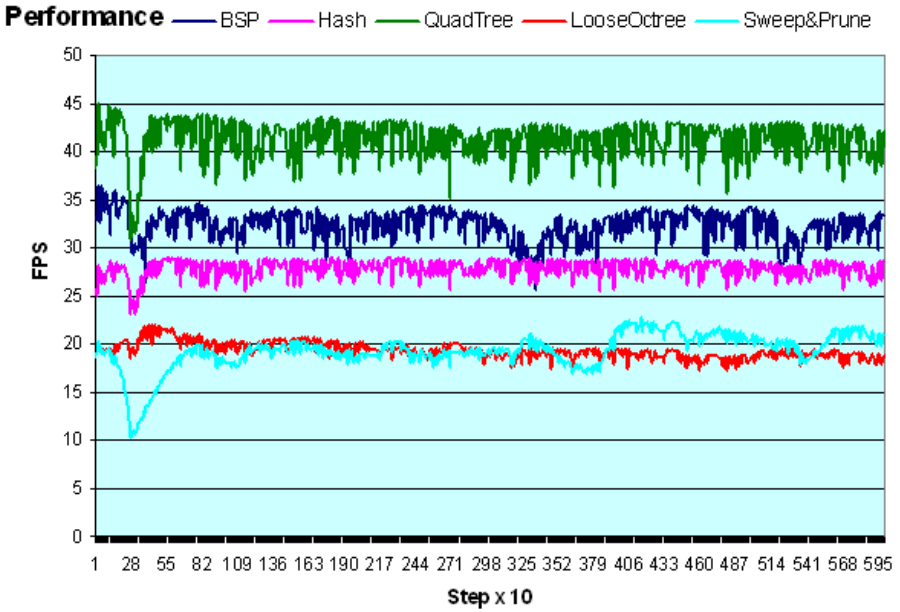
Výsledky testovania - Simulácia 1

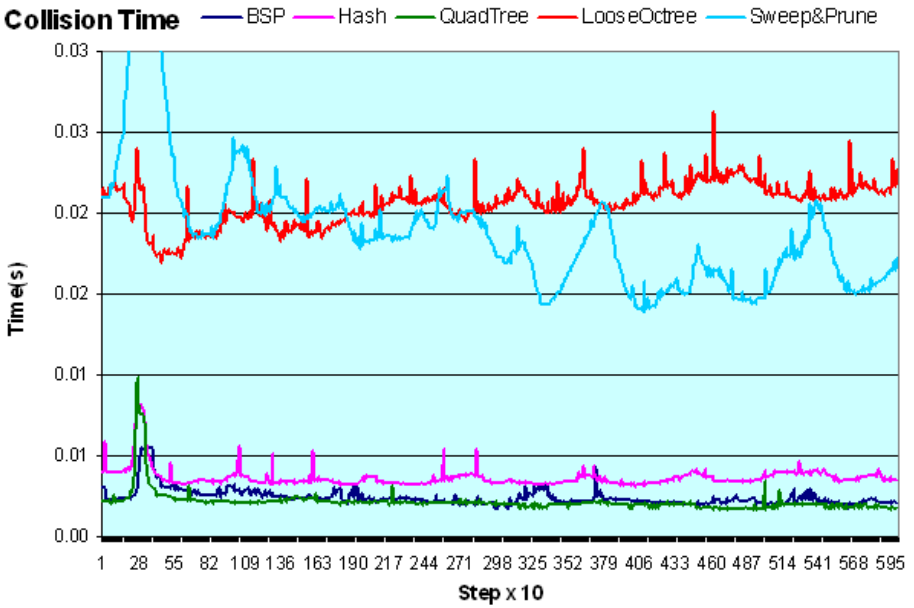


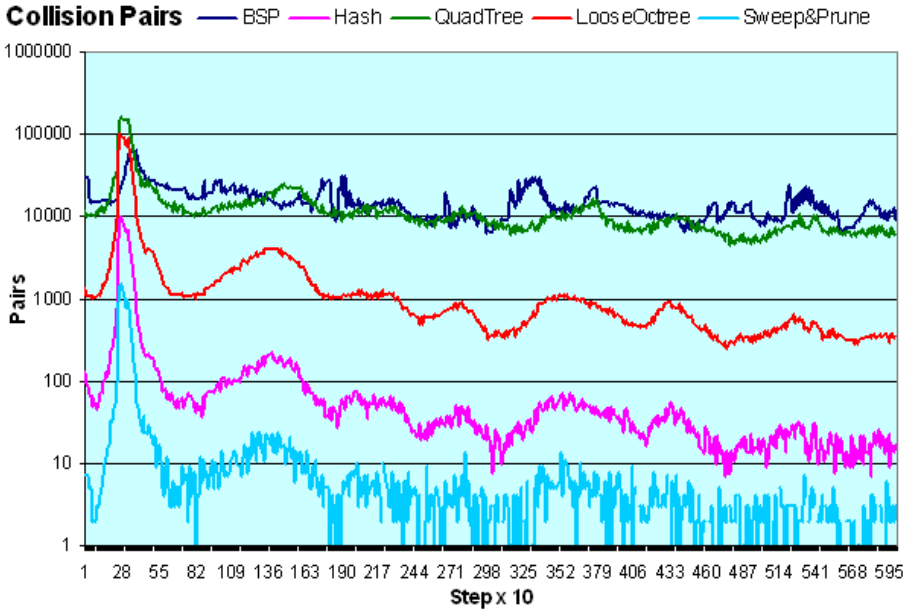




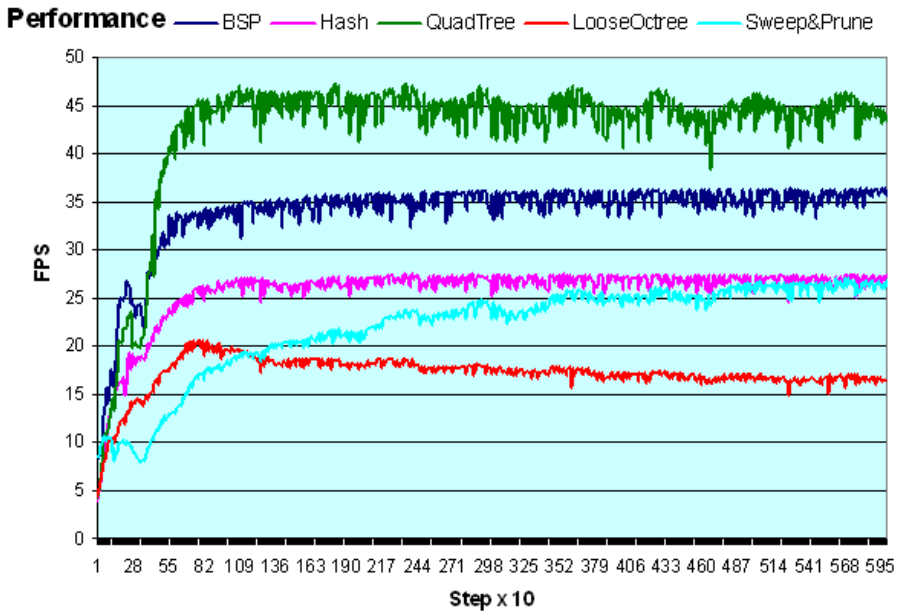
Výsledky testovania - Simulácia 2

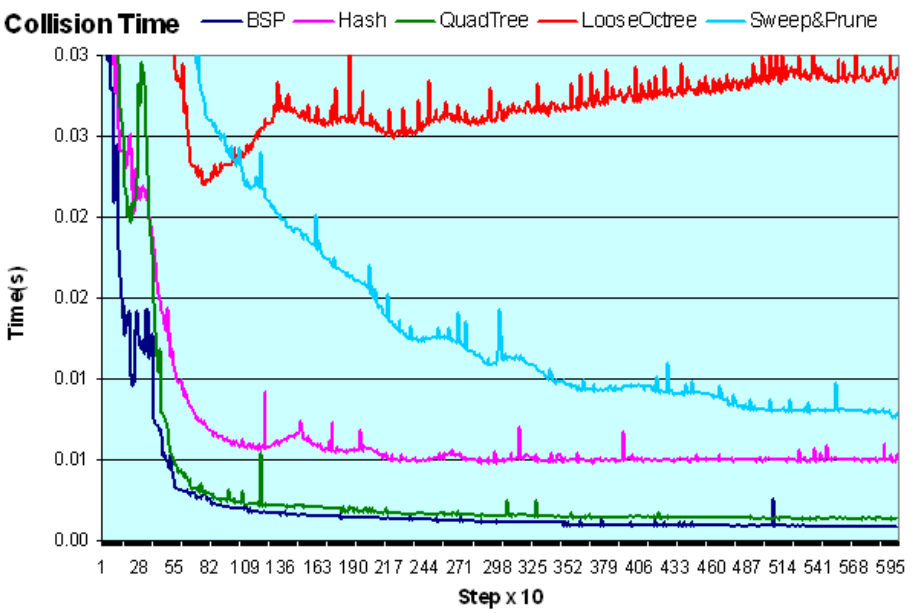


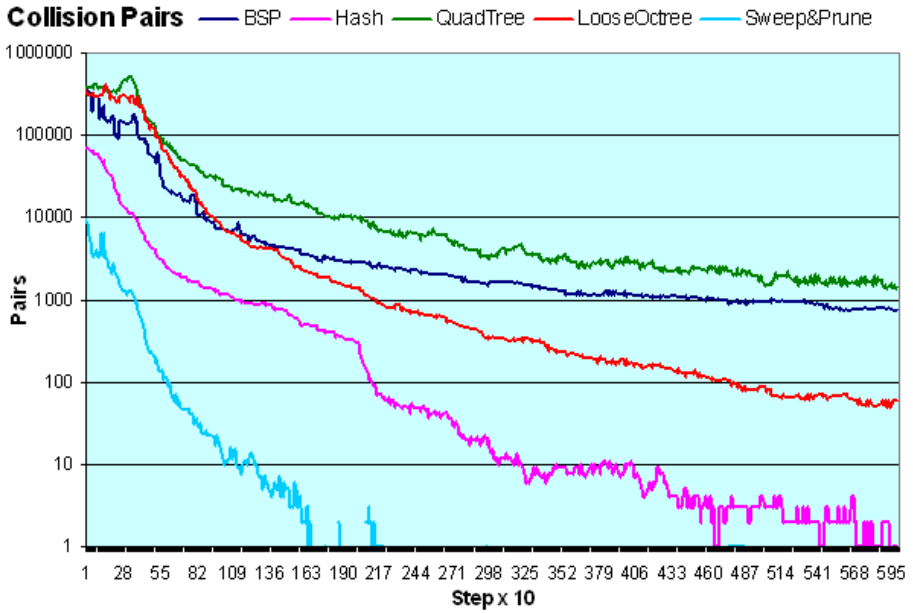




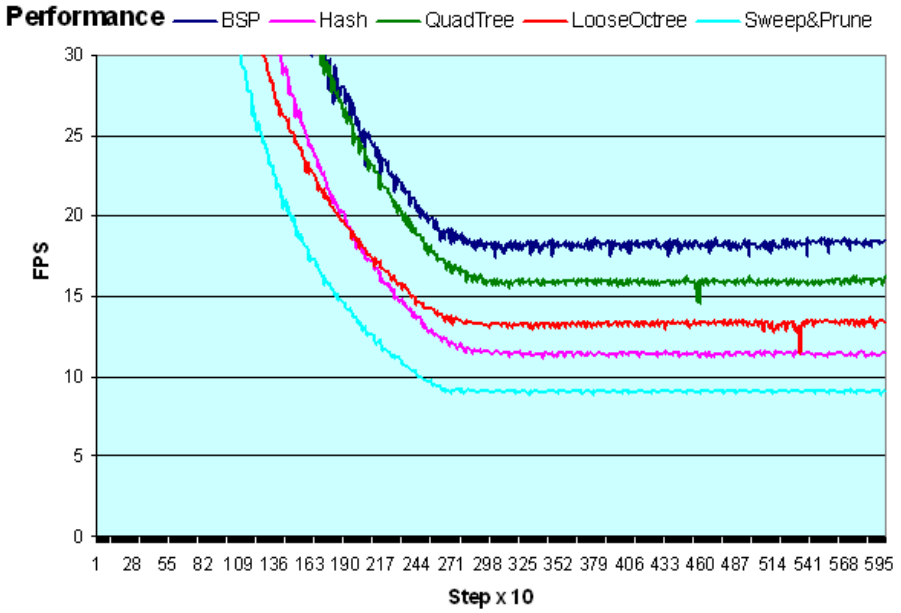
Výsledky testovania - Simulácia 3

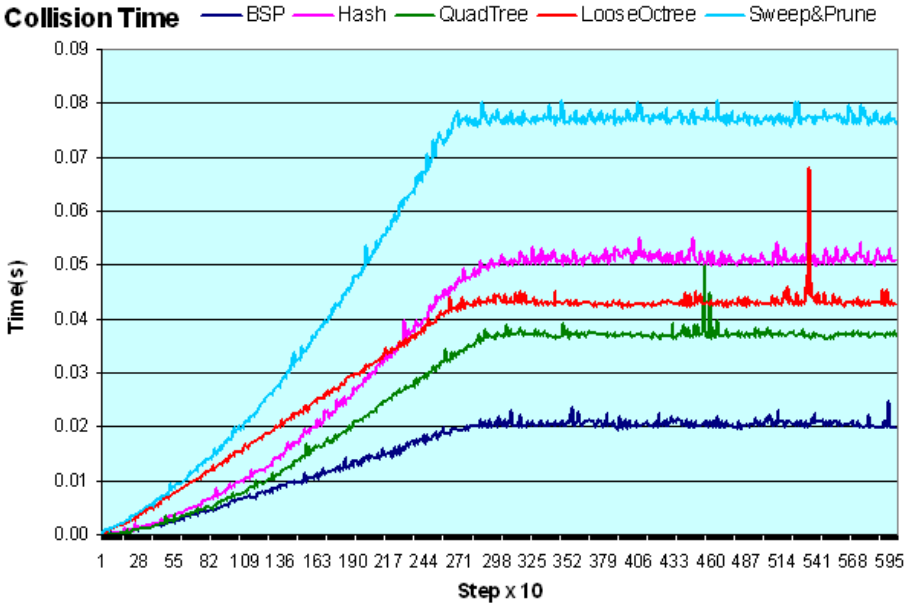


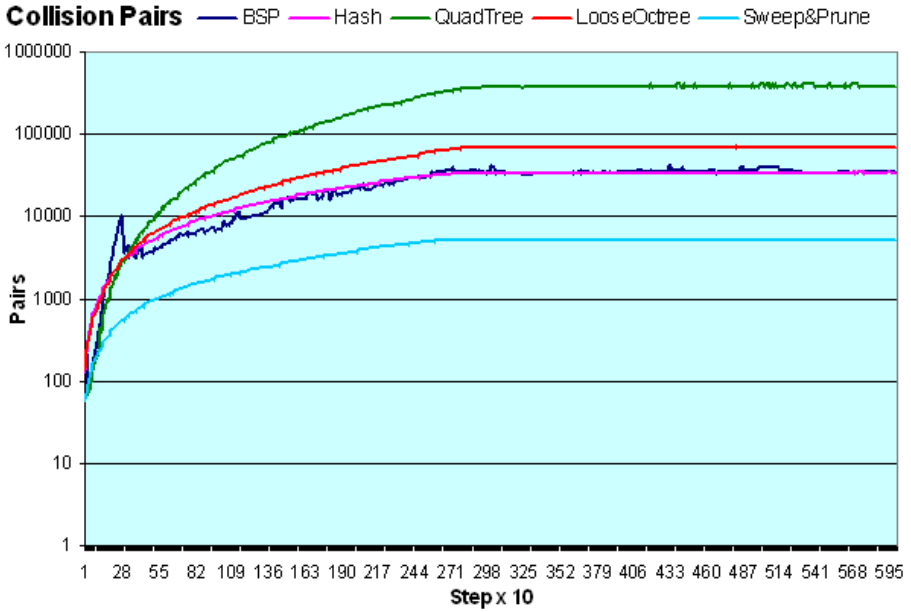




Výsledky testovania - Simulácia 4







Záver

- lepší výkon ako spatial hash, loose octrees a sweep-and-prune
- podobný počet kolidujúcich párov ako quadtree, ale lepší čas
- vhodné najmä pre simuláciu s časťou statických objektov alebo pre prostredie s priveľkým množstvom kolízií