projekt do předmětu **GMU – Grafické a multimediální procesory**

Delění scény pomocí BVH na GPU

řešitelé: **Pavel Frýz**, xfryzp00

Zadání

Implementujte metodu dělení prostoru prostřednictvím BVH na GPU

- vizualizujte výsledek pomocí frustum-cullingu a OpenGL
- srovnejte výsledek s jednoduchou CPU implementací
- popište způsob akcelerace na GPU

Objekty jsou náhodně generovány v předem definované oblasti a mají definovný vlastní bounding box.

Použité technologie

- C++ 11
- SDL 2
- OpenGL
- OpenCL
- glm
- OpenCL C++ wrapper
- GLEW

Použité zdroje

- Výpočet mortonova kódu

https://fgiesen.wordpress.com/2009/12/13/decoding-morton-codes/

- Tvorba BVH stromu

http://devblogs.nvidia.com/parallelforall/wp-content/uploads/sites/3/2012/11/karras2012hpg_paper.pdf

- Extrakce rovin z projekcni matice

http://www.cs.otago.ac.nz/postgrads/alexis/planeExtraction.pdf

- Frustrum culling

http://www.iquilezles.org/www/articles/frustumcorrect/frustumcorrect.htm

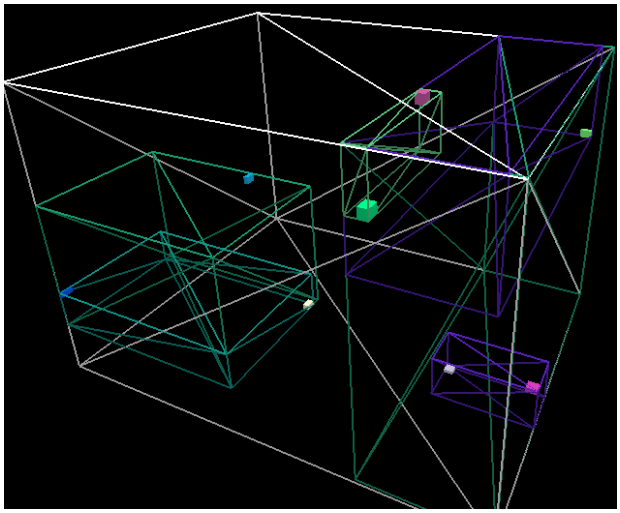
- Funkce pro výpočet počtu nulových bitů

http://www.docjar.com/html/api/java/lang/Integer.java.html

- Cvičení z GMU a úkoly z PGR

Nejdůležitější dosažené výsledky

- Generovani BVH stromu, paralelni vytvareni struktury stromu a vypocet mortonova kodu pro jednotlive objety, které vede na zrychleni algoritmu.
- Zobrazeni BVH stromu, koren stromu ma bilou barvu, ostatní ozly mají barvu generovanou náhodně, potomci stejného uzlu, ale sdílejí stejnou barvu



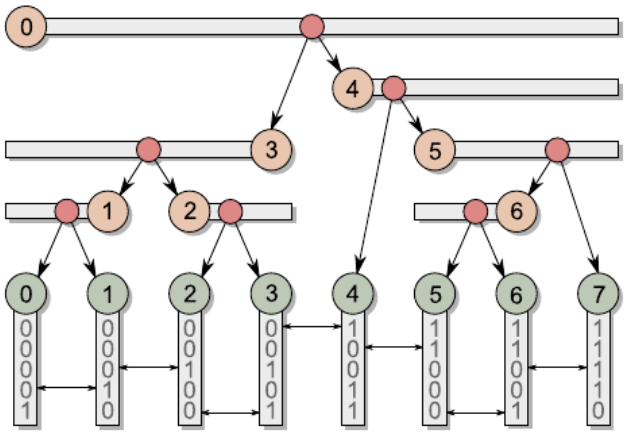
Ovládání vytvořeného programu

- Leve nebo prave tlačítko myši: rotace kamery
- Kolecko mysi: priblizeni kamery
- a: pridani objektu (pokud je objektu 100 a vice pridava naraz 100 objektu)
- d: odebrani objektu (pokud je objektu vice nez sto odebere naraz 100 objektu)
- w: prepina zobrazeni wireframu
- x: prepina vykreslovani bounding boxu vnitrnich uzlu stromu
- z: zapina a vypina vypocet BVH na cpu
- c: prepina frustrum culling

Zvláštní použité znalosti

- Mortonuv kod – funkce která mapuje vicedimenzionální data na jednodimenzionální se zachováním lokality bodů, výpočet lze provést proložením binárních souřadnic jednotlivých os, např.: pro bod $P(x_2x_1x_0, y_2y_1y_0, z_2z_1z_0)$ s přesností tři bity na souřadnici získáme devítí bitový kód $z_2y_2x_2z_1y_1x_1z_0y_0x_0$.
- Tvorba BVH stromu:
  1. Pro všechny objekty se vypočte mortonuv kod(paralelně)
  2. Seřadí se objekty podle mortonova kódu
  3. Vytvori se vnitrni uzly stromu
  4. Pro každý vnitrni uzel se zjistí jeho potomci(paralelně)
  5. Projde strom od listů ke kořeni a vypočtou se bounding boxy vnitrnich uzlů
- Ukázka stromu vytvořeného stromu je na následujícím obrázku. Kořen stromu je uzel 0.

  Jeho potomci jsou potom určení pozicí na kterém se liší klíče.



Co bylo nejpracnější

Nejvíce času zabralo nalezení vhodných článků a nastudování problematiky pro vytvoření bvh stromu, a jak lze tento proces dělat paralelně. Velmi těžké bylo donutit se a vyhradit si dostatek času pro řešení projektu.

Zkušenosti získané řešením projektu

- Výpočet mortonova kódu
- Tvorba BVH stromu
- Práce s OpenCL

Autoevaluace

Ohodnoťte vaše řešení v jednotlivých kategoriích (0 – nic neuděláno, zoufalství, 100% – dokonalost sama). Projekt, který ve finále obdrží plný počet bodů, může mít složky hodnocené i hodně nízko. Uvedení hodnot blízkých 100% ve všech nebo mnoha kategoriích může ukazovat na nepochopení problematiky nebo na snahu kamuflovat slabé stránky projektu. Bodově hodnocena bude i schopnost vnímat silné a slabé stránky svého řešení.

Technický návrh: 58% (analýza, dekompozice problému, volba vhodných prostředků, …)

Problém byl rozdělen do několika částí, kdy každá je implementována samostaně.

Programování: 52% (kvalita a čitelnost kódu, spolehlivost běhu, obecnost řešení, znovupoužitelnost, …)

Myslím, že je kód dost čitelný, kód je rozdělen do funkcí, názvy funkcí a proměných jsou výstižné,

mohlo by tam být více komentářů, a nemusel bych v průběhu měnit jazyk komentáře, kdy něco je popsáno česky a něco anglicky.

Vzhled vytvořeného řešení: 62% (uvěřitelnost zobrazení, estetická kvalita, vhled GUI, …)

Uživatelské rozhraní je jednoduché a intuitivní.

Využití zdrojů: 84% (využití existujícího kódu a dat, využití literatury, …)

S využitím literatury a existujícího kódu jsem velmi spokojen, práci jsem si usnadnil použitím kostry, kterou jsem použil z úkolů z PGP.

Hospodaření s časem: 72% (rovnoměrné dotažení částí projektu, míra spěchu, chybějící části řešení, …)

S využitím času jsem spokojen, až na to, že jsem mohl začít dělat dřív, ale nemám pocit, že bych nějak výrazně spěchal.

Celkový dojem: 80% (pracnost, získané dovednosti, užitečnost, volba zadání, cokoliv, …)

Celkově jsem s prací spokojen, ale je tam několik věcí, které by šli zlepšit. Jelikož jsem dělal projekt sám byl poměrně pracný, ale kdybych ho dělal ve více lidech, našla by se práce i pro ně. Volbu zadání bych neměnil. Zkusil jsem si a naučil pracovat s OpenCL, což očekávám, že se mi bude hodit v budoucích projektech.