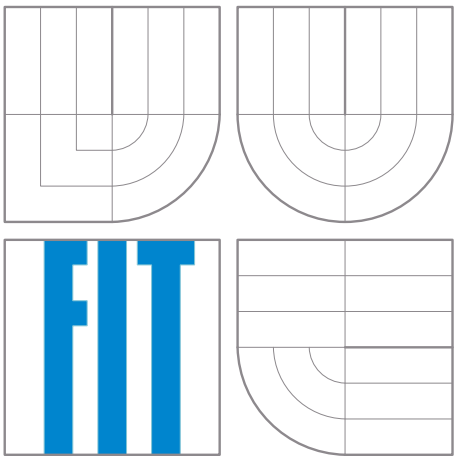


Projekt do předmětu GMU – Grafické a multimediální procesory

Výpočet siluet pomocí GPU

4. ledna 2015

Řešitelé: Zdeněk Biberle (xbiber00@stud.fit.vutbr.cz)
Vít Hodes (xhodes00@stud.fit.vutbr.cz)
Fakulta Informačních Technologií
Vysoké Učení Technické v Brně

1 Zadání

- Výpočet stínových těles
  - Implementace na GPU
  - Implementace na CPU pro porovnání rychlosti
- Vytvoření demonstrační aplikace
  - Vykreslení scény se stíny pomocí patřičně modifikovaného z-fail algoritmu
  - Měření rychlosti
  - Různé vizualizační pomůcky

  vypočítaná stínová tělesa na jendoduché scéně

2 Použité technologie

3 Technologie potřebné pro běh

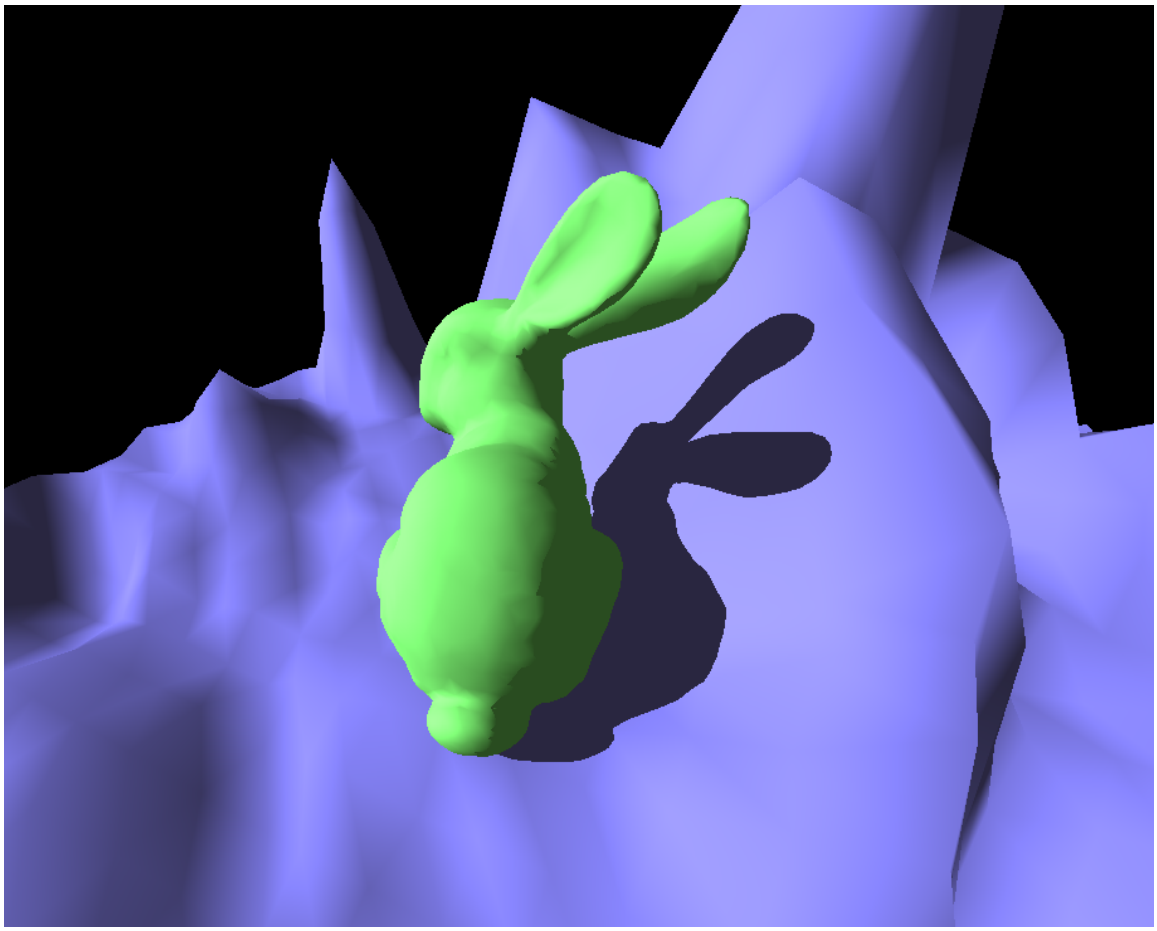
- Grafický akcelerátor s podporou OpenGL 4.3
  - SSBO
  - Image load/store
  - Compute shadery
- SDL2
- GLM
- glew

4 Technologie použité pro tvorbu

- Blender
- Textový editor/vývojové prostředí vlastní volby

5 Použité zdroje

- Stanford bunny
- Utah teapot
- Byungmoon Kim, Kihwan Kim, Greg Turk - Real Time Shadow of Transparent Casters Using Shadow Volume, 2007
- Dokumentace OpenGL 4.3



Obrázek 1: Králíček vrhající stín na terén

6 Nejdůležitější dosažené výsledky

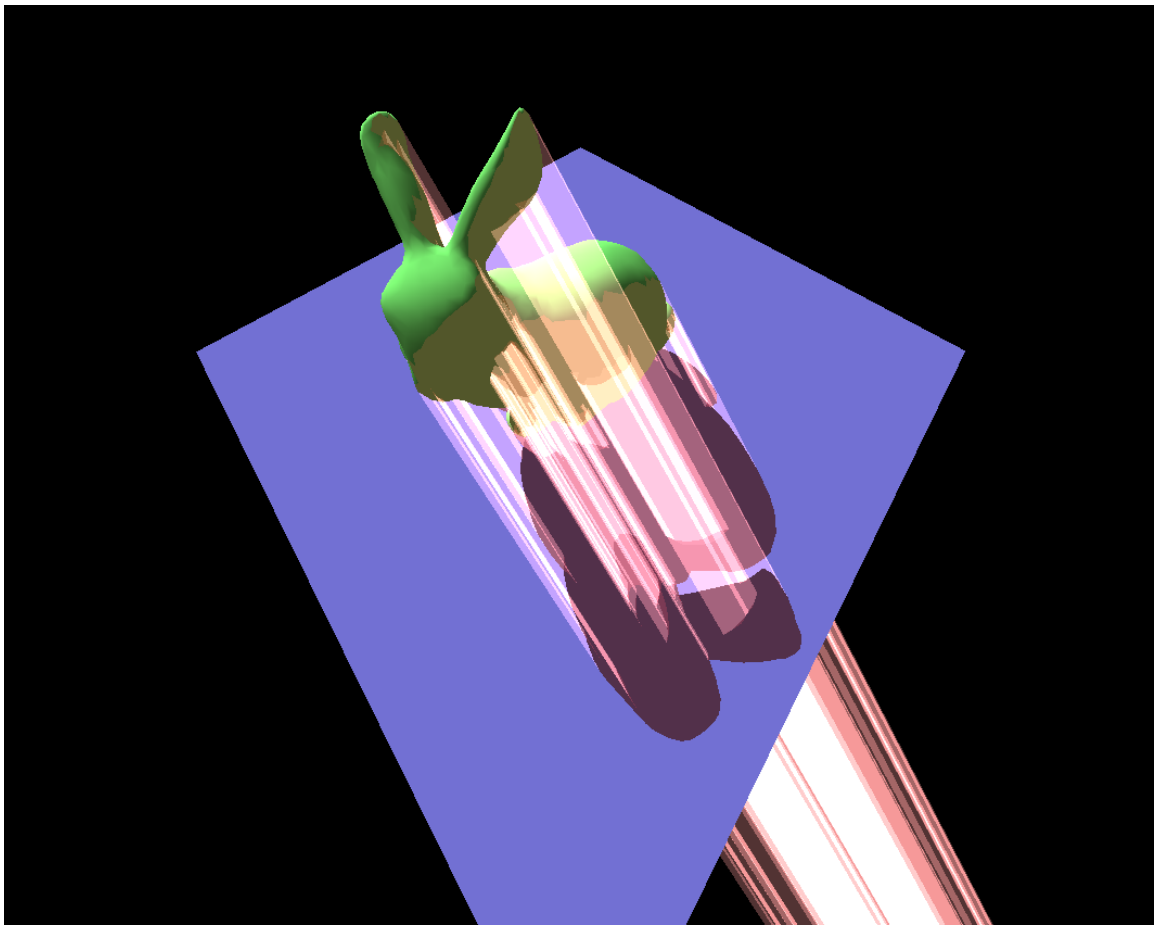
Aplikace provádí generování stínového tělesa pro libovolný model na CPU i na GPU a scénu poté zobrazuje s jeho použitím. Dodatečně umožňuje i zobrazení samotného stínového tělesa. Ovšem dle našich zkušeností nefunguje generování stínového tělesa na grafických akrtách od společnosti AMD.

6.1 Generování stínových těles i pro non-manifold geometrii

Zadaný algoritmus umožňuje generování stínových těles i pro non-manifold geometrii. Na obrázku 3 vidíme několik čtverců vrhajících korektní stín. Každý tento čtverec je složen pouze ze dvou trojúhelníků a není tedy možné na ně aplikovat běžný algoritmus generování stínových těles (tj. protažení hran mezi dvěma trojúhelníky, kde jeden je natočen ke světlu a druhý je od světla odvrácen) tak, aby dosáhl správného výsledku.

7 Ovládání vytvořeného programu

- Levé tlačítko myši - Společně s pohybem myši umožňuje změnu úhlu pohledu na scénu.
- Kolečko myši - Přiblížení a oddálení zobrazené scény.



Obrázek 2: Králíček a jeho stínové těleso

- Klávesa R - Přepíná autonomní otáčení zobrazené scény.
- Klávesa T - Přepíná zobrazení vypočteného stínového tělesa.
- Klávesa C - Přepíná mezi CPU a GPU implementací výpočtu stínového tělesa.

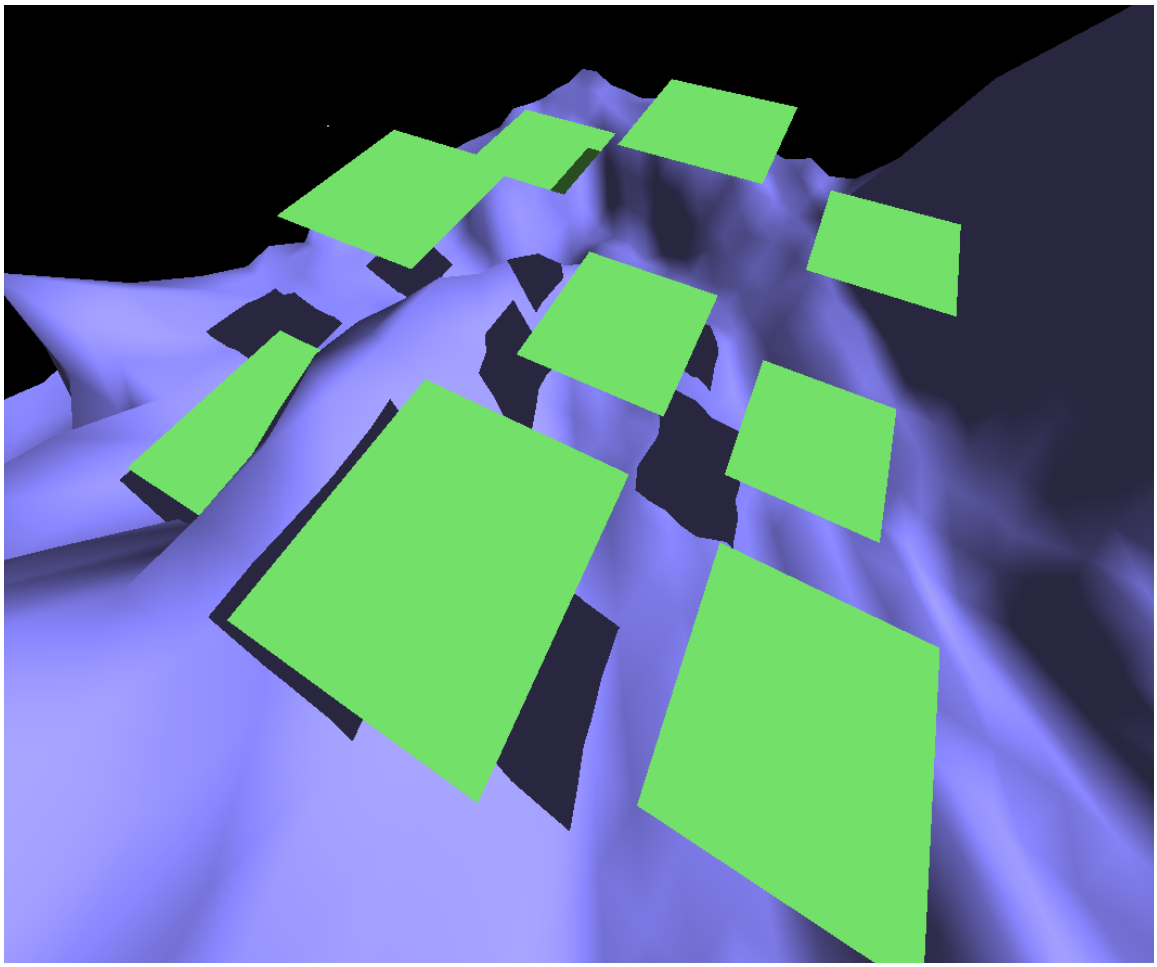
8 Zvláštní použité znalosti

Použití OpenGL compute shaderů se různě liší od použití OpenCL či CUDA. To vyžadovalo získání dodatečných informací o jejich využívání a o komunikaci s němi. Využití image load/store v shaderech openGL pro vytvoření vlastního ”stencil bufferu”.

9 Rozdělení práce v týmu

Zdeněk Biberle Základ aplikace, GPU výpočet stínového tělesa.

Vít Hodes Vykreslení scény za použití stínového tělesa, nahrávání modelů, CPU výpočet stínového tělesa.



Obrázek 3: Geometrie složená z několika non-manifold částí vrhající stíny

10 Co bylo nejpracnější

Zdeněk Biberle Značnou část času jsem strávil nad komunikací mezi aplikací a compute shaderů. Později jsem zjistil, že implementace SSBO na grafických kartách od AMD se v současnosti vyznačuje různými problémy a tudíž jsem řešil nevyřešitelné.

Vít Hodes Přijít na to, jak impementovat celočíselný "stencil buffer"s obecným přístupem ze shaderů. Po prozkoumání několika slepých cest se dospělo k řešení pomocí iimage2D a atomického sčítání. Taky implementovat stíny nad shadow volume, který se mi na AMD kartě negeneruje správně se moc nedá.

11 Zkušenosti získané řešením projektu

Zdeněk Biberle

- Compute shadery
- SSBO

Vít Hodes

- Framebuffer object, color/depth attachment
- v GLSL existují i shadow samplery a image samplery
- image load/store, atomické operace obecně v shaderech

12 Autoevaluace

Technický návrh (70%): (analýza, dekompozice problému, volba vhodných prostředků, . . .)

Programování (70%): (kvalita a čitelnost kódu, spolehlivost běhu, obecnost řešení, znovupoužitelnost, . . .) Vzhledem ke kratšímu rozsahu se moc neuplatnuje zapouzdření ve vykreslovací části. Znovupoužitelnost spíše znalostí než konkrétního kódu.

Vzhled vytvořeného řešení (40%): (uvěřitelnost zobrazení, estetická kvalita, vhled GUI, . . .) Estetická kvalita nebyla cílem řešení.

Využití zdrojů (70%): (využití existujícího kódu a dat, využití literatury, . . .) Bez dokumentací OpenGL API by tento projekt nevznikl.

Hospodaření s časem (50%): (rovnoměrné dotažení částí projektu, míra spěchu, chybějící části řešení, . . .) Přiměřené.

Spolupráce v týmu (80%): (komunikace, dodržování dohod, vzájemné spolehnutí, rovnoměrnost, . . .) Komunikace dobrá.

Celkový dojem (70%): (pracnost, získané dovednosti, užitečnost, volba zadání, cokoliv, . . .) Projekt byl zajímavý a potenciálně velmi užitečný. Škoda problémů s různými platformami(AMD x nVidia).

13 Doporučení pro budoucí zadávání projektů

Témata projektů byla dostatečně pestrá, ale příště by mohla být zadána dříve.

14 Různé

Ještě něco by v dokumentaci mělo být? Napište to sem! Podle potřeby i založte novou kapitolu.