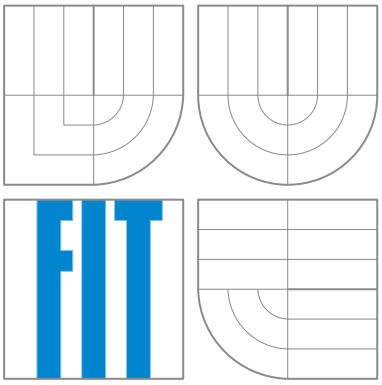


Multimédia

Postprocessing OpenGL scény

13. května 2014

Autor: Pavel Macenauer, xmacen02@stud.fit.vutbr.cz
Jan Bureš, xbures19@stud.fit.vutbr.cz

Fakulta Informačních Technologií
Vysoké Učení Technické v Brně

Obsah

Cíl projektu

Cílem projektu bylo vygenerovat OpenGL scénu. Tu následně vyrenderovat do textury, modifikovat pomocí GLSL pixel shaderů a zobrazit. Implementovat různé efekty. Demonstrovat jejich použití na vygenerované scéně. Jednotlivé efekty mělo být možné volit za běhu aplikace.

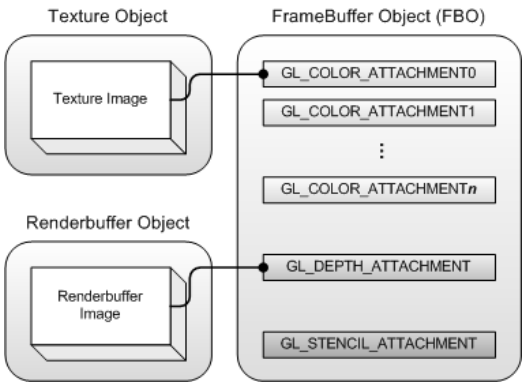
Popis řešení

K řešení projektu jsme použili jazyk C++ a vývojové prostředí Visual studio 2012. Pro práci s OpenGL jsme využili knihovny GLUT a GLEW. Samostatné shadery jsou implementovány v jazyce GLSL a každý shader je umístěn v samostatném souboru. Dohromady jsme implementovali 10 shaderů, z toho dva proměnné v čase. Dohromady tedy 10 postprocessing efektů.

OpenGL Framebuffer Object

Nejvhodnější metodou k renderování do textur v nových verzích OpenGL je využití tzv. Framebuffer Objectu. Samotný OpenGL kontext přichází již s výchozím framebufferem, který se následně využívá k zobrazení na obrazovce, nicméně umožňuje nám vytvořit si i vlastní.

Součástí framebufferu jsou color buffer, depth buffer a stencil buffer. Tyto buffery se musí všechny manuálně alokovat a krom color bufferu jsou nepovinné. Framebuffer umožňuje renderovat i na více lokací naráz a to z toho důvodu, že přehazování samotných framebufferů by byla velmi drahá operace. Max. počet color bufferů je dán grafickou kartou, resp. konstantou `GL_MAX_COLOR_ATTACHMENTS`.



Výhodou vlastního framebufferu je tzv. off-screen rendering, kdy si scénu můžeme vyrenderovat mimo obrazovku do nějaké textury a pracovat pak už pouze ze samotnou texturou. To nachází využití ve 2 oblastech:

- postprocessing textury
- virtuální pohledy na scénu

Jednotlivé funkce shaderů

K postprocessingu aplikace využívá následujících efektů:

Blur Rozostření pomocí matice 3x3. Vybere se okolí pixelu na vstupu do shaderu a výsledná hodnota je průměr okolí.

Noční vidění Převedení obrazu do zeleného spektra.

Černobílá Převedení obrazu do černobílého spektra.

Inverze Invertování barev.

Sin Změna souřadnic pomocí funkce sinus.

Bloom Jsou zvýrazněny osvětlené plochy a následně rozmazány.

Sobel Shader na detekci hran pomocí Sobelova operátoru - jak v horizontálním, tak ve vertikálním směru.

Kruhové Vlnění Kruhové vlnění závislé na čase.

Šum Šum závislý na čase.

Dithering Dithering efekt nebo-li redukce počtu barevných složek.

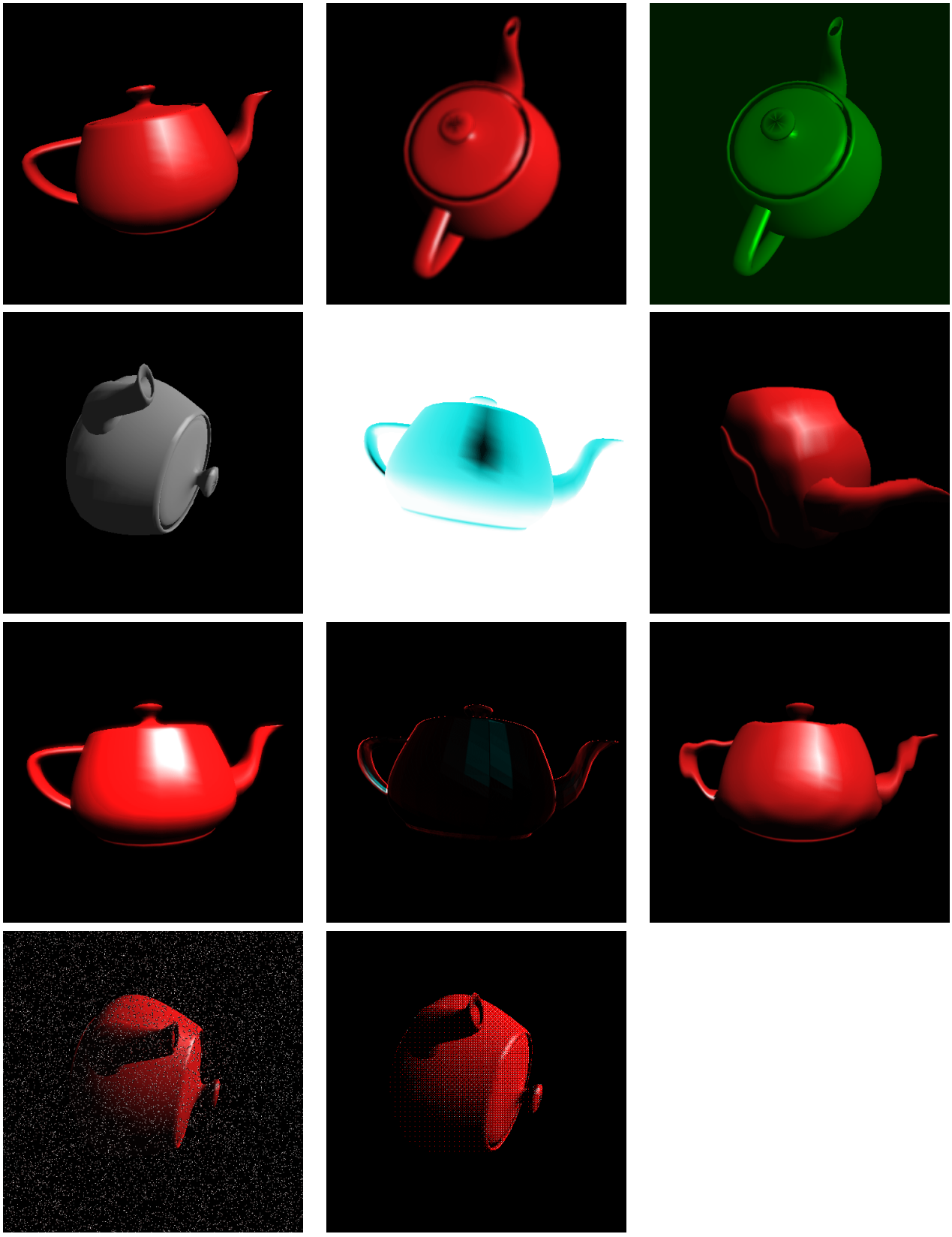
Ovládání aplikace

Vše potřebné ke spuštění aplikace na platformě Windows je ve složce `exe` a spouští se pomocí `Postprocessing.exe`. Aplikaci je možné ovládat pomocí kláves. Jednotlivé shadery se změní podle stlačené klávesy následovně:

- 0 Bez efektu.
- 1 Rozostření scény pomocí matice 3x3.
- 2 Efekt nočního vidění.
- 3 Převod do odstínů šedé.
- 4 Inverze barev.
- 5 Změna souřadnic texelu pomocí funkce sinus.
- 6 Zvýraznění odlesků a jejich rozmazání.
- 7 Detekce hran pomocí sobelova operátoru.
- 8 Simulace vlnění.
- 9 Náhodný šum.
- d Dithering.

Galerie implementovaných efektů

Efekty jsou vykresleny v pořadí jednotlivých stlačitelných kláves viz. Ovládání aplikace.



Překlad

Projekt byl vypracován v MS Visual Studiu 2012, je tak dělaný pro platformu Windows. Ve složce `src` je obsažen i projektový soubor, pomocí kterého je projekt preložitelný. Využívá externích knihoven `GLEW` a `GLUT`, které jsou volně ke stažení. Celý projekt je ke stažení na
https://github.com/mmaci/vutbr-fit-mul-postprocessing-opengl
kde jsou přiloženy i všechny potřebné knihovny.

Závěr

Povedlo se implementovat projekt dle zadání. OpenGL scénou je červený GLUT čajník, který se vyrenderuje do textury a ta je pak vykreslována přes průchod shaderem na obrazovku.

Do budoucna by bylo vhodné implementovat i rozhraní pro řetězení renderování a shaderů (umožňilo by tak fungovat s OpenGL scénu na způsob práce s grafickým editorem, kdy shadery by byly jednotlivé úpravy).

Literatura

[1] Alexei Sholik. Ripple fragment.
https://gist.github.com/alco/3070640.

[2] blitzbasic.com. image processing with glsl shaders.
http://www.blitzbasic.com/Community/posts.php?topic=85263.

[3] Martin Christen. Loading, Compiling, Linking, and Using GLSL Programs.
http://www.opengl.org/sdk/docs/tutorials/ClockworkCoders/loading.php.

[4] open.gl. Framebuffers.
http://open.gl/framebuffers.

[5] wikibooks.org. OpenGL Programming/Post-Processing.
http://en.wikibooks.org/wiki/OpenGL_Programming/Post-Processing.