projekt do předmětu **PGR – Počítačová grafika**

Náročné demo v OpenGL - ogl01

řešitelé: **Pavel Koupý**, xkoupy00

Zadání

Obecné zadání

- Navrhněte vhodné demo zatěžující řádně grafickou kartu vykreslováním přes OpenGL.
- Implementujte, demonstrujte.

Specifikace zadání

- Zadaní nebylo konzultováno s vyučujícím.
- Výpočet a vykreslení fraktálního útvaru akcelerované grafickou kartou. Konkrétním příkladem v tomto projektu je Mandelbrotova množina komplexních čísel.
- Grafická karta potažmo Fragment Shader bude provádět určený počet výpočetních iterací pro každý pixel vykresleného primitiva a stanovovat příslušnost (barvu) jednotlivých pixelů k Mandelbrotově množině.
- Výpočet pro zjištění příslušnosti je vhodný pro paralelizaci na GPU, protože při výpočtu nezáleží na okolních bodech
- Měnitelné parametry umožnující stížení výpočtu posunutím hranice počtu iterací a dalších parametrů poskytuje možnost měnit náročnost za běhu dema.
- Vykreslení běží jako animace a je tak možné sledovat změny struktury dle parametrů v reálném čase

Použité technologie

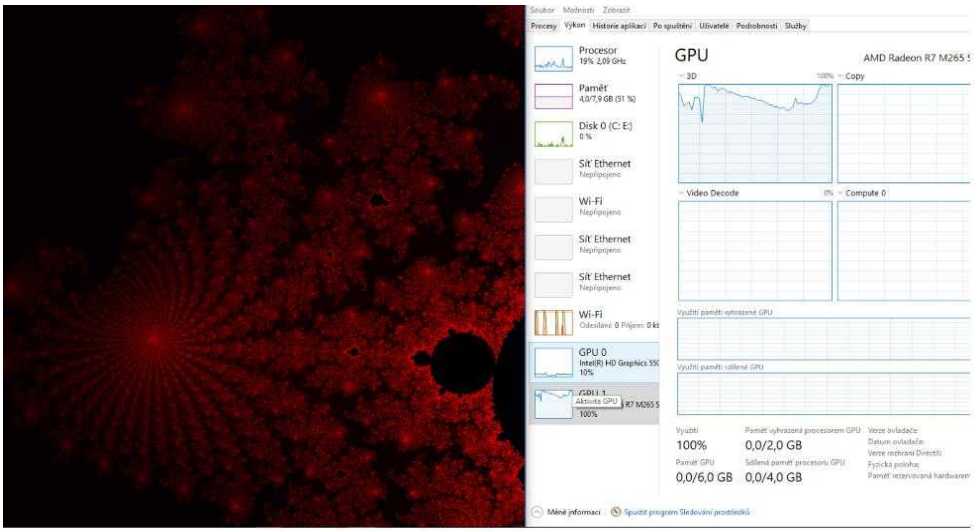
- Při tvorbě bylo použito knihoven OpenGL >=3.3, GLSL >= 1.30, SDL 1.2.15, GLEW a knihovnu „pgr“ ze cvičení.
- Program běží a je odzkoušen na platformách Windows 10 64-bit a Ubuntu 18.05 64-bit VM.
- Pro tvorbu bylo použito textového editoru Sublime a posléze IDE Visual Studio 2012.

Použité zdroje

[1] https://en.wikipedia.org/wiki/Mandelbrot_set
[2] https://stackoverflow.com/questions/4405381/how-to-zoom-mandelbrot-set
[3] http://kmlinux.fjfi.cvut.cz/~pauspetr/html/skola/fraktaly/reserse.htm#_Toc73066115
[4] https://cs.wikipedia.org/wiki/Barnsleyho_kapradí
[5] https://github.com/donqustix/MandelbrotGL
[6] http://www.relativitybook.com/CoolStuff/julia_set.html
[7] http://mrob.com/pub/muency/escaperadius.html

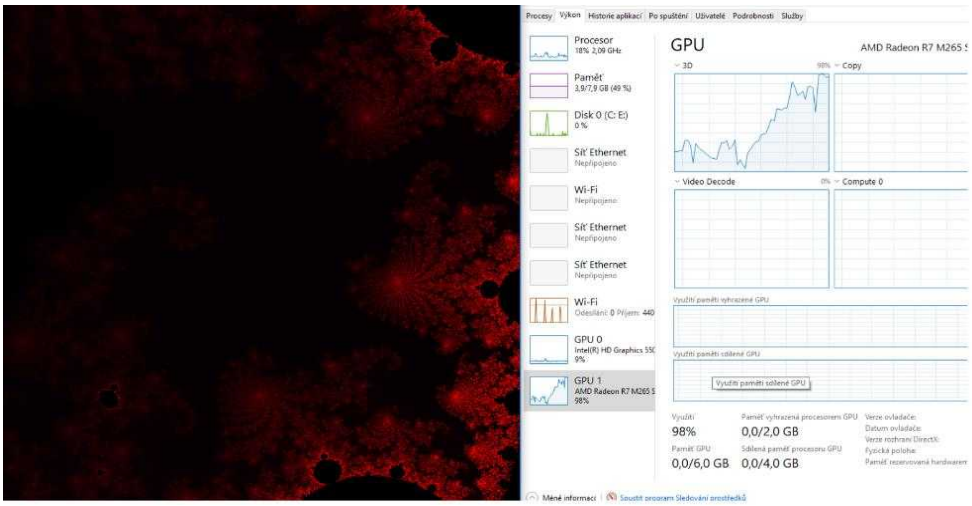
Nejdůležitější dosažené výsledky

Obecně nejdůležitějším výsledkem je úspěch při vytížení grafické karty. V závislosti na parametrech jako maximální počet iterací, množství snímků za vteřinu a velikost zobrazené plochy. Tyto parametry jsou nastavitelné za běhu.



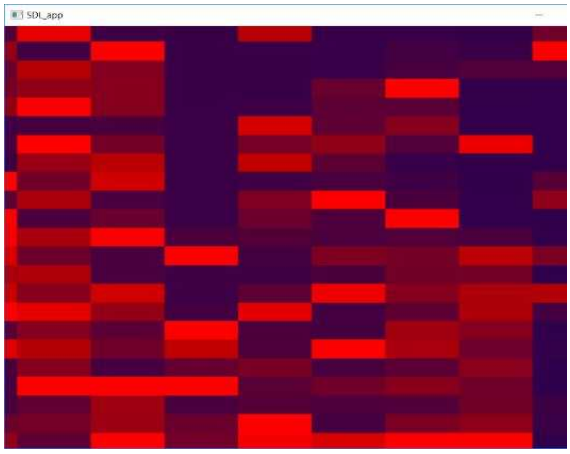
Obrázek 1: Demostrace vytížení AMD Radeon R7 265M

Nejobtížnější se při vysokém počtu iterací stávají body, které do množiny patří, protože pro takový bod je nutné vypočítat všechny iterace než je možné bod považovat za součást množiny. Tedy výpočetně náročnými a zároveň nejpůsobivějšími částmi fraktálu jsou kraje množiny v závislosti na počtu iterací.



Obrázek 2: Demostrace vytížení mac. iterací : 1000, zoom : 400

Nakonec se stane přiblížení tak velkým, že nestačí datový typ **float** pro jemnější zobrazení.



Obrázek 3: Dosažení limitu float dat. typu

Ovládání vytvořeného programu

Animaci potažmo parametry výpočtu je možné nastavovat pomocí klávesnice.

Počet iterací:

Num 1 – maximální počet iterací +10

Num 2 – maximální počet iterací -10

Num 3 – maximální počet iterací +100

Num 4 – maximální počet iterací -100

Přiblížení:

Num 6 – násobek přiblížení x10

Num 7 – násobek přiblížení x0.1

Zajímavé body:

Num 8 – iterace skrze pole bodů +

Num 9 - iterace skrze pole bodů -

Obecné parametry:

Num 5 – c, max. iterací, přiblížení

Esc - ukončení

Zvláštní použité znalosti

Fraktály

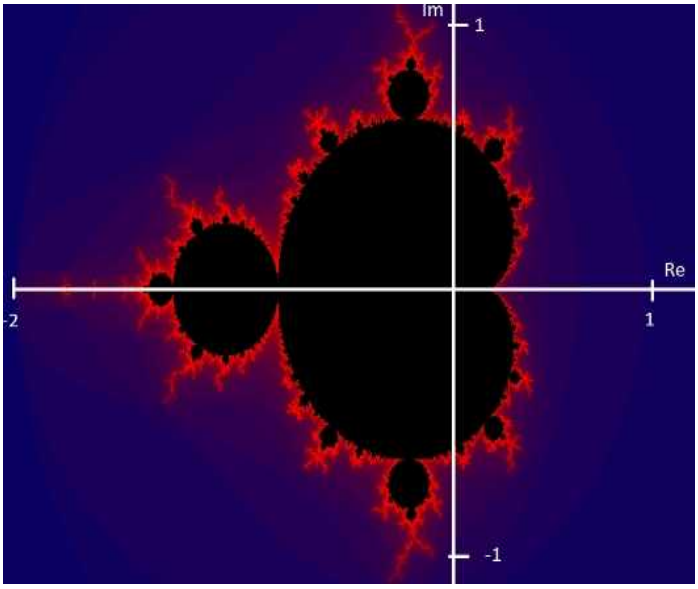
Jedná se o geometrický objekt , který po rozdělení případně přiblížení vykazuje soběpodobnost těchto jednotlivých částí. Lze jej definovat jako nekonečně členitý útvar.

Soběpodobnost tedy invariance ke změně měřítka je taková vlastnost objektu, že při jákemkoliv zvětšení vypadá objekt stále stejně či podobně. Jedná se o jeden ze základních stavebních principů v přírodě a je tedy možné soběpodobnost pozorovat takřka všude. Příkladem můžeme uvést větvení stromů, rostliny, mraky, cevní a žilní systém, povodí řek, ulity korýšů a další. Velmi zajímavým důkazem principu opakování téže struktury v přírodě je Bansleyho kapradí a modelování těchto struktur pomocí fraktální geometrie.

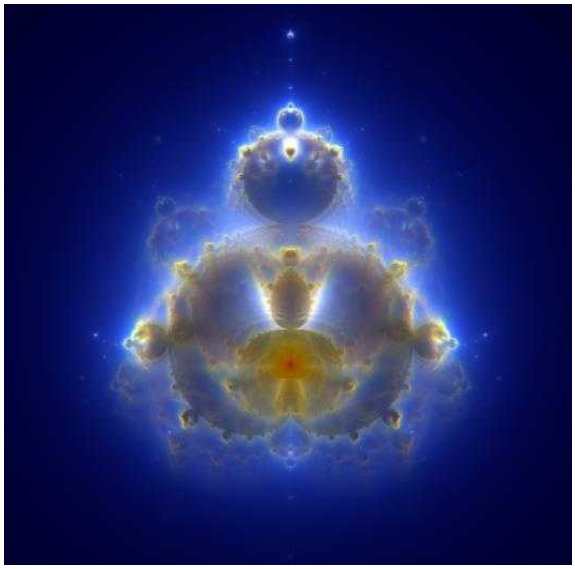
Obrázek 4: vlevo : Bansleyho kapradí , vpravo : Slezinník netíkovitý (zdroj: Wikipedia)

Mandelbrotova množina

Mandelbrotova množina komplexních čísel byla matematicky popsána francouzským matematikem Pierrem Fatou a později poprvé vygenerováná Benoitem Mandelbrotem v roce 1979.



Obrázek 6: Mandelbrotova množina komplexních čísel


Obrázek 5: Buddhabrot - Nebulabrot

Výpočet náležitosti bodu a vizualizace

Náležitost se kontroluje pozorováním posloupnosti generované pomocí vzorce :

$$z_{n+1} = z_n^2 + c$$

Kde **z** a **c** jsou komplexní čísla.

- Komplexní čísla lze napsat rozepsat jako $z = x + iy$.
- Pro **c** platí to stejné, jen zavedeme index $c = x_0 + iy_0$.
- **Z^2**, zde uplatníme vzorec pro druhou mocninu dvojčlenu a dostaneme

$$z^2 = x^2 + i\,2xy - y^2 \quad .$$

- Po rozdělení na reálnou a imaginární část dostaneme

$$zRe = x^2 - y^2 + x_0$$

$$zIm = 2xy + y_0$$

Pro posouzení konvergence či divergence počítané posloupnosti poslouží vzdálenost nově vypočteného **z** použijeme omezenost této množiny kruhem se středem (0,0) a poloměrem 2 . Tedy porovnáme absolutní hodnotu čísla **z,** zda leží uvnitř kruhu (též „úniková podmínka“) a pokud ano pokračujeme v další iteraci. Jinak iteraci končíme a počtem provedených iterací obarvíme aktuální fragment.

Únikový algoritmus je nejjednodušší algoritmus pro obarvení bodů dané množiny. Na základě počtu iterací nutných k překročení únikové podmínky je obarven jednotlivý fragment. Pro demonstraci tento algoritmus postačuje, ale program, který by byl určený pro objevování a zkoumání fraktálu, tedy jeho vizualizace, by vyžadovala lepší algoritmus pro obarvení. Pro vyhlazení je dobré diskrétní hodnoty nahradit spojitou funkcí. Další možností je barvení pomocí histogramu počtu bodů se stejným počtem iterací apod..

Vlastní výpočet spočívá v namapování souřadnic okna na intervaly z obrázku č.5. a ty případně přiblížit nebo posunout do žádaného místa při zvětšování.

Rozdělení práce v týmu

Jednočlenný tým

Co bylo nejpracnější

Příprava pracovního prostředí byla spíše zdlouhavá než pracná. Pracnější částí práce bylo pochopit samotný výpočet náležitosti bodu do Mandelbrotovi množiny a další náležitosti toho matematického konceptu v souvislosti s výpočtem na GPU.

Zkušenosti získané řešením projektu

Získané zkušenosti budou z oblasti fraktálů a jejich vykreslení. Pomocí fraktální geometrie je možné generovat celá prostředí a objekty v něm s nevídanou komplexností. Pokud se do výpočtu vnese nějaký šum či chyba, jako v reálném prostředí, je pak možné velmi přesně napodobit skutečné objekty jako stromy a další vegetací s věrohodnou nedokonalostí. V průběhu řešení projektu se ukázali některé problémy s pracovním prostředím jako zastaralé verze ovladačů a různé další komplikace spojené s testovacím strojem, kterým byl notebook se dvěma přepínatelnými GPU jednotkami Intel HD5000 a AMD Radeon R7 M265. Nicméně všechny problémy, které vyvstaly byly spíše triviálního rázu.

Autoevaluace

Technický návrh: 65%

Analýza a dekompozice problému nebyla obtížná po konečné volbě zadaní. Prostředky byly zvoleny z počátku trochu nešťastně, protože IDE Visual Studio nabízí větší komfort než obyčejný textový editor. Z hlediska návrhu se jedná pouze o vykreslení primitiva se správně obarvenými fragmenty.

Programování: 55%

Kód není nijak obtížně čitelný je poměrně krátký a vhodně okomentovaný, jedná se o upravenou kostru ze cvičení. Hlavní princip se ukrývá v Fragment Shaderu.

Vzhled vytvořeného řešení: 50%

Zde je možné komentovat pouze estetickou stránku. U fraktálu velmi těžko uchopitelná stránka, protože jsou to „nekonečně“ krásné útvary. Procentuální hodnocení nemá příliš smysl. Pokud zahrneme estetiku fraktálu, tak se blížíme stovce a pokud ne tak není co hodnotit a tedy nula, protože aplikace nemá žádné GUI, které by zbytečně kazilo dojem z pohledu na fraktální struktury.

Využití zdrojů: 65%

Největším oporou byla Wikipedie[1] a rešerše[3], kde bylo možné najít všechny potřebné informace, včetně pseudokódu pro použité algoritmy.

Hospodaření s časem: 20% (rovnoměrné dotažení částí projektu, míra spěchu, chybějící části řešení, …)

Míra spěchu byla vysoká, protože na projekt se uvolnil čas až poslední týden před odevzdáním a nedá se tak hovořit o nějakém rovnoměrném dotažení apod.. V řešení jsou všechny podstatné části , ale existuje spoustu dalších možných rozšíření.

Spolupráce v týmu: 90%

Spolupráci nekomentuji.

Celkový dojem: 55%

Pracnost a obecně náročnost všechny části projektu nebyly vysoké . Nicméně výsledek je vizuálně velmi atraktivní a fraktály obecně jsou velmi zajímavými útvary a při správně zvolených parametrech je i takto jednoduchá aplikace schopna jednoduše vytížit použitou GPU jednotku na maximum.

Doporučení pro budoucí zadávání projektů

Organizace projektů je v naprostém pořádku. Možná bych u jednotlivých zadaní zveřejnil fotografie či alespoň slovní popisy projektů, které již byly na toto téma studenty vytvořeny, aby bylo možné z něčeho vyjít a případně vědět jaké úrovně by specifikace jednotlivých zadaní měly dosahovat. Případně by sloužily jako odrazový bod pro konzultaci.