BRNO UNIVERSITY OF TECHNOLOGY
FACULTY OF INFORMATION TECHNOLOGY

Excel@FIT 2023
http://excel.fit.vutbr.cz

Zobrazení rozsáhlých volumetrických dat na CPU

Jan Svoboda

Abstrakt

Tato práce se zabývá návrhem a implementací systému, který umožňuje zobrazovat rozsáhlá volumetricka data v reálném čase na CPU běžného počítače. Práce si klade za cíl řešit jak problematiku samotného zobrazování, kdy tato data často nelze celá umístit do operační paměti stroje, tak i problematiku úložiště těchto dat, kdy v případě rozsáhlých datasetů může být jejich uchovávání v úložišti cílového počítače nežádoucí. Navržené řešení sestává ze dvou aplikací, klientské a serverové. Serverová část byla implementována na technologii ASP.NET Core a slouží jako vzdálené úložiště volumetrických dat, která jsou po malých blocích a v různých kvalitách poskytována klientské aplikaci. Klientská aplikace, implementována v jazyce C++, tato data zobrazuje metodou vrhání paprsků a dle vytvořených strategií řeší načítání a uchovávání potřebných bloků v lokální paměti. Při implementaci klientské aplikace byl kladen důraz na paralelizaci klíčových procesů pro dosažení vysokého výkonu. Výsledný systém umožňuje uživateli zobrazovat rozsáhlá volumetrická data na běžném počítači, bez nutnosti mít tato data fyzicky dostupná v rámci jeho lokálního úložiště, avšak s předpokladem propojení se serverem pomocí vysokorychlostní internetové sítě. Taktéž umožňuje uživateli provádět vzdálenou správu jednotlivých datasetů prostřednictvím jednoduchého grafického rozhraní.

*xsvobo2b@vutbr.cz, *Faculty of Information Technology, Brno University of Technology*

1. Úvod

Volumetrická data jsou využívána napříč širokou škálou technických oborů, ve kterých mají nejrůznější uplatnění. Nejčastěji se lze setkat s volumetrickými daty, která reprezentují výstupy zobrazovacích metod ve vědě a lékařství, jako jsou například počítačová tomografie, magnetická rezonance nebo kryoelektronová mikroskopie, dále jsou využívány ve vědeckých simulacích. Své uplatnění taktéž nalézají ve filmovém i herním průmyslu.

S tím jak se s technickým pokrokem zvyšuje kvalita zobrazovacích metod a simulací, úměrně také roste rozsah jimi produkovaných volumetrických dat, což zvyšuje jak požadavky na jejich zobrazování, tak i na kapacitu úložiště cílového počítače.

Z těchto důvodu je stále oblíbenější využití klient-server architektur, kdy server může sloužit buď jako pouhé vzdálené úložiště volumetrických dat, nebo může vykonávat i samotné zobrazováni. Pro zobrazování volumetrických dat existuje mnoho metod a mohou být implementovány na GPU, nebo na CPU. GPU obecně poskytuje vyšší zobrazovací výkon, avšak u běžných strojů často nedisponuje příliš velkou pamětí v porovnání s běžně dostupnými velikostmi pamětí RAM. Výhodou klient-server architektury je zjednodušení manipulace s rozsáhlými datasety a možnost sdíleného přístupu k datům, avšak za podmínky kvalitního internetového propojení mezi klientským a serverovým zařízením.

Navržené a implementované řešení sestává ze dvou aplikací, klientské a serverové, kdy serverová část slouží jako úložiště volumetrických dat, která jsou rozdělená do menších bloků v různých kvalitách. Tyto bloky jsou prostřednictvím API poskytovány klientským zařízením, která tato data zobrazují metodou vrhání paprsků. Při zobrazování je kladen vysoký důraz na co nejvyšší výkon, i na úkor mírně horších vizuálních výsledků. Řešení disponuje přívětivým grafickým uživatelským rozhraním, pomocí kterého lze snadno provádět správu jednotlivých datasetů a zobrazovat je.

2. Návrh systému

Cílem práce bylo navrhnout a vytvořit systém, který umožní zobrazovat rozsáhlá volumetrická data na

CPU běžného počítače v reálném čase, kdy tato data mohou značně přesahovat kapacitu operační paměti tohoto stroje. Jelikož i samotné uchovávání těchto dat může být značně problematické, kdy je data navíc často vhodné pro účely zobrazení předzpracovat, což však nároky na úložiště značně zvyšuje, kladla si práce za cíl vyřešit i umístění těchto dat mimo cílový počítač.

Systém byl tedy navržen pomocí klient-server architektury, kdy server, implementovaný na technologii ASP .NET Core, uchovává volumetrická data rozdělená do malých bloků, v různých kvalitách. Klientksá aplikace, implementována v jazyce C++ s grafickou knihovou Dear ImGui, dle vytvořených strategií načítá potřebné bloky ze serveru a provádí jejich zobrazování. Úlohou serveru je také převod vstupních dat do blokové podoby a umožňovat klientským zařízením provádět správu projektů.

Obecné schéma systému je znázorněno na **Obrázku 1**.

3. Server

3.1 Předzpracování dat

Předzpracování dat je proces, který je spuštěn pouze jedenkrát nad každým datasetem. Nejdříve jsou data rozdělena do bloků o velikosti mocniny dvou, které jsou následně rekurzivně ukládány a škálovány vždy na poloviční rozlišení. Bloky jsou ukládány do samostatných souborů s využitím bezztrátového kopresního formátu *GZIP*. Ten má schopnost provádět kompresi a dekompresi přímo na datových streamech, za použití pouze malého množství dodatečné paměti [1]. Data jsou zde serializovaná po řádcích a k serializaci do jiných typů dochází až na klientském zařízení. K tomuto postupu byl přistoupeno, protože nabízí více možností z hlediska komprese, umožňuje lepší ladění a taktéž usnadňuje zpracování dat serverem.

3.2 API

API umožňuje komunikaci klientkých aplikací se serverem nad protokolem HTTP. Je založeno na návrhovém vzoru RPC (volání vzdálených procedur). Poskytuje operace pro správu projektů serveru a pro získávání dat a metadat. Jednotlivé metody a princip jejich použití jsou znázorněny na **obrázku 5**.

4. Klient

4.1 Načítání dat ze serveru

Před zobrazením jsou nejdříve přednačteny všechny bloky v nejnižší kvalitě, poté, již při zobrazování, dochází k postupnému načítání bloků ve vyšších kvalitách dle potřeby.

Pro načítání dat ve vyšších kvalitách byly vytvořeny strategie, které umožňují kvalitní zobrazení i při omezené velikosti operační paměti. Potřebnou kvalitu dat zjišťují zobrazovací vlákna, ale samotné načítání kvalitnějších bloků je prováděno nezávisle v několika samostatných vláknech. Postupné načítání datasetu lze zaznamenat na **obrázku 4**.

Serializace dat je prováděna pomocí Mortonovy křivky. Ta společně s křivkou Hilbertovou poskytuje dobré výsledky z hlediska prostorové lokality [2]. Oproti té však disponuje jednodušším způsobem mapování. Mapování je zde prováděno pomocí instrukcí pro bitovou manipulaci.

V případě nedostatku paměti dochází ke snižování kvality bloků, u kterých je kvalita pro dané zobrazení zbytečně vysoká, a taktéž jsou v případě potřeby odstraňovány bloky nepotřebné.

4.2 Zobrazování dat

Data jsou zobrazována metodou vrhání paprsků (ray-casting), u jejíž implementace byl kladen důraz na rychlost, i za cenu horších vizuálních výsledků. Pro výpočet kroku zde bylo využito algoritmu DDA, kdy nedochází k opakovánému čtení hodnoty voxelu (trojrozměrný pixel) v rámci paprsku. Tohoto algoritmu pro průchod voxelovou mřížkou využívá ve své práci také například Hilton [3]. Pro účely optimalizace výpočtu indexů voxelů je na scénu aplikována transformace, která umožní zobrazování realizovat nad jednotkovou voxelovou mřížkou. Další využitou optimalizací je předčasné ukončení paprsku. Zobrazování bylo taktéž paralelizováno.

Při zobrazování dochází ke konfliktům s načítacími vlákny v sekcích, které jsou kritické z hlediska výkonu. Tyto sekce musely být proto řešeny pomocí bezzámkového programování.

5. Výsledky

Výsledný systém, jehož graifké rozhraní je vyobrazeno na **obrázku 6**, dosahoval na testovacím stroji při zobrazování scén, které jsou znázorněny na **obrázcích 2**, **3**, **4**, a pří různém počtu vláken snímkové frekvence, zobrazené v grafu na **obrázku 7**. Byla také ověřena funkčnost načítacích strategií, kdy časový průběh podílů bloků v potřebné kvalitě a velikosti dat v paměti jsou znázorněny grafem na **orbázku 8**. Experiment byl proveden na datasetu o velikosti 7 GB při maximální paměti dat 2GB.

Poděkování

Tímto bych rád poděkoval panu
Ing. Michalu Španělovi, Ph.D. za skvělé vedení této práce a za poskytnutí cenných odborných informací.

Literatura

[1] L. Peter Deutsch. GZIP file format specification version 4.3. RFC 1952, 5 1996.

[2] Toshihide Ibaraki, Naoki Katoh, and Hirotaka Ono. On the locality properties of space-filling curves. In *Algorithms and Computation*, volume 2906. Springer Berlin / Heidelberg, Germany, 2003.

[3] J.E. Hilton. Dynamic modelling of radiant heat from wildfires. 12 2017.