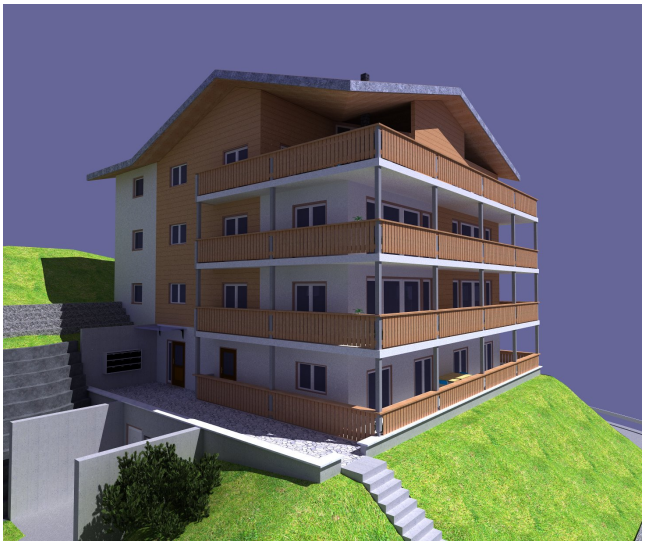
Souhrnná výzkumná zpráva k projektu Vývoj softwaru v oblasti CAD systémů, 3D grafiky a vizualizace grafických scén

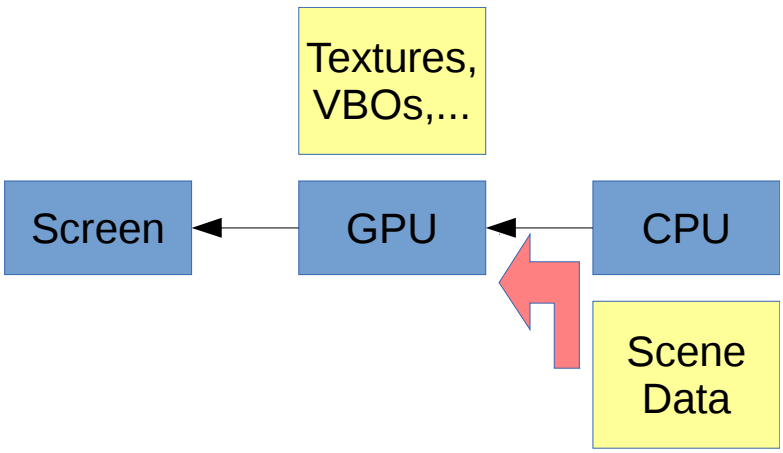
Jan Pečiva, Tomáš Milet, Michal Tóth, Tomás Starka, Jozef Kobrtek*
Fakulta informačních technologií, Vysoké učení technické v Brně

Vizualizace rozsáhlých modelů v CAD oblasti je náročným úkolem jednak z pohledu množství zobrazovaných dat, jednak z pohledu jejich fragmentace. Zákazníci v oblasti CAD návrhů, designu a architektury nezřídka vytvářejí velmi rozsáhlé soubory dat, které je potřeba vizualizovat v realném čase. Navíc tyto soubory dat jsou často fragmentovány do velmi mnoha malých elementárních částí. Zpracovat je tradičními vizualizačními postupy, které používají dnešní běžně dostupné zobrazovací knihovny, včetně počítačových her, je nevhodné. Požadavek plynulé vizualizace takovýchto dat ukázal na potřebu speciálně orientovaných algoritmů pro vizualizaci CAD modelů. Tato architektura byla vyvinuta a je prezentována touto souhrnnou výzkumnou zprávou.



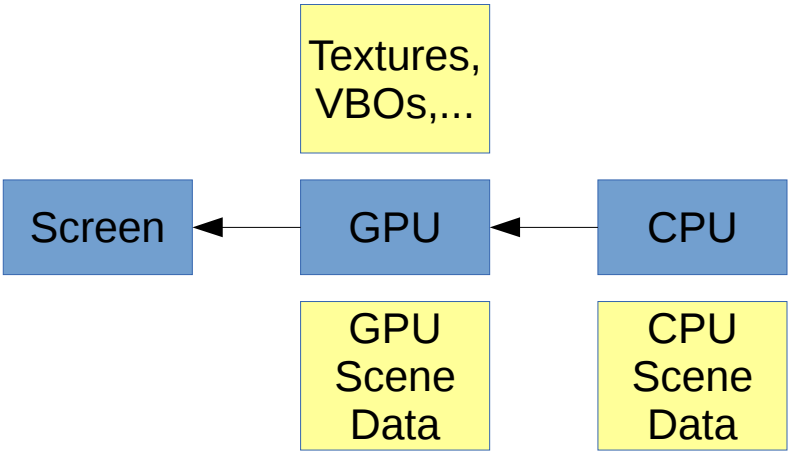
Obrázek 1: CAD model s velkým množstvím malých elementů

Hlavní řešený problém je optimalizace vizualizačního procesu. Především se pak jedná o optimalizaci zpracování dat na grafiké kartě (GPU). Pro její optimální vypočetní výkon je vhodné umístit co nejvíce dat do její grafické paměti a minimalizovat její komunikaci s hlavním procesorem počítače (CPU). Textury a geometrická data (VBO) jsou už dnes téměř výlučně umísťována automaticky na grafickou kartu, avšak jednotlivé příkazy pro vykreslování dat na obrazovku jsou typicky posílány z procesoru počítače, neboť data scény, její struktura a vykreslovací příkazy jsou obyčejně umístěna v hlavní paměti počítače a zpracovávána hlavním procesorem. Pro každý vykreslený snímek na obrazovku je potřeba projít celou scénu hlavním procesorem a poslat vykreslovací příkazy do grafické karty (GPU), jak je ukázáno na obrázku 2.



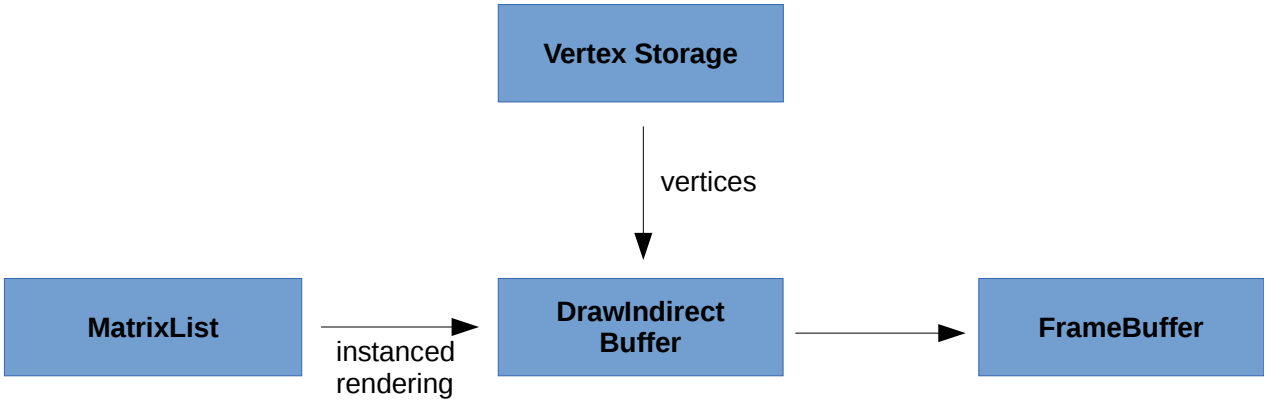
Obrázek 2: Klasický přístup k zobrazování scény. CPU řídí celý proces vykreslování.

* peciva,imilet,starka,itoth,ikobrtek@fit.vut.cz



Obrázek 3: Navrhovaný přístup zefektivnění zobrazování CAD modelů. Výkonově kritická data scény jsou přenesena na GPU a tam zpracovávána námi navrženými algoritmy.

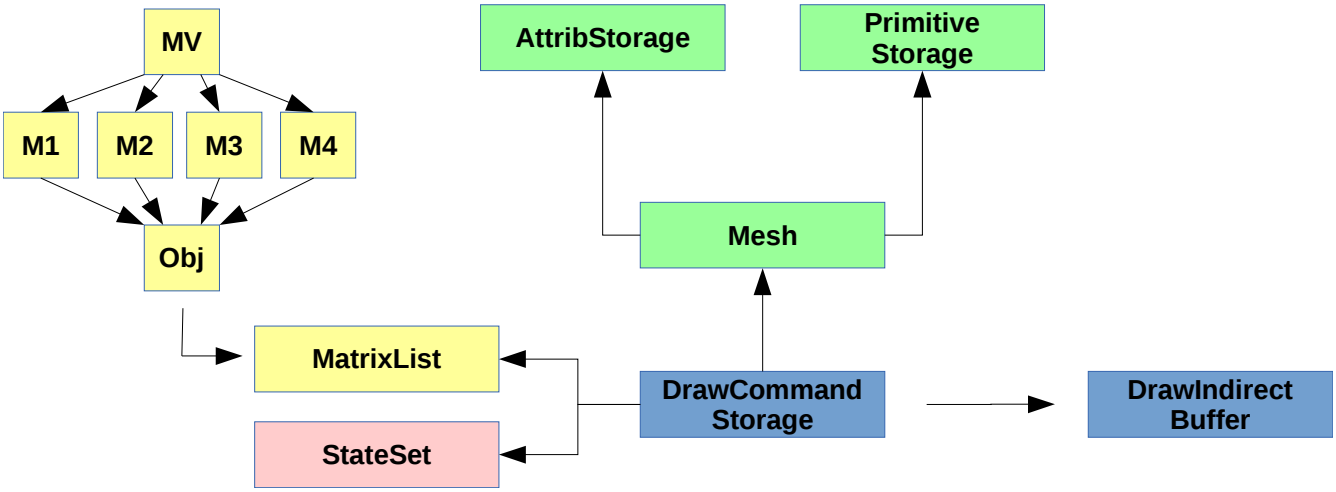
Nově navrhovaný přístup přesouvá většinu dat scény z hlavní paměti od hlavního procesoru (CPU) do paměti grafické karty, kde je může přímo využívat grafický procesor (GPU). Na straně procesoru tak zůstanou pouze kritické struktury pro alokaci, dealokaci a údržbu objektů. Celá scéna je však zobrazitelná téměř výhradně skrz grafický procesor s minimem práce na hlavním procesoru (CPU) a minimem interakce s ním. Výkonově kritická data jsou tedy umístěna na grafickou kartu a tam zpracována námi navrženými algoritmy pro rychlé zobrazení.



Obrázek 4: Princip nové architektury pro rychlé rozbrazování CAD modelů

Základní idea rychlého zobrazování CAD modelů na grafické kartě je představena na obrázku 4. Geometrická data vizualizovaného modelu jsou uložena v datových strukturách na grafické kartě, zde souhrnně nazvaných jako Vertex Storage. Geometrická data potřebují ke svému zobrazení geometrickou transformaci, zjednodušeně řečeno místo v prostoru, kde je model umístěn. Jako příklad nám může sloužit zaparkované auto, které má svoje prostorové souřadnice podle toho, kde bylo zaparkováno. Model může mít i více transformací. Například kolo auta má hned čtyři transformace. Je to zpravidla jeden model, ale umístěn či "instanciován" čtyřikrát ve scéně odpovídající čtyřem pozicím, kde má auto kolo.

Tyto transformace jsou souborně dodány ve struktuře nazvané MatrixList. Následně je spuštěn náš vlastní algoritmus na grafické kartě, který zkombinuje geometrii z Vertex Storage a transformační matice z MatrixListu a další zobrazovací informace. Na konci algoritmu je vytvořen takzvaný DrawIndirectBuffer, který obsahuje předzpracované informace pro zobrazení scény. Hlavní procesor (CPU) pak už jen, zjednodušeně řečeno, dá finální pokyn a celá scéna je zobrazena. Takto je značná část vypočetních nároků přenesena z hlavního procesoru na grafickou kartu, která má mnohem větší vypočetní kapacitu a je schopna tuto úlohu zvládnout podstatně rychleji a navíc zvládnout i úkoly, které by nebyly na hlavním procesoru výkonově řešitelné.



Obrázek 5: Detailní popis architektury pro rychlé zobrazování grafických scén a CAD modelů

Detailní popis celé architektury je uveden na obrázku 5, kde jsou jednotlivé bloky z obrázku 4 rozkresleny detailně. Geometrická data jsou uložena ve strukturách AttribStorage s popisem jejich zobrazení v PrimitiveStorage. Vazbu mezi těmito strukturami zastřešuje třída Mesh zapouzdřující funkcionalitu zobrazování trojúhelníkové sítě. Ve žluté barvě jsou znázorněny struktury týkající se prostorových transformací. Transformace jsou organizovány v grafu (Mx). K určité transformaci je pak vždy připojen MatrixList, který je vyplněn spočítanými transformačními maticemi. Třída StateSet reprezentuje nastavení zobrazovací pipeline, tedy jakým způsobem je daný objekt vykreslován. Spojením Mesh, MatrixListu a StateSetu pak vznikne DrawCommand, který je uložen v DrawCommandStorage na grafické kartě. Zpracováním DrawCommandBufferu grafickým procesorem pak vzniká DrawIndirectBuffer (terminologie OpenGL), který je přímo zpracovatelný grafickou kartou a způsobí vykreslení scény.

Projekt je k dispozici na internetu jako open source (https://github.com/Rendering-FIT/GPUEngine) a je možné si jej stáhnout a vyzkoušet.