Hardwarový fotorealismus

Grafické akcelerátory prodělaly za poslední tři roky neuvěřitelný pokrok. Hlavní podíl na této skutečnosti má slibně se rozvíjející trh s počítačovými hrami a zejména s herními konzolami. Obrovský rozvoj herního průmyslu vedl k zásadním rozšířením průmyslových standardů DirectX a OpenGL, což usnadnilo a otevřelo cestu pro vývoj na poli grafických akcelerátorů. V následujícím povídání si ukážeme nové možnosti grafických akcelerátorů vzhledem k posledním verzím standardů, tedy vzhledem k verzím DirectX 9.0 a OpenGL 2.0.

Grafickým akcelerátorem rozumíme grafickou kartu, která umožňuje kromě rastrového výstupu na obrazovku i hardwarové urychlení některých grafických operací. Je naprosto samozřejmé, že dnešní grafický akcelerátor (dále jen GA) urychluje operace, jako jsou práce s okny (vykreslování, posuv, zvětšování, překrývání), s obrázky (BitBlt, zvětšování, převody barevných palet), s fonty (vyhlazování, rasterizace, rotace...), s vektory, s videem, s kurzorem myši aj., tedy stručně řečeno 2D obrázkové a vektorové operace.

Pro kancelářskou práci tyto operace jistě dostačují, ale nároční uživatelé a pokročilé programové nástroje již vyžadují více. Výrobci hardwaru si uživatele pohotově rozdělili na kancelářsky orientované uživatele (GA má jen 2D podporu), na průmyslové uživatele (GA podporuje i 3D operace, ale bez nároku na vizuální kvalitu výstupu) a na hráče (GA podporuje mimořádně náročné 3D operace s důrazem na grafickou kvalitu výstupu). Paradoxně největší nároky na HW vybavení mají právě "hráči", což ale není v souladu se skutečností, že průmysloví uživatelé za vyšší výkon v CAD aplikacích připlatí snáze než domácí uživatelé. Postupem času se tedy hranice přirozeně posunuly a výkon GA pro průmyslovou a herní sféru je srovnatelný. Jako vedlejší efekt modernizace a pokroku v operačních systémech se posunula laťka i u kancelářských aplikací, a tak se pokročilé GA s 3D akcelerací vyskytují i v čistě "kancelářské" komunitě. V praxi to znamená, že dnes již nenajdete GA bez alespoň základní podpory 3D zobrazení.

3D akcelerace operací je založena na průmyslových standardech DirectX (Microsoft) a OpenGL (SGI). Bez těchto standardů by nebylo možné programovat aplikace s tím, že při použití příslušného GA budou jejich 3D a 2D operace významně urychleny. Pro inženýrské aplikace je pravidlem požadavek standardu OpenGL, zejména pro jeho portabilitu (je původně napsán pro unixové prostředí a dnes je navíc k dispozici na platformách Mac i PC) a přehlednost (vývojářům se jednodušeji tvoří nové aplikace). V herním průmyslu se setkáváme zejména s požadavkem na DirectX, kde je největší devizou široká podpora firmy Microsoft a s tím spojené konsekvence. Oba zmiňované standardy mají prakticky shodné vlastnosti a vývojáři jsou často nuceni vyvíjet pro obě platformy. Je velmi smutné, že se nepodařilo realizovat projekt Fahrenheit, který měl za úkol oba standardy spojit v jeden (Microsoft z projektu odstoupil s odkazem na nedostatečně rychlý vývoj na straně SGI...).

3D AKCELERACE "MINULOSTI"

Základem 3D akcelerátoru je grafický procesor - GPU (Graphics Processor Unit) -, který je uložen spolu s grafickou pamětí na kartě grafického akcelerátoru. Úkolem GPU je zpracování operací s geometrickými a rastrovými daty a zobrazovat je na monitoru. Zobrazení 3D scény pomocí GA předpokládá, že povrch 3D objektu je aproximován výhradně pomocí trojúhelníků (nebo N-úhelníků). Při zobrazování grafickým akcelerátorem se pro každou 3D scénu opakuje následující postup: Trojúhelníky složené z vrcholů, texturovacích souřadnic a normálových vektorů (kolmice k povrchu) jsou spolu s texturami přeneseny do GA, kde jsou nejprve transformovány do obrazových souřadnic, pak rasterizovány (tj. jsou vykresleny do paměti hloubky), následně jsou stínovány (je použita textura a zjednodušené stínování) a výsledek je prolnut s již vykreslenou scénou (je řešena viditelnost a základní obrazové operace jako zvětšení apod.).

Uvedený způsob zobrazení 3D scény pomocí paměti hloubky (Z-buffer nebo také Depth-buffer) se v HW akceleraci používá již od raných začátků grafiky (osmdesátá léta). Standardní 3D zobrazení povrchu těles pomocí GA bylo dříve založené pouze na zjednodušeném stínování pomocí Phon gova a Gourardova modelu. Tyto matematické modely jsou však na hony vzdáleny od fotorealistického zobrazení. Pro dosažení kvalitativně vyššího výsledku výrobcům her nezbylo než rozšiřovat zobrazení nejrůznějším nestandardním obcházením GA nebo kompletním výpočtem zobrazení na CPU počítače.

V roce 2000 však došlo k významnému zlomu na poli GA, který umožnil programátorům zasáhnout do výsledného 3D zobrazení scény pomocí CPU. Tento zlom byl zapříčiněn zejména vydáním balíku

Microsoft DirectX 8, který byl logickým důsledkem přípravy herní konzoly Microsoft XBox ve spolupráci s předními výrobci grafického hardwaru. Balík DirectX 8 významně posunul i vývoj (do té doby značně strnulého) standardu OpenGL, jehož rozšíření do verze 1.2 a později 2.0 (zejména na popud firmy Apple a vývojářů grafického hardwaru) vytvořilo úplně nové možnosti multiplatformních grafických aplikací.

Firmy ATI a nVidia, hlavní "hráči" na trhu, jako první implementovaly nový koncept do svých grafických akcelerátorů a tím začal jejich boj o nejrychlejší implementaci grafického procesoru, sběrnic a pamětí. Společnost ATI zasadila firmě nVidia zásadní ránu, když překvapila včasnou a kvalitní HW implementací rozšíření DirectX 9.0 (a OpenGL 2.0), ale nVidia svou ztrátu postupně dotáhla a nové grafické procesory obou renomovaných výrobců se drží na srovnatelné výkonové hladině.

NOVÁ GENERACE GRAFICKÝCH AKCELERÁTORŮ

Architektura nové generace grafických akcelerátorů (obr. 1) je založena na proudovém zpracování dat (data streams), kde lze pevně zakotvené funkce pro geometrická data ovlivnit pomocí tzv. Vertex Shaders (VS, obr. 2) a pro rastrová data pomocí tzv. Pixel Shaders (PS, obr. 3). Vývojáři tak mohou pomocí VS a PS aktivně programátorsky zasáhnout do architektury předchozí generace GA a tím mít plnou kontrolu nad grafickým výstupem.

Vlastní idea programování VS a PS je značně převzata z jazyků pro stínování, jako je Pixar RenderMan (RIB) nebo Stanford Real-time Shading Language (SRSL). Tyto jazyky umožňují kompletně přeprogramovat elementární stínovací algoritmy pro existující grafické programy. Díky nim lze dosáhnout maximální obecnosti úrovně renderingu, a to od jednoduchého zobrazování plošek až po nejsložitější simulaci anizotropického nebo skleněného povrchu. Právě psaní "shaders", jak se těmto elementárním stínovacím programům říká, je základem vývojových center uvnitř velkých firem, jako jsou Pixar, ILM, Dream Works, Digital Domain, Weta Digital aj.

VS a PS jsou elementární programy určené pro zmíněné samostatné výpočetní jednotky GA, které pracují s omezenou instrukční sadou podobnou assembleru. Vzhledem k tomu, že VS a PS jsou součástí složitější, proudové struktury, označují se tyto programy jako fragmentové programy (Fragment Programs, Codes).

Vstupem fragmentových programů je tzv. vertex stream (sada vrcholů trojúhelníků pro VS) a pixel stream (pixely trojúhelníků pro PS). VS a PS jsou omezeny nejen velikostí vstupních dat, ale i pamětí GA a maximální délkou programu, protože hardware GA musí provést elementární program shaderu pouze v omezeném čase. Problém kapacity programu je řešen na úrovni paralelního zpracování více výpočetními jednotkami (GA zpravidla obsahuje čtyři a více paralelních jednotek pro PS a pro VS). Instrukční sada pro VS a PS pracuje plně vektorově (tj. barva je reprezentována čtveřicí RGBA, vrchol trojúhelníku je dán souřadnicemi XYZ a případně dalším parametrem), navíc je přizpůsobena pro stínování, tudíž lze v jednom kroku počítat elementární operace, jako je vzdálenost vektorů, maticové násobení, úbytek vzdáleností aj.

Výsledný výstup otevřené architektury GA ovlivňují oba fragmentové programy VS a PS současně, ale přesto jsou některé výstupy dominantně vytvořeny jedním ze shaderů. V následujících odstavcích se pokusíme ukázat úlohu obou fragmentových programů odděleně.

VERTEX SHADERS

Vstupem pro VS je proud vrcholů zobrazovaných dat (vertex streams) s příslušnými parametry, výstupem VS je přeměněná struktura vektorů s příslušnými normálami, barvami vrcholů, texturovacími souřadnicemi apod. Interpretace vstupních dat tedy plně závisí na vlastním podprogramu VS. Vytvořený program může chápat zadaná vektorová data jako vrcholy trojúhelníků (asi nejčastěji), ale také jako řídicí body vytvářené plochy nebo procedurálního tělesa. Pro ilustraci obecnosti této implementace si uvedeme několik příkladů vlastností výstupů VS:

Vržené stíny objektů. Ze znalosti pozice světla a geometrie tělesa se vytvoří neviditelné stínové těleso (tzv. shadow volume extrusion), které se v další fázi použije při výpočtu zastíněných bodů.

Zrcadlení objektů. Pro plošná zrcadla platí, že odraz v nich je totožný s obrazem, který vznikne, pokud se na scénu díváme ze správného pohledu "uvnitř zrcadla". Pomocí VS tak lze vytvořit "novou scénu", která je vůči pozorovateli zrcadlová, a vznikne tak efekt zrcadlového povrchu, ve kterém se odrážejí okolní objekty. Snadno tak lze například vytvořit vodní hladinu, kterou lze navíc procedurálně zvlnit.

Komprese geometrie. Sběrnicí projdou komprimovaná data, která díky VS na výstupu "expandují" do plné šíře. Příkladem mohou být parametrické plochy, jako například BSpline, NURBS nebo stále více používané subdivision surfaces.

Procedurální tělesa a deformace. Na vstupu VS je pouze "kostra" geometrie s příslušnými parametry a výsledek je do výsledné podoby modifikován podle interakce, dynamiky apod. Nejčastější je vytvoření deformací v kolmém prostoru povrchu (reálné vlnění a změny geometrie těles, vítr). Speciálním

případem jsou pak procedurální tělesa (fraktály, stromy, voda, oheň), která se vytvoří až na základě VS počítaného na GPU.

Zobrazení chlupů. Povrch tělesa se speciálním VS programem pokryje "chlupy" (vytvoří se speciální elementární těleso ve směru normál).

Speciální stínování v závislosti na pohledu pozorovatele. Například osvětlování scény speciálním stínovacím algoritmem, konturové stínování (zdůraznění hraniční geometrie - hran těles) aj. K uvedeným příkladům je třeba si uvědomit, že pro VS jsou stále omezené prostředky programování (zatím max. 128 instrukcí fragmentového programu). I přes omezení lze vytvářet celé procedurální plochy typu rostliny, louky, postavy, bubliny apod., a to vše na GPU, tedy nezávisle na CPU počítače, který již může paralelně zpracovávat jiné úlohy.

PIXEL SHADERS

Výstupní proud dat z VS přechází do rasterizeru, který vykresluje každý zobrazovací troj úhelník do paměti GA. Proud dat z rasterizeru (tzv. pixel stream) je vstupem pro Pixel Shader (PS). Vstupní proud dat obsahuje mj. texturovací souřadnice, polohu bodu, příslušnosti k trojúhelníku aj. Výstupem zpracování PS jsou již výstupní barvy pixelů s příslušnou průhledností (alfa kanál) a hloubkou (Z-buffer). Pomocí podprogramu PS lze ovlivnit celkovou interpretaci výsledného zobrazení a vytvořit tak naprosto speciální algoritmus stínování každého bodu, který bude vytvořen jen pro daný účel. Nejlépe si ukážeme působnost PS opět na příkladech. Pomocí PS lze vytvářet například následující stínování povrchů: metalické, broušené, textilní, hrbolaté (bump), průsvitné, Fresnelovy nebo obecně anizotropické a zářivé povrchy. Mezi speciality současných aplikací patří simulace kůže, mraků, oblohy, vody, vln, reflexí, refrakcí, okolního mapování apod. PS lze navíc výhodně využít pro post-processing image a videoefekty, jako například detekce hran, kolorování, morfologii, segmentaci obrazu apod. V reálném čase tak lze provádět řadu operací, které byly do té doby výsadou nákladných specializovaných karet. Pro herní průmysl je významná aplikace PS pro procedurální textury, kde je možné použít například fraktální šumové povrchy, buněčné automaty, texture-bombing aj. Pro PS fragmentové programy platí (podobně jako pro VS) značné omezení co do počtu instrukcí (zatím maximálně 22), které je dáno mj. i mnohonásobně vyšším počtem zpracovávaných dat. Uvnitř architektury GA je většinou alespoň dvakrát více PS (resp. texturovacích) jednotek než pro VS.

POKROČILÉ ZOBRAZOVÁNÍ POMOCÍ GPU

Spojením PS i VS vznikají opravdu pokročilé algoritmy stínování, pro něž se vžilo označení GPU rendering. Pomocí GPU lze nyní realizovat řadu nadstandardních zobrazovacích technik v reálném čase. Za všechny uveďme několik příkladů.

Techniky **NPR** (Nonphotorealistic Rendering) představují stále žádanější "nefotorealistická zobrazení" 3D scény. Pomocí NPR je scéna zobrazena například technikou připomínající ruční malbu, perokresbu, tužku nebo obecně konturovou kresbu (cartoon). NPR zobrazení lze pomocí GPU dosáhnout například díky hranovým filtrům PS v kombinaci se stínovým tělesem VS (generátoru kontur rozhraní těleso-stín).

Techniky **IBR** (Image-Based Rendering) zobrazují 3D scény pomocí sady fotografií nebo obecně 2D obrazů. IBR zahrnuje i osvětlovací modely, pro něž se využívají světelné mapy okolí (tzv. Light-Probes). Důkazem toho, že výpočty pomocí PS a VS jsou opravdu mocné, je demo od průkopníka IBR Paula Debevece (obr. 5). Scéna se skleněnými a zrcadlovými koulemi, nasvětlená reálným okolním světlem, je počítána v reálném čase v rozlišení 1280 x 1024. Při výpočtu je využit kompletně přepracovaný Wardův stínovací model. Stejný výpočet trvá programu Radiance řádově minuty na jeden obrázek...

Procedurálnost zobrazení pomocí GPU umožňuje například algoritmicky "ochlupit" povrch libovolného tělesa i s příslušnými dynamickými vlastnostmi. Demo vývojáře Tomohideho Kana (obr. 4) používá VS pro generování příslušné normálové plochy, PS se pak starají o textury a věrné zobrazení anizotropického povrchu každého zobrazeného chloupku. Procedury PS umožňují generovat i procedurální obrazy, jakými jsou Mandelbrotovy nebo Juliovy fraktály. Dalším příkladem pokročilého výstupu jsou zvlněné (bump), průhledné a zrcadlové povrchy, které využívají kromě standardních textur tzv. normálové mapy a pomocný buffer (stencil plane) pro zobrazení zrcadlených povrchů.

Na GPU se podařilo přenést i nejpokročilejší techniky zobrazení, jako je **sledování paprsků** (ray-tracing) a metody **globálního osvětlení** (global illumination). Díky těmto technikám je možné v současné době zobrazit (téměř) v reálném čase (asi 5 snímků/s) "přesné" vržené stíny, radiozitu, mnohonásobné odrazy, lomy světla, měkké stíny aj.

PROGRAMOVÁNÍ GPU

Programování GPU je možné od standardů OpenGL 1.2 a DirectX 8. V prvních verzích standardů nalezneme řadu omezení co do velikosti vstupních dat i co do přesnosti proměnných a vektorů (celočíselné hodnoty jednoduché přesnosti). Řadu omezení překonávají poslední verze standardů, tj. OpenGL 2.0 a DirectX 9.0, kde vektorová data jsou obecné matice a vstupní data jsou v obecném neceločíselném formátu. Uvedené vlastnosti dělají z GPU obecný, vektorově a rastrově orientovaný koprocesor kompetitivní s CPU.

Vytvářet fragmentové programy pro PS a VS není snadné. Je nutné si uvědomit, že program je vždy spouštěn jen "jednosměrně" (proudově), musí se provést v předem omezeném čase a navíc se musí počítat na více výpočetních jednotkách najednou. Proudový způsob provádění neumožňuje přímým způsobem implementovat smyčky a větvení (je třeba vytvářet víceprůchodové "lineární" programy). Při psaní programů je třeba dbát i na omezenou velikost texturovací paměti (dnes až 256 MB), kam se musí vejít nejen geometrie, textury, pomocné buffery a vlastní fragmentové programy, ale též vytvořená procedurální tělesa a obecně všechny výstupy PS a VS.

Pro psaní fragmentových programů si lze vybrat hned několik jazyků. Microsoft integroval do svého Visual Studia jazyk HLSL (High-Level Shader Language), který svou strukturou připomíná takový "maticový assembler". Lze však využít i trochu "lidštější" jazyky, jako je CG (podporován firmou nVidia), který připomíná jazyk C a umožňuje programátorský komfort včetně generování smyček, větvení aj. Další možností je zmiňovaný jazyk SRSL nebo specializovaná prostředí, jakými jsou ATILLA, ShadeLab nebo knihovna SUSHI od vývojářů firmy ATI.

Výrobci HW se předhánějí v otevřenosti svých produktů a dodávají zdarma speciální nástroje pro ladění PS a VS. Například firma ATI dodává zajímavý nástroj Render Monkey, který mj. umožňuje importovat 3D scény včetně textur a interaktivně zkoušet naprogramované nebo vlastní VS a PS shadery. Podobné (i když mnohem jednodušší) prostředí pro zkoušení fragmentových programů nazvané CgFX Viewer má i firma nVidia. V brzké době se jistě objeví i oficiální nástroj pro přímý převod jazyka RenderMan (RIB) do fragmentových programů. Naznačuje to skutečnost, že nVidia nedávno zakoupila společnost Exluna, která mj. vyráběla populární zobrazovací program BMRT, interpretující právě rozšířený jazyk RIB.

VYUŽITÍ GPU V PRAXI

Je zřejmé, že pokročilé zobrazování pomocí GPU využívají zejména vývojáři počítačových her (např. poslední verze her Doom nebo Unreal Tournament). Obrovskou sílu pokročilých výpočtů na GPU však dnes využívají i "seriózní" průmyslové aplikace, například špičkové 3D modelovací a kompoziční programy Maya (Alias) a Softimage (Avid). Oba zmíněné programy obsahují možnost tzv. real-time preview, při nichž jsou scény zobrazovány pomocí GPU a příslušných real-time shaderů (PS a VS programy). Výsledek real-time preview je mnohdy kvalitativně shodný s opravdovým programovým zobrazením (software rendering), přičemž je minimálně dvacetkrát rychlejší. Omezujícím faktorem pro plné využití je pouze velikost texturovací paměti a "šíře" sběrnice. Využití GPU pro preview již implementují i další vývojáři, jako například NaN (Blender 3D), Maxon (Cinema 4D) aj.

Již tento rok se objevily i první implementace grafických akcelerátorů pro PDA a mobily. První portace standardu OpenGL pro PDA je na světě, a tak se s VS a PS setkáme i na přenosných kapesních počítačích. Zde bude vývoj ještě ovlivněn omezeným rozlišením, které vyžaduje další standardizaci.

CO BUDE DÁL...

Konstatování, že už prakticky každý kancelářský PC je vybaven 3D grafickým akcelerátorem, již dnes asi nikoho nepřekvapí. Přesto si jistě vzpomínáte, že před třemi lety to ještě nebylo běžné a pro mnohé to bylo i těžko představitelné. Současné grafické karty a standardy však začínají překonávat nejednu predikci grafických optimistů a neuvěřitelné se stává skutečností nové akcelerátory obsahují obecné implementace pokročilých zobrazovacích algoritmů, které se dnes používají téměř výhradně pro speciální efekty ve filmech a v reklamách.

Rychlost sběrnic grafických adaptérů dosahuje 2,1 GB/s, GPU je již dnes výpočetně rychlejší než CPU, počet paralelních aritmetických jednotek je čtyři a více, grafická paměť je 256 MB... Zdálo by se, že kromě paměti, paralelních jednotek a rychlosti GPU již není co vylepšovat. Logicky vyvstává otázka: Co bude dál?

Odpověď je nejasná, ale směr vývoje naznačují poslední vědecké konference, jako je SIGGRAPH nebo Rendering Workshop 2003. Na nich se již neprobírala klasická témata zobrazení 3D scén pomocí grafických adaptérů. Hlavním tématem se stalo využití GPU jako specializovaného, vektorově orientovaného koprocesoru pro "seriózní výpočty" a simulace. Moderní architektura umožňuje pomocí GPU počítat kolize objektů, nelineární optimalizace, nebo dokonce soustavy lineárních rovnic. Ale o tom zase někdy příště...

Jan Buriánek

INFOTIPY

www.ati.com/developer Oficiální stránky ATI
www.nvidia.com Oficiální stránky nVidie
www.debevec.org Real-time implementace HDR renderingu pro Radeon 9700 PRO
www.opengl.org Stránky OpenGL
www.microsofr.com/directx Stránky DirectX
www.pixar.com Stránky společnosti Pixar (autoři programu RenderMan)
www.bmrt.org Stránka programu Blue Moon Rendering Tool (nyní zakoupen nVidií)