REYES using DirectX 11

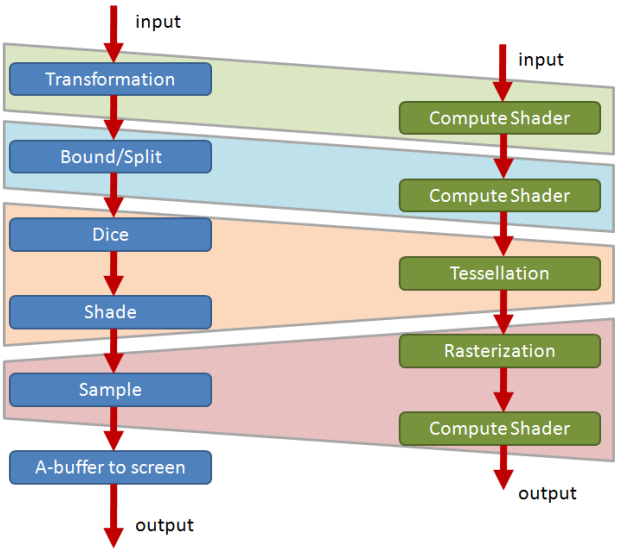
Andrei Tatarinov
NVIDIA

REYES is a common approach to generating fine-quality pictures, which is widely used in movie industry for creating CG and cartoons. REYES pipeline is a micro-polygonal pipeline, which subdivides and dices any input primitive into sub-pixel primitives, which are shaded in object-space and sampled in screen-space by a huge number of samples, thus achieving high-quality analytical anti-aliasing and precise attribute interpolation, which help to create fine images that can be used in movies and cartoons.

REYES is a highly parallel pipeline by its nature, performing similar actions on a huge set of primitives, making it a good task for GPU to solve. Several publications were made, describing different ways of implementing REYES on GPU, including

- Kun Zhou et al., "RenderAnts: Interactive REYES Rendering on GPUs", 2009
- Anjul Patney and John D. Owens, "Real-Time REYES Style Adaptive Surface Subdivision", 2008
- Andrei Tatarinov, Alexander Kharlamov, "Alternative Rendering Pipelines on NVIDIA CUDA", 2009

These implementations are mostly using Compute capabilities of GPU, without using GPU's graphics pipeline, or any particular part of graphics pipeline. Nowadays DirectX11-capable GPUs are becoming available, making it possible for software engineers to use DirectX11 features to accelerate existing and develop new rendering approaches and techniques. DirectX11 presents new rendering pipeline, which includes Tessellation stage, and new feature, which is called Compute and is a natural addition to graphics pipeline, providing fast interop between compute and graphics. The main purpose of this talk is to show that REYES can benefit from both of these features, achieving better performance.



The scheme above shows how REYES pipeline can be mapped on DirectX11 pipeline. Subdivision stage requires recursion – it performs recursive subdivision of input primitives, and does bounds-check on every iteration. Tessellation with stream-out can be used to perform this stage using graphics pipeline, or DirectCompute can be used. In case when DirectCompute is used, persistent threads are implemented which read and write data to the same input buffer, allowing recursion. Performance details and analysis of both approaches are provided during the talk.

Tessellation stage of DirectX11 pipeline looks like a perfect candidate to be used for dicing stage of REYES pipeline, since dicing stage commonly uses tessellation to generate sub-pixel quads from a set of subdivided primitives. DirectX11 Hull and Domain shaders are used to compute dicing factors based on a shading rate, and non-programmable tessellation unit is used to generate UV-coordinates for micro-quads.

Dicing stage of REYES pipeline alone usually generates huge amounts of data, which can't fit into GPU's on-board memory. This is why this stage needs to be pipelined with subsequent stages – shading and sampling, so that data generated during dicing would be processed immediately and not be stored anywhere. Shading in REYES is done per-vertex, not per-pixel, this is why Domain shader perfectly fits to be used to implement this stage. Rasterization stage of GPU graphics pipeline is used to perform sampling. All primitives are rendered to a highly-supersampled rendertarget, providing high levels of anti-aliasing and solving a problem of GPU being inefficient at rendering polygons of sub-pixel size (due to one pixel being expanded into a whole region of pixels in a supersampled surface).

One of the main difficulties in Compute-based implementation of dicing, shading and sampling stages is the need to combine them into a pipeline (due to memory pressure problem described above), which involves solutions like uberkernels, data storage structures and task-switching schemes. The idea of using DirectX11 tessellation to perform dicing is to fully utilize hardware features of GPU, since GPU already has intermediate primitive storage structures and scheduling mechanisms which can be used for REYES purposes. This eliminates the overhead required for maintaining custom memory structures and performing custom scheduling schemes. Transition from Compute to DirectX11 allowed author to achieve speed-ups up to 50% (with equal shading rates and sample counts). While subdivision stage runs at approximately the same speed for both CUDA and Compute, it adds negligible value to the overall performance. However, using DirectX11 tessellation to pipeline dicing, shading and sampling stages allows significant performance gain.

During the talk author is going to show the advantages of using DirectX11 and graphics pipeline to implement REYES compared to purely compute-based approaches (both from development time and execution time perspectives). Author provides performance analysis of different stages of REYES pipeline, including comparison of software and hardware rasterization and benefits of using natural pipeline of GPU.

SIGGRAPH 2010, Los Angeles, California, July 25 – 29, 2010.
ISBN 978-1-4503-0210-4/10/0007