Tiled and Clustered Forward Shading

Supporting Transparency and MSAA

Ola Olsson, Markus Billeter and Ulf Assarsson
Chalmers University of Technology

Figure 1: *Shot from the Crytek Sponza scene with semi-transparent bubbles added, lit by 1024 random lights. With 8x MSAA at 720p resolution, Tiled Deferred runs at 53 FPS (without bubbles), Tiled Forward at 52 FPS and Clustered Forward at 161 FPS on a GTX 680. The diagrams illustrate how transparent geometry affects clustered and tiled forward shading.*

1 Abstract

We present details of Tiled and Clustered Forward Shading in its application to rendering transparent geometry and using Multi Sampling Anti Aliasing (MSAA). We detail how transparency and MSAA is supported, and present performance results measured on modern GPUs.

Previous techniques for handling large numbers of lights are usually based on deferred shading [Andersson 2009; Lauritzen 2010]. However, deferred shading techniques struggle with impractically large frame buffers when MSAA is used, and make supporting transparency difficult. In addition, deferred shading makes it more difficult to support custom shaders on geometry.

Tiled Forward Shading is a new and highly practical approach to real-time shading scenes with thousands of light sources, introduced by Olsson and Assarsson in 2011 [2011]. Their results, measured on an GTX 280 GPU, indicated that tiled forward shading was impractically slow. Performance on more recent GPUs has improved considerably (approaching that of tiled deferred), which opens up the possibility of using the technique to support transparency and MSAA.

Clustered Shading further extends tiled shading by adding depth partitioning [Olsson et al. 2012]. We show how Clustered Forward Shading can be extended to support transparency efficiently.

Forward shading naturally supports both transparency and MSAA, which has been shown in previous work. However, the performance and implementation details have not previously been investigated.

We provide details on how to construct the light grid for use with transparency. When the transparent geometry is considered, the depth range optimization cannot be fully used. Instead, only a more conventional hierarchical depth test can be used. The grid structure can be built once, and quickly pruned to prepare a more efficient instance for opaque geometry. However, as each transparent layer must consider all the lights in the tile, performance does not scale linearly with the depth complexity, but far worse (Figure 1, right).

To improve on this we extend clustered forward shading by constructing the grid using a pre-pass over all geometry (not just opaque), and flagging clusters as a side effect. This allows us to quickly find the unique clusters used. As clusters contain only space around actual samples that need shading, efficiency is much better (Figure 1, left).

For deferred shading a single 1080p, 16x MSAA, 16-bit float RGBA buffer requires over 250Mb of memory. In addition, each sample may need to be shaded individually, effectively running shading at a per-sample frequency. For forward shading, no G-Buffers are required and MSAA is trivially enabled.

A brief performance and memory comparison is shown in Figure 2, showing that clustered forward outperforms tiled forward by more than 2 times, and also outperforms tiled deferred, if MSAA is used.

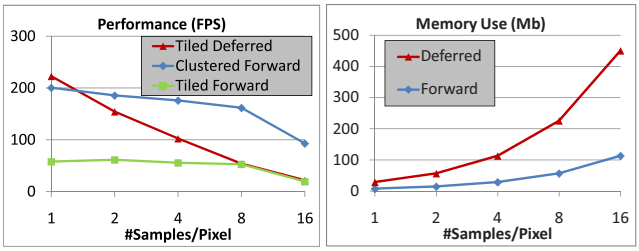

Figure 2: *Left, performance for a view similar to Figure 1 (deferred without bubbles). Right, memory use of deferred vs. forward at 720p, assuming 32-bit depth and color targets, and 3 × 64-bit G-buffers.*

References

ANDERSSON, J., 2009. Parallel graphics in frostbite - current & future. SIGGRAPH Course: Beyond Programmable Shading.

LAURITZEN, A., 2010. Deferred rendering for current and future rendering pipelines. SIGGRAPH Course: Beyond Programmable Shading.

OLSSON, O., AND ASSARSSON, U. 2011. Tiled shading. *Journal of Graphics, GPU, and Game Tools 15*, 4, 235–251.

OLSSON, O., BILLETER, M., AND ASSARSSON, U. 2012. Clustered deferred and forward shading. In *HPG '12: Proceedings of the Conference on High Performance Graphics 2012*.

SIGGRAPH 2012, Los Angeles, California, August 5 – 9, 2012.
ISBN 978-1-4503-1435-0/12/0008