BibTeX:  
@INPROCEEDINGS{7251842,   
author={C. F. Chang and K. W. Chen and C. C. Chuang},   
booktitle={2015 IEEE International Conference on Digital Signal Processing (DSP)},   
title={Performance comparison of rasterization-based graphics pipeline and ray tracing on GPU shaders},   
year={2015},   
pages={120-123},   
abstract={The interactive rendering of 3D computer graphics has approached the photorealistic quality, as evident by the vivid shading effects and lush outdoor scenes in recent game engines. Clearly, the traditional 3D graphics APIs are reaching their limits, and the need to switch to more extendable ray-tracing based techniques has arisen. In this work, we explore the fundamental differences between ray tracing based and rasterization based techniques, including how they are supported by the processors and by the programming platforms. We duplicate the typical shading effects in both ray tracing and rasterization, starting from the simple Phong lighting, to slightly more complex Whitted-style shadow and reflection. Although the rasterization-based techniques clearly outperform ray tracing in current generations of graphics processors, we show by more precise quantitative analysis that the performance gaps are not as wide as thought. And the gap may narrow further when the requirement of image

quality increases in the future.},

BibTeX [contd..]

keywords={graphics processing units;ray tracing;rendering (computer graphics);3D computer graphics;GPU shaders;Phong lighting;Whitted-style shadow;graphics processors;lush outdoor scenes;rasterization-based graphics pipeline;ray tracing;vivid shading effects;Graphics processing units;Lighting;Ray tracing;Rendering (computer graphics);Three-dimensional displays;GPU shaders;rasterization;ray tracing},   
doi={10.1109/ICDSP.2015.7251842},   
ISSN={1546-1874},   
month={July},}

**Bibliography**

In the graphics pipeline, apart from using the rasterization methods onto each primitive, there has arisen a necessity to switch to more extendable ray-tracing based techniques.

To understand the fundamental difference in these two techniques, I have chosen this article which demonstrates a conceptual over view and their implementation patterns. The methods in which they are supported by the processors and by the programming platform is discussed providing the pros and cons of each techniques.

We duplicate the typical shading effects in both ray tracing and rasterization, starting from the simple Phong lighting, to slightly more complex Whitted-style shadow and reflection. Although the rasterization-based techniques clearly outperform ray tracing in current generations of graphics processors, we show by more precise quantitative analysis that the performance gaps are not as wide as thought.

In my understanding, rasterization-based approaches run much faster than ray tracing because in the past, the GPUs were specially designed hardware to support rasterization-based 3D graphics pipelines. However, ray tracing followed a more general framework that supports more complex effects such as shadow and reflection natively but were not supported by the graphics processors. I am willing to follow more of these articles and contribute in the field of computer graphics.

**References:**

* <http://ieeexplore.ieee.org.libproxy.uml.edu/xpls/icp.jsp?arnumber=7251842>
* <http://ieeexplore.ieee.org.libproxy.uml.edu/search/searchresult.jsp?newsearch=true&queryText=shaders%20in%20computer%20graphics>
* UML Library guides

"This is entirely my own work, except as disclosed in the documentation. I gave help to the following persons:  
None  
Signed Kiran C Shettar"