Bulletin of Networking, Computing, Systems, and Software – www.bncss.org, ISSN 2186-5140
Volume 7, Number 1, pages 48–49, January 2018

Acceleration Techniques for Ray Tracing based on Parallel Methods

Jin Okaze, Chikatoshi Yamada, Kei Miyagi
National Institute of Technology, Okinawa College
905 Henoko, Nago-shi, Okinawa
{ac164603@edu., cyamada@, k.miyagi@}okinawa-ct.ac.jp

Shuichi Ichikawa
Toyohashi University of Technology
1-1 Hibarigaoka, Tempaku-cho, Toyohashi
ichikawa@tut.jp

***Abstract*—Ray tracing is a method of rendering for 3DCG (3-Dimensional Computer Graphics), and it is possible to produce realistic image. However, high-speed processing is required because its cauculation of lights become huge. In this article, we suggest structure about assigning SP (Streaming Processors), aim for to speed up ray tracing algorithm by combining screen separation on GPUs and screen mapping effectively.**

I. Introduction

3DCG (3-Dimensional Computer Graphics) are graphics that use three-dimensional spatial information. In recent years, an opportunity of using 3DCG techniques has increased due to enhance of a performance of computing technology[1]. There is a method of a 3DCG technique called ray tracing. Ray tracing simulates a behavior of light, and it is possible to produce realistic Computer Graphics. However, high-speed processing is required because its calculation of lights becomes huge[2]. So, we suggest structure about assigning SP (Streaming Processors), aim for to speed up ray tracing techniques by combining screen separation on GPUs and screen mapping effectively.

II. Ray tracing

Figure 1 shows the mechanism of ray tracing. Ray tracing is one of the methods of a 3DCG technique. It calculates a behavior of light by using a cross decision processing as follows. The vector from vantage point to any point at a rays, $P(t)$, is obtained by formula (1) : where e represents vantage point, t represents distance to any point, and d represents a direction vector.

$$\vec{P}(t) = e + t\vec{d} \tag{1}$$

Below is a variation of formula (1) : where r represents a radius of a circle.

$$t^2 - 2t(\vec{v} \cdot \vec{d}) + (\vec{v} \cdot \vec{v}) - r^2 = 0 \tag{2}$$

Following formula (2), you can calculate a cross decision processing as formula (3).

$$D = (\vec{v} \cdot \vec{d}) - (\vec{v} \cdot \vec{v}) + r^2 \geq 0 \tag{3}$$

III. Methods

We aim for to speed up ray tracing techniques using the following methods.

- Screen separation on GPUs
- Screen mapping
- Assigning SP(Streaming Processors)

A. Screen separation on GPUs

Figure 2 shows screen separation on GPUs. Using parallelism of ray tracing algorithm, a method of Screen Separation parallelizes calculation of luminance values[3].

B. Screen mapping

Screen mapping means pre-calculation of an existence of an object. If there are no objects, the calculation in the extent can be reduced[4].

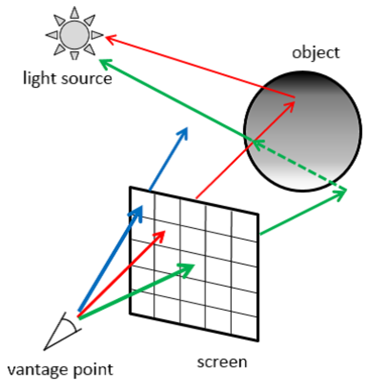

Fig. 1. ray tracing

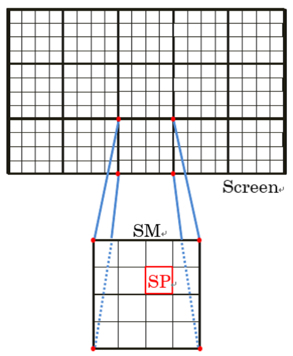

Fig. 2. Screen separation on GPUs

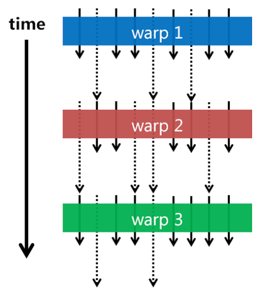

Fig. 3. Before assigning

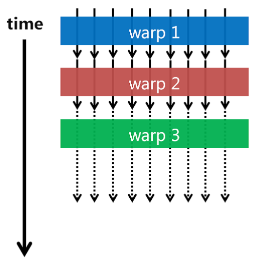

Fig. 4. Intentional assigning

TABLE I
IMPLEMENTATION ENVIRONMENT

OS	Windows7 Enterprise
CPU	Intel Core i7-860 2.80GHz
GPU	NVIDIA GeForce GTX TITAN
Memory	16GB
Development (CPU)	Microsoft Visual Studio 2015(C++)
Development (CPU+GPU)	CUDA 8.0

C. Assigning SP(Streaming Processors)

Assigning SP means intentionally assigning SP depending on the result of the screen mapping as you can see in Fig. 3 and Fig. 4.

IV. Experiments and Results

A. Implementation environment

The implementation environment is shown in Table I.

B. Experiment 1

We implemented the ray tracing on a CPU. Figure 5 shows the image generated by executing the ray tracing on a CPU. We succeeded generating the correct image, however it took time to executing the ray tracing on a CPU, about 4 minute.

C. Experiment 2

Then, we conducted the preliminary experiment for speeding up ray tracing on GPUs. The method of Experiment 2 is shown in Fig.6. By assigning SP in the different parts of the calculation (not a ray tracing), we investigated effects on the execution time. Table II displays the result of Experiment 2. It indicates a speed-up of 23% by different way of assigning.

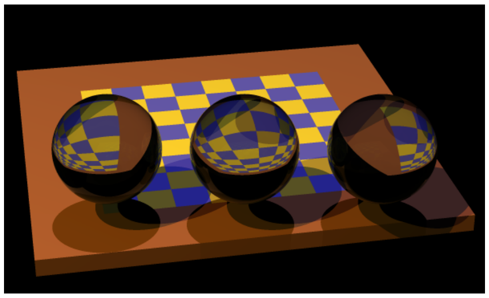

Fig. 5. Results of Experiment 1

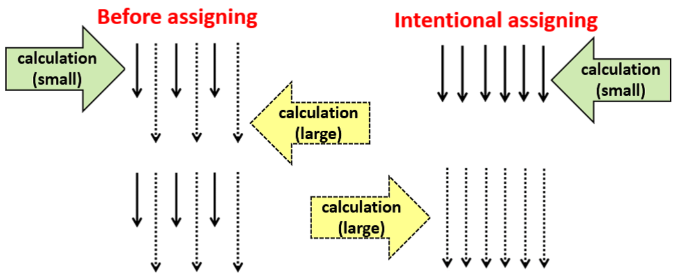

Fig. 6. Experiment 2

TABLE II
RESULTS OF EXPERIMENT 2

status	speed($msec$)	performance($Gflop/s$)
Before assigning	6.082	21.551
Intentional assigning	4.9334	26.57

V. Discussion and Conclusion

As shown in Table II, to change of the way of assigning SP may bring the increased execution speed. Although we haven ’t executed the ray tracing on GPUs yet, we have a good prospect to speed up the ray tracing on GPUs in these experiments.

Acknowledgments

This work was supported by JSPS KAKENHI Grant Number JP17K01063.

References

[1] S. Masuda, et al.: ”Speedup of Raytrac-ing Using GPU,” IPSJ Kansai branch meeting (2012).
[2] K. Ueno, et al.: ”Implementing Real-time Raytracing using GPU,” The 74th National Convention of IPSJ (2012).
[3] L. Meng, et al.: ”Paralleization of Real-time Ray Tracing Using GPU,” FIT2012 (2012).
[4] D. Okazaki, et al.: ”Real-time Raytracing using Space Partitioning on GPUs,” IPSJ Kansai branch meeting (2012).