PROCEEDINGS

of the Union of Scientists - Ruse

Book 5

Mathematics, Informatics and Physics

Volume 9, 2012

RUSE

The Ruse Branch of the Union of Scientists in Bulgaria was founded in 1956. Its first Chairman was Prof. Stoyan Petrov. He was followed by Prof. Trifon Georgiev, Prof. Kolyo Vasilev, Prof. Georgi Popov, Prof. Mityo Kanev, Assoc. Prof. Boris Borisov, Prof. Emil Marinov, Prof. Hristo Beloev. The individual members number nearly 300 recognized scientists from Ruse, organized in 13 scientific sections. There are several collective members too – organizations and companies from Ruse, known for their success in the field of science and higher education, or their applied research activities. The activities of the Union of Scientists – Ruse are numerous: scientific, educational and other humanitarian events directly related to hot issues in the development of Ruse region, including its infrastructure, environment, history and future development; commitment to the development of the scientific organizations in Ruse, the professional development and growth of the scientists and the protection of their individual rights.

The Union of Scientists – Ruse (US – Ruse) organizes publishing of scientific and popular informative literature, and since 1998 – the "Proceedings of the Union of Scientists- Ruse".

BOOK 5

"MATHEMATICS, INFORMATICS AND PHYSICS"

VOLUME 9

CONTENTS

SOFT SHADOWS FOR GPU BASED RAY-TRACING

Tzvetomir Vassilev

Angel Kanchev University of Ruse

Abstract: *This paper presents an approach for generating soft shadows with distributed ray-tracing. It uses a disk light source and unlike the classic algorithm where a set of shadow rays are cast to a collection of random locations on the area light source, the points on the disk are pseudo random and aim at achieving a wider penumbra with a smaller number of shadow rays. Results of the tests are presented at the end of the paper. The software was implemented using the NVidia OptiX ray tracing engine, which allows rendering reasonable quality scenes at 10-15 frames per second.*

Keywords: *Ray tracing, GPU computation, Soft Shadows*

INTRODUCTION

Shadows play an important role for increasing the realism in computer graphics. Ray-tracing was introduced as a method for visibility calculation and shadow determination. It produces very realistic scenes, but is very slow and unsuitable for real time rendering.

Over the past few years the continual development of graphics processing unit (GPU) technology made these devices attractive for more than just rendering. The today's modern GPUs are very powerful computationally, capable of parallel processing. They outperform the modern central processing units (CPUs) in floating-point calculations. In addition high-level programming languages have been developed for GPUs, which made them popular for speeding up all kinds of computational tasks.

Recently NVIDIA has released their OptiX ray-tracing engine. OptiX programs are executed on the GPU, computations for each ray are performed in parallel, which speeds up the ray-tracing the makes it possible to render complex scenes in real time. This work is based on distributed ray tracing and it proposes an approach to sampling the light source, which produces better quality shadows.

The rest of the paper is organized as follows. The next section reviews previous work on shadows generation. Section 3 describes the nature of ray-tracing and how hard and soft shadows are generated. Section 4 presents an approach to computing pseudo random sampling points on an area light source for distributed ray-tracing. Section 5 gives results of the experiment and Section 6 concludes the paper.

PREVIOUS WORK

A good classic survey of shadow algorithms was done by Woo et al [1]. Hasenfratz et al. gave a more up to date review on recent work [2], with a focus on hardware accelerated shadows.

Point light source used with ray tracing results in a hard shadow. The principle behind ray tracing for hard shadow determination is very simple: a shadow ray is cast from the intersection point to the light source. If the ray intersects an object between its origin and the light source, then it is in shadow, otherwise it is not. Cook et al. [3] proposed a distributed ray tracing method for soft shadow generation. A collection of shadow rays is shot from the intersected point to randomly generated locations on the light source. The number of rays is proportional to the illumination of the region if the rays were completely unoccluded and proportional to the projected area of the light source as seen from the surface. The intensity of the penumbra depends on the number of intersections of those rays with occluding objects. Because the points on the light source are randomly generated, in order to achieve good quality of penumbra usually the number of shadow rays is quite high. Recently, Lane et al. [4] presented a soft shadow algorithm that uses a

single ray and edge visibility computation to reconstruct the shadowed area. The algorithm is significantly faster than distributed ray tracing in many situations, but the visibility algorithm depends on geometric complexity, and the advantage over classic ray tracing degrades significantly as the number of edges in the scene increases.

Lischinski et al. [5] described ray tracing layered depth images (LDI) for computing secondary rays in image based rendering. However, the source of their LDI is often range data captured from views other than those of the light sources, so their method was predisposed to light leaks.

Agrawala et al. [6] introduced ray tracing of multi-view shadow maps for creating soft shadows from area lights. This method delivers high quality soft shadows for regular opaque geometry, but the cost of algorithm is linear in the number of views, so it is quite expensive.

Xie et al. [7] introduced multilayer transparent shadow maps, which can be used to produce high quality soft shadows. The algorithm is suitable for scenes with extremely complex geometry, fur, and volume objects.

One of the commonly used approaches is path tracing. The first description of a path tracing algorithm was given by Kajiya [8] in his paper "The rendering equation", where he proposes an approximation to the rendering equation which uses ray tracing. In distributed ray tracing, lights are sampled directly when a diffuse surface is hit by a ray. In path tracing, a new ray is randomly generated within the hemisphere of the object and then traced until it hits a light. This type of path can hit many diffuse surfaces before interacting with a light. A path tracer continuously samples pixels of the image. It starts to become recognisable after quite a few samples per pixel, perhaps 100, which makes the approach rather slow.

RAY-TRACING AND SHADOWS

Let us assume we have a 3D scene and want to render it in a RGB buffer of size (width x height) pixels. In ray-tracing we cast as many rays as the number of pixels (in our case width x height) from the camera to a rectangle of pixels situated behind the scene. Each ray is traced and collisions are sought with the objects in the scene. If a collision is found with an object, its material shader is called. In OptiX the material shader is a CUDA program that computes the colour of the pixel where the ray hit the object surface. In this way the output RGB buffer is filled with RGB colour values. As OptiX programs are executed on the GPU, the computations for each ray are done in parallel, which increases the performance.

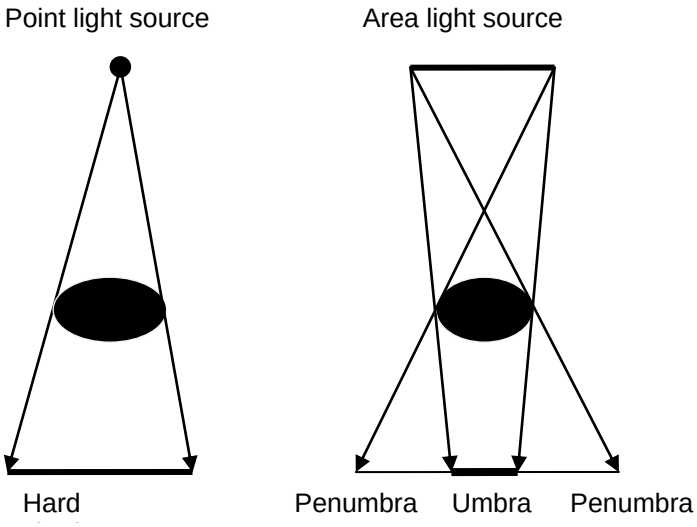

Fig. 1. Hard shadow (left) and soft shadow (right)

If a point light source is used then all generated shadows in the scene are hard shadows as shown in Fig. 1, left. In order to generate soft shadows, one has to use an area light source, Fig. 1, right, in which the shadow is split in two areas: umbra – no light reaches the object (darker area) and penumbra, which is partially dark. The modelling of area light sources in distributed ray tracing is achieved in the following way. A set of shadow rays is shot from the intersection point of a ray and an object to random locations on the area light source. These rays are traced; the lighting for each of them is calculated and averaged to get the resulting shading on the shadowed object. In practice people usually use a rectangle, circle or spherical light sources. Specialists [9] usually recommend to use spherical or circle (disk) light sources, as they are closer to real life ones. In this work we decided to use a disk source. Evenly distributed random points on a unit disk can be generated using the following formula [10]:

$$\begin{aligned} x &= \sqrt{r}\cos\theta \\ y &= \sqrt{r}\sin\theta \end{aligned}, \qquad (1)$$

where r is a random value in the interval [0, 1] and θ is a random angle in [0, 2π]. However, as Fig. 2 shows, when we use equation (1) to sample the disk, the resulting penumbra is not wide enough, although the size of the light source is close to reality. This requires many random points to be used, which of course slows down the rendering.

THE PROPOSED ALGORITHM

In this work we use disk light sources for rendering soft shadows. When an intersection is found of a camera ray with an object in the scene, its material shader is called, whose main purpose is to compute the colour of the corresponding pixel at the intersection point. As explained in the previous section, for each light source a set of shadows rays are cast from this point to a set of locations on the disk. However, in our approach these locations are not completely random that is why we call them pseudo random values. Let us assume we want to generate n points on a disk with a radius of one. In our approach the first point is always the centre of the disk. The next (n-1)-th are generated in the following way. The angles θ in equation (1) are not random, but evenly distributed, using the equation:

$$\theta_i = i * 360/(n-1), i = 1,..., n-1. \qquad (2)$$

The radiusses are not completely random numbers either and the points are calculated with the following equations:

$$\begin{aligned} x &= 0.3 + 0.7\sqrt{a}\cos\theta \\ y &= 0.3 + 0.7\sqrt{a}\sin\theta \end{aligned}, \qquad (3)$$

where a is a random value in the interval [0, 1]. In this way one point is in the central part of the disk and all others are distributed over equal angles, and their radiusses are evenly distributed random numbers in the interval [0.3, 1]. The main purpose of this distribution is to generate a larger penumbra with a smaller number of shadow rays per pixel. Someone might be tempted to use evenly distributed values without any randomisation. This however leads to an aliased image (a pattern in the penumbra), that is why it is always recommended to use random numbers for antialiasing [11].

In addition if the random radiusses are generated every time in the CUDA shader the rendering will be too slow. That is why they are pre-computed in advance and stored in a buffer in the GPU memory. When the shadows rays are shot the coordinates are multiplied by the radius of the light source.

RESULTS

The algorithms were implemented in C++ under Windows 7 and were tested on a laptop with a 2.20 GHz Intel core-i7 processor, 16 GB RAM and NVidia GeForce GTX 560M graphics card. The NVidia OptiX ray tracing engine was used for rendering. The images were rendered using a disk light source with a diameter of 4 cm, which is the usual size of a domestic light bulb.

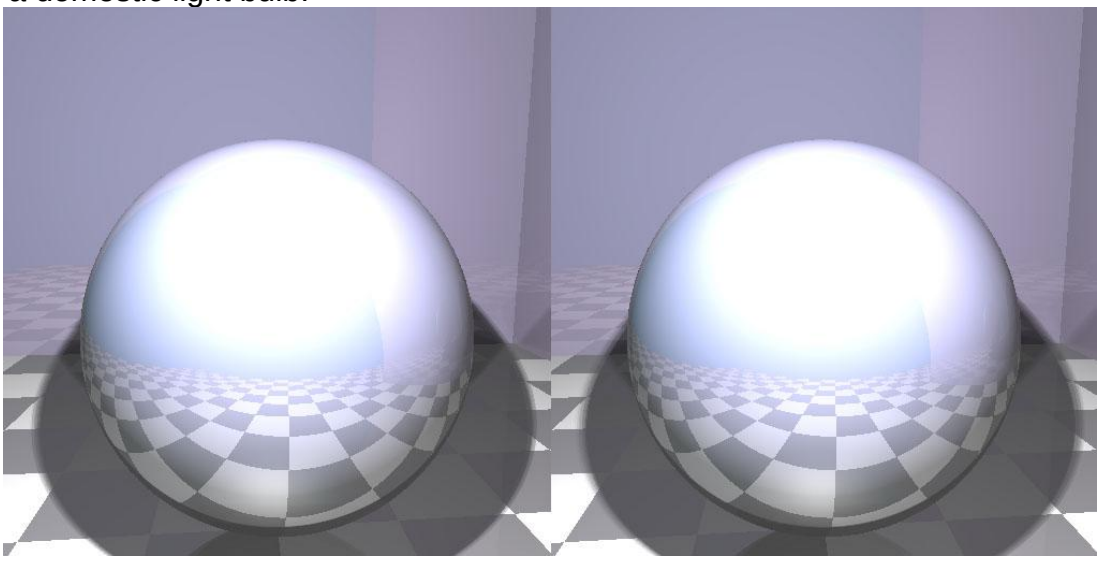

Fig. 2. A sphere rendered with 16 shadow rays per pixel
Left: random location on the light source; Right: pseudorandom locations

Fig. 3. A ray-traced dressed human body with soft shadows

Fig. 2 compares two images of a sphere rendered with 16 shadow rays per pixel. The left image is produced using randomly generated points on a disk with equation (1), while the right one is a result of the pseudo random approach using equation (2) and (3). The quality of penumbra is the same but for the right image it is clearly larger. A ray-traced dressed virtual human body using the same number of shadow rays per pixel is shown in Fig. 3.

In order to evaluate the speed of the proposed approach, time was measured for 100 rendering frames for two objects, sphere and a dressed human body, using one and two light sources. Measurements in frames per second are shown in Table 1, and the speed can be considered in real time.

Table 1. Performance of the rendering technique

Rendered object	1 light source	2 light sources
Sphere	15.95 fps	9.39 fps
Human body	8.36 fps	6.13 fps

CONCLUSIONS

This paper presented a fast distributed ray-tracing approach for producing soft shadows using disk light source. The programs were implemented on GPU using the NVidia OptiX ray tracing library. The main difference to the classic distributed ray tracing soft shadow technique is that the shadow rays are shot to a set of pseudo random locations on the light source, which are generated in such a way that a larger penumbra can be achieved with a smaller number of rays, which speeds up the rendering. The following conclusions can be drawn:

- The proposed equations (2) and (3) for generating pseudo random points on a disk light source produce good quality soft shadows at relatively high speed with as few as 16 rays per pixel;
- Performing parallel computations for each ray on the GPU, using NVidia OptiX engine, allows good quality ray tracing at a speed of 15 frames per second.

REFERENCES

[1] Woo A., Poulin P., Fournier A.: A survey of shadow algorithms. IEEE Computer Graphics and Applications 10, 6 (11 1990), 13–32.

[2] Hasenfratz J.-M., Lapierre M., Holzschuch N., Sillion F. X.: A survey of real-time soft shadow algorithms. In Proceedings of EUROGRAPHICS 2003 (2003).

[3] R. Cook, T. Porter, L. Carpenter, Distributed Ray Tracing, Computer Graphics, 18(3), July 1984, pp. 137-145.

[4] Lane S., Aila T., Assarsson U., Lehtinen J., Akenine-Moller T.: Soft shadow volumes for ray tracing. In SIGGRAPH ’05: ACM SIGGRAPH 2005 Papers (2005), Computer Graphics Proceedings, Annual Conference Series, ACM, pp. 1156–1165.

[5] Lischinski D., Tampieri F., Greenberg D. P.: Image based rendering for non-diffuse synthetic scenes. In Eurographics Rendering Workshop 1998 Proceedings (1998).

[6] Agrawala M., Ramamoorthi R., Heirich A., MOLL L.: Efficient image-based methodsvfor rendering soft shadows. In Proceedings of SIGGRAPH 2000 (2000), Computer Graphics Proceedings, Annual Conference Series, ACM, pp. 375–384.

[7] Xie, F., Tabellion, E. and A. Pearce. Soft Shadows by Ray Tracing Multilayer Transparent Shadow Maps. In Eurographics Symposium on Rendering (2007).
[8] Kajiya, J. The Rendering Equation, Computer Graphics, 20(4), August 1986, pp. 143-150.
[9] Teoh, S.L. Ray Tracing: Soft Shadows, Lecture Notes in Computer Graphics, 2008, http://www.cs.sjsu.edu/~teoh/teaching/previous/cs116b_sp08/lectures/
[10] Wiesstein, E. W. Disk point picking, MathWorld--A Wolfram Web Resource. http://mathworld.wolfram.com/DiskPointPicking.html
[11] Manocha, D. Soft Shadows. Lecture Notes in Computer Graphics, 2009, http://www.cs.unc.edu/~dm/UNC/COMP236/lectures.html

CONTACT ADDRESS
Assoc. Prof. Tzvetomir Ivanov Vassilev, PhD
Department of Informatics, FNSE
Angel Kanchev University of Ruse
7017 Ruse, Studentska Str. 8, Bulgaria
Phone: (++35982) 888 475
Email: TVassilev@uni-ruse.bg

МЕКИ СЕНКИ С ТРАСИРАНЕ НА ЛЪЧИ БАЗИРАНО НА ГРАФИЧЕН ПРОЦЕСОР

Цветомир Василев

Русенски университет "Ангел Кънчев"

Резюме: *Тази статия представя подход за генериране на меки сенки с използване на разпределено трасиране на лъчи. Използван е кръгов източник на светлина и за разлика от класическия алгоритъм, при който множество от лъчи се изстрелват към съвкупност от случайни точки на източника на светлина, точките от кръга са псевдо-случайни с цел да се получи по-широка полусянка с по-малко на брой лъчи. Резултати от тестовете са показани в края на статията. Софтуерът е реализиран с използване на библиотеката NVidia OptiX ray tracing engine, което позволява рендиране на сцени със сравнително добро качество за 10-15 кадъра за секунда.*
Ключови думи: *Трасиране на лъчи, изчисления с графичен процесор, меки сенки*

ISSN 1314-3077