International Journal of Electrical and Computer Engineering (IJECE)
Vol. 5, No. 3, June 2015, pp. 562~568
ISSN: 2088-8708

Implementation of Cloth Simulation Using Parallel Computing on Mobile Device

Jae Hong Jeon*, Se Dong Min, and Min Hong*****
* Department of Computer Science, Graduate School, Soonchunhyang University, Korea
** Department of Medical IT Engineering, Soonchunhyang University, Korea
*** Department of Computer Software Engineering, Soonchunhyang University, Korea

Article Info

Article history:

Received Jan 13, 2015
Revised Apr 20, 2015
Accepted May 5, 2015

Keyword:

Cloth simulation
GPGPU
Mass spring system
Shader
Transform feedback

ABSTRACT

Physically based modeling and simulation is an important technique for deformable object simulation, which is widely used to represent the realistic shape change and movement of objects for mobile game or 3D simulation. However, they require the high computational cost for representing the physical phenomenon on deformable objects when it applied on mobile device. In this paper, we designed and implemented the cloth simulation for deformable object simulation using the parallel technique on mobile device to optimize the computational burden. We especially applied GPU parallel technique for the integration solving process such as Euler, Midpoint, 4th-order Runge-Kutta method to estimate the particles' next status using positions and velocities. Also we applied multi-thread parallel technique for calculating the spring force. Then we compared the performance of each integration methods between under only CPU and CPU with GPU on mobile device. Also we compared the computing time of spring calculation between only CPU and using CPU multi-thread.

Corresponding Author:

Min Hong,
Department of Computer Software Engineering,
Soonchunhyang University,
Unit 507, Multimedia Bld., Soonchunhyang Univ., Eumnae-ri, Sinchang-myeon, Asan-si,
Chungcheongnam-do, 336-745 Rep. of Korea
Email: mhong@sch.ac.kr

1. INTRODUCTION

Although various mobile applications have been implemented with recent advanced mobile device technologies, most applications are based on 2D contents due to the limited performance of mobile devices. Thus, the implementation of game, animation, virtual reality, augmented reality, and advertisement which require realistic representation of 3D objects is difficult to implement on mobile devices. In addition, the advent of recent HMD (Head Mounted Display) devices such as Oculus life and Gear VR have been received the public attention in VR (Virtual Reality) world. The real objects that have rigid or deformable characteristics for realistic representation of movement and interaction between objects highly require physically-based simulation.

A 3D deformable object simulation can represent the deformed objects like real objects, but it requires the high computational power due to represent the 3D object world or related calculation. It has to compute lots of physically related calculations in each simulation time step, so the physically based simulation is not trivial to perform on mobile devices. To solve these problems, many researches have been widely studied on numerical integration methods, collision check and response methods [1], multi-core and parallel approaches [2] on mobile devices. However most researches have been focused on the regular PC environments not mobile environments.

Recent mobile devices provide the GPU that has relatively low clock speed, but it provides lots of ALUs (Arithmetic Logic Unit) which can be applied to parallel processing. However, there are no official parallel processing related libraries for mobile devices until now which can conveniently support the implementation of mobile application. The GPU is simultaneously working with ALUs, so when GPU got work process, it separates and launches the number of ALUs and performs together in same time. This is we call GPGPU (General-Purpose computing on Graphics Processing Units).

In this paper, we proposed and implemented the physically based cloth simulation [3] using a vertex shader that computes lots of particle positions with GPGPU simultaneously. We proposed a new algorithm that using a transform feedback [4] which can capture the data from a shader and can reuse them in next simulation time. Also we proposed new cloth simulation data structure and multi-thread [5] [6] method for calculating the spring forces. In addition, we performed the cloth simulation using only CPU and the proposed GPU parallel computing and compared the performance of spring calculation time using multi-thread CPU and only CPU.

2. CLOTH SIMULATION USING PARALLEL COMPUTING

The computation of proposed cloth simulation using GPGPU method can be classified into two parts: CPU computation and GPU computation. The basic cloth simulation properties such as nodes, spring connections, external forces and so on are initialized and stored on CPU. Then it computes all spring forces and sum up these forces with external forces at each nodes using CPU. In next step, it computes the next status of node positions using GPGPU using GPU. All next positions of each node are independently computed by GPGPU, so it has to be arranged with suitable format for GPU and CPU.

2.1 Data Structure of Cloth Simulation

Data structure of cloth simulation in our paper is different from the traditional approach. Node information in traditional cloth simulation includes the position, velocity, and force in each unit. This system should continuously manage the data transmission to send these data to GPU memory in every time step. In the proposed algorithm, the position, velocity, and force information are managed with exactly same format in GPU memory by list and when these information are required to be calculated, they are referred using list by pointer. Therefore, the proposed data structure can prevent the unnecessary data transmission for rearrangement of position, velocity, force and mass in every time step. Figure 1 shows the comparison of data structure between the traditional cloth simulation and the proposed cloth simulation.

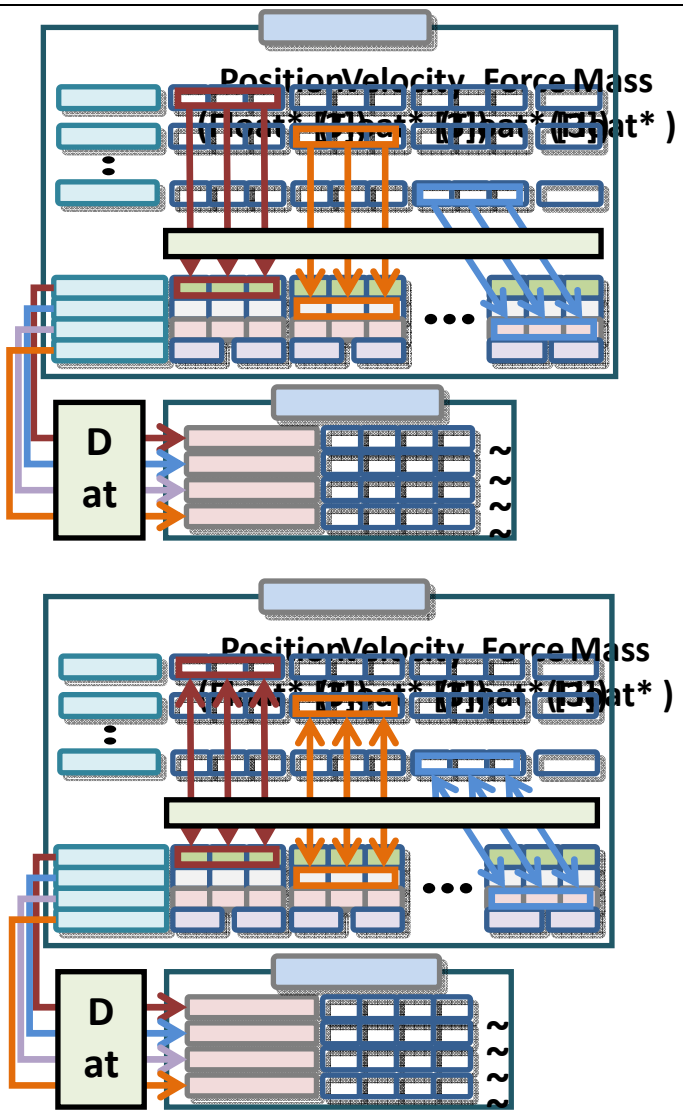


Figure 1. Data structure comparison between the traditional and proposed cloth simulation

2.2 Calculate of Spring Force using Multi-Thread of CPU

Generally, calculation of spring forces in cloth simulation requires lots of computational time and the traditional cloth simulation used only one CPU for this computation that leads the decreased simulation performance. To solve the problem, the proposed method utilized the multi-thread approach on CPU to calculate the spring forces in parallel manner. The proposed algorithm splits n springs into 16 thread groups and each group independently calculates spring forces with its own thread. When the calculation of spring forces is finished in every thread, the proposed algorithm calculates the next status of node positions using numerical integration. Since each spring is connected with two nodes, the calculated spring forces are stored in associated node information. Because several threads can access to same node at the same time, we applied the mutex (mutual exclusion) to prevent this problem. Figure 2 shows the flowchart of implemented cloth simulation with multi-thread of CPU and with one CPU.

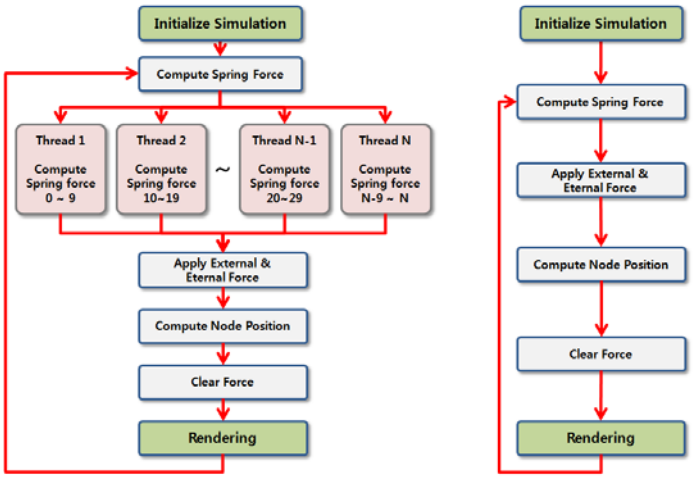


Figure 2. Flowchart of cloth simulation using multi-thread CPU and one CPU

2.3 Cloth Simulation using the Transform Feedback Buffer

The CPU passes all node data into GPU memory, and vertex shader concurrently calculates the next node positions using GPGPU and then updated node position data should be transferred to into CPU. In this research, we applied the transform feedback buffer that is recently supposed by open GL ES. Using the transform feedback buffer, we can capture the node data after calculation on GPU and then push these data into the transform feedback buffer. Therefore, the implemented cloth simulation with the transform feedback buffer can efficiently transfer the related data from GPU to CPU. Figure 3 shows the flowchart of implemented cloth simulation using GPU.

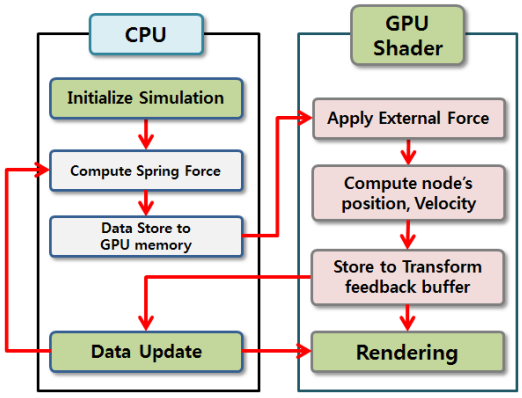


Figure 3. Flowchart of the implemented cloth simulation with GPU

3. RESULTS AND ANALYSIS

Figure 4 shows the result of experimental test which was performed on IPad Air Retina. We performed Semi-Euler, Midpoint, 4th-order Runge-Kutta method to compare the performance of GPU and CPU environments under from 3,600 to 250,000 vertices.

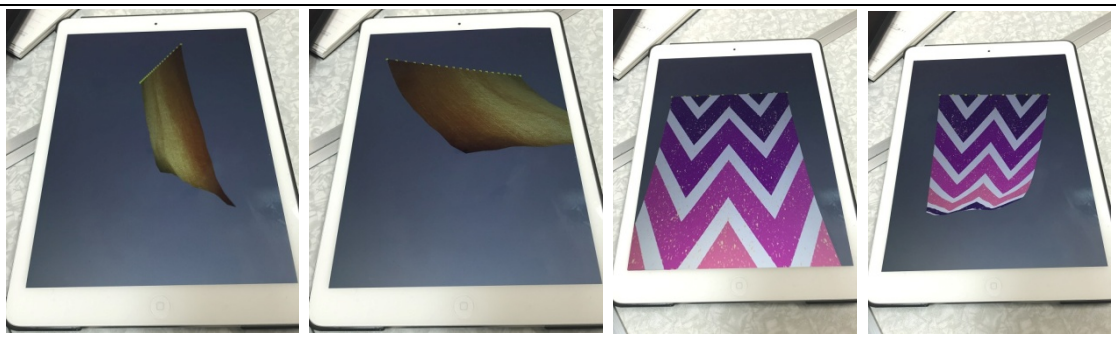


Figure 4. Snapshots of implemented cloth simulation

As a result, the proposed GPU parallel technique with the transform feedback buffer is much faster than only CPU, when the number of nodes is over the specific breaking point. Since cloth simulation with GPU parallel computing has to transfer the data from CPU to GPU memory, it requires the transfer latency. However, when the number of nodes is over 6,000 vertices, the proposed method is much faster than only CPU method. In addition, 4th-order Runge-Kutta method which requires more complex computing operations has more advantage with GPU. Figure 5 shows the performance result (ms) of comparison with 3 different integration methods using GPU and CPU.

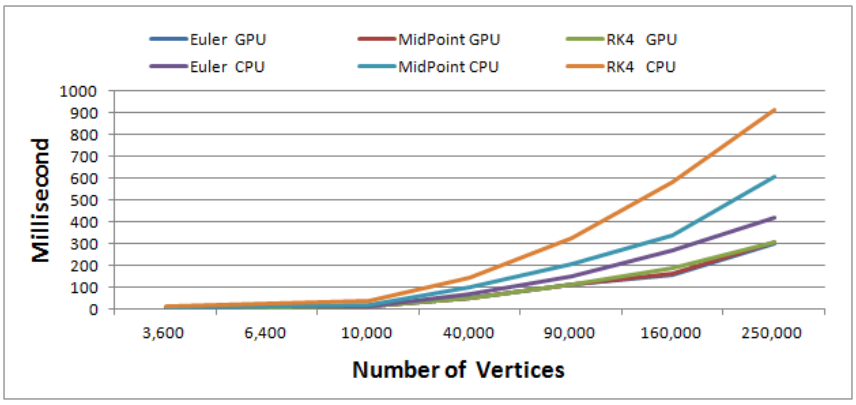


Figure 5. The performance result with different integration methods with CPU and GPU

The proposed multi-thread CPU parallel technique method is much faster than CPU only, when the number of springs is over 40,000. Spring force calculations can be quickly processed by splitting them into multi-thread CPU. Although there is no significant improvement of performance when the number of spring is low, when the number of springs is getting increased, the multi-thread divides the springs into 16 groups and quickly calculates them using multi-core in mobile device. Therefore, we believe that the proposed method is well suitable for some simulation applications that require plausible real-time performance of cloth simulation for mobile devices. Figure 6 shows the result of performance test for cloth simulation when multi-thread CPU was applied to the calculation of spring forces.

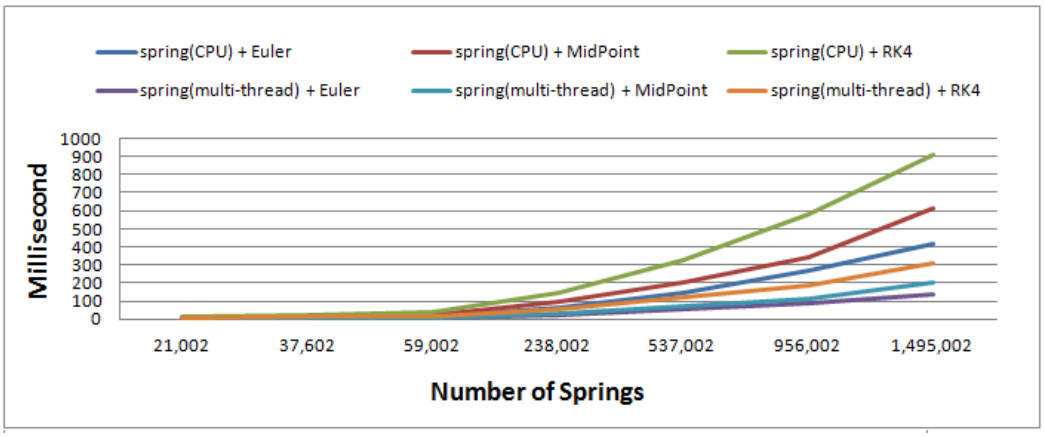


Figure 6. The performance result of spring force calculation using multi-thread CPU and only CPU

4. CONCLUSIONS

In this paper, we proposed the cloth simulation using parallel computing methods that are based on the multi-thread with CPU and the transform feedback buffer with GPU. Using the transform feedback buffer which can captures node information, when shader is working on mobile device. The multi-thread CPU can divide and execute the calculation of spring forces with some of groups. To update the position and velocity of nodes, we used Semi-Euler, Midpoint, 4th-order Runge-Kutta integration methods based on CPU and GPU. The proposed GPU parallel technique is much faster than CPU, when the number of nodes is relatively high enough. Also the multi-thread CPU parallel technique reduces the spring force computing time in cloth simulation. In this paper, although we confirmed the performance of the CPU and GPU parallel computing approach, we did not combine two proposed methods yet. The integrated cloth simulation using both CPU and GPU parallel processing is expected to provide much improved performance of cloth simulation, when the spring force computation with multi-thread CPU and with GPU using transform feedback buffer are combined. In addition, real-time deformable 3D object simulation can be a good candidate in mobile environment to achieve the realistic and plausible performance and behavior of deformable objects for these CPU and GPU parallel processing methods.

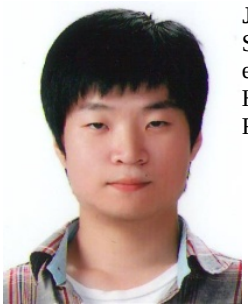
ACKNOWLEDGEMENTS

This work was supported by the Soonchunhyang University Research Fund.

REFERENCES

[1] J. Mosegaard, "A GPU accelerated spring mass system for surgical simulation", *Studies inHealth Technology and Informatics*, vol. 111, pp. 342–348, 2005.
[2] J.H. Jeon, M.H. Choi, Y.S. Jeong and M. Hong, "Hierarchical Bounding Sphere FFD-AABB Algorithm for Fast Collision Handing of 3D Deformable Objects on Smart Devices", *Journal of Internet Technology*, Vol 14, No. 5, pp. 843-850, 2013.
[3] D. Baraff and W. Andrew, "Large steps in cloth simulation", *Proceedings of the 25th annual conference on Computer graphics and interactive techniques. ACM*, pp.43-54, 1998.
[4] D. Ginsburg, B. Purnomo, D. Shreiner, and A. Munshi, "Open GL ES 3.0 Programming Guide," 2014.
[5] S. Kazuki, and T. Furumoto. "Grand central dispatch", *Pro Multithreading and Memory Management for iOS and OS X. Apress*, 139-145, 2012.
[6] V. Nahavandipoor, "Concurrent Programming in Mac OS X and iOS: Unleash Multicore Performance with Grand Central Dispatch", O'Reilly Media, Inc., 2011.

BIOGRAPHIES OF AUTHORS



Jae Hong Jeon received BS degree in Department of Computer Software Engineering from Soonchunhyang University in 2012. Now he is undertaking a master degree of computer engineering course as a member of the Computer Graphics Lab at Soonchunhyang University. His research interests are computer game development, computer graphics, AR (Augmented Reality) and embedded motion capture

Se Dong Min was born in Seoul, Korea, in 1975. He received the M.S. and Ph.D. degrees in electrical and electronic engineering from the Department of Electrical and Electronics Engineering, Yonsei University, Seoul, in 2004 and 2010, respectively. He is currently an Assistant Professor at the Department of Medical IT Engineering, Soonchunhyang University, Asan, Korea. His research area includes biomedical signal processing, healthcare sensor application, and mobile healthcare technologies.

Min Hong is an Associate Professor at the Department of Computer Software and Engineering, Soonchunhyang University in Asan, Korea. He received BS in Computer Science from Soonchunhyang University in 1995. He also received MS in Computer Science and PhD in Bioinformatics from the University of Colorado in 2001 and 2005, respectively. His research interests are in Computer Graphics, Mobile Computing, Physically-based Modeling and Simulation, Bioinformatics Applications, and u-Healthcare Applications. In present, he is a Director of Computer Graphics Laboratory at Soonchunhyang University.