Multi-Threaded Application for Marching Cubes Algorithm

*[1]Soydan Serttas, [1]Kayhan Ayar, [1]Guluzar Cit and [1]Cemil Oz
[1]Faculty of Computers and Informatics, Department of Computer Engineering, Sakarya University, Turkey

Abstract

Marching cubes is an algorithm used for obtaining triangle models by voxels. Simulation drawings in areas such as medical imaging, special effects and 3-dimensional modeling are generated as voxel maps. In addition, voxels are used for collision detection and deformation modeling in computer graphics. Drawings of models in these areas are created by extracted polygonal meshes of iso-surfaces from voxels. Marching cubes algorithm tends paralleling on account of it is applied to each voxel separately. In this study, marching cubes algorithm is implemented by using multi-threads by a single CPU. It's observed that, if the number of voxels which create the model increase, then efficiency of using threads is also increased.

Key words: Marching cubes, voxelization, parallel computing.

1. Introduction

Designers have used computers to design 3 dimensional (3D) models for many years. Computer technologies have been improved day by day. Virtual environments have been developed which allows 3D modeling by using virtual reality tools. In recent years, voxel based models [1] come into prominence in many areas such as virtual sculpting, flow dynamics, medical imaging etc.

This work is a part of the real time virtual sculpting which has implemented on one CPU. This system uses marching cubes algorithm [2,3] for cutting edge after the voxelization process. Our sculpting simulation reads 3D virtual models in a variety file formats such as raw and stl. These models consist triangle polygon meshes, and our simulation voxelize models' outer surface and interior to generate their volumetric data. The surface is reconstructed using marching cubes algorithm.

The aim of the study is to decrease the computation time of the marching cubes algorithm. For the purpose of saving time we have divided the job into multiple threads. The process time for different thread counts have been observed and compared.

The remainder of the paper is organized as follows: in section 2, literature review is placed. In section 3, we present the materials and methods. In section 4, we mention about the results. Last, we explore the significance of the results in section 5.

*Corresponding author: Address: Faculty of Computers and Informatics, Department of Computer Engineering, Sakarya University, 54187, Sakarya TURKEY. E-mail address: sserttas@sakarya.edu.tr, Phone: +902642955916 Fax: +902642955454

2. Literature Review

Marching cubes algorithm intends to create a surface from a 3D array of data. The idea of the algorithm is to create a triangular mesh that will approximate the iso-surface and calculate the normals to the surface at each vertex of the triangle [4]. Firstly, the surface is located in a cube of eight pixels. Second, normals are calculated, and last, it's marched to next cube. Marching cubes has been applied in many application areas, including digital sculpting [5,6], deformable modeling [7,8], dynamics and mechanics [9], biomedicine [10,11] and natural events [12].

A multi-core processor has two or more processors for efficient simultaneous processing of multiple tasks. Multi-threaded processors support the execution of many simultaneous threads via a combination of the multiple cores [13,14]. Since the start of having multiple cores, central processing units (CPUs) begin multithreading that aims to increase utilization of a single CPU [15,16]. Multi-threading can be used on the CPU for mobile applications [17], power efficiency [18] or the purpose of increasing the speed of simulations [19].

There are some techniques about parallelization such as cloud computing [20] which is based on distributed system architecture. A distributed system is a collection of independent computers. These computers need some data management dependencies to accomplish the task and communication enhancements can cause time consuming if you have a large data.

3. Materials and Method

In this study marching cubes algorithm is used to generate 3D triangle mesh from voxelized models. To generate a triangular mesh, marching cubes algorithm has to be performed on every voxel of the 3D object and create suitable triangles for that voxel. When all the voxels are processed, the triangular model is formed. Figure 1 shows different resolution voxelize models and triangular models generated by using marching cubes algorithm.

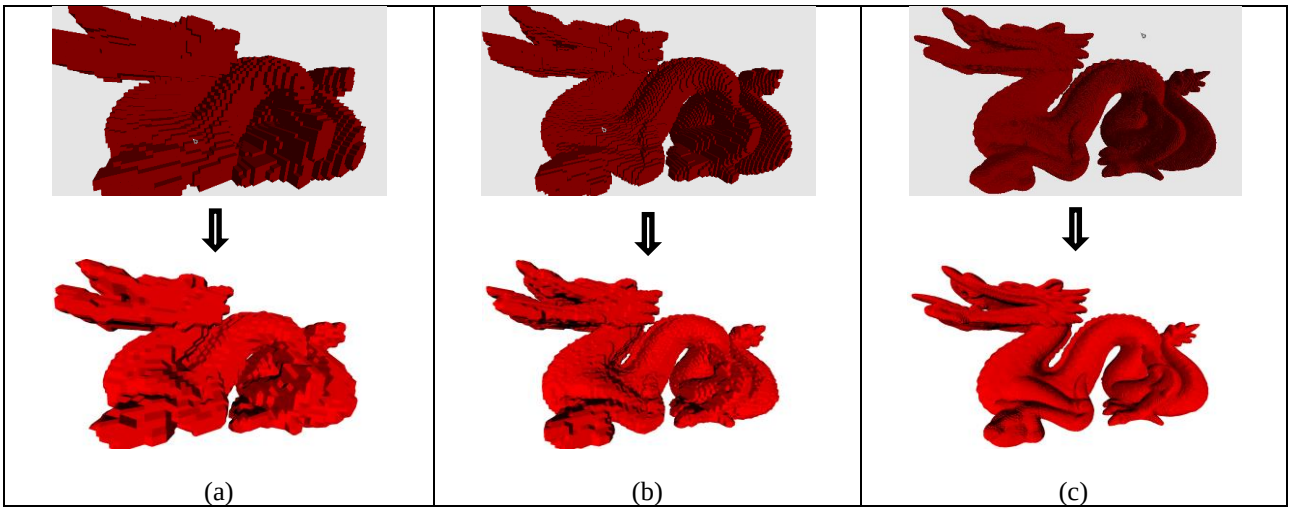

Figure 1. Applying marching cubes in different resolutions. (a) 64x64x64 resolution, (b) 128x128x128 resolution, (c) 512x512x512 resolution

The work has simulated on an Intel Core Quad 2.50 GHz CPU. Marching cubes algorithm can be time consuming in higher resolution. In this study this process has performed by multiple threads simultaneously to decrease the processing time. First, voxel's data have been split into parts according to number of threads. Figure 2 shows how voxel's data have been split into parts. Then a buffer has been created to store triangles to be generated. After all preparation is done, threads start running.

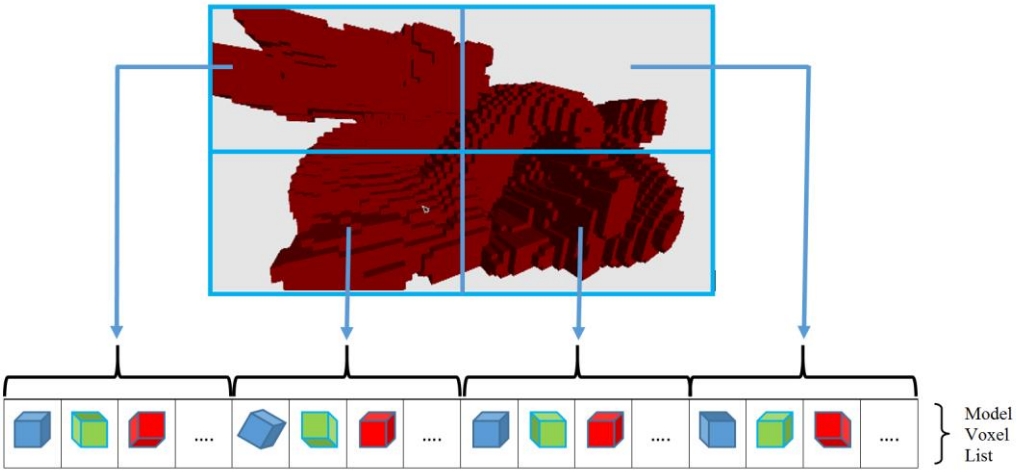

Figure 2. Voxel's data

Each thread will work only on dedicated buffer parts. Because of the nature of marching cubes algorithm there is no race condition between threads. Every thread can work freely without knowing other threads.

4. Results

Figure 3 shows four thread marching cubes processes. "Thread Control Module" is responsible for creating threads and distributing voxel's data among them.

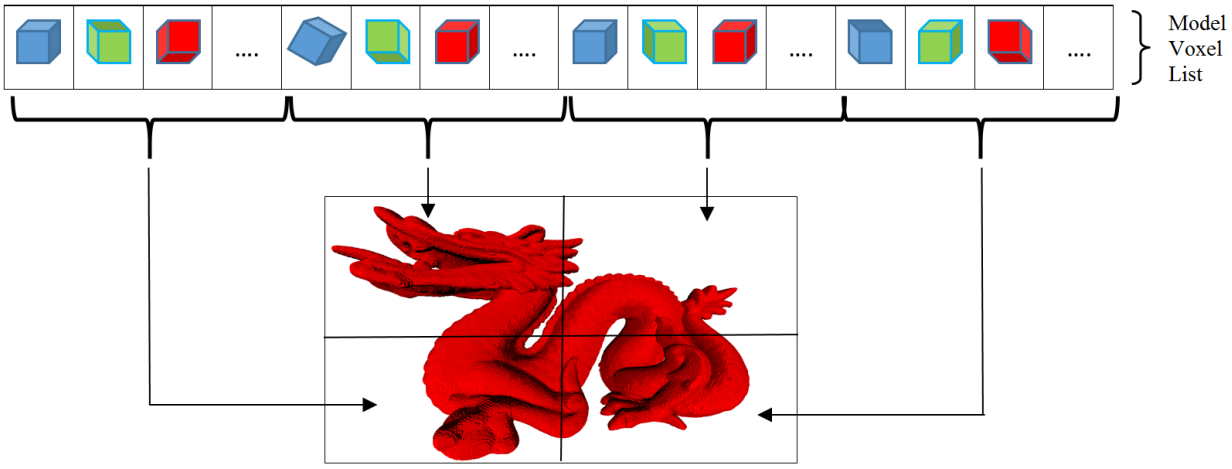

Figure 3. Processing marching cubes with four threads

5. Discussion

As shown in table 1, using multiple threads is actually decreased marching cubes processing time. In high resolutions, number of threads is becoming more important to decrease marching cubes time. When thread number is equal to CPU's core count, time difference is maximum. But, if thread number is higher than core count then time difference is becoming to disappear. Main bottleneck in this method is the working CPU's core count.

Table 1. Marching cubes processing time(in seconds) with different thread count

Resolution / Thread Count	64x64x64	128x128x128	256x256x256	512x512x512
1	0,122	0,383	1,515	6,949
2	0,093	0,130	0,651	3,842
4	0,041	0,717	0,390	1,450
6	0,037	0,596	0,287	1,331
16	0,034	0,501	0,261	1,301

Conclusions

The purpose of the study is to decrease the computation time of the marching cubes algorithm by dividing the job into multiple threads using a single CPU. Marching cubes algorithm is suitable for parallelized threads. While the thread count increases with the CPU 's core count, the process time of marching cubes algorithm decreases. In this study, only the calculation power of the CPU has been used on account of the aim to increase efficiency of a single CPU.

A real time simulator shows 30 frame for a second. With a view to simulate a 3D model which has 512x512x512 resolution, simulator displays (512x512x512) x 30 = 3840MB data. With the inclusion of the graphics processor, the processing time decreases in more. For this operation, OpenCL library can be used. In addition, cloud computing can be used, on the other hand it requires network connection which negatively affects simulation performance.

References

[1] Kaufman A, Cohen D, and Yagel R. Volume Graphics. IEEE Computer 1993, 26(7);51-64.
[2] Weber G, Scheuermann G, Hagen H, Hamann B. Exploring scalar fields using critical isovalues. In: Proceedings of visualization'02, Boston, 2002;171–8.
[3] Newman TS, Yi H. A survey of the marching cubes algorithm. Computers & Graphics 2006; 30:854–79.
[4] Lorensen WE, Cline HE. Marching Cubes: A High Resolution 3D Surface Construction Algorithm, SIGGRAPH '87 Computer Graphics, 21(4):163-9.
[5] Ferley E, Cani MP, Gascuel JD. Practical volumetric sculpting. Visual Computer 2000;

16(8):469–80.
[6] Cit G, Ayar K, Serttas S¸ Oz C. A real-time virtual sculpting application with a haptic device. TWMS J. App. Eng. Math. 2013;3(2):223-30.
[7] Lin F, Seah HS, Lee YT. Deformable volumetric model and isosurface: exploring a new approach for surface boundary construction. Computers & Graphics 1996;20(1):33–40.
[8] Keeve E, Girod S, Kikinis R, Girod B. Deformable modeling of facial tissue for craniofacial surgery simulation. Comput Aided Surg. 1998;3(5):228-38.
[9] Matsuda H, Cingoski V, Kaneda K, Yamashita H, Takehara J, Tatewaki I. Extraction and visualization of semitransparent isosurfaces for 3D finite element analysis. IEEE Transactions on Magnetics 1999;35(3):1365–74.
[10] Yim P, Vasbinder G, Ho V, Choyke P. Isosurfaces as deformable models for magnetic resonance angiography. IEEE Transactions on Medical Imaging 2003;22(7):875–81.
[11] Narkbuakaew W, Sotthivirat S, Gansawat D, Yampri P, Koonsanit K, Areeprayolkij W, Sinthupinyo W, and Watcharabutsarakham S. 3d surface reconstruction of large medical data using marching cubes in VTK. The 3rd International Symposium on Biomedical Engineering, ISBME' 2008; 64-7.
[12] Trembilski A. Two methods for cloud visualization from weather simulation data. Visual Computer 2001;17:179–84.
[13] Olukotun K. et al. The Case for a Single-Chip Multiprocessor. Int. Conf. on Architectural Support for Programming Languages and Operating Systems, 1996; 2–11.
[14] Tullsen D, Eggers S, and Levy HM. Simultaneous Multithreading: Maximizing On-Chip Parallelism. Int. Symp. on Computer Architecture, 1995; 392–403.
[15] Liu T, Curtsinger C, Berger ED. Dthreads: efficient deterministic multithreading. SOSP '11 Proceedings of the Twenty-Third ACM Symposium on Operating System Principles; 327-36.
[16] Blake G, Dreslinski RG, and Mudge T. A Survey of Multicore Processors. IEEE Signal Processing Magazine, November 2009; 26-37.
[17] Texas Instruments Inc. OMAP 4: Mobile applications platform. 2009, Online available: http://focus.ti.com/lit/ml/swpt034/swpt034.pdf
[18] Nawathe UG, Hassan M, Yen KC, Kumar A, Ramachandran A, and Greenhill D. Implementation of an 8-Core, 64-Thread, Power-Efficient SPARC Server on a Chip. The IEEE Journal of Solid-State Circuits, 2008; 43(1): 6-20.
[19] Öz C, Çit G, Ayar K. Multithreaded voxelization method for virtual sculpting. In:European Conference of Technology and Society; 2013.
[20] Armbrust M, Fox A, Griffith R, Joseph AD, Katz R, Konwinski A, et al. A View of Cloud Computing. Communications of the ACM April 2010; 53(4):50-8.