ISSN : 0974 - 7435 Volume 10 Issue 23

2014

BioTechnology

An Indian Journal

FULL PAPER

BTAIJ, 10(23), 2014 [14426-14431]

Pickup of large scale point cloud based on GPU

Ming Huang[1], Yanmin Wang[1], Yong Zhang[1]*, Xinle Fu[2]

[1]Beijing University of Civil Engineering Architecture , Key Laboratory for Urban Geomatics of National Administration of Surveying, Mapping and Geoinformation,100044, Beijing, (CHINA)

[2]GIS Application Development Department, Sichuan Remote Sensing Geomatics Institute, 610100, Chengdu, (CHINA)

E-mail: 1158373292 @qq.com

ABSTRACT

With the rapid development of three-dimensional laser scanning technology, ultra-large fine point cloud data has been gradually become an important data source of three-dimensional model. In an interactive computer graphics application system, the interactive pickup of graphics is an important method. However, the traditional pick-up algorithm is limited to a situation that CPU-based ray intersection algorithm can only pick up a small amount of data triangular facets. Because the speed of picking up large scale point cloud data is slow, a GPU-based point cloud picking algorithms was presented to solve the problem. The basic idea of the algorithm is that by spatial transformation to convert the point cloud to screen space, and then, the point was calculated which is the nearest to the mouse click point in screen space. The GPU's parallel computing capabilities were used to achieve spatial transformation and distance comparison by Computer Shader in this algorithm. So the speed of the pickup has been increased. The experimental results show that compared with the CPU, the pickup method based on GPU parallel computing has greater speed advantage. Especially for the point cloud over 10 million points, the speed of the pickup has been increased 2-3 times faster. This use of GPU parallel computing capabilities have significant advantages in terms of handling large scale data volumes.

KEYWORDS

Point cloud; Pickup; Compute shader; GPU parallel computing; Large scale data volumes.

INTRODUCTION

In an interactive computer graphics application, such as 3D games, virtual reality, CAD / CAM, etc., it requires the user to interact with the system via an input device. These interactions include rotation, translation, delete, view object status information. Interactions need to locate the object in the scene by picking operation. Pickup technology has been become one of the most basic and a very important component of these systems. Because there is such a wide range of pickup applications, many scholars have conducted in-depth research. Yao Jiquan and Wang Jian put forward that adding a third dimension to the two-dimensional coordinates of pickup to inverter exchange for World Space to calculate pick-rays. According to the depth of the order of objects make-ray - object intersection judgment in the world space[1]. Zhu Mingliang put forward picking algorithm[2] based viewport space. In recent years, graphics hardware gradually has a powerful programmable function. There has been the use of hardware (GPU) acceleration pickup algorithms[3]. As HANRAHAN P and HAEBERLI P raised WYSIWYG method [4]; Zhang Jiahua and other proposed GPU picking algorithms which use Geometry Shader to implement ray - geometry intersection operation [5]; Zhao Hanli and Jin Xiaogang and other proposed FRMP method [6] .

In recent years, the method of a point which represented as a primitive object's surface and be drawn has gradually become the focus of the study. On the one hand due to the rapid development of three-dimensional laser scanning technology, super-large scale models can be get through laser scanning, such models are initially represented as a three-dimensional point discrete (point cloud). On the other hand the computing power of the computer has reached a new level, at this level, the point as drawing primitives become meaningful [7, 8]. However, the study agree that pickup algorithms are for triangular elements, but not for point primitives, these algorithms are not suitable for point clouds, and because of the large point cloud data, the speed of these pickup methods are very slow.

In this paper the pickup point cloud GPU-based algorithm was presented. The algorithm has the following characteristics: (a) The algorithm processing point cloud, and is suitable for large-scale point cloud model pickup.(b) The algorithm is based on the screen space. The point is transformed from local space to screen space, and then compared with the distance and consistent with the rendering pipeline rendering order, it is not involved in the transformation matrix inversion. The algorithm is not restricted by the problem that the inverse matrix of the transformation matrix which does not exist. (c) the algorithm is based on the GPU. DirectX 11 Shader (Compute Shader) was used to make point conversion and distance comparison operations. Thanks to the powerful GPU's parallel computing capabilities[9, 10]. Compared with the algorithm running on the CPU, Pick up speed is 2-3 times faster. The larger point cloud, the more obvious advantages of speed.

GPU-BASED ALGORITHM FOR POINT CLOUD PICKUP

Point cloud picking algorithm based on screen space

Currently in computer graphics, the most widely used algorithm is the classic Ray picking algorithm. The basic principle is that firstly users click on the screen to take a screen coordinates, then the coordinates is transformed into a viewport coordinate of display system. Next add the depth value to the point. Through a series of coordinate space conversion, reversely calculate the coordinates of the pickup point in the world coordinate system. Lead a ray by viewpoint (camera position) to the pickup point. In the three-dimensional space rays and objects conduct intersection. If they intersect, the object has been picked up [11, 12]. Since the point cloud model is composed by a large number of points and point is no radius or size, so it cannot judge whether a point and the ray intersect each other. Therefore ray picking algorithm does not apply to the pickup point cloud data. Point cloud is picked up through the method which is based on the screen space.

The basic idea of the algorithm is that make a series of space conversion to the point of points cloud. Calculate projection coordinates on the screen. In screen coordinates, obtain point which is nearest the points that user clicks on the screen. Specific steps are as follows:

a:Users click on the screen to get the screen coordinates of the point S(s_x , s_y).

b:Strike the screen coordinates of the points which are from point cloud. The coordinates of the point P_i (x_i , y_i , z_i)from point cloud is known . The point is multiplied by the world matrix. And the point is transformed from model space to world space. Next the point is multiplied by the view matrix M_{view} transformation from world space to the observation space. And then it is multiplied the projection matrix, the transformation is made from observation space to the projection space. The last moments it is multiplied by viewport transformation; the transformation is made from projection space to screen space. In the screen coordinate system coordinates $P_i^{'}$ ($x_i^{'}$, $y_i^{'}$, $z_i^{'}$),

$$P_i^{'} = P_i * M_{world} * M_{view} * M_{project} * M_{viewpoint} \quad (1)$$

Viewport transformation matrix:

$$M_{viewpoint} = \begin{bmatrix} w/2 & 0 & 0 & 0 \\ 0 & -h/2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ w/2 & h/2 & 0 & 1 \end{bmatrix} \quad (2)$$

where w is the width of viewport ,h is the height of the viewport.

c:In screen coordinates, calculate the point $P_i^{'}(x_i^{'}, y_i^{'}, z_i^{'})$ (which does not participate in operation) which is the nearest point $P^{'}(P_x^{'}, P_y^{'}, P_z^{'})$ from S(s_x, s_y).The corresponding point $P(p_x, p_y, p_z)$in model space is the selected point. Please read through the following sections for more information on preparing your paper. However, if you use the template you do not have to worry about setting margins, page size, and column size etc. as the template already has the correct dimensions.

The processes of GPU-based pickup point cloud algorithm

The algorithm of pickup point cloud is based on the screen space. For large-scale point cloud (cloud point number greater than 15 million), the algorithm operations are performed on the CPU, and the speed of picking up is limited. This limitation can be solved by a powerful GPU parallel computing power. Microsoft's DirectX 11 compute shader newly added can be used to perform graphics unrelated general-purpose computing[13]. The above picking algorithms are ported to compute shader. By calculating shader's powerful parallel computing capabilities, you can solve the shortage of the pickup algorithm and improve pick up speed.

The basic idea of the GPU-based point cloud pick up algorithm is that the transformation of point and the distance calculation of clicking screen point are implemented in Direct3D 11[14, 15]. Using GPU parallel computing to improve the speed of picking up, the whole process flow chart shown in Figure 1.

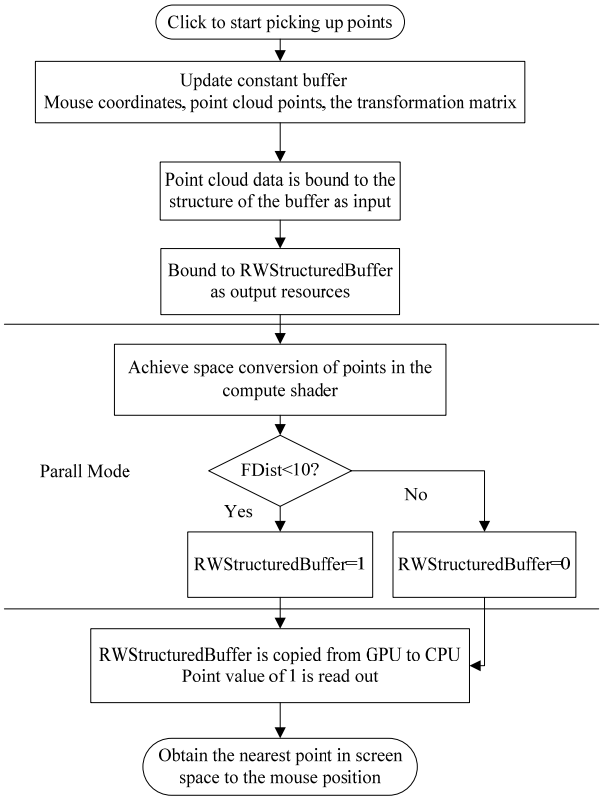

Figure 1: GPU-based algorithm of pickup point cloud flowchart

The Implementation of GPU-based pickup point cloud algorithm

The main process of the algorithm is implemented on a compute shader. Specific implementation process includes these elements: Creating resources for compute shader, Initializing resources and binding resources. Setting computes shader, calling Dispatch execution.

Create a resource for the compute shader, initialize resources and bind resources

Compute shader requires the use of three resources: Constant buffer, the buffer structure and RWStructuredBuffer[16, 17]. As shown in Figure 2.

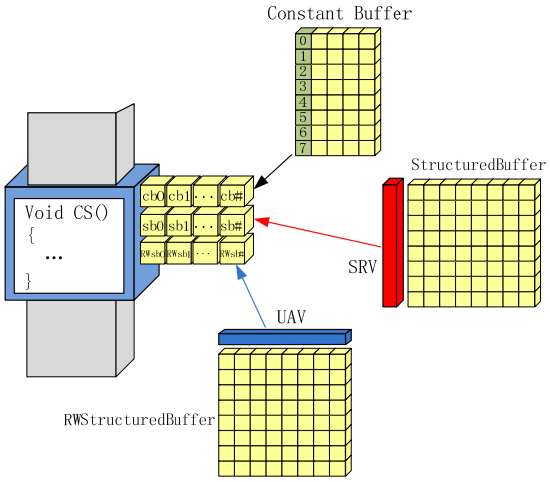

Figure 2: Schematic of compute shader binding resources

Since the algorithm needs known the screen-clicked point, the numbers of points and the transformation matrix, these values are stored with constant buffer. Before compute shader execution, the value of these variables needs to be passed to the constant buffer (Constant Buffer). Bind coordinate point cloud data into the structure buffer (StructuredBuffer). When the compute shader's calculations are completed, the result of the calculation needs to return the CPU by RWStructuredBuffer. RWStructuredBuffer is CPU not only bound to buffer data for the GPU to use but also can copy the buffer to the CPU for reading.

Compute shader setup and execution

A certain number thread group can be opened by Dispatch (X, Y, Z) function in shader file. Specify the number of threads of a thread group containing bynumthreads (X, Y, Z) function. As shown in Figure 3, a thread group may have a composition of n threads. But during the actual hardware implementation, these threads are divided into warps (Each warp is composed by 32 threads). Each wrap is processed by multiprocessor on the SIMD32. You can specify multiple threads within a thread group number are not 32. However, for performance reasons, the number of threads in the thread group should be the warp (32 threads) multiples. Program specifies a thread group has 256 threads composition. Because each thread handles one point, the program open (NumOfPoint + BLOCK_SIZE -1) / 256 thread group. NumOfPoint represents the total number of points. The number of BLOCK_SIZE is256 to ensure that each point is processed. As shown in Figure 7,for simplicity reasons, through numthreads (BLOCK_SIZE, 1, 1) and Dispatch ((UINT) (NumOfPoint + BLOCK_SIZE -1) / 256,1,1) the threads and thread groups are set to a dimension.

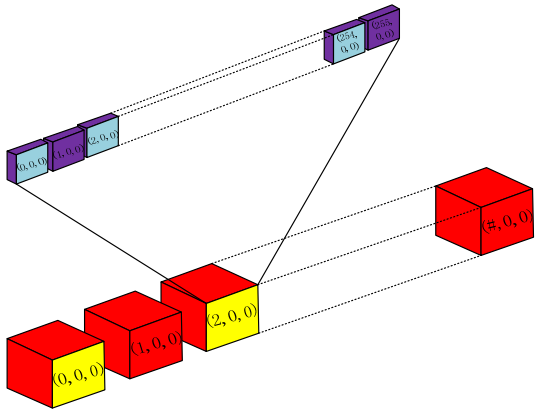

Figure 3: Compute shader thread group and thread schematic

The number of threads which compute shader open is the same number of point clouds, so a thread is responsible for operation of a point. Shown in Figure 4, each thread correspond to the index point will make space conversion. It is converted from local coordinate system to the screen coordinate system. Determine the distance from point to each screen-clicked point is less than the set value.

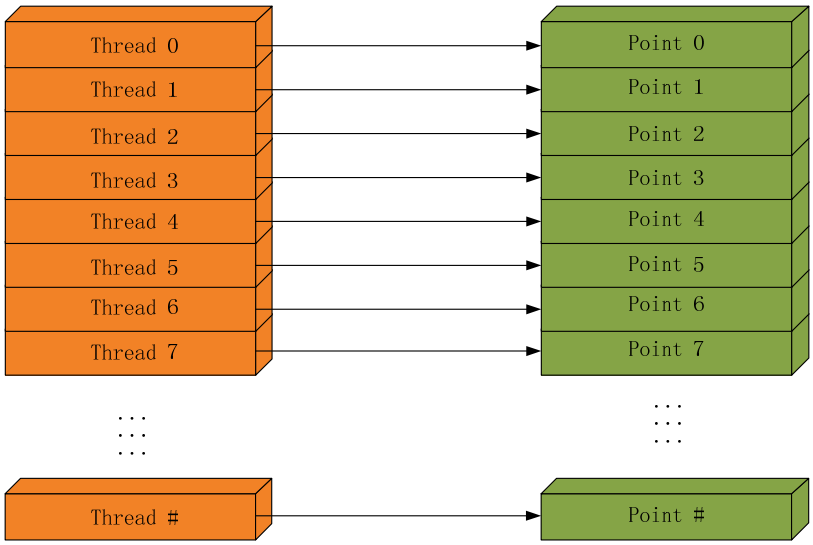

Figure 4 : Schematic of each thread is responsible for a point of operation

Retrievethe compute shader operation result

After the calculation of compute shader is completed, RWStructuredBuffer need to be copied from the GPU to CPU, read by the CPU. You need to create a buffer, the usage of Buffer tag is D3D11_USAGE_STAGING, And CPU buffer access method is D3D11_CPU_ACCESS_READ. Use Copy -Resource method to copy the GPU resources to the system memory. Finally, we can use the Map and Uncap function mapping buffer to read CPU resources. At last, a small amount is calculated on the CPU to obtain the selected point.

ANALYSIS OF EXPERIMENTAL RESULTS AND THE RESULTS

This paper is based on the configuration CPU: Intel Core 26600 and GPU: the NIVIDIA GT610 2G computer, and Based on Visual Studio 2010 environment, combination of using C ++ language and DirectX 11 programming. The picking algorithm which is running on the CPU is based on screen space .It is the combination of the method of picking-up point cloud based on CPU and the GPU-based point cloud pickup algorithms. The point cloud number in the experimental data is in the range of 130,000 to 18, 000, 000. By drawing crosshairs on the point selected to judge the accuracy of selection, as shown in Figure 5.

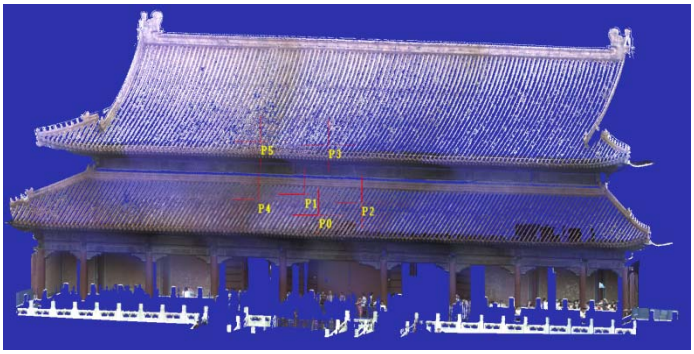

Figure 5: Example of pickup point

By testing different sizes of data, respectively obtain the amount of picking time which is CPU-based or GPU-based. After statistical methods to obtain statistics which is shown in Table 1:

For picking algorithm based on screen space, the main time-consuming is to make space conversion and distance comparison. A major time-consuming of GPU-based algorithm for picking is to copy the results from the GPU to the CPU. For the small amount of data of the point cloud (the number of points less than 2 million), while picking method based on GPU is faster, the time consuming of two methods ‘s difference is not large. For large amounts of data point cloud (the number of point is greater than five million), GPU-based picking method is more than two times the speed of the methods which is based onscreen space . And with the increasing amount of data, the speed advantage of GPU-based method of picking becomes more and more obvious. Thus, for modern high-performance of GPU, it is suitable for performing a large number of general purpose computing. For large point cloud, use the GPU for pickup has a large speed advantage.

TABLE 1: Experimental statistics table

the number of points from point cloud (Ten thousand)	time consuming of the method of picking-up point cloud based on CPU (A millisecond)	time consuming of the method of picking-up point cloud based on GPU (A millisecond)
13.1516	11.806	9.345
28.2361	21.554	16.342
57.1459	42.917	25.429
79.0590	58.134	34.365
139.1901	105.674	72.547
228.2259	203.952	105.149
440.0833	384.890	180.702
608.3357	556.826	240.427
836.0005	706.340	313.959
965.2038	816.199	372.684
1791.3302	1296.009	560.421

ACKNOWLEDGEMENT

This paper was supported by plan projects National Administration of Surveying, Mapping and Geoinformation of China (Grant NO.2013CH-15) and by the National Natural Science Foundation of China (Grant No. 41301429).

REFERENCES

[1] YAO Ji-quan, LI Xiao-huo.Research on 3-dimension pick-up of human-computer interaction in computer graphics. Journal of Engineering Design,(2006).
[2] ZHU Ming-Liang,DONG Bing,WANG Yi,XIE Bu-ying .Algorithm for Picking in 3D Scenes Based on Viewport Space.Journal of Engineering Graphics, (2008).
[3] Abate, A.F., Nappi, M., Ricciardi, S., etc. GPU accelerated 3d face registration / recognition. Lecture Notes in Computer Science, Advances in Biometrics. (2007).
[4] HANRAHAN P,HAEBERLI P. Direct WYSIWYG painting and texturing on 3D shapes.ACM SIGGRAPH Computer Graphics,(1990).
[5] ZHANG Jia-hua, LIANG Cheng, LI Gui-qing.3D Primitive Picking on GPU. Journal of Engineering Graphics,(2009).
[6] ZHAO H,JIN X,SHEN J,etal. Fast and Reliable Mouse Picking Using Graphics Hardware. International Journal of Computer Games Technology,(2009).
[7] AKENINE-MOLLER T,HAINES E. Real-Time Rendering,Third Edition. USA：A K Peters,(2008).
[8] Bosch J, Goswami P, Pajarola R. Simple and efficient terrain rendering on the GPU. Proceedings of EUROGRAPHICS,(2009).
[9] Moslah, A. Valles-Such, V. Guitteny, S. Couvet and S. Philipp-Foliguet, Acceleratedmulti-view stereo using parallel processing capababilities of the GPUs, 3DTV Conference, (2009).
[10] Kim J, Hong S, Nam B. A performance study of traversing spatial indexing structures inparallel on GPU. Proceedings of the 14th IEEE International Conference on HighPerformance Computing and Communication, .United kingdom: (2012).
[11] GUO Yan-xia, HOU Tong-pu, DU Yuan-yuan. Picking Entities in 3D Scene Based on DirectX. Journal of Liaoning University of Petroleum & Chemical Technology, (2009).
[12] Nie Hui. Interactive mapping of pixel pickup. Journal of Northwest Institute of Textile Science And Technology, (1997).
[13] Microsoft Corporation. DirectX 11 SDK. USA: Microsoft Corporation,(2010).
[14] FRANK D. Introduction to 3D Game Programming with DirectX 11.USA: Mercury Learning & Information,(2012).
[15] ZINK J,PETTINEO M,HOXLEY J. Practical Rendering and Computation with Direct3D 11. USA：A K Peters,(2011).
[16] Boyd, C. Direct3D 11 Computer Shader: More generality fro advanced techniques. USA. (2008).
[17] Wendy Jones, Allen Sherrod. Beginning Directx 11 Game Programming. Delmar Cengage Learning, (2010).