УДК 004.92 + 004.942

HIGH-QUALITY HARDWARE ACCELERATED VISUALIZATION OF PARTICULATE MATERIALS

Volkov V.P. [1], *Dosta M.* [1], *Heinrich S.* [1], *Svjatnyj V.A.* [2]
[1]*Hamburg University of Technology, Germany;*
[2]*Donetsk National Technical University, Ukraine.*

This contribution describes a novel subsystem for the visualization of particulate materials. It is based on a rendering method that takes advantages of modern graphics hardware and performs interactive representation of large particle assemblages. The method, which has been proposed in this work, allows to reach high level of visual quality with further data export into image and video files. The results of performance tests have shown high calculation efficiency of the novel subsystem.

Introduction

More than 60% of all products which are sold by such companies as DuPont, BASF or ICI to the customers are amorphous, crystalline or polymeric solids. In order to optimize production processes, minimize energy consumption and develop effective control strategies a numerical simulation of them is performed. The simulation of these processes can be done on the different time and length scales [1]. Simultaneously with empirical or semi-empirical models which are used on the macroscale, the microscale models can be effectively applied.

Contrary to the empirical models, the microscale simulations allow to consider the material microproperties and apparatus geometry. In the last two decades the usage of the discrete element method (DEM) on the microscale is a state of the art [2]. In the DEM each particle is considered as a separate entity. To receive an appropriate simulation results the number of the modeled discrete elements in the system should be large enough and in many cases it can be greater than millions of particles.

The visualization of the DEM simulation results plays an important role in studying the behavior of solids processes. However, the visualization of millions of moving particles is a computationally complex task, which requires special technique for the real-time rendering (ability to represent more than 30 frames per second). The simplified implementation of the visualization subsystem leads to the enormous high computational effort. Therefore, the advanced approaches have been developed and implemented in the novel software tool.

1 Visualization of the discrete elements

The granular material which is simulated with the DEM can be effectively represented as a set of spherical objects. These spheres are spatially distributed in the three-dimensional space, have different diameters and change their positions with time. The most common way to visualize such systems is the usage of the OpenGL [3] graphical library, where particles are rendered as polygonal spheres. However, this visualization mode has a relatively small scalability factor related to the number of spheres. The increase of particles number leads to the sufficient decrease of rendering performance. Therefore, it is problematically to use this mode for the DEM simulations. In order to find the root of the problem and to use alternative visualization approaches the image generation pipeline should be analyzed.

The video card generates images through a pipelined sequence of operations as it is shown in Figure 1. A limitation factor of a is its slowest stage (pipeline bottleneck). Therefore, the first step in optimizing of graphical applications is to identify the slowest graphics pipeline stage.

OpenGL Graphic Pipeline (data advances from left to right)						
Runs on CPU		Runs on GPU				
Application	OpenGL Driver	Vertex Shading	Geometry Shading	Primitive Assembly	Fragment Shading	Frame Buffers
					Light Operations	
					Texture Operations	

Figure 1. Graphics pipeline stages

Nowadays, the one commonly used software tool to identify the bottleneck of the graphics pipeline is a program gDEBugger [4]. It allows to disable various stages of the OpenGL graphic pipeline and to observe the changes in performance. This methodology requires the usage of some performance counters that will enable measuring the overall performance. CPUs and GPUs utilization along with frames/seconds counters are usually the best metrics for this purpose.

From the performance analysis the conclusion has been drawn, that the bottleneck of polygonal spheres is in the stage of the vertex shading. Each polygonal sphere is represented as a set of vertices which afterwards should be visualized. Therefore, the number of rendered primitives in most cases is increased in two orders of magnitudes.

The best way to optimize this graphics pipeline stage is to represent the spheres as "point sprites" [5]. The main idea of this optimization is to display a sphere from a single vertex using special vertex and fragment shaders. Such approach reduces total number of vertices in the dozens of times, increases frame rate and can lead to the generation of more realistic picture. In Figure 2 the different visualization modes are compared.

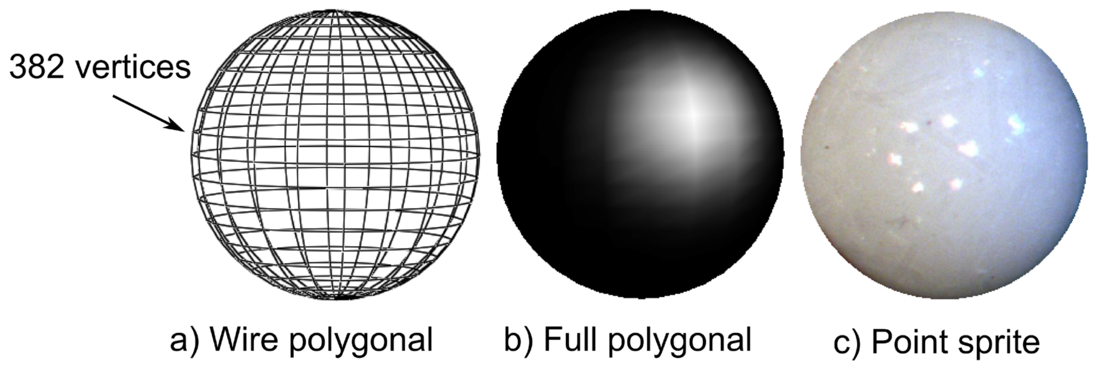

Figure 2. Sphere models

2 Performance tests

Results of performance tests for polygonal and point sprite modes are compared in Figure 3. Polygonal spheres were drawn using function gluSphere with 12 slices and 12 stacks (134 points pro sphere). The performance tests were carried out on the graphics card Intel GMA X4500.

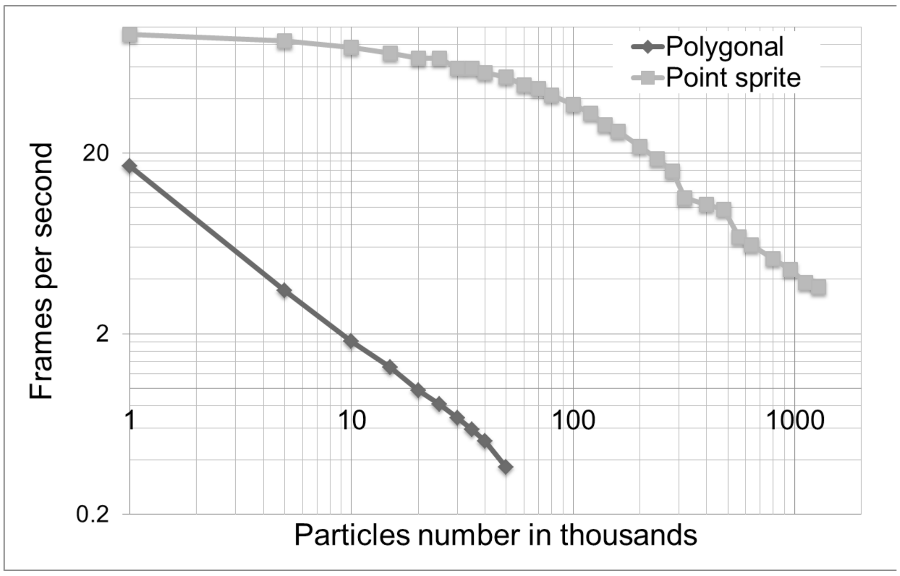

Figure 3. Performance test result

As it can be seen in Figure 3, the performance of the point sprite mode is significantly higher than the performance of the standard polygonal mode. On the graphic card GMA X4500 the point sprite mode allows to visualize up to 100000 spheres in real-time (more than 30 frames per second).

3 Selection of an individual particle

For many research purposes it is important to monitor the position of an individual particle. That is the particle selection mode has been implemented why into the system. In this mode the particle or particle assemblies can be specified by mouse click. Afterwards, the information about selected species is printed.

The main difficulty related to the selection mode is to determine required particle by given mouse coordinates (hit test). In the first step of hit test algorithm, the equation of the ray hit test in the three-dimensional space should be determined. This ray is limited by near and far clipping planes (points N and F), as it is shown in Figure 4.

The coordinates of the points N and F are calculated as follows:

$$\begin{pmatrix} X_{obj} \\ Y_{obj} \\ Z_{obj} \\ U \end{pmatrix} = (P_m \times MV_m)^{-1} \times \begin{pmatrix} \frac{2X_{win}}{W_{view}} - 1 \\ \frac{2Y_{win}}{H_{view}} - 1 \\ 2Z_{view} - 1 \\ 1 \end{pmatrix} \qquad (1)$$

where X_{obj}, Y_{obj}, Z_{obj} are the coordinates of the point; U is unused variable, included for consistent matrix notation; P_m is an OpenGL projection matrix, which specifies view frustum parameters; MV_m is an OpenGL modelview matrix to specify eye position; X_{win}, Y_{win} are the two-dimensional coordinates of mouse cursor; W_{view}, H_{view} is a viewport width and height; Z_{view} is a parameter which should be equal to -1 or to +1 to obtain the coordinates of N and F accordingly.

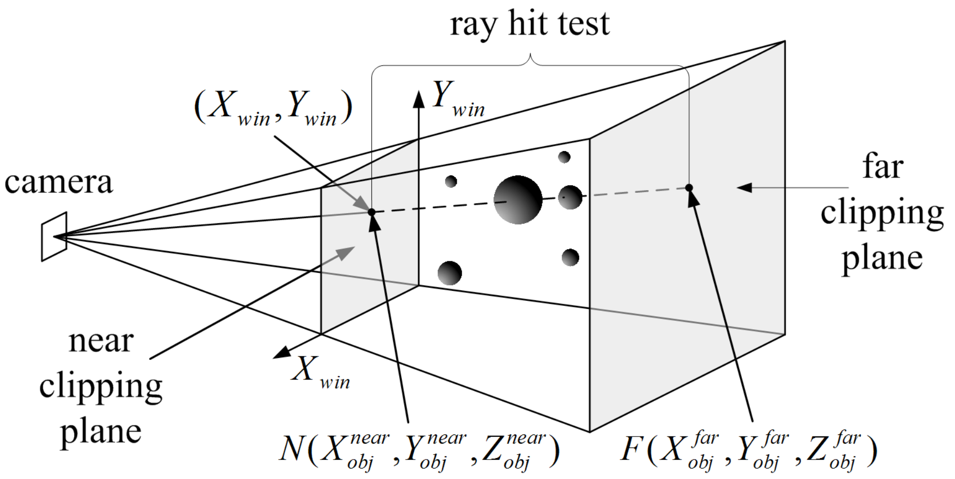
ray hit test
(X_{win}, Y_{win})
Y_{win}
camera
far clipping plane
near clipping plane
X_{win}
$N(X_{obj}^{near}, Y_{obj}^{near}, Z_{obj}^{near})$
$F(X_{obj}^{far}, Y_{obj}^{far}, Z_{obj}^{far})$

Figure 4. General representation of the ray hit test

In the second step of the algorithm the intersection between the line of sight and particles is determined. Firstly, the additional parallelepiped is used to minimize calculation effort. The diagonal of it is determined by previously calculated ray hit test. Secondly, for each particle which is situated in the parallelepiped volume the following equation system is solved:

$$\begin{cases} \dfrac{X_{cross} - X_{obj}^{near}}{X_{obj}^{far} - X_{obj}^{near}} = \dfrac{Y_{cross} - Y_{obj}^{near}}{Y_{obj}^{far} - Y_{obj}^{near}} = \dfrac{Z_{cross} - Z_{obj}^{near}}{Z_{obj}^{far} - Z_{obj}^{near}} \\ (X_{cross} - X_{center})^2 + (Y_{cross} - Y_{center})^2 + (Z_{cross} - Z_{center})^2 = R^2 \end{cases} \tag{2}$$

where X_{cross}, Y_{cross}, Z_{cross} are coordinates of intersection point; X_{center}, Y_{center}, Z_{center} are sphere coordinates; R is a sphere radius.

If the solution of an equation system (2) exists and intersection point lies between points N and F, then this indicates that current sphere is intersected by ray hit test. In this case the distance between intersection point and point N is determined as follows:

$$D_i = \sqrt{\left(X_{obj}^{near} - X_{cross}\right)^2 + \left(Y_{obj}^{near} - Y_{cross}\right)^2 + \left(Z_{obj}^{near} - Z_{cross}\right)^2} \tag{3}$$

Finally, particle, that should be selected, must satisfy the following condition:

$$D_{selected} = min(D_i); \quad i \in [1; n] \tag{4}$$

where n is the total number of particles.

4 Functionality of the novel subsystem

The presented in this contribution visualization subsystem allows to:
– perform particles rendering in polygonal and point sprite method;
– specify granules color in accordance to physical parameter (particle mass, velocity,

size, etc.);

- display coordinate system and coloring palette;
- save current view as a high resolution image file;
- save transient process behavior as video file of VMW (Windows Media Format) format;
- mark specified particles or agglomerates;
- visualize liquid bridges and solid bonds;
- perform rendering on a large number of different modern graphics cards. The specific behavior of some OpenGL drivers is considered to provide correct visualization.

In Figure 5 the visualization example is shown. In this figure the agglomerate structure is illustrated.

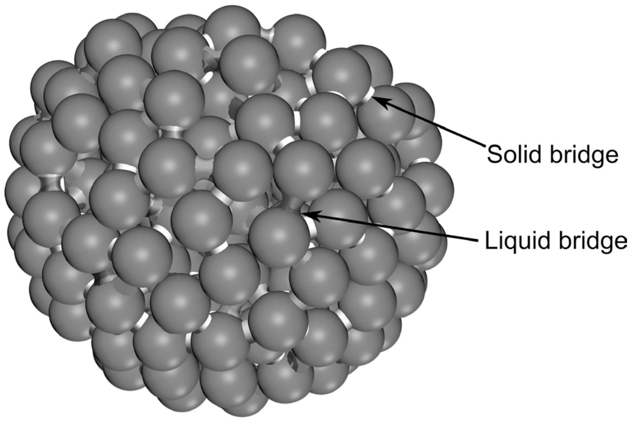

Figure 5. Visualization example

Conclusions

The presented in this contribution software subsystem is able to perform high-quality visualization of granular materials. Such particulate materials can be consisting of a large amount of particles. Therefore, the one of the most important demands on the novel tool is the optimal usage of the graphic cards.

In order to increase the performance characteristics of the novel software package, the advanced rendering approach has been proposed and implemented. This approach is based on the point sprite rendering mode. The test investigations which have been performed have underlined the effectiveness of the new program.

References

[1] M. Dosta, S. Antonyuk, S. Heinrich, „Multiscale simulation of fluidized bed granulation process”, Chemical Engineering & Technology, 2012, Vol. 35, 1373-1380.

[2] H. P. Zhu, Z. Y. Zhou, R. Y. Yang and A. B. Yu, "Discrete particle simulation of particulate systems: A review of major applications and findings," Chemical Engineering Science, 2008, Vol. 63, pp. 5728-5770.

[3] OpenGL home page. Electronic recourse. Access mode:

[4] http://www.opengl.org/

[5] gDebugger home page. Electronic recourse. Access mode:

[6] http://www.gremedy.com/

[7] G. Reina, T. Ertl, “Hardware-Accelerated Glyphs for Mono- and Dipoles in Molecular Dynamics Visualization”, Proc. of Eurographics Symposium on Visualization, 2005, 171-182.