Abstract

Real-time large-scale granular flow scene simulation has become a challenging task because massive amounts of geometric and graphical computation need to be rapidly processed. The paper presents a new simulation method that compute Discrete Element Method (DEM) entirely on GPU. To achieve this goal, Spatial subdivision algorithm is firstly proposed to partition space into a uniform grid, we also propose parallel radix sort algorithm and then sort the particles based on their hash values. With the above methods, we can effectively increase speed for searching neighboring particles and interacting with boundaries. To reach the requirement of real-time simulation of large-scale granular flow based on the Discrete Element Method (DEM), the implementation is made with the NVIDIA Compute Unified Device Architecture (CUDA). The experimental results show that the method proposed not only can simulate large-scale granular flow scene in real-time, but can simulate various kinds of granular flow behaviors in real-time by adjusting the very few experimental parameters.

1. Introduction

Granular flow is composed of discrete particles which display liquid and solid lubricant behavior in sliding contacts [1]. Granular flow problems are very common in many agricultural and industrial areas such as agricultural production, food industry, chemical engineering, pharmacy, metallurgy, source of energy, etc. They are also in natural disaster such as debris flow, landslides, avalanche, sand dune evolution, etc.
Since granular materials have characteristics of variability and dynamic nature with the change of external factor, an uneven distribution of chain-like mechanical structure indicates many unique features different from regular solid and fluid. So the study of granular materials, recently, has become a hot topic in the world, which has the important scientific significance and signification of engineering application [2~4]. For now, the numerical simulation based on DEM was used to simulate and analyze granular flow problems, whose fundamental idea is: the discontinuous bodies are separated into the combination of rigid element to make these rigid element satisfy the equation of motion, and then the equations of motion of these rigid element are solved to seek mass motion of discontinuous body by using iterative methods at each time step [5].

In Geotechnical Engineering and Powder Engineering fields, many studies have been done for granular flow. But most of these works for granular flow focus on the cause of the motion, the movement mechanism, and the numerical simulation. The study goal in this paper is to simulate large-scale granular flow scene by constructing a granular flow model based on DEM. However, with the growing number of particles, it is difficult to simulate real-time large-scale granular flow scene on CPU.

With the rapid improvement of computer hardware performance and computing speed, GPU (Graphic Process Unit) Performance improvement, especially in recent years, almost is in the double in each year. GPU not only can be used to render 3D scene but supports more and more complicated calculation. More and more researchers focus on the research of larger amount of computation and more realistic simulation such as real-time large-scale granular flow scene simulation and fluid simulation [6]. Many previously unavailable real-time computing algorithms on the CPU currently have been realized on the GPU due to GPU performance improvement [7], which makes it possible to simulate large-scale granular flow scene based on complex physical models.

In this paper, the simulation of large-scale granular flow scene based on DEM is performed on the GPU, and accelerated computing and rendering is realized by using GPU techniques. However, computational complexity of simulation scene increases rapidly when the interaction of granular flow and boundary occurs in 3D scene, which causes the problem to be resolved about how to simulate granular flow in real-time and keep reality of large-scale granular flow scene in interactive virtual scene.

2. Related work

In recent years, the energy consumption and the frequent occurrence of natural disasters which are caused by granular flow problems had brought tremendous financial damage for people, so the simulation of granular flow has become hot topic in geotechnical engineering, agricultural and industrial areas. On the other hand, DEM, in the micro-scale of particle, is used to study the change of various kinds of quantities, and can reflect basic qualities of some processes. It is an analytical tool in mechanics of granular media and especially suited for the study of granular flow. So the study of granular flow using DEM, in recent years, is developed quickly in geotechnical engineering, agricultural and industrial areas.

DEM is rooted in molecular dynamics, and is first put forward by Cundall [8]. Since the DEM theory was put forward, it had played an irreplaceable role in numerical methods in the two traditional domains [5]: geotechnical engineering and powder engineering. Originally, the mechanical behaviors of discontinuous media such as rocks are its object of study. Later, Cundall and Strack put forth two-dimensional computer simulation for granule [9], which is the beginning of simulating the motion of granule by using DEM. Soon afterwards, Cundall and Strack joined ITASCA USA International Engineering Consulting Corporation. This Corporation launched two-dimensional circular disk program BALL and three-dimensional pellet program TRUBALL, which were developed into Commercial software PFC-2D/3D [10, 11]. The study of DEM starts relatively late in our country, but is developing rapidly. In recent years, many domestic scholars have made many contributions in this study fields. In 1986, the basic principles of DEM and its application examples were first introduced to rock mechanics...
and engineering by Yong-Jia Wang in China [12]. In the following over twenty years, a lot of experts and
scholars in China applied DEM to geotechnical engineering [13~17], and DEM becomes an effective way
to simulate granular flow for fields of geotechnical engineering.

However, most of the recent studies about granular flow using DEM pay more attention to numerical
simulation and the mechanism, applying DEM to the study on the modeling and rendering of large-scale
granular flow dynamic scene has been still scarce. So this paper presents a new method to construct a
computer model to simulate real-time large-scale granular flow scene based on DEM.

3. Equation of motion of granular flow based on DEM

DEM is kind of numerical simulation according to Newton's second law of motion and its
constitutional relationships between force and displacement on the points of contact between particles.
The equations of motion for a random particle i is [18, 19]:

\[ m_i \frac{dv_i}{dt} = \sum f^c_{ji} + f_i \]  \hspace{1cm} (1)

Eqs.(1) give the equations of motion of force. Where \( m_i \) is the mass of particle i, \( v_i \) is translational
velocity, \( f^c_{ji} \) is contact force, \( f_i \) is the force is exerted on the particle i.

\[ I_i \frac{d\omega_i}{dt} = \sum (R_i \times f^c_{ji} + M^B_{ji}) + M_i \]  \hspace{1cm} (2)

Eqs.(2) give the equations of motion of moment. Where \( I_i \) moment of inertia, \( \omega_i \) is rotational speed, \( R_i \)
is vector from centre of particle i to adherent point, \( M^B_{ji} \) is rolling moment of resistance due to uneven
normal force distribution on interface.

When two particles make contact with each other, the contact force \( f_{c} \) is generated between particles.
The contact forces \( F_c \) can be decomposed into the normal contact force \( F_{cn} \) perpendicular to interface
and the tangent contact force \( F_{cs} \) parallel to interface.

\[ F_c = F_{cn} + F_{cs} = F_{cn}n + F_{cs}s \]  \hspace{1cm} (3)

Where \( n \) and \( s \) is the normal vector and the tangent vectors between particle i and particle j, respectively.
A contact model between particles is core of DEM. Now the popular and mature application is that
particles are exerted on deformed elements such as Spring, damper etc. Particle j exerts contact normal
force \( F_{cn} \) on particle i and the contact normal force \( F_{cn} \) are given by:

\[ F_{cn} = f_{ls} + f_{ld} = -k(d - |r_{ij}|) \frac{r_{ij}}{|r_{ij}|} + \eta v_{ij} \]  \hspace{1cm} (4)

Where \( f_{ls} \) is repulsive forces between particle i and particle j. A repulsive force \( f_{ls} \) is modeled by a
linear spring. A damping force \( f_{ld} \) is modeled by a dashpot—which dissipates energy between
particles—are calculated for a particle i colliding with a particle j, where \( k, d, r_{ij}, \) and \( v_{ij} \) are the spring
coefficient, particle diameter, and relative position and velocity of particle j with respect to particle i,
respectively [20]. A shear force \( F_{cs} \) is also modeled as a force proportional to the relative tangential velocity \( v_{ij,t} \):
Where the relative tangential velocity $v_{ij,t}$ is calculated as

$$v_{ij,t} = v_{i,j} - \left( v_{i,j} \cdot \frac{r_{i,j}}{|r_{i,j}|} \right) \frac{r_{i,j}}{|r_{i,j}|}$$

(6)

Then the force and torque applied to a rigid body are the sums of the forces exerted on all particles of the rigid body:

$$F_c = \sum_i (f_{i,s} + f_{i,d} + F_{cs})$$

(7)

$$T_c = \sum_i (r_i \times (f_{i,s} + f_{i,d} + F_{cs}))$$

(8)

Where $r_i$ is the current relative position of particle $i$ to the center of mass.

4. Collision detection and reaction

A very time-consuming step in most physics simulations is collision detection and reaction, so this undertaking is a primary target for optimization, which has a direct impact on real-time of large-scale granular flow scene simulation. To detect collision between particle and boundary, we can partition space into a uniform grid, and then distance field is sampled on scene. When the distance between particle and boundary is smaller than particle radius, we can determine that the collision between particle and boundary has occurred. To detect collision between particles, we construct hash table of particle ID and cell ID, and then sort the particles based on their hash values. By using this hash table, we can effectively increase speed for searching neighboring particles and interacting with boundaries because pairs of particles that cannot possibly collide have been quickly culled away.

4.1. Collision detection and reaction between particle and boundary

After the boundary of granular flow has been defined, we need to detect collision for granular flow and boundary. Our method is similar to literature [21] but is somewhat adjusted. We partition space into a uniform grid and sample distance field on scene, and then store sampled-information into cell. When distance ($d$) value between particles and boundary is less than threshold ($T$) value of collision boundary, we can determine that the collision between particle and boundary has occurred. In the process of this collision, we need to calculate velocity ($v$) of colliding particles relative to the velocity of boundary. We just process the stationary boundary in this paper. So we just process the particles velocity themselves when collision has been detected and particles are located within the boundary. Because energy will be dissipated between particle and boundary in the process of collision, we set up damping coefficient $D$ ($D = -0.6$ in this paper), which can simulate the movement of the particles according to velocity component in $x$, $y$, $z$ coordinates.
4.2. Collision detection and reaction between particles

To calculate interactive forces caused by all the other particles, we not only need to sort and search the particles in entire space but to detect collision between particles. However, we can speed up collision detection between the particles once the entire space has been divided into a uniform grid. The collision detection in this paper is performed according to distance between the particles. When the distance between two particles is smaller than their diameter, collision occurs. However, large-scale granular flow scene involves a larger number of particles, whose collisions must all be computed. If we don’t employ an optimization algorithm for these numerous particles, we have to check $n^2$ pairs for $n$ particles, whose time complexity is $O(n^2)$ [22,23]. This approach is impractical, plus it is difficult to solve such a large problem in real time even with current GPUs [24]. Thus, we use a three-dimensional uniform grid cell to reduce the computational burden. The grid initialized covers the whole computational domain, and its cell size is the same as the size of the particle. We can calculate a hash value for each particle based on its cell id, we then sort the particles based on their hash values when collision occurs. With the above methods, we can effectively increase speed for searching neighboring particles and interacting with boundaries because pairs of particles that cannot possibly collide have been quickly culled away. Consequently, the efficiency is improved from $O(n^2)$ to $O(n \log n)$.

5. Implementation of simulation on the GPU

It is suitable for DEM to simulate real-time large-scale granular flow scene on GPU. It is also relatively easy to parallelize particle systems and the massive parallel computation capabilities of modern GPUs, which makes it possible to simulate large systems at interactive rates. In our simulation system, the computation of each particle is assigned to each thread, which can take advantage of the massive parallel computation capabilities of modern GPUs to meet real-time requirement.

5.1. Processing procedures

The change of motion state of granular materials is caused by the changes of external force and the internal stresses. The 3D scene requires hundreds of thousands of particles to improve the details of particles movement. If calculations of these particles are entirely performed on CPU, the real-time and mutual requirements can’t be met for large-scale granular flow scene simulation [25].

This paper presents a kind of granular flow simulation method based completely on GPU, which makes full use of high performance computing to simulate granular flow in real-time using DEM. Granular flow simulation is shown in the flow chart (see Fig1):
5.2. Initialization of granular flow system

Initialization of granular flow system includes the numbers of particles, particles distributions, the colour of particles, it also includes some DEM parameter such as gravitational acceleration, damping coefficient, time slice etc. The distance (d) value of collision between particles is also the part of initialization. To render particles, we need to initialize the size of point particles and the size of ball particles. Apart from that, the background and boundary conditions need to be initialized.

5.3. Spatial subdivision

Spatial subdivision partitions space into a uniform grid, such that a grid cell is at least as large as the largest object. Each cell contains a list of each particle whose centroid is within that cell [26]. We use a
grid where the cell size is the same as the size of the particle (double its radius). A collision test between
two particles is then performed only if they belong to the same cell. If a grid where the cell size is more
than the size of the particle, we should not only test whether the two particles locate the same cell but test
whether one of two particles centroid is within that cell when we test the collision between particles,
which is not necessary.

After uniform grid has been partitioned, we need to create a list of particle IDs along with a hashing of
the cell IDs in which they reside, sorting this list by cell ID. The positions of particles, under the action of
force, will change after each iteration. The hash table will also be updated with the change of particles
position.

5.4. Computation of basic physical quantities

According to the initialized parameters and equation of motion, we can calculate relative position of
particles, relative velocity between particles, relative tangential velocity etc. The hash table then is
constructed to find a relationship between particles and cells. With the above methods, we can compute
the coordinate of the position of each particle at any given time.

5.5. Parallel Radix Sort Algorithm based on GPU

The main computation process of DEM is as follows [5]:

a) Detection of contacts, the interaction relation, the calculation of interactional physical quantities
   (use relational data as the operating object)

b) Analysis of equation of motion, the update of cell physical quantities (uses cell data as the
   operating object)

c) The calculation of other equivalent physical field (stress, strain etc)

d) The calculation of incremental time and moving on to next time step

The key, for all calculation above, is to how to calculate interaction force between particles, and how to
construct the hash table between particle and cell.

An efficient Sort algorithm can ensure the calculation of interaction force between particles, So it is the
key to realize the scene simulation using DEM.

The choice of sort algorithm is limited to a stable sort because we have already ordered the cell ID
array first by object ID. The stability of a radix sort guarantees that this expected direction of sorting
works. So we choose The Parallel Radix Sort Algorithm.

Our simulation scene is performed on PC, whose type of Graphics card is NVIDIA GeForce 8800GT.
There is 16 K of shared memory available per multiprocessor on this GeForce 8800 GTX.

Because we are targeting the GeForce 8800 GTX with its 16 multiprocessors, we use Num_Blocks =
16 thread blocks. n threads per thread block is divided into n/R thread groups. A radix sort is sorted by
keys (the 32-bit cell IDs).

After all cell IDs have been created, we need to count them to compute their total number for use
during the creation of the collision cell list. Each thread requires its own set of counters, or else threads
could potentially corrupt each other's counters. To avoid read-after-write or write-after-read conflicts
between threads of the same thread group, we have to construct a serialized increment of the radix
counters for each thread group.
The resulting array of radix counters is then processed to convert each value into offsets from the beginning of the array. Finally, these offsets are used to determine the new positions of the keys for this pass, and the array of keys is reordered accordingly.

Our method is similar to literature [27]. The difference is that we first calculate the prefix sum for Thread Block in Shared memory. In this way, Thread Block has been correct order before global memory is written back, which minimize confliction of the transmitting data caused by hash and make full use of memory Bandwidth of GPU.

5.6. Rendering of granular materials

The grid data structure is generated from scratch each time step. Each particle is assigned to a grid cell based on its center point. The index of the particle is stored in the cell. Spatial coordinates of particles is Output at each time step. Two renderer in our simulation scene is used to render granular flow: point particles renderer and ball particles renderer, particles renderer render is used to render ball particles, point particles renderer is used to render transparent point particles.

6. Result

The software used in the experiment of 3D terrain and rainfall rendering includes Windows XP, Visual Studio 2005, and OpenGL. The experiment is conducted on an ordinary Personal Computer. Hardware configuration is listed as follow:

- CPU: Intel E8400, 3.0 GHz 1333 MHz FSB, 6MB L2 cache;
- Graphics card: NVIDIA GeForce 8800GT, 512M memory size;
- Total host memory: 2.0 GB.

Granular flow motion simulation in this paper is just on gravitational effects, not considering wind-force effects and electromagnetic field effects.

The experiments are divided into two groups. Our first set of experiments is shown in Figure 2: Cubic block composed of numerous particles, on gravitational effects, falls to the slope with an inclination of 30 degrees, which forms flying particles.

![Fig 2. Falling cubic block composed of numerous particles falls to the slope with an inclination of 30 degrees](image)

Our second set of experiments is shown in Figure 3: Spherical block composed of numerous particles falls to the surface of Granular flow, which has both a ripple and an echo effect.
Fig 3. Spherical block composed of numerous particles falls to the surface of Granular flow

The relation of the number of particles and FPS are shown in Table 1. When the number of particles reaches 200,000, the rendering speed of large-scale granular flow scene can still reach as high as 37.9 FPS.

Table 1. Relation table of the number of particles and FPS

<table>
<thead>
<tr>
<th>Particle number ($10^4$)</th>
<th>FPS (frames per second)</th>
<th>Each frame time/s</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5</td>
<td>190.6</td>
<td>0.0052</td>
</tr>
<tr>
<td>0.8</td>
<td>180.6</td>
<td>0.00553</td>
</tr>
<tr>
<td>1.6</td>
<td>139.4</td>
<td>0.00717</td>
</tr>
<tr>
<td>4.0</td>
<td>119.1</td>
<td>0.00840</td>
</tr>
<tr>
<td>6.0</td>
<td>84.6</td>
<td>0.0118</td>
</tr>
<tr>
<td>8.0</td>
<td>56.8</td>
<td>0.0176</td>
</tr>
<tr>
<td>12</td>
<td>45.8</td>
<td>0.0218</td>
</tr>
<tr>
<td>16</td>
<td>40.8</td>
<td>0.0245</td>
</tr>
<tr>
<td>20</td>
<td>37.9</td>
<td>0.0264</td>
</tr>
<tr>
<td>40</td>
<td>25.4</td>
<td>0.0394</td>
</tr>
<tr>
<td>60</td>
<td>19.7</td>
<td>0.0508</td>
</tr>
<tr>
<td>100</td>
<td>12.9</td>
<td>0.0775</td>
</tr>
</tbody>
</table>

7. Conclusion and discussion

The basic concept of DEM is firstly introduced in this paper. And then the simulation about granular flow problem using DEM is introduced. Finally, how to simulate real-time large-scale granular flow scene on GPU by using DEM is given. Experiment results demonstrate that the method proposed not only can simulate large-scale granular flow scene in real-time, simulate various kinds of granular flow behaviors in real-time by adjusting the very few experimental parameters. Our future works is as follows:

1. Simulate granular flow composed of uneven particles, which ensure more realistic granular flow simulation.
2. Simulate more phenomena based on our method, and render Multi-phase Flow scene using the technique of extraction of surface.
3. Make more use of performance of GPU, enhance computational efficiency and improve realistic 3D effect.

Acknowledgements
This work was supported by the National Basic Research Program of China (973 Program) (No. 2010CB731504). It was also supported by the Youth Talent Team Program of IMHE, Chinese Academy of Sciences (No.SDSQB-2010-03).

References


**Jian Ye** is currently working toward the Ph.D. degree at Institute of Mountain Hazards and Environment, Chinese Academy of Sciences. Current research focus is on the 3D mountain disaster simulation.