Discrete Element Simulation of Mill Charge in 3D using the BLAZE-DEM GPU framework.

Nicolin Govender*[1,3] , Raj K Rajamani[2], Schalk Kok[1], Daniel N Wilke[1]

[1] *University of Pretoria, Department of Mechanical and Aeronautical Engineering, Pretoria, 0001, South Africa*

[2] *Metallurgical Engineering Department, University of Utah 135 South 1460 East Salt Lake City, Utah-84112, USA*

[3] *Advanced Mathematical Modeling CSIR, Pretoria, 0001, South Africa*

Abstract

The Discrete Element Method (DEM) simulation of charge motion in ball, semi autogenous (SAG) and autogenous mills has advanced to a stage where the effects of lifter design, power draft and product size can be evaluated with sufficient accuracy using either two-dimensional (2D) or three-dimensional (3D) codes. While 2D codes may provide a reasonable profile of charge distribution in the mill there is a difference in power estimations as the anisotropic nature within the mill cannot be neglected. Thus 3D codes are preferred as they can provide a more accurate estimation of power draw and charge distribution. While 2D codes complete a typical industrial simulation in the order of hours 3D codes require computing time in the order of days to weeks on a typical multi-threaded desktop computer. This paper introduces a 3D GPU code based on the BLAZE-DEM framework that utilizes the Graphical Processor Unit (GPU) via the NVIDIA CUDA programming model. Utilizing the parallelism of the GPU a 3D simulation of an industrial mill with four million particles takes 1.16 hour to simulate one second (12 FPS) on a GTX 880 laptop GPU. This new performance level makes 3D simulations a routine task for mill designers and researchers. Furthermore the shorter compute time can elevate computations to a higher level wherein ore particle breakage and slurry transport can be included in the simulation. In this paper we verify our GPU code by comparing charge profiles and power draw obtained using the CPU based code Millsoft and pilot scale experiments. Finally, we show computations for plant scale mills.

1. Introduction

1.1. Background and Motivation

Since the first application of the discrete element method (DEM) for the simulation of grinding mills by Rajamani [1] in 1990 there has been a phenomenal growth in the variety of ways this technique is used in the mining industry. Prior to DEM, Powell's [2] single ball trajectory in rotary mills was a key advancement in understanding lifter relief angle on the trajectory of charge. This algorithm continues to serve the mining industry even today.

In the late 90s two-dimensional DEM codes where the norm, due to the ease of execution on a personal computer with a single central processing unit (CPU). On the other hand, three-dimensional simulations promise greater accuracy of simulated results at the expense of computing time. To alleviate heavy computing burden, parallel and multi node computing is pursued. As a result the three-dimensional simulation is not a typical everyday tool used by metallurgists and mill design teams. At the outset it is useful to discuss the merits of 3D code in comparison with 2D code. The 2D code is easy to execute in a matter of hours on a CPU. It has been heavily used in hundreds of mining operations for annual or semiannual replacement of shell lifters [3]. This code has impacted the production, capacity and liner life of ball mills, autogenous mills and semi-autogenous (SAG) mills. The three-dimensional simulation code animations are more accurate because the momentum of balls and rock particles in the axial direction of the mill is accounted for. It has been used for the simulation of slurry and pebble motion in a pulp lifter amongst other things **(author?)** [4]. More importantly, we have not discovered the insights possible with a 3D code. It has not been readily available to researchers and mill designers since execution times are of the order of weeks on a single CPU for a typical plant size mill, which then becomes awkward and laborious to pursue on a routine basis.

1.2. Computational Aspects

A full 3D simulation of a mill will give valuable insights into the dynamics within a mill which can improve energy efficiency resulting in a saving of thousands of dollars. An emerging trend of the past few years is the implementation of scientific and engineering solutions on a new classes of processors termed General Purpose Graphical Processor Units (GPGPU) [5, 6, 7] which offers CPU cluster computing performance at a fraction of the cost. Rajamani et.al shows a speed up of 50x over CPU implementations for mill charge motion while Govender et.al [8] showed a speed up of 132x for polyhedral particles.

1.2.1. GPU hardware

The Graphic Processor Unit (GPU) was initially developed to reduce the computational burden on the CPU during the rendering process which involves the manipulation of millions of pixels on a screen simultaneously. This required that it be made up of mostly Arithmetic Logic Units (ALU) enabling it to perform these arithmetic operations in bulk. This is opposed to control and logic in the case of the CPU [9] which must be able to perform the complex logical operations that is needed to run an operating system. A Streaming Multiprocessor (SM) on a GPU is equileveant to a core on the CPU. Each SM on a Kepler GK110 GPU can launch 2048 threads which are only capable of performing identical task (Single Instruction Multiple Data (SIMD))[10]. Each CPU core is capable of launching two threads which can work independently and perform different tasks in each thread.

Figure 2 illustrates the type of tasks that each unit excels at performing. The limited GPU outperforms the versatile CPU in spite of the considerably lower clock rate when processing identical tasks. In order to realize a speed up on the GPU we need to ensure that our DEM algorithm is completely decoupled and expressed as a SIMD task, in that we carry out the same instructions on different data elements which are particles in the case of tumbling mills.

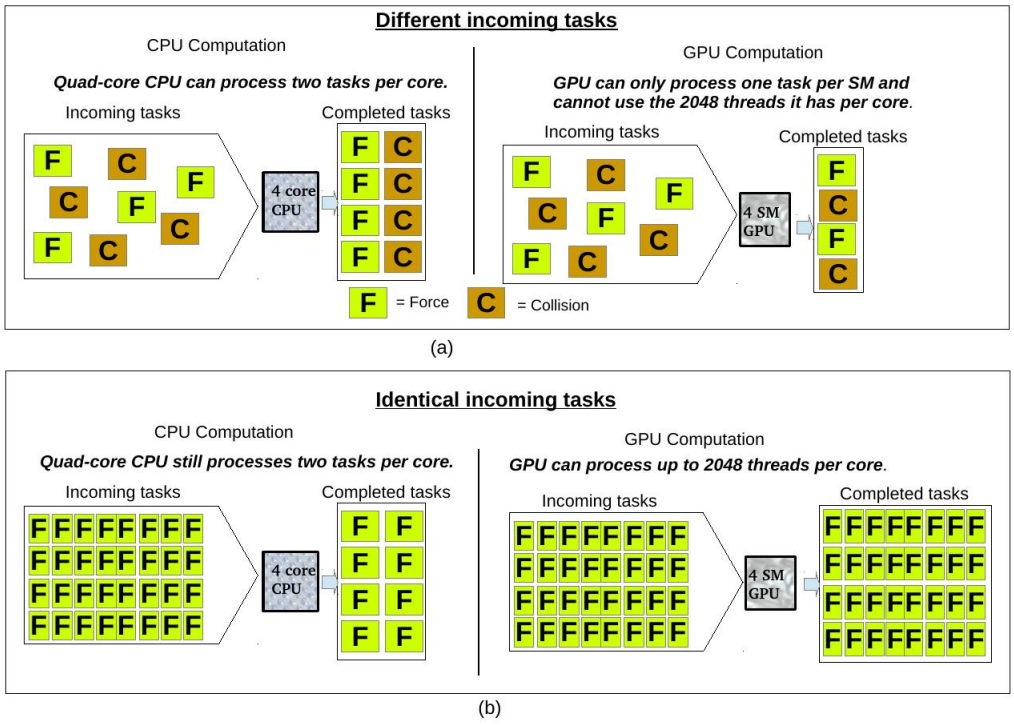

Figure 1: Comparison between CPU and GPU task processing for the case of (a) different incomming tasks and (b) identical incomming tasks.

1.2.2. GPU software platform

The NVIDIA developed CUDA programming model [10] provides access to the GPU from a variety of high level programming languages such as C++, Java and Python. CUDA batches threads into blocks (maximum 1024 threads) for execution on a SM. Threads within blocks can access fast shared memory with each thread in turn having access to its own 32 bit registers (fastest memory available). CUDA allows us

to create thousands of thread blocks containing millions of threads which get scheduled for execution on the hardware as SMs become available (we don't have control of the execution order of blocks). The execution of a block will only complete once all threads within the block have reached an end point. This is very important and requires us to design algorithms that require similar times to complete for all threads to best utilize the parallelism on the GPU. In this manuscript we show how plant scale mill simulation can be done in a matter of hours and 10s of hours for very large mill simulations.

1.3. Discrete Element Method

The details of discrete element calculations are described in a large number of publications. Here, we describe the computational steps necessary to execute the algorithm and how this algorithm is implemented on the GPU. The flow-diagram in Figure 2 describes the DEM process that we model. The bulk of computational time is spent on neighbour searching and collision detection.

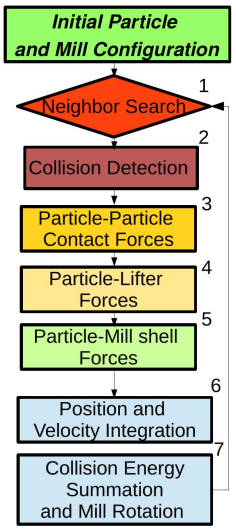

Figure 2: Flow chart of DEM simulation procedure.

An assumption in many DEM simulations is that particles are considered to be perfectly rigid for the duration of collision contact. In reality perfectly rigid particles do not exist, as all bodies will experience (to some extent) local deformations during contact. These deformations however occur on a time scale which is much smaller than what is required for capturing the macroscopic behavior of a system. Thus it is often sufficient to use a constitutive law, such as a linear spring to model contact forces. Computing the time evolution of the system requires us to solve simultaneously Newton's equations of motion for all contacting particles, which on current hardware (2014) is only possible for a few thousand rigid bodies. Hence, we assume that there are only binary contacts between particles at any given time. The total force acting on a particle is obtained by summing the individual contributions of all the binary contacts of a particle per time-step, as illustrated in Figure 3. This is a good approximation of reality provided the particles are of a similar size and move very little during a time step. Hence, computations can be carried out in parellel for all particles independent of each other.

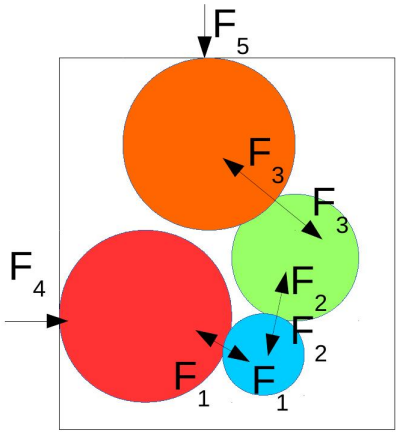

Figure 3: DEM force assumption.

In summary the assumptions that we make are :

1. Rigid bodies with single point contact,
2. Binary contact.
3. Local short-range interactions,
4. Non-incremental friction model and
5. Similar particle size.

In summary, these assumptions result in a system that is completely decoupled and can be expressed as a Lagrangian type process in which we are able to simulate the motion of individual particles independently of each other [11] resulting in our algorithim being ideally suited to the GPU.

1.4. Mill Simulation with DEM

In 2001 Venugopal and Rajamani [12] presented the DEM 3D computational methodology and showed comparison with power draft in a laboratory scale 90 cm diameter mill. In the same year Rajamani and Mishra [13] used the same 3D code for the prediction of power draft in plant scale mills. Herbst and Nordell [14] combined 3D DEM with smoothed particle hydrodynamics (SPH) and the finite element method (FEM) to simulate slurry and solid charge motion, ore particle breakage and liner wear. Cleary (2001b) demonstrated the sensitivity of charge behavior and power draft of a 5m diameter ball mill to liner geometry and charge composition using 3D code. There are continued advances in the simulation of breakage and slurry flow incorporating all the details in three-dimensional simulations. Morrison and Cleary [15] describe the evolution of "Virtual Comminution Machine", a simulation code that simulates breakage and slurry transport in tumbling mills. In their simulation both the discrete element method and smoothed particle hydrodynamics are employed for slurry and pebble flow through the grate slots and the pulp lifter. Cleary and Morrison [4] show that 3D DEM combined with SPH is a tool for analyzing mineral processing equipment such as mills, twin deck screens and so on. In a more recent study Alatalo et al. [16] compared the experimental deflection of a lifter in a ball mill with 3D predictions made with EDEM **(author?)** [17], a commercial DEM code. They concluded that 3D predictions were closer to experimental values than 2D.

2. Verification of GPU DEM code

The CPU code used in simulations shown here is the 2D DEM code Millsoft developed by Rajamani et al. [1]. The GPU code used is based on the BLAZE-DEM framework developed by Govender et al. [18, 8] with the additional algorithms for mill simulations described in Section 2.2.2.

2.1. Physical Model

A linear spring dash-pot model is used to calculate the normal force given by :

$$\mathbf{F}_{\mathrm{N}} = (\mathrm{K_n}\delta)\bar{\mathbf{n}}_{\mathrm{s}} - \mathrm{C_n}\mathbf{v}_{21}. \tag{1}$$

where δ is the penetration depth, $\mathbf{v}_{21} = \mathbf{v}_1 - \mathbf{v}_2$ is the relative translational velocity, $\mathrm{K_n}$ is the spring stiffness, $\mathrm{C_n} = \frac{2\ln(\epsilon)\sqrt{\mathrm{K_n m}}}{\sqrt{\ln(\epsilon)^2+\pi^2}}$ is the viscous damping coefficient and $\bar{\mathbf{n}}_\mathrm{s}$ the normal at contact where ϵ is the coefficent of restitution and m the mass of the particle. The model parameters should be choosen such that the penetration depth is not too much. Typical DEM simulations have a maximum penetration set at 5% of the radius **(author?)** [19].

Millsoft uses a history dependent friction model given by :

$$\mathbf{F}_\mathrm{T} = -\min\left[\mu \parallel \mathbf{F}_\mathrm{N} \parallel, \mathrm{K_T}\sum \mathbf{v}_\mathrm{T}\triangle\mathrm{t} + \mathrm{C_T}\mathbf{v}_\mathrm{T}\right]. \tag{2}$$

where $\mathbf{v}_\mathrm{T} = (\mathbf{v}_\mathrm{R} - (\mathbf{v}_\mathrm{R}.\bar{\mathbf{n}}_\mathrm{s}))\bar{\mathbf{n}}_\mathrm{s}$ is the relative tangential velocity , μ the coefficient of friction, $\mathrm{K_T}$ the tangential spring stiffness and $\mathrm{C_T} = \frac{2\ln(\epsilon)\sqrt{\mathrm{K_T}m}}{\sqrt{\ln(\epsilon)^2+\pi^2}}$ the tangential damping coefficient. However this is not possible on the GPU currently as discussed by Govender et al. [18]. Thus our GPU code uses an history independent friction model [6, 20] .This makes GPU computations many times faster than CPU computations [6, 18, 21] as looking up previous time step information on the GPU would be very inefficient. Therfore the GPU friction model is simplified to

$$\mathbf{F}_\mathrm{T} = -\min\left[\mu \parallel \mathbf{F}_\mathrm{N} \parallel, \min\left[\,\mu\|\mathbf{v}_\mathrm{T}\|\,, \min\left[\,\mathbf{v}_{1\mathrm{T}}, \mathbf{v}_{2\mathrm{T}}\,\right]\right]\right]. \tag{3}$$

This model has been shown to match experiment very well in the previous works by the authors [18]. The key difference between (2) and (3) is that during a time step (3) can only slow down a particle but not reverse its direction, which is sufficient for dynamical simulations.

2.2. Additions to the BLAZE-DEM framework

2.2.1. Calculation of power drawn by a mill

Grinding in the mineral processing industry is performed primarily by a ball mill which consumes a vast amount of energy and can account for as much as half of the processing cost. Thus understanding grinding mechanisms and estimating the power drawn by a mill can give guidance to improve energy efficiency. The harsh environment inside the mill makes obtaining experimental data difficult. Thus the physical quantities calculated in a DEM simulation provide valuable insight to improve efficiency. Since the power drawn by a mill is largely determined by the dynamics of the charge within the mill, we can obtain a good estimate by analyzing the energy loss mechanisms in a DEM simulation.

The total energy consumed by a mill is simply the net sum of the energy dissipated through contact $E_{diss} = \sum(F_{diss}.\triangle\mathrm{x})$. Here F_{diss} is given by the damping and friction terms in Equations (1)-(3) respectively. $\triangle$x is the distance over which the force acts and it is estimated by v$\triangle$t since we do not store contact history in our GPU implementation. Thus Power $= \frac{E_{diss}}{\triangle\mathrm{t}}$.

2.2.2. Collision Detection

The collision detection algorithim is based on the geometry class classification as described by Govender et al. [8] for the GPU architure:

1. We firstly do a broad phase check if the particle is beyond the cylinder that bounds all the lifters **r.**
2. If a particle passes this check then we loop over all lifters checking if there is intersection between the particle and a bounding cylinder $\mathbf{r}_\mathrm{lifter}$ of a lifter, as depicted in Figure 4(a).

Heuristic (1) requires O(N) computations and heuristic (2) requires O(K) computations where K is the number of lifters and N is the number of particles.

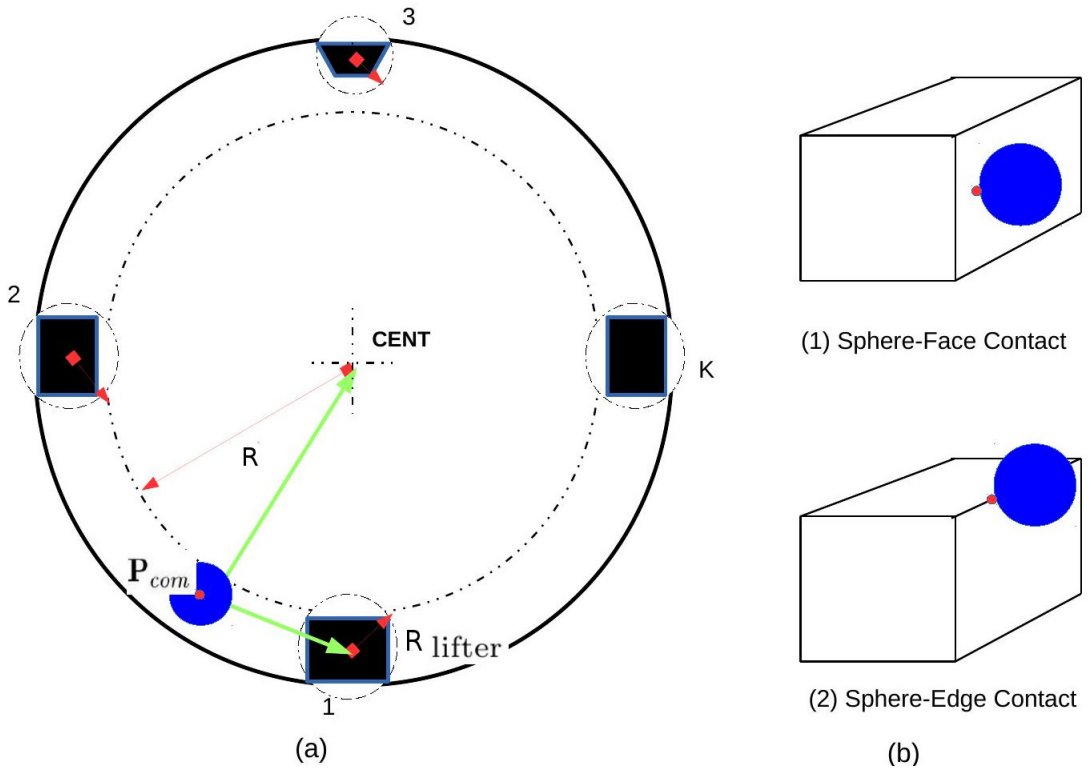

Figure 4: (a) Particle-lifter broadphase collision detection and (b) detailed collision detection.

If there is an intersection between a ball and a lifter we then find the contact normal and penetration distance as described in Algorithim 1.

Algorithm 1 Particle-lifter detailed collision detection.

1. **Loop** over all **lifter faces**
   (a) Compute the distance $\mathrm{d} = \mathbf{n}_{face} \cdot (\mathbf{P}_{com} - \mathbf{C})$ between the lifter face and particle. Here $\mathbf{P}_{com}$ is the center of mass position of the particle, $\mathbf{C}$ is the center of the face and $\mathbf{n}_{face}$ the face normal.
   (b) If this distance d is less than the particle radius we have possible contact.
      i. The contact point is given by $\mathbf{P}_{contact} = \mathbf{P}_{com} + \mathrm{d}.\mathbf{n}_i$.
      ii. We now check if the contact point is actually on the face as (a) is for an infinite plane.
      iii. We check if the penetration is valid (max 10% of radius) ($\delta = \mathrm{d} - \mathrm{r} < 0.10\mathrm{r}$)
      iv. We have contact at point $\mathbf{P}_{contact}$ with normal $\mathbf{n}_{face}$ and penetration distance δ.
2. If there is no contact with the faces we do a check for contact with edges, which is computationally more expensive.
3. **Loop** over all **lifter edges**
   (a) Compute the vector $\mathbf{L_{PE}} = (\mathbf{P}_{com} - \mathbf{E}_{\mathrm{i}}^{0})$ that gives the shortest distance between the particle and lifter edge, where $\mathbf{E}_{\mathrm{i}}^{0}$ is a vertex on the lifter edge.
   (b) We now check if this vector is valid. If $(\mathbf{E}_{i}^{\mathrm{Dir}} \cdot \mathbf{L}_{\mathrm{PE}}) > 0$, where $\mathbf{E}_{i}^{\mathrm{Dir}}$ is the direction of the lifter edge, then
      i. We compute the contact point $\mathbf{P}_{contact} = (\mathbf{E}_{\mathrm{i}}^{0} + L_{PE}.\mathbf{E}_{i}^{\mathrm{Dir}})$.
      ii. We check if the distance $\mathrm{d} = \| \mathbf{P}_{com} - \mathbf{P}_{contact} \|$ between the point and the particle is less than the radius.
      iii. We check if the penetration is valid (max 10% of penetration) ($\delta = \mathrm{d} - \mathrm{r} < 0.10\mathrm{r}$).
      iv. We have contact at point $\mathbf{P}_{contact}$ with normal $\mathbf{n} = (\mathbf{P}_{com} - \mathbf{P}_{contact})/\mathrm{d}$ and penetration distance δ.

2.3. Charge motion in a two-dimension mill.

In this simulation we vary the size of the balls maintaining the same load level and compare the profiles between CPU and GPU at steady state which is typically reached after 3 revolutions. The mill diameter is 5.16m , with length set equal to ball diameter for simulating motion in 2D when using a 3D code. Thirty two rows of rectangular lifters with a height of 9.5 cm and width of 15 cm was used. Table 1 summarizes the model parameters used in the simultions in Section 2.3.

Table 1: Model Parameters used in simulation for a 2D mill.

Parameter	$K_n(N.m^{-1})$	ϵ	K_T	$\mu_{particle}$	μ_{Lifter}	$\triangle t$
GPU	4×10^5	0.45	-	0.70	0.30	1×10^{-5}
CPU	4×10^5	0.45	3×10^5	0.70	0.70	1×10^{-5}

Figures 5-7 depicts the charge profiles obtained with both codes. We see a good match between both codes with the position of the shoulder being slightly lower in the GPU simulations due to the different shear force model used.

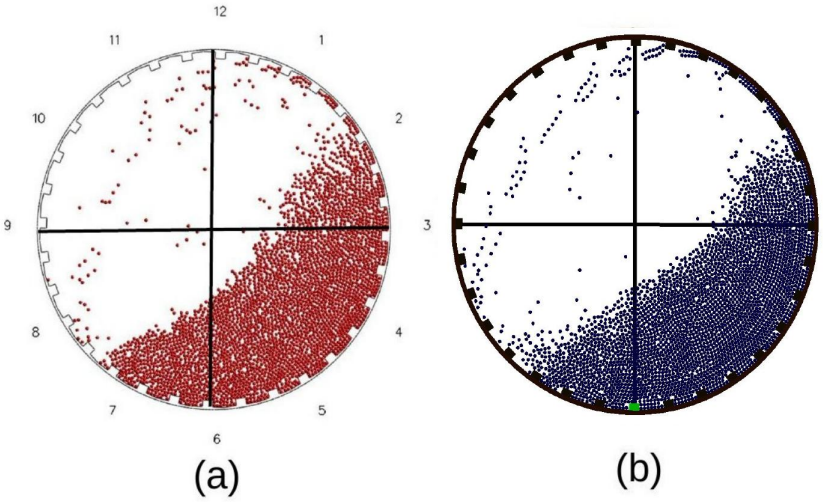

Figure 5: (a) CPU and (b) GPU charge profiles, N= 2916 (radius=2.5 cm).

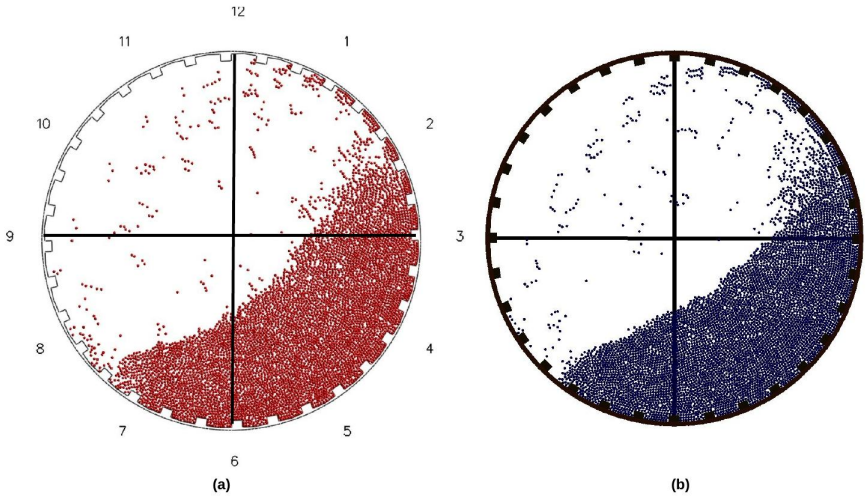

Figure 6: (a) CPU and (b) GPU charge profiles. N= 5344 (radius=1.85 cm).

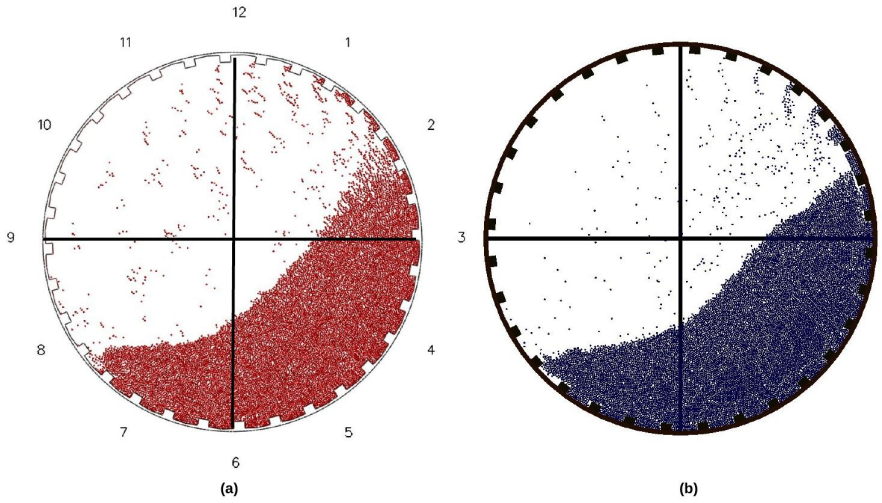

Figure 7: (a) CPU and (b) GPU charge profiles. N= 11664 (radius=1.25 cm).

2.4. Charge motion and power draw for a slice mill

In this simulation we use the experimental data as given by Moyes et al. [22]. To gauge the differences that exist between DEM and experiment we keep the spring parameters the same as we vary both the mill-speed and load. It is thus expected that the error with DEM will increase with super-critical mill speeds as the maximum penetration distance increases making the assumptions in Section 1.3 invalid. The mill diameter is 55cm, the is length of 2.35cm and is filled with steel balls having a radius of 2.2 cm. Twelve rows of 22 cm square lifters are used. Table 1 summarizes the model parameters used in the simultions in Section 2.4.

Table 2: Model Parameters used in simulation for slice mill.

Parameter	$K_n(N.m^{-1})$	$\epsilon_{particle}$	ϵ_{lifter}	K_T	$\mu_{particle}$	μ_{lifter}	$\triangle t$
GPU	4×10^4	0.90	0.80	-	0.14	0.39	2×10^{-5}
CPU	4×10^5	0.66	0.66	4×10^5	0.14	0.39	2×10^{-5}

Figure 8 shows the how the power varies as a function of rotation speed. We see a good match for sub-critical speeds which is the normal operation mode of a mill between both DEM codes and experiment. At super-critical speeds there is a slight difference due to the time-step being too large for the penetration distances as a result of the increased speed making the DEM assumptions "less valid".

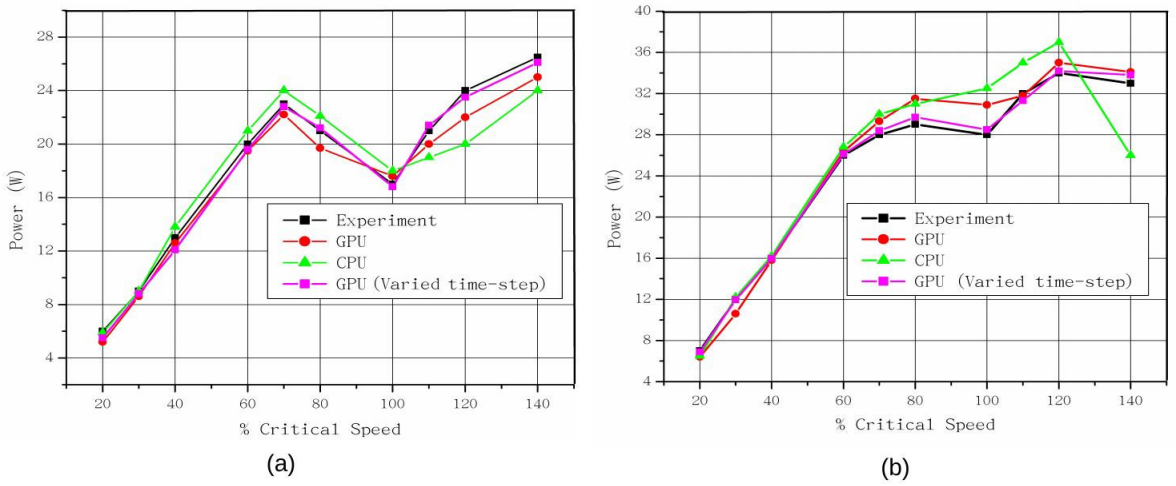

Figure 8: Power draw for (a) 25% and (b) 35% loading between experiment [22], GPU and CPU simulations.

Figures 9 and 10 depicts the charge profile for sub-critical speeds. We see a good match which is expected as the power values are similar. Note that it is difficult to do an accurate frame matching.

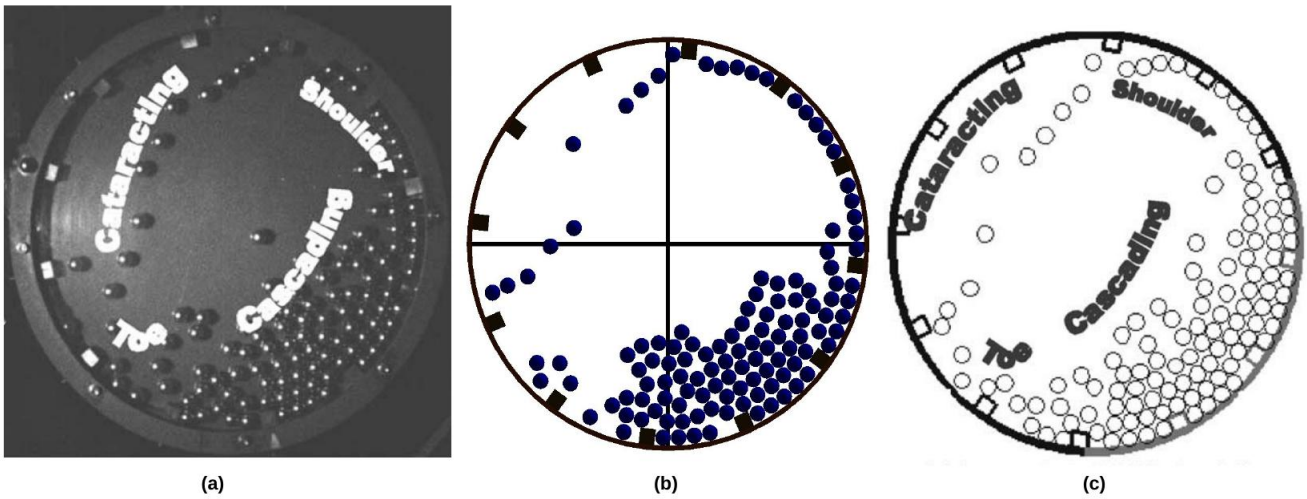

Figure 9: (a) Experiment [22](b) GPU and (c) CPU charge profiles. N= 120 (25% filling), rpm = 40.81 (70% critical speed).

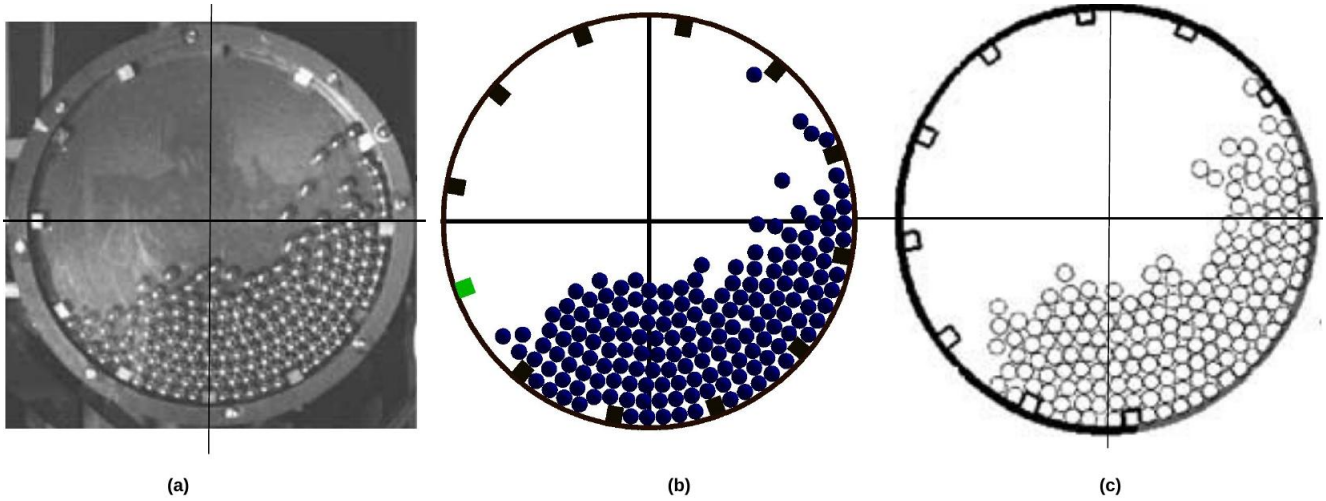

Figure 10: (a) Experiment [22] (b) GPU and (c) CPU charge profiles. N= 169 (35% filling), rpm = 17.50 (30% critical speed).

Figures 11 and 12 depict the charge profile for super-critical speeds. We note that DEM correctly predicts 1 and 2 layers of centrifuging particles for 100 and 160 percent of critical speed respectively. While experiment predicts zero power draw as all particles centrifuge, this is not possible in DEM as there will always be vibrations due to the spring model used. However the mills we are interested in simulating do not operate at these speeds. We have included these results to demonstrate the limitations of DEM.

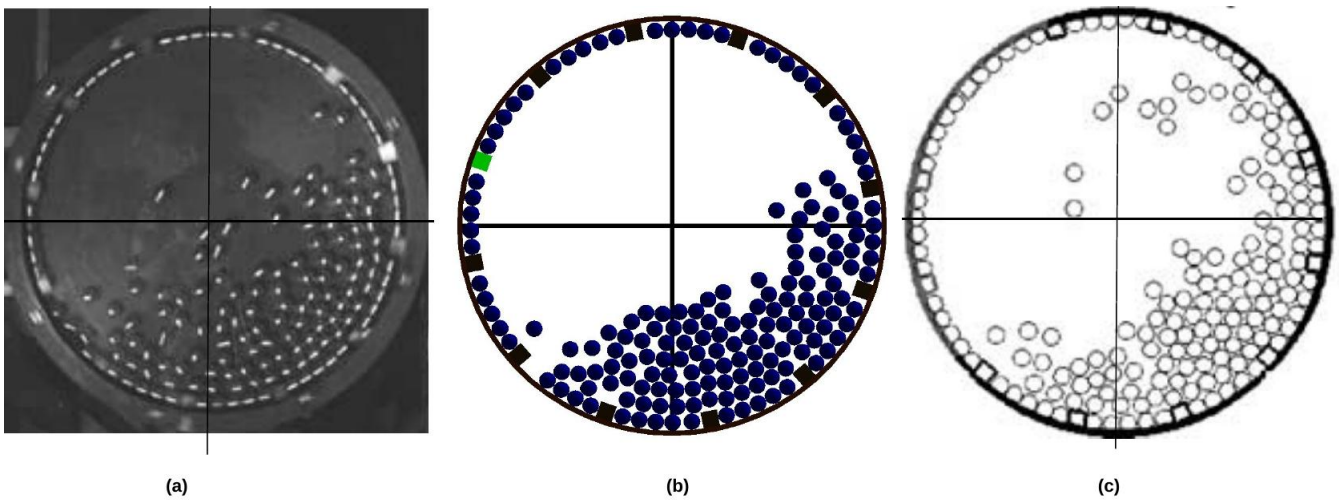

Figure 11: (a) Experiment [22] (b) GPU and (c) CPU charge profiles. N= 169 (35% filling), rpm = 58.30 (100% critical speed).

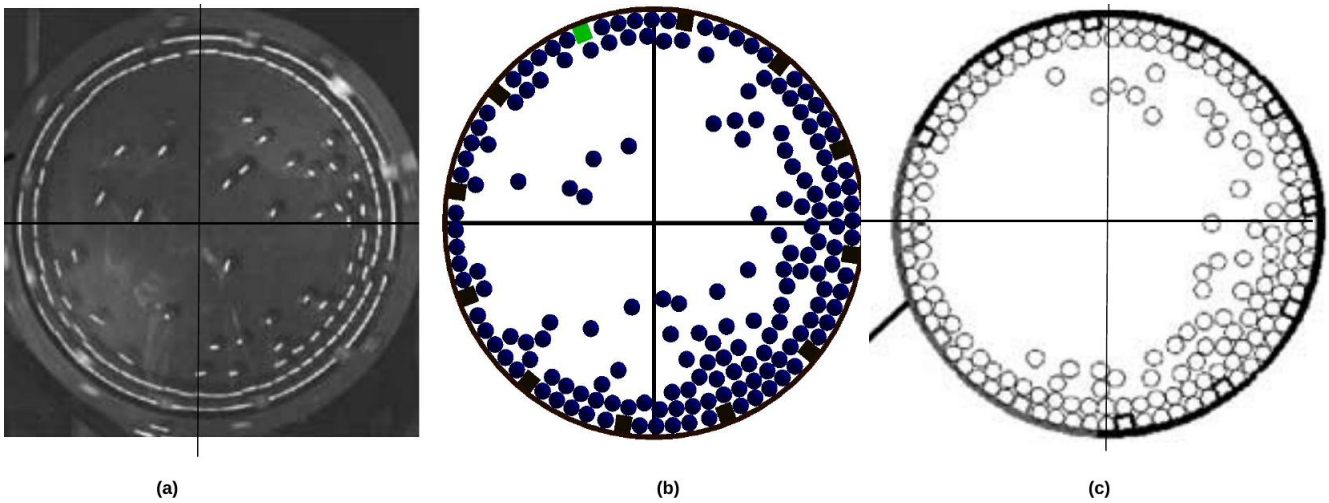

Figure 12: (a) Experiment [22] (b) GPU and (c) CPU charge profiles. N= 169 (35% filling), rpm = 93.30 (160% critical speed).

2.5. Charge motion and power draw in a three-dimensional mill

We use the experimental data obtained by Venagopal and Rajamani [12] using a 90 cm diameter by 15.0 cm length mill containing eight 4.0 cm square lifters. The face of the mill was made of Plexiglas^{TM} as to enable photographing of the tumbling charge. The mill was operated at 30%, 50% and 70% of critical speed for two levels of mill filling, 20% and 30% by volume respectively. The charge was photographed with a video camera. Representative snapshots of the GPU DEM predicted charge profiles alongside the still camera images for each of the experiments are shown in Figures 13 to 16 with the associated power draft in Table 4. Again to gauge the differences that exist between DEM and experiment we keep the spring parameters the same as we vary both the mill-speed and load. Charge profiles predicted by the GPU DEM code are consistent with observed charge profiles. The positions of the toe and shoulder of the charge is also reasonable with a slightly lower toe position and lower mass distribution which is expected due to the simpler friction model used. An exact match between charge trajectories is difficult to obtain due to simplifying assumptions made in the DEM model as well as the mechanical losses and the geometry of the mill not being exactly the same due to wear and manufacturing processes. Table 3 summarizes the model parameters used in the simultions in Section 2.5.

Table 3: Model Parameters used in simulation for a 3D mill.

Parameter	$K_n(N.m^{-1})$	$\epsilon_{particle}$	ϵ_{lifter}	K_T	$\mu_{particle}$	μ_{lifter}	$\triangle t$
GPU	4×10^5	0.65	0.65	-	0.20	0.20	1×10^{-5}

Table 4 shows the power draw obtained with experiment and our GPU code. We notice a good match with GPU DEM with a maximum difference of 12 % for the case of 30% filling.

Table 4: Power draw with experiment and GPU DEM for 3D mill.

RPM	Filling (20%)		Filling (30%)	
	Power(w)			
	Experiment	GPU DEM	Experiment	GPU DEM
14	301	288	393	345
22	459	456	617	605
32	532	518		

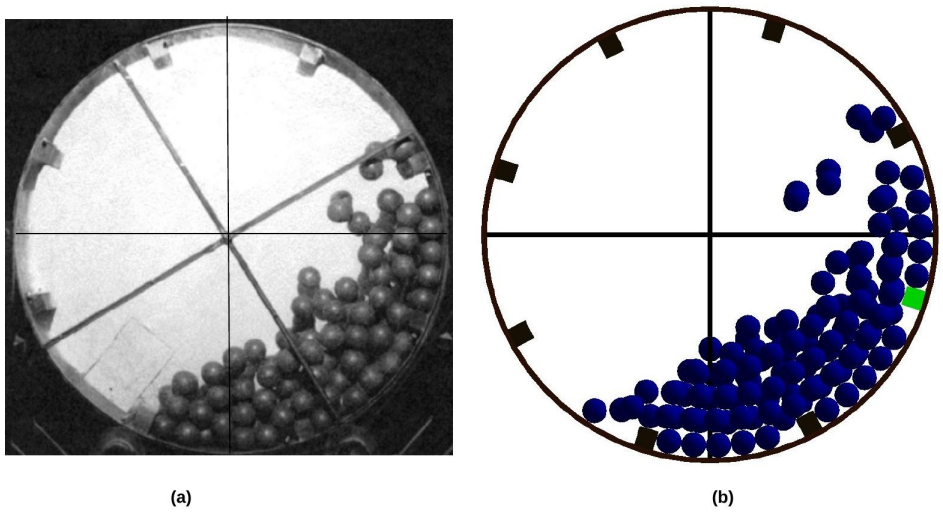

Figure 13: (a) Experiment (b) GPU charge profiles. N= 168 (20% filling), rpm = 14 (30% critical speed).

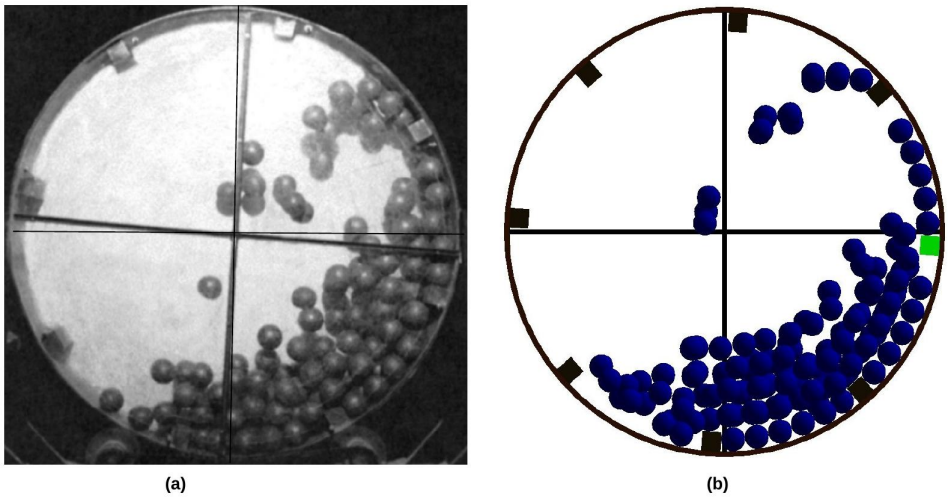

Figure 14: (a) Experiment [12] (b) GPU charge profiles. N= 168 (20% filling), rpm = 22 (50% critical speed).

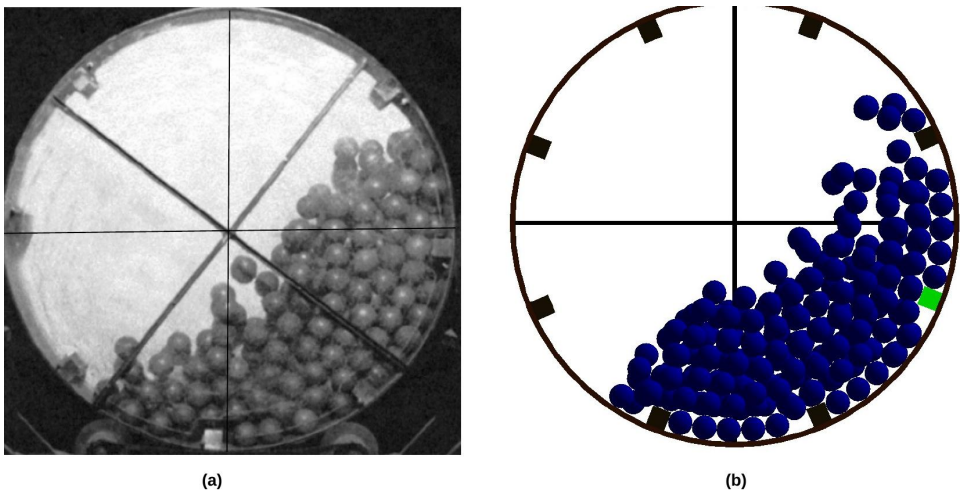

Figure 15: (a) Experiment [12] (b) GPU charge profiles. N= 243 (30% filling), rpm = 22 (30% critical speed).

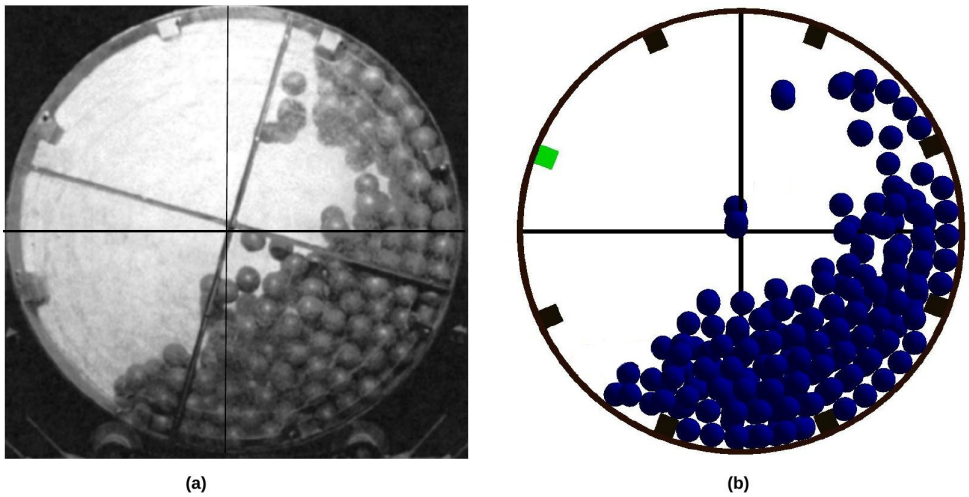

Figure 16: (a) Experiment [12] (b) GPU charge profiles. N= 243 (30% filling), rpm = 22 (50% critical speed).

3. Industrial Mill Simulation

The Los Bronces SAG mill is a 10.12m diameter by 4.7m long mill rotating at 10 rpm . The mill charge is made up of 31% ore and 10% balls by volume. The power draft reported for this mill (Malcolm et al. 2011) is 7.1 MW. Since, no further information about mill internals was available, typical values of operating parameters for this type of mill was used. It is presumed that the mill would be fitted with 64 rows of high low lifters as shown in Figure 17.

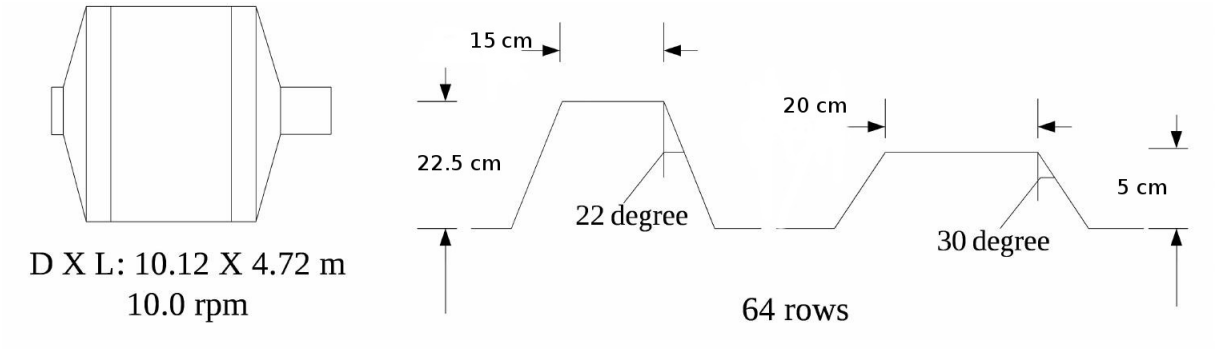

Figure 17: Lifter design Los Bronces semi autogenous mill.

Equilibrium ball size distribution with top ball size of 12.5 cm was used in the simulation. The ore in the mill charge was approximated as a Gaudin-Schuhmann distribution with slope 0.6 and top size of 15.0 cm. The combined weight distribution and number of ore and ball particles in the charge mix is shown in Table 5. Table 6 summarizes the model parameters used in the simultions in Section 3.

Table 5: Charge distribution for Los Bronces mill.

Diameter(cm)	Density(kg/m^3)	Weight (%)	Number
	ORE		
15.0	2850	9	6320
12.5	2850	9	9020
10.1	2850	30	81881
	BALLS		
12.4	7800	21	9800
10.0	7800	19	16171
8.7	7800	12	16220
TOTAL			**139392**

Table 6: Model Parameters used in simulation of Los Bronces mill.

Parameter	K_n(N.m^{-1})	$\epsilon_{particle}$	ϵ_{lifter}	K_T	$\mu_{particle}$	μ_{lifter}	$\triangle$t
GPU	1.5×10^6	0.75	0.75	-	0.40	0.70	2×10^{-5}

Figure 18 shows the charge profile in the mill. The charge profile is what one would anticipate, for such large filling of 41%. The charge shoulder is nearly at 2 o'clock and the toe is between 7 and 8 o'clock points on the mill circle. We also see radial segregation with the smallest particles moving to the mill shell as predicted by theory and experiment.

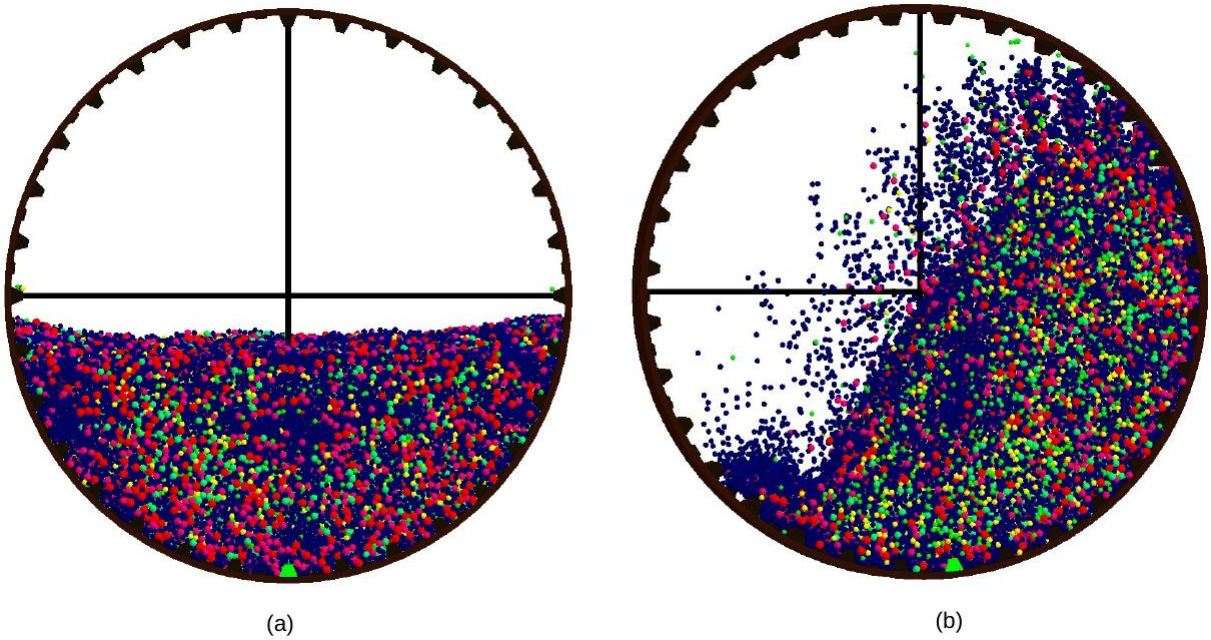

Figure 18: (a) Initial conditions N= 139392 (41% filling) (b) Charge profile of Los Bronces mill.

The computed power is stablizes from one revolution to the next as shown in Figure 19 (a). The large number of spheres in the simulation smooth out the energy consumed in collisions per revolution. Hence, accurate dependence of contact parameters on contact velocity is unnecessary here. The DEM value of Ø6.8 is slightly lower that the experimintal of 7.1MW as expected since mechanical losses from the drive shaft and permanent deformation is not taken into account. Figure 19 (b) shows the contribution to power draw by particle to particle collisions, particle to mill shell collisions and particle to lifter collisions. Around 68% of the power draw comes from the lifter since it lifts the majority of the load to the shoulder of the mill and hence excessive forces occur at the points of contact on the lifters. Likewise, around 18% is contributed by the mill shell as it supports the load between two lifters. Thus less than 14% is contributed by particle-particle collision.

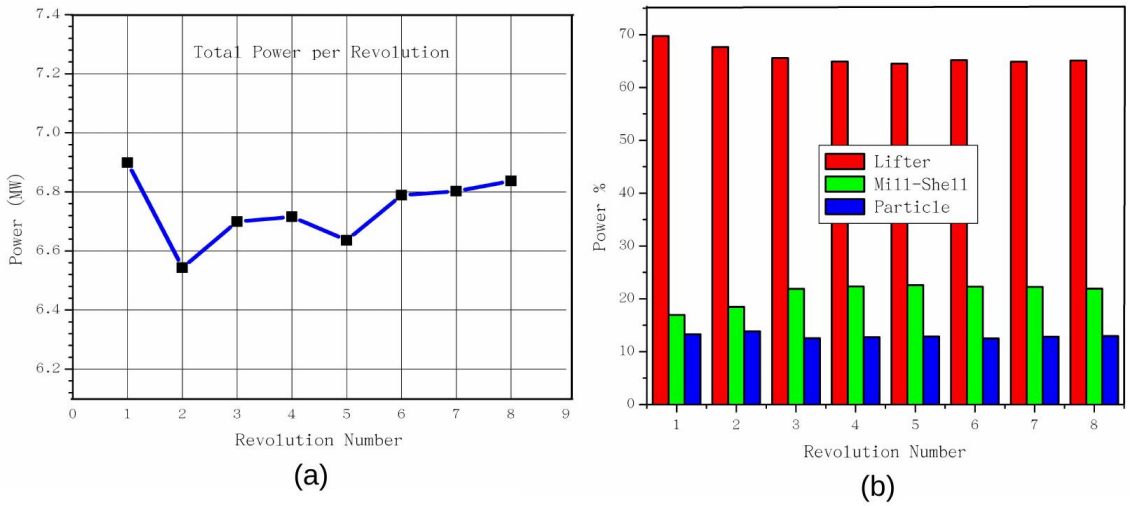

Figure 19: (a) Total power draw (b) Power distribution over time of Los Bronces mill.

3.1. Performance scaling

To gauge the performance of our code we increased the length of the mill to 2800 cm to accommodate four million mono-sized steel balls with a diameter of 6 cm. Figure 20 shows the charge profile.

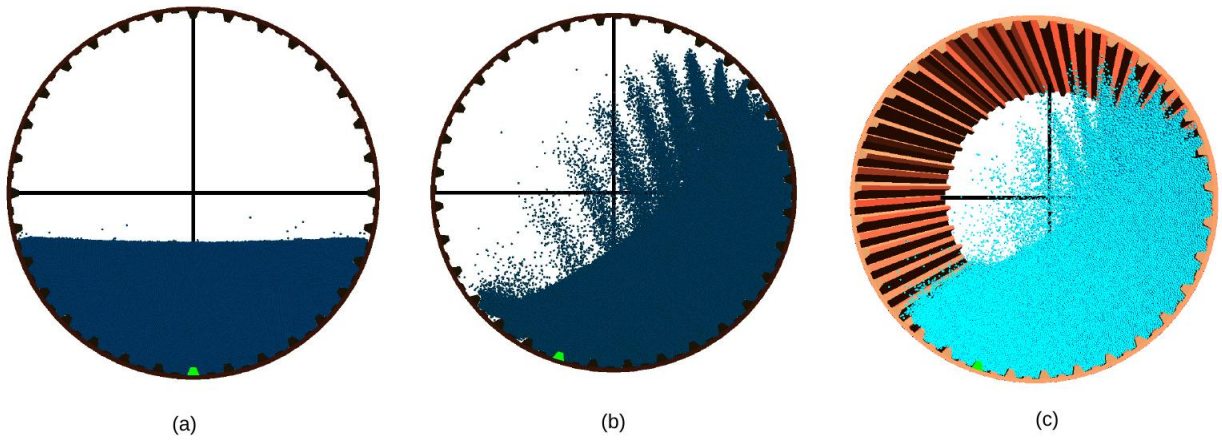

Figure 20: (a) Initial conditions N=4×10^6 (35% filling) (b) steady state profile (orthognal view) (c)steady state profile (isometric view) .

The power draft is distributed with 47 % to particle-particle collisions, 44 % to particle-lifter collisions and 9 % to particle mill-shell collisions. The shear number of particles simulated results in particle-particle collisions consuming the most amount of energy. The GPU compute time for one revolution (6 seconds) using a time step of 2×10^{-5} with a NVIDIA Kepler GPU is 7 hours (12 FPS). To show the benefits of a full 3D simulation we performed the same simulation in 2D with the same parameters and obtained a very different charge profile as depicted in Figure 21 (a). Figure 21 (b) depicts the charge profile for a slice of 10% length. We notice that the pattern is very similar to the full length of the mill. Clearly the effect of the axial direction cannot just be neglected as the 2D case cannot fully reproduce the dynamics.

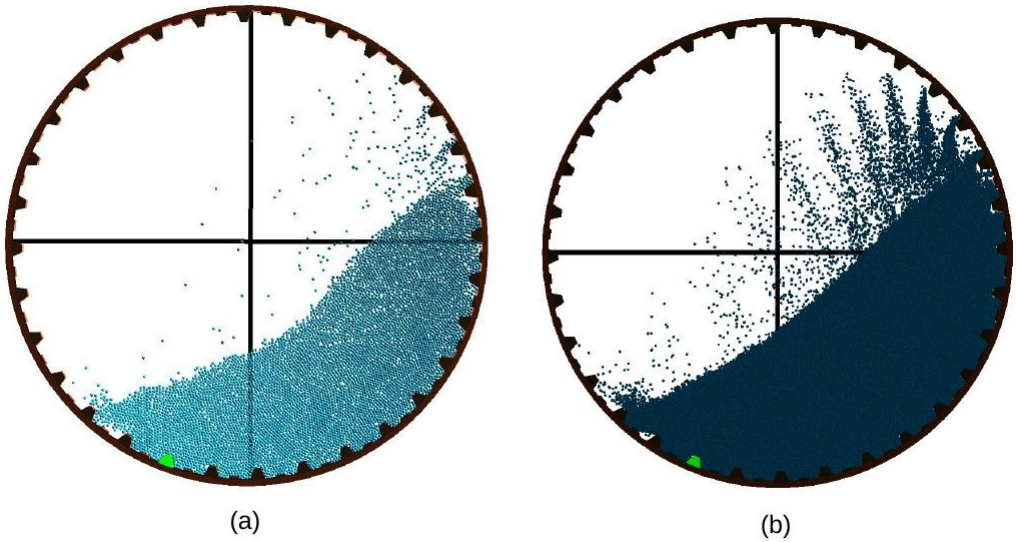

Figure 21: (a) 2D steady state profile, N=6744 (b) Steady state profile for a slice 10% of the length, N=385534.

Figure 22 shows the scaling of our code with increased particle number, we see a trend of linear scaling which is good indicating of the scalability of our code. The GPU compute time for one revolution (6 seconds) of a simulation consisting of 10 million particles using a time step of 2×10^{-5} with a NVIDIA Kepler GPU will take just 18.5 hours with a simulaton of 100 million particles taking just over a week.

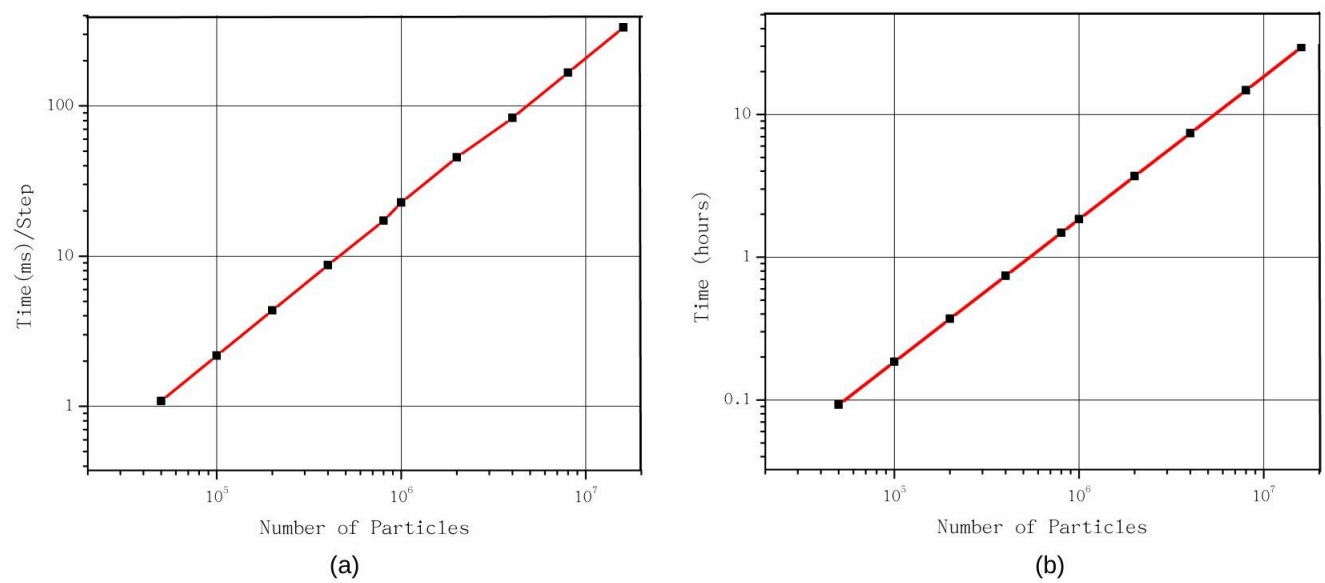

Figure 22: Scaling of GPU code with number of particles (N) .

4. Conclusions

In this paper we have presented a novel approach for modeling tumbling mills utilizing the GPU architecture. The modular approach of our code allows us to analyze the distribution of power amongst different collision types in the mill amongst other things which lends insight that may be exploited to improving the energy efficiency of mills. We achieve a new performance level in DEM modeling of mills by simulating 16 million particles at 3 FPS on a laptop GTX 880 GPU. Our code can handle 1 billion particles using the 12GB of memory availble on a NVIDIA K40 GPU. However such a large simulation should ideally be run with multiple GPUs and is currently under development.

5. Acknowledgments

The first author acknowledges the support of the university of Utah in providing research funds for his stay at the university to do this work. We gratefully acknowledge the support of NVIDIA Corporation with the donation of the GPU used for this research.

References

[1] B. Mishra, R. Rajamani, Numerical simulation of charge motion in a ball mill., 7th European Symposium on Comminution. 1 (1990) 555–563.

[2] M. Powell, The effect of liner design on the motion of the outer grinding elements in a rotary mill, International Journal of Mineral Processing 31 (1991) 163–193.

[3] M. Dennis, R. Rajamani, Evolution of the perfect simulator, International Autogenous and Semiautogenous Grinding Technology Proceedings 31 (2001) 163–193.

[4] P. W. Cleary, R. Morrison, Particle methods for modelling in mineral processing, International Journal of Computational Fluid Dynamics 23 (2009) 137–146.

[5] M. Hromnik, Masters Thesis: A GPGPU implementation of the discrete element method applied to modeling the dynamic particulate environment inside a tumbling mill, University of Cape Town, www.uct.ac.za, 2013.

[6] J. Longmore, P. Marais, M. Kuttel, Towards realistic and interactive sand simulation: A GPU-based framework, Powder Technology 235 (2013) 983–1000.

[7] T. Harada, GPU Gems 3: Real-time rigid body simulation on GPUs, Vol. 3, 2008.

[8] N. Govender, D. Wilke, S. Kok, Collision detection of convex polyhedra on the NVIDIA GPU architecture for the discrete element method, Journal of Applied Mathematics and Computation http://dx.doi.org/10.1016/j.amc.2014.10.013.

[9] NVIDIA, NVIDA KEPLER GK110 architecture whitepaper (2012).
URL `http://www.nvidia.com/NVIDIA KEPLER GK110 Architecture Whitepaper`

[10] J. Sanders, E. Kandrot, CUDA by example, Vol. 12, 2010.

[11] P. Cundall, Strack, A discrete numerical model for granular assemblies, Geotechnique 29 (1979) 47–65.

[12] R. Venugopal, R. Rajamani, 3d simulation of charge motion in tumbling mills by the discrete element method., Powder Technology 115 (2001) 157–166.

[13] B. K. Mishra, R. Rajamani, Three dimensional simulation of plant size SAG mills, International Conference on Autogenous and Semiautogenous Grinding Technology 31 (2001) 48–57.

[14] J. Herbst, L. Nordell, Optimization of the design of SAG mill internals using high fidelity simulation, International Conference on Autogenous and Semiautogenous Grinding Technology 31 (2001) 150–164.

[15] R. Morrison, P. W. Cleary, Towards a virtual comminution machine, Minerals Engineering 21 (2008) 770–781.

[16] J. Alatalo, K. Tano, Comparing experimental measurements of mill lifter deflections with 2D and 3D DEM predictions, DEM5 Proceedings, Queen Mary University, London, UK, 1 (2010) 194–198.

[17] J. Favier, EDEM (2014).
URL `http://www.dem-solutions.com/`

[18] N. Govender, D. Wilke, S. Kok, R. Els, Development of a convex polyhedral discrete element simulation framework for NVIDIA Kepler based GPUs, JCAM 270 (2014) 63–77.

[19] P. W. Cleary, Recent advances in DEM modelling of tumbling mills, Minerals Engineering 14 (2001) 1295–1319.

[20] N. Bell, Y. Yu, Particle-based simulation of granular materials, Eurographics/ACM SIGGRAPH Symposium on Computer Animation 25 (2005) 29–31.

[21] G. Neubauer, C. A. Radek., GPU Based Particle Simulation Framework With Fluid Coupling Ability, NVIDIA GTC 2014, San Jose,USA, 2014.

[22] O. Hlungwani, J. Rikhotso, H. Dong, M. Moys, Further validation of DEM modeling of milling: effects of liner profile and mill speed, Minerals Engineering 16 (2003) 993–998.