processes

MDPI

Article

Chrono::GPU: An Open-Source Simulation Package for Granular Dynamics Using the Discrete Element Method

Luning Fang, Ruochun Zhang, Colin Vanden Heuvel, Radu Serban and Dan Negrut *

Department of Mechanical Engineering, University of Wisconsin-Madison, Madison, WI 53706, USA; lfang9@wisc.edu (L.F.); rzhang294@wisc.edu (R.Z.); colin.vandenheuvel@wisc.edu (C.V.H.); serban@wisc.edu (R.S.)

* Correspondence: negrut@wisc.edu

Abstract: We report on an open-source, publicly available C++ software module called Chrono::GPU, which uses the Discrete Element Method (DEM) to simulate large granular systems on Graphics Processing Unit (GPU) cards. The solver supports the integration of granular material with geometries defined by triangle meshes, as well as co-simulation with the multi-physics simulation engine Chrono. Chrono::GPU adopts a smooth contact formulation and implements various common contact force models, such as the Hertzian model for normal force and the Mindlin friction force model, which takes into account the history of tangential displacement, rolling frictional torques, and cohesion. We report on the code structure and highlight its use of mixed data types for reducing the memory footprint and increasing simulation speed. We discuss several validation tests (wave propagation, rotating drum, direct shear test, crater test) that compare the simulation results against experimental data or results reported in the literature. In another benchmark test, we demonstrate linear scaling with a problem size up to the GPU memory capacity; specifically, for systems with 130 million DEM elements. The simulation infrastructure is demonstrated in conjunction with simulations of the NASA Curiosity rover, which is currently active on Mars.

Keywords: granular material; Discrete Element Method; physics-based simulation; GPU computing

check for updates

Citation: Fang, L.; Zhang, R.; Vanden Heuvel, C.; Serban, R.; Negrut, D. Chrono::GPU: An Open-Source Simulation Package for Granular Dynamics Using the Discrete Element Method. *Processes* **2021**, *9*, 1813. https://doi.org/10.3390/pr9101813

Academic Editor: Joanna Wiącek

Received: 7 September 2021
Accepted: 8 October 2021
Published: 13 October 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction: State of the Art

The Discrete Element Method (DEM) [1] is a widely-adopted scheme for predicting the dynamics of large granular systems [2], from mixing [3] and particulate flows [4] to landslides [5,6] and astrophysical processes [7]. Due to the small step size necessary for numerical stability and the large number of particles that might be required to capture the physics of interest, DEM can be computationally expensive. Until very recently, Central Processing Unit (CPU)-only parallel computing techniques have been implemented to accelerate large-scale DEM simulations. These approaches drew on (1) OpenMP for single multiprocessors with shared memory architectures [8]; (2) the Message Passing Interface (MPI) standard for clusters with distributed memory [9]; and (3) hybrid MPI–OpenMP parallelism [8,10].

An alternative architecture for parallel computing is provided by the Graphics Processing Unit (GPU). Over the last decade, owing to its high bandwidth and fast global memory, the GPU has anchored the intense arithmetic computations demanded, for instance, by artificial intelligence applications, linear algebra, and molecular dynamics simulation. Several researchers have since established DEM codes leveraging the GPU architectures [11,12]. Nonetheless, whether using CPU or GPU computing, the number of DEM elements used in experiments reported in the literature has been rather small—in the vicinity of 10^3 to 10^5 elements [13–27]; for comparison, in one cubic meter of sand, there are on the order of two billion elements. However, one should look beyond the maximum number of elements handled to assess a DEM simulator; the element size, normal contact stiffness, and element density also come into play. In this context, the traits of a difficult DEM problem include a

large number of elements, a small element size, high stiffness, and low density. For instance, a DEM code will have a significantly easier task simulating the dynamics of 1 million DEM elements, each of radius 100 mm, than when the radius is 100 µm. This is because, despite a change of units, a large value of the contact stiffness leads to deformations that are difficult to track accurately in double precision when the particle size is very small; also, for handling small particles and their deformations, the time steps will have to drop substantially. Thus, the review of the literature below, which is organized chronologically, pays attention to more than simply the number of elements and run times; it also reports the hardware configurations used and, whenever possible, other metrics that speak to the prowess of the DEM simulator used.

Our own DEM simulations consider up to 123 million elements on one GPU card; the Young's modulus was 213×10^6 Pa, the density was 2800 kg/m^3, and the size of the spherical particles was 0.64 mm. Our simulator, called Chrono::GPU, currently handles only monodisperse systems. The review of the literature below places the performance of Chrono::GPU in a broader context.

In [28], benchmark DEM simulations were run for 2D systems with 0.2 million particles using a 64-node supercomputer. The integration step size was of the order 1×10^{-4} s, and one second of simulation required approximately 200 s of run time using 64 nodes. In [29], a mixing via tumbling mill of one million particles required one week to simulate a 1.5 s rotation of the mixer on a cluster with 32 processors. The particles were spheres of radius 8 mm, Young's modulus 2.16×10^6 Pa, and density 2500 kg/m^3. In [30], 150,000 glass beads were considered in the "high fill" scenario for a spheronizer simulation. The Young's modulus was reduced to 4.87×10^6 Pa, the Poisson ration was 0.2, the density was 1500 kg/m^3, and the bidisperse mixture had particles of 2 mm and 4 mm radii. Ten seconds of simulation required 375 hours of run time on a Dell SC1425 cluster with 16 Intel Xeon (3.6 GHz) processors. In [31], the problem size for a hopper simulated via DEM went up to 400,000 bodies. The simulation drew on MPI [32] and a cluster with 36 processors. The monodisperse material had spheres of 1 mm radius; no material information was provided regarding the stiffness of the spheres or the integration step size. In [33], an 81,000 DEM element simulation took 35 days on an MPI-managed 32-core architecture to simulate 120 s of system dynamics. The granular material was polydisperse with radii of 1.5 mm for 15% of material, 2 mm for 35% of material, 2.5 mm for 35% of material, and 3 mm for 15% of material. The Young's modulus was relaxed from 68.9×10^9 Pa to 200×10^6 Pa for the sake of increasing the simulation time step, which was $\Delta t = 1.2 \times 10^{-6}$ s. In [34], the authors switched to a multi-GPU solution for systems with more than one million particles owing to GPU memory exhaustion; a settling simulation with 10 million bodies was run with up to 32 MPI-managed GPUs. The one million body simulation in [34] was carried out on one GPU using a monodisperse system with spheres of radius 2.5 mm, Young's modulus 10^7 Pa, and density 733 kg/m^3. No information was provided regarding the amount of time required to complete a simulation. However, a speed-up of 40 was reported relative to a CPU implementation discussed in [35]. In the latter, the number of bodies was 130,000, with a particle radius of 2.5 mm, Young's modulus of 10^8 Pa, and density of 2500 kg/m^3. Results pertaining to a powder compaction simulation are reported in [36]. Therein, the authors used mixtures of spheres with up to three different radii, with the smallest being 0.1 mm. The Young's modulus and particle density were 9×10^9 Pa and 5170 kg/m^3, respectively; the simulation time steps used were in the range of 10^{-8} s to 10^{-7} s. Strong scaling analysis results were presented for the dynamics of one million bodies using from 4 to 20 GPUs. A compaction analysis was carried out using particles of radius 0.1 mm stored in a 3D cell of size 1 cm. This led to simulations of systems with up to 0.563 million elements. In [37], the authors ran powder simulations. To reduce the number of particles and the computational time, the particle diameter was scaled from 0.01–0.25 mm to 0.4–0.7 mm. The simulation time step was 2×10^{-6} s, and the number of particles in the simulation was 4.03 million. For a 2.8 million particle system run on an Nvidia GTX 1080Ti GPU, the simulation took roughly 0.03 s per time step; model simplifications were made to

shorten the computation, which would have otherwise required 32 days to complete. In [12], a DEM solver running on the GPU was stated to handle systems with 60 million elements. Significant simplifications were made to reach this particle count; e.g., the arithmetic was carried out in single precision, which is known to be inadequate in almost all DEM simulations. The normal force model could only be Hookean; moreover, the friction force model did not keep track of history, which is known to lead to inaccurate results, see, for instance [38]. A simplified, no-history friction force model was also used in [39], where the implementation was reported to handle approximately one million spheres clumped in sets of four to form 0.256 million aggregate bodies of a more complex shape.

Switching to terradynamics applications, the number of elements in DEM simulations is typically smaller. In [40], a commercial tool was used to simulate a low number of particles. Two cases reported therein used particles that had 5 mm and 10 mm nominal radii (actual radii were 0.95 to 1.05 times the nominal size). The density of the sand particles was 2600 kg/m^3, and the shear modulus was 43×10^9 Pa. Spherical particles were used in [41] to compare DEM and FEM in relation to the soil–tool interaction. The commercial tool used in [40] was also used in [41]. Owing to licensing constraints, the number of particles used in DEM increased to 250,000 in a polydisperse setup with particles of 1.5 mm, 3 mm, 6 mm, and 12 mm. The density of the sand particles was 2600 kg/m^3, and the shear modulus was 5×10^7 Pa. The hardware used was a commodity, relatively high-clocked CPU processor with four cores. In [42], a set of four deformable wheels interacted with a granular terrain made up of 0.9 million elements. The granular material was made up of spheres with radius 12 mm, density 2500 kg/m^3, and contact force stiffness $k_n = 10^6$ N/m. The simulation proceeded at a time step of $\Delta t = 3.5 \times 10^{-5}$ s, and the completion of the 7.64 s long simulation of the vehicle operating on granular terrain required 5.5 days of computation time. A similar problem was analyzed in [43], where the terrain was made up of 90,304 spheres of radii between 6 mm and 7 mm. The Young's modulus was 75×10^9 Pa, the density was 2600 kg/m^3, and the integration time-step used was $\Delta t = 3.5 \times 10^{-5}$ s. The settling of the particles reportedly took 46 hours to simulate, while the rolling of one tire for approximately 5 s took 52 h. In [44], the DEM particle was the union of three spheres of identical radii. The monodispersed tri-sphere particles could be circumscribed with a sphere with a radius of about 4 mm. A wheel–terrain interaction problem was simulated using 392,049 such particles; the bulk shear stress was 50×10^6 Pa and density was 2875 kg/m^3. No simulation times were reported.

To the best of our knowledge, the largest granular dynamics simulation of practical relevance to date contained 2.4 billion elements. It was run on 16,384 CPUs (131,072 cores) of Japan's K-supercomputer [45–47]—the fastest supercomputer in the world in 2012 [48]. For reference, in [46], the problem size increased to 1.9 billion particles of radius 0.114 mm, density 2600 kg/m^3 and Young's modulus 10^7 Pa. The simulation was run on 4096 CPUs (32,768 cores); no simulation times were provided.

The rest of the manuscript is organized as follows. Section 2 summarizes the DEM model (equations of motion and contact force model). Section 3 highlights software implementation aspects; e.g., mixed-data types, domain decomposition, support for co-simulation, and checkpointing. Section 4 focuses on validation. Section 5 reports on a scaling analysis. Section 6 demonstrates the DEM solver via a simulation of a rover operating on granular terrain. We close with conclusions and directions of future work.

2. The DEM Model

In DEM, the fundamental assumption is that particles are allowed to have small penetration, δ_n, at the point of contact, contrary to the non-smooth contact method [49],

where no penetration is allowed. The dynamics of the particles are explicitly updated by summing the applied forces, torques and gravity; see Equation (1),

$$m_i \frac{d\boldsymbol{v}_i}{dt} = m_i \boldsymbol{g} + \sum_{j=1}^{nc} (\boldsymbol{F}_n + \boldsymbol{F}_t + \boldsymbol{F}_c), \tag{1}$$

$$I_i \frac{d\boldsymbol{\omega}_i}{dt} = \sum_{j=1}^{nc} (\boldsymbol{r}_i \times \boldsymbol{F}_t + \boldsymbol{M}_r). \tag{2}$$

When two bodies, i and j, are in contact, as shown in Figure 1, by means of various contact laws, such as Hertzian contact theory, the Mindlin friction law [50] and rolling resistance models [51], the interaction between particles, namely the normal force $\boldsymbol{F}_n$, cohesive force $\boldsymbol{F}_c$, tangential friction force $\boldsymbol{F}_t$, and rolling friction torque $\boldsymbol{M}_r$, can be evaluated based on material properties and local deformations. Specifically, normal and tangential forces can be described as spring-dashpot mechanisms, with the tangential force, $\boldsymbol{F}_t$, being capped to satisfy the Coulomb condition through the friction coefficient μ_s:

$$\boldsymbol{F}_n = k_n \boldsymbol{u}_n - \gamma_n \boldsymbol{v}_n, \tag{3}$$

$$\boldsymbol{F}_t = k_t \boldsymbol{u}_t + \gamma_t \boldsymbol{v}_t, \quad \|\boldsymbol{F}_t\| \leq \mu_s \|\boldsymbol{F}_n\|. \tag{4}$$

Here, the stiffness and damping coefficients k_n, k_t, γ_n, and γ_t can either be user-defined [52] or dependent on material properties: Young's modulus, Poisson ratio, and coefficient of restitution [38]. Both flavors use the Hertzian contact force model [53]. Local deformation, namely normal penetration $\boldsymbol{u}_n = \delta_n \boldsymbol{n}$, tangential displacement history $\boldsymbol{u}_t$, and relative velocity at the contact point $\boldsymbol{v}_{rel} = \boldsymbol{v}_n + \boldsymbol{v}_t$, are evaluated as

$$\boldsymbol{v}_{rel} = \boldsymbol{v}_j + \boldsymbol{\omega}_j \times \boldsymbol{r}_j - \boldsymbol{v}_i - \boldsymbol{\omega}_i \times \boldsymbol{r}_i, \tag{5}$$

$$\boldsymbol{v}_n = (\boldsymbol{v}_{rel} \cdot \boldsymbol{n})\boldsymbol{n}, \tag{6}$$

$$\boldsymbol{v}_t = \boldsymbol{v}_{rel} - \boldsymbol{v}_n, \tag{7}$$

where $\boldsymbol{v}_i$, $\boldsymbol{\omega}_i$ and $\boldsymbol{v}_j$, $\boldsymbol{\omega}_j$ are the velocity at the center of mass and the angular velocity of bodies i and j, respectively. Position vectors $\boldsymbol{r}_i$ and $\boldsymbol{r}_j$ point from the center of mass of bodies i and j to the contact point. In [50], the authors state that the friction force $\boldsymbol{F}_t$ is dependent on the loading history and should be updated incrementally to reproduce results from physical tests [38]. Thus, the tangential displacement $\boldsymbol{u}_t$ is accumulated throughout the duration of contact. To enforce the Coulomb friction law, at any step where $\|\boldsymbol{F}_t\| > \mu_s \|\boldsymbol{F}_n\|$, the tangential displacement $\boldsymbol{u}_t$ is updated using

$$\boldsymbol{u}_t = \frac{\mu_s \|\boldsymbol{F}_n\|}{k_t} \frac{\boldsymbol{u}_t}{\|\boldsymbol{u}_t\|}. \tag{8}$$

The cohesion model adds a constant attractive force along the contact normal direction,

$$\boldsymbol{F}_c = -C\boldsymbol{n}. \tag{9}$$

When a particle rolls over ground or another particle, rolling resistance can arise from an asymmetric normal stress profile at the contact patch [54]. Currently, Chrono::GPU uses a constant torque model where the magnitude of $\boldsymbol{M}_r$ is proportional to the rolling friction coefficient μ_r, the magnitude of normal force $\boldsymbol{F}_n$, and the particle radius r_i, while its direction opposes the relative rotational velocity $\boldsymbol{v}_{rot}$:

$$\boldsymbol{M}_{r,i} = \mu_r r_i \frac{\boldsymbol{F}_n \times \boldsymbol{v}_{rot}}{\|\boldsymbol{v}_{rot}\|}, \tag{10}$$

$$\boldsymbol{v}_{rot} = r_j(\boldsymbol{\omega}_j \times \boldsymbol{n}) - r_i(\boldsymbol{\omega}_i \times \boldsymbol{n}). \tag{11}$$

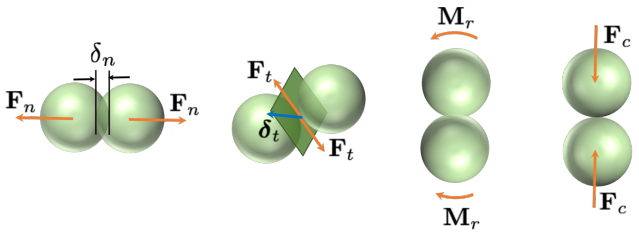


Figure 1. Contact forces that describe particle–particle interaction: from left to right, normal force $\boldsymbol{F}_n$, tangential friction force $\boldsymbol{F}_t$, rolling friction torque $\boldsymbol{M}_r$, and cohesion $\boldsymbol{F}_c$.

3. Aspects Related to the Design of the Software

3.1. Mixed Data Types

On modern hardware architectures, the speed of 32-bit floating point operations such as addition and multiplication is at least twice as fast as the performance of 64-bit operations [55]. Moreover, the memory required to store a value in 32 bits will be half of that required to store it in 64 bits. This is also relevant for cache performance—the typical cache line on a CPU is 64 bytes wide, which means that more variables will be found in cache if they store their values using 32 bits instead 64 bits. Finally, upon a cache miss, moving data from memory to cache is also more effective if the variables that are accessed store their values using 32 instead of 64 bits. Therefore, 32 bit data types such as `int` and `float` are preferred over 64 bit types such as `long int` and `double`. However, this raises the question of whether or not single precision ($\epsilon_{float} \approx 10^{-7}$ vs. $\epsilon_{double} = 10^{-16}$) is enough for DEM problems. Indeed, DEM particles are modeled with large contact stiffness, which can lead to small deformations in normal and tangential directions. Take one sphere of radius $R = 1\,\text{mm}$ resting on the ground as an example, with normal stiffness $k_n = 1 \times 10^8\,\text{N/m}$, density $\rho = 2.5 \times 10^3\,\text{kg/m}^3$ and gravity $g = 9.8\,\text{m/s}^2$; using the Hertzian contact model, the normal deformation due to gravity is about $\delta_n = 1.02 \times 10^{-8}\,\text{m}$, which is too small to be captured in single precision.

There is one more salient point in relation to using floating point numbers: they are "designed" to capture all numbers on the real axis. If one uses 32 bits in floating point to capture real numbers, there are only 2^{32} machine numbers used as proxies for an infinite number of numbers on the real axis. However, why should one care in DEM about values such as 8.5^{47}? All the physics of interest takes place in a relatively well defined range of values: in SI units, the speeds are between zero and several hundreds, and the particles are located in space in a range from -100 to 100, for example, etc. Therefore, there are bits out of the 32 bit budget for a floating point number that go unused since the DEM physics does not consider them. In our implementation, we decided to use our 32 bit budget differently. Indeed, we use variables of type `int` to specify the position of the particles in the 3D space, with three `int` values for the x, y, and z coordinates of a DEM element. In other words, we scale position quantities to be covered by the whole range of `int`, (−2,147,483,647, 214,7483,647), essentially slicing the domain in each direction into as many pieces as a single `int` can represent. The minimum length unit (space resolution) is defined as

$$l_{unit} = \frac{\max(B_x, B_y, B_z)}{N_{int} - 1}, \tag{12}$$

where $\{Bx, By, Bz\}$ is the size of simulation domain in each direction, and $N_{int} = 2^{31}$. For a soil bin of length 1 m, l_{unit} is about 4.66×10^{-10}—less than the order of normal penetration, and therefore sufficient to account for micro-deformation.

Another advantage of using `int` for position is that, for the same data size, integer operations are faster than floating point arithmetic, thus reducing the computation cost of position-related calculations, such as broadphase contact detection (Section 3.2).

The rule of thumb is that we only use the 8 byte `double` data type when necessary and with variables stored in very fast memory (registers) and use 4 byte or less data types

for variables stored in slow memory. Table 1 lists the different data types encountered in Chrono::GPU and their memory location.

Table 1. Various data types in Chrono::GPU and their memory location.

Data Type	Variable	Memory Type
`unsigned int`	Subdomain owner index	Global
`int`	Local coordinate within subdomain	Global
`float`	Kinematics quantity, friction history, mass, etc	Global
`double`	Penetration	Register
`float`	Contact force calculation	Register

3.2. Domain Decomposition and Local Coordinates

Chrono::GPU uses a Linked-Cell method [56], which is commonly relied upon for the contact detection of monodisperse elements. The entire domain is decomposed into small subdomains, and each particle is assigned to a subdomain (SD) based on the location of its center of mass. The size of an SD is defined relative to the size of the DEM element; currently, the size of the SD is 3.5 times the diameter of the element (this value can be adjusted by the user)—a decision dictated by the size of the CUDA thread block and the amount of shared memory available on the targeted GPUs [57]. As a rule of thumb, for packed granular material, we typically see between 60–100 DEM elements touching an SD. When searching for potential contacts, only the SDs that touch the particle of interest can contain potential contacts. Note that each particle can touch at most 8 SDs.

Apart from contact detection, the SDs are also used to represent the particle position, P_{global}, indirectly as a combination of the local position with respect to the origin of the owner SD, P_{loc}, and the location of SD origin, P_{SD}; see Figure 2 for illustration. The costs of storing particle positions are 4 bytes for the index of the owner SD (unsigned int) and $4 \times 3 = 12$ bytes for local coordinates (int), for a total of 16 bytes. This is less than $3 \times 8 = 24$ bytes if 64 bit data types (long int or double) are used. Therefore, positions are stored efficiently in the global memory, significantly reducing the time required to move data. During contact detection, the SD owner index and local positions are brought from the device global memory and the absolute positions are computed. For each contact pair, the absolute global positions are cast as doubles to ensure accuracy when computing penetration. These operations access fast local memory instead of global memory, improving memory bandwidth and cache utilization.

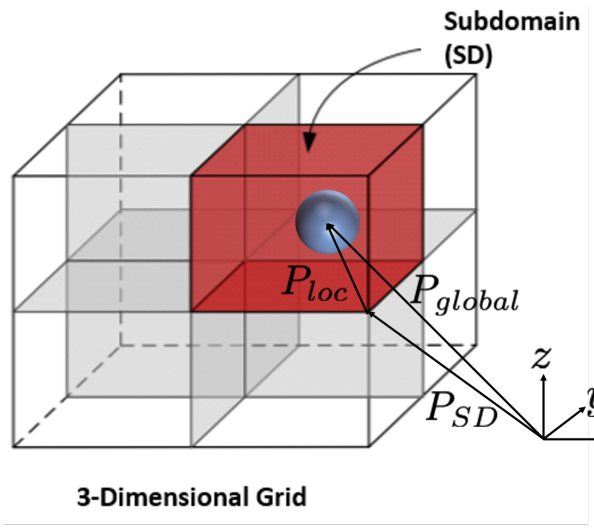


Figure 2. Representing particle position indirectly through owner SD as $P_{global} = P_{loc} + P_{SD}$.

Additionally, we group DEM elements by their owner SDs, which maps well to the GPU architecture. Thus, the CUDA execution configuration used is such that one SD is associated with a CUDA block of threads, and depending on the phase of analysis, one thread in a block is associated with either a DEM element or a contact event. We draw heavily on the use of shared memory by bringing all elements from one SD into shared memory and repeatedly using the variables therein to compute the quantities of interest; e.g., penetration, normal force, friction force, etc.

3.3. Co-Simulation

Chrono::GPU is designed to interact with the Chrono simulation engine [58] through a co-simulation approach. Apart from supporting basic shapes such as planes and cones to enforce boundary conditions, the user can import meshes to represent the geometry of an implement—e.g., complex wheel or track shoe—interacting with the granular material. Chrono::GPU decomposes imported meshes into a collection of triangle facets to make use of GPU parallelization. Each of those facets interacts with the granular material individually, yet their combined effects on the mesh-represented objects are registered by Chrono::GPU. See Figure 3 for an illustration of one such force pair between a triangle facet and a particle. The tangential frictional force that the particle experiences is $\boldsymbol{F}_s$, and the torque applied on the particle is calculated as $\boldsymbol{\tau}_s = \boldsymbol{F}_s \times \boldsymbol{r}$, where $\boldsymbol{r}$ is the vector from the particle center to the point of contact. Likewise, the torque applied on the meshed object is calculated as $\boldsymbol{\tau}_m = \boldsymbol{F}_m \times \boldsymbol{R}$, where $\boldsymbol{F}_m$ is the tangential frictional force that the mesh experiences, and $\boldsymbol{R}$ is the vector from the user-specified center of the meshed object to the point of contact.

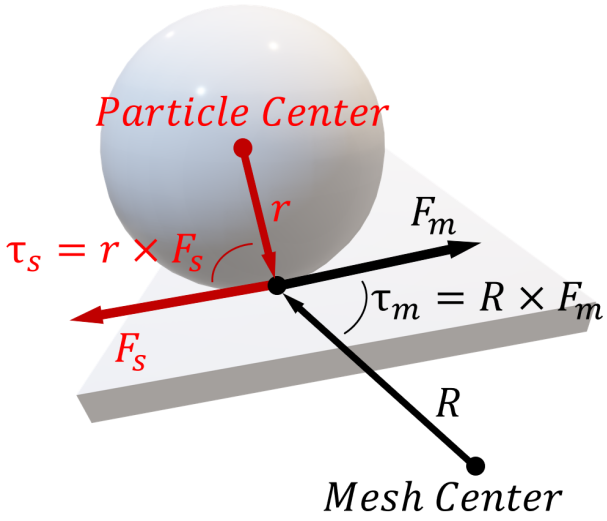


Figure 3. The tangential force F_s and F_m (normal forces are not shown in this figure) and torque τ_s and τ_m in a particle–triangle contact pair.

The force and torque exerted from a granular particle to the object can then be transferred to the Chrono core module via an external force accumulator API. The Chrono core then simulates the dynamics of the objects in question, updating their position and velocity information. These updates, in turn, are picked up by Chrono::GPU through mesh manager APIs and affect the granular physics in the next time step. The Chrono::GPU co-simulation workflow is shown in Figure 4. This workflow enables the external objects to have their own physics simulated rather than relying on prescribed motion. One co-simulation example can be found in Section 6.

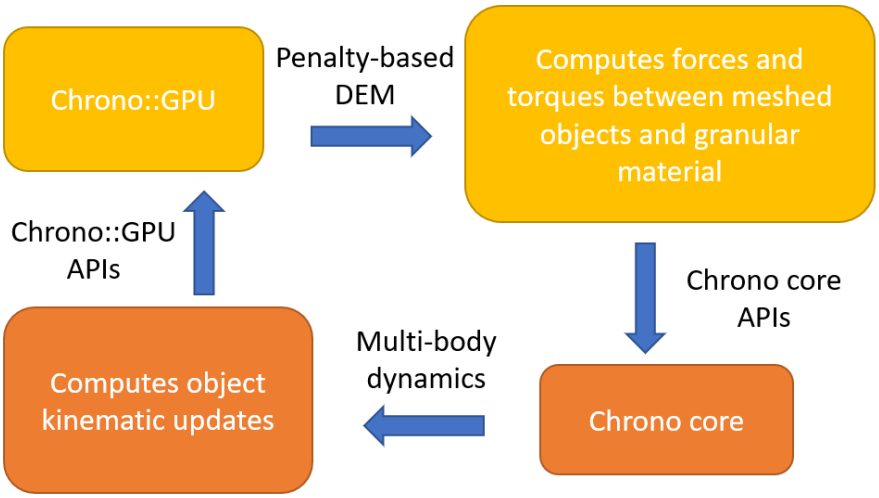


Figure 4. The co-simulation workflow in one time step.

3.4. Checkpointing

Chrono::GPU has full checkpointing support as it can output the current simulation state to a file and then restart the simulation at a later time by reading in this state file. Checkpointing is helpful in long DEM simulations. For instance, the rover dynamics simulation in Section 6 starts with a settling phase to create the material bed, which is checkpointed and subsequently reused across multiple runs. Note that the friction history is also stored in the checkpoint file, rendering this file relatively large. If losing the contact history is not a concern for the user, Chrono::GPU also supports restarting based on particle position and velocity only.

4. Validation Tests

To ensure the soundness of Chrono::GPU, various types of validation tests were carried out; for each type of test, parametric studies were also conducted to further check the robustness of the solver. In Section 4.1, small-scale tests were first carried out to validate the contact force models. Therein, we present the results of oblique impact (Section 4.1.1) and a three-sphere stacking test (Section 4.1.2) and how they compare to analytical and numerical solutions reported elsewhere. In Section 4.1.3, individual contact forces are investigated. Additional small-scale validation tests are available in [59]. During the second validation phase, typical benchmark tests were compared with numerical or experimental results: direct shear (Section 4.2.1), two types of cratering test (Sections 4.2.2 and 4.2.3), and rotating drum (Section 4.2.4). The numerical results for comparison were obtained using a different module, Chrono::Multicore [60]—an OpenMP-based implementation for granular dynamics handling both smooth and non-smooth contact [61]. Unless mentioned otherwise, all validation tests use the same experimental setup, contact force model, and simulation parameters as the corresponding reference, see Table 2 for details.

Table 2. Simulation parameters for large-scale tests: ρ—density, R—radius, E—Young's Modulus, ν—Poisson ratio, COR—Coefficient of Restitution, μ_s^{s2s}—sphere-to-sphere sliding friction coefficient, μ_s^{s2m}—sphere-to-mesh sliding friction coefficient, μ_s^{s2w}—sphere-to-wall sliding friction coefficient, μ_r—rolling friction coefficient, Δt—step size.

Parameters	Direct Shear	Cratering	Low-Velocity Cratering	Rotating Drum
ρ(kg/m^3)	2.55×10^3	2.5×10^3	2.48×10^3	2.5×10^3
R(cm)	0.3	0.1	0.5	0.0265
E (Pa)	4×10^7	7×10^7	7×10^8	7×10^7
ν	0.22	0.24	0.24	0.24
COR	0.87	0.9	0.9	0.97
μ_s^{s2s}	0.18	0.3	0.16	0.16
μ_s^{s2m}	0.4	[-]	[-]	[-]
μ_s^{s2w}	[-]	0.3	0.45	0.45
μ_r	0	0	0.09	0.09
Δt (s)	1×10^{-5}	5×10^{-6}	1×10^{-6}	1×10^{-6}

4.1. Small-Scale Tests

4.1.1. Oblique Impact

The contact force models are first validated by a set of oblique impact tests, where a sphere of initial velocity $v_i = v_{i,n} + v_{i,t}$ collides with the ground at different impact angles θ under zero gravity. The sphere bounces back with velocity $v'_i = v'_{i,n} + v'_{i,t}$ and angular velocity ω'_i. If the friction force F_t is saturated during the collision—i.e., $F_t = \mu_s F_n$—the kinematics after rebound can be derived analytically based on rigid body dynamics [62],

$$e_t = 1 - \frac{\mu_s(1+e)}{\tan\theta}, \tag{13}$$

$$\omega'_i = \frac{5}{2}\frac{\mu_s(1+e)v_{i,n}}{R}. \tag{14}$$

Here, $e_t = v'_{i,t}/v_{i,t}$ is the tangential coefficient of restitution (COR) and ω'_i is the angular velocity. The critical value of the impact angle for sliding regime to occur is derived in [63] as

$$\theta^\star = \arctan\left(\frac{7}{2}\mu_s(1+e)\right). \tag{15}$$

In Figure 5, both analytical and numerical tangential COR e_t are plotted for various impact angles θ.

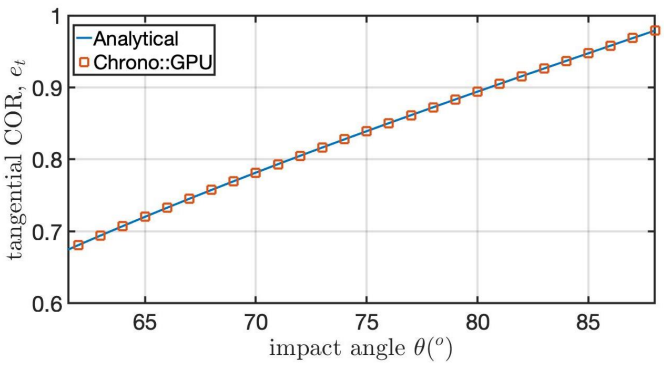


Figure 5. Analytical and numerical tangential COR e_t for impact angle θ, where $e = 1$, $\mu_s = 0.3$.

4.1.2. Sphere Stacking

To validate the rolling friction model and examine how a pyramid-like structure can withstand this, a set of small pyramid tests were carried out. For each test, two identical spheres of mass $m = 1$ kg and radius $R = 0.15$ m with a small gap in between were settled

on a flat surface, where the velocity of each sphere was less than 1×10^{-4} m/s. A third sphere of the same radius R but a different mass m_{top} was placed between and above the bottom spheres with zero initial velocity. To minimize the influence from impact, the third sphere was initialized in contact with the bottom ones. Depending on m_{top}, the gap, and rolling friction coefficient μ_r, two scenarios can occur: the top sphere drops to the ground, or it moves down slightly but the structure eventually stabilizes with the bottom spheres supporting the top sphere. This type of physics comes into play on a larger scale in angle of repose experiments. For each combination of sphere gap and μ_r listed in Table 3, the mass of the top sphere was increased by 0.01 kg to find the critical mass m^c_{top} for the pile to collapse, as demonstrated in Figure 6.

Table 3. Parameter space of sphere stacking test.

Parameter	Values
Rolling friction coefficient μ_r	$[0.07, 0.08, \ldots, 0.19, 0.2, 0.3, 0.4]$
Gap between bottom spheres	$[0.2R, 0.25R, 0.3R, 0.35R, 0.4R]$

A similar trend is observed for different gap values: m^c_{top} increases with μ_r up to a certain point, after which the critical mass decreases slightly until it reaches a plateau, indicating that increasing μ_r no longer influences the stability of the stack.

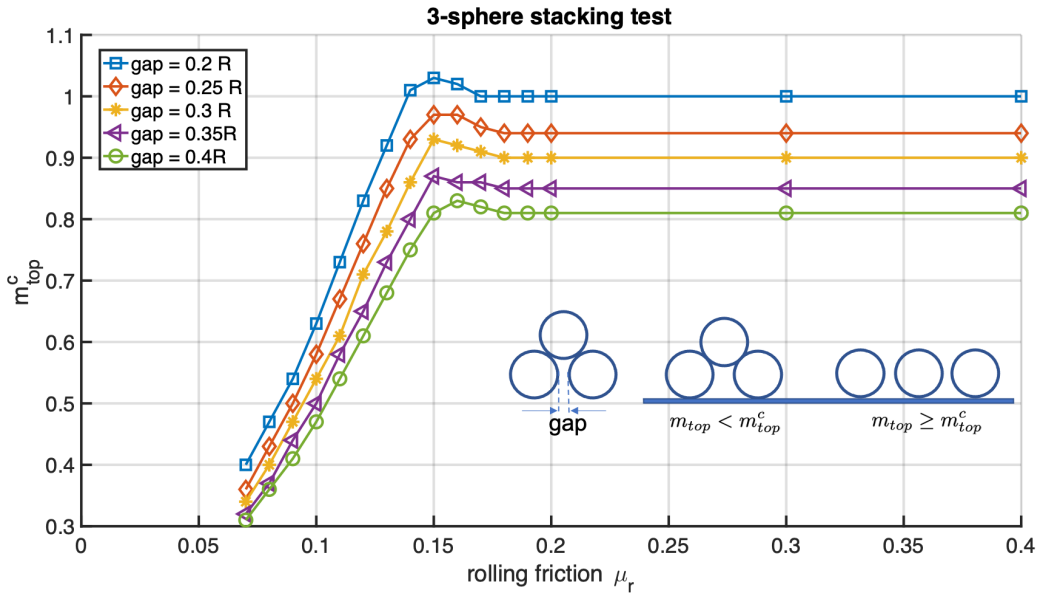


Figure 6. Minimum mass of top sphere m^c_{top} for the pile to collapse.

The same numerical experiment was first conducted in [64] using a spring-damper-type rolling friction model and simulated with Chrono::Multicore. Both tests show similar trends; however, the spring-damper-type rolling friction model can support a heavier weight. Therein, rolling resistance can be nonzero when the relative motion between objects is zero.

4.1.3. Wave Propagation

To validate the contact force at a microscopic level, a system of particles arranged on a triangular lattice was loaded with a downward external force, F_{ext}, at the top center. The system consisted of 15 horizontal layers of spheres, with each layer alternating between 60 and 61 spheres; see Figure 7.

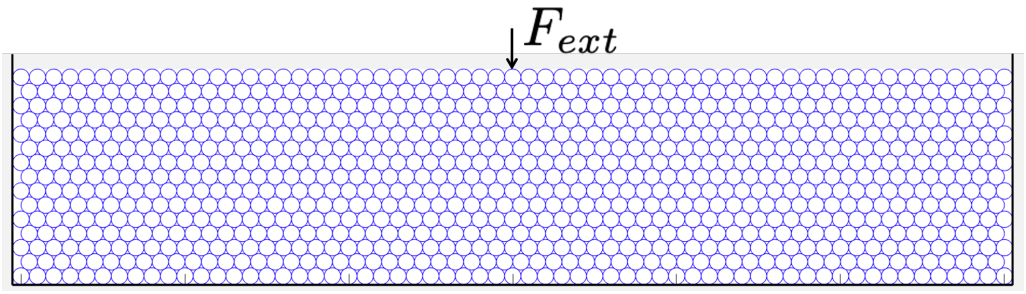
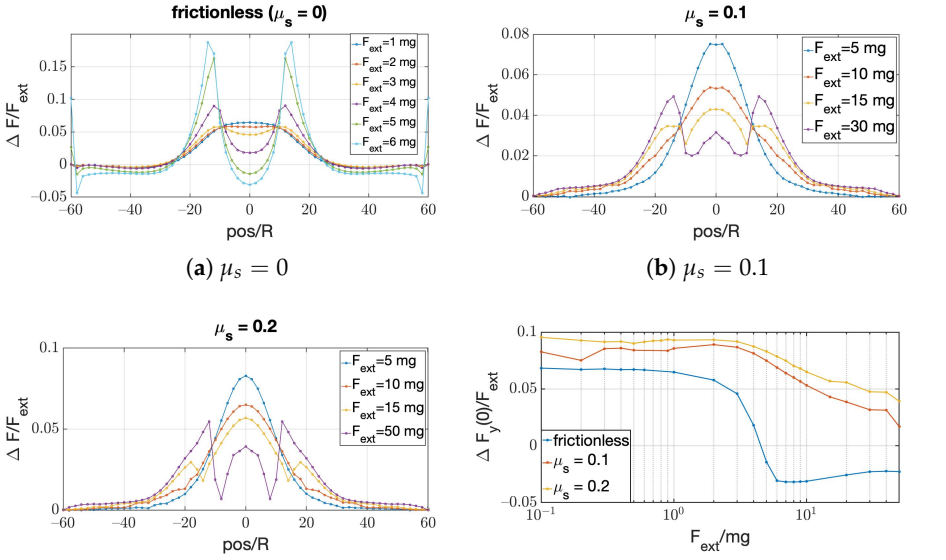


Figure 7. Wave propagation test setup.

This experiment originated from the work of Goldenberg et al. [65,66], in which the authors used 2D disks; the simulations herein were conducted using 3D spheres. The work investigated how friction influences the response of a granular assembly under a localized force. The test consisted of three steps. First, particles were settled with gravity into a box of width $122R$, where R is the particle radius, and the vertical contact forces between the container bottom and each sphere were recorded as F_0. Next, an external force F_{ext} was gradually applied on the top center particle until it reached the desired value. Eventually, with F_{ext} being constant, the system was allowed to settle again, and the contact force between each sphere and the bottom of the box was recorded as F_y.

An interesting force pattern can be observed by using various values of F_{ext} and the sphere–sphere friction coefficient μ_s. The normalized reaction force, $\Delta F/F_{ext}$, where $\Delta F = F_y - F_0$, is plotted as a function of the normalized particle position. Figure 8a–c are results using sphere–sphere friction coefficients of $\mu_s = 0, 0.1$ and 0.2, respectively. The force distribution exhibits two profile types. One type has only one peak, and the maximum force occurs directly below where the force is applied (pos $= 0$). In contrast, the other type can have two peaks. The first type indicates an isotropic elastic response, while the latter represents a hyperbolic response. Figure 8d plots the normalized contact force increment at the bottom center, $\Delta F_y(0)/F_{ext}$, as a function of the normalized external force F_{ext}/mg. For each lattice-grid system, the elasticity disappeared with increasing external force F_{ext}, and the introduction of friction extended the elastic range (flat part of the curve), consistent with the observation from [66].



(**a**) $\mu_s = 0$ (**b**) $\mu_s = 0.1$

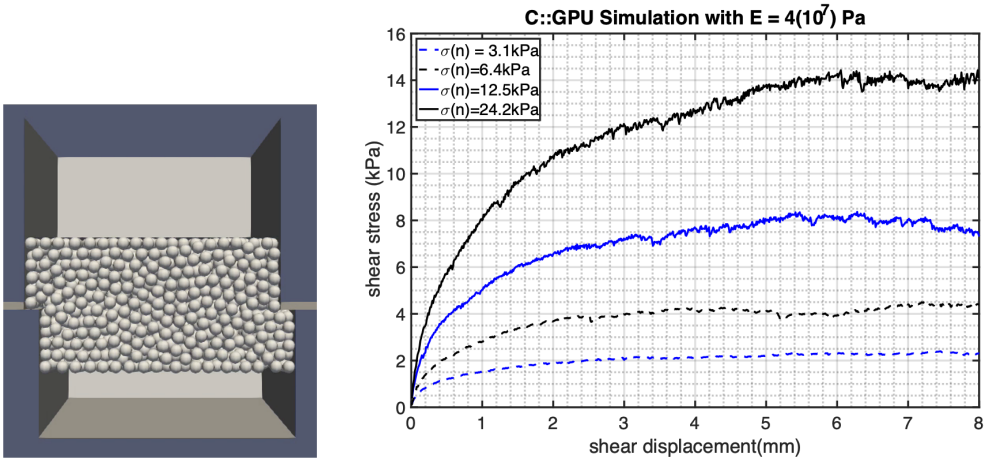
(**c**) $\mu_s = 0.2$ (**d**) Vertical force at bottom center for various F_{ext}

Figure 8. Force profile on the bottom in response to different values of applied force, F_{ext}, applied at the top of the grid at pos $= 0$; the effect of gravity is excluded.

4.2. Benchmark Tests

4.2.1. Direct Shear Test

Several shear tests were carried out and compared with the numerical results reported in [38]. As shown in Figure 9a, a 12 cm × 12 cm box was first filled with identical particles. Next, a plate was placed on top for compression. Four direct shear tests were performed at a shearing velocity of 1 mm/s and under a constant normal stress: 3.1, 6.4, 12.5, and 24.2 kPa. The relation between shear stress and shear displacement is illustrated in Figure 9b. The results match closely to those reported in [38,67].



(**a**) Setup (**b**) Shear stress vs. shear displacement

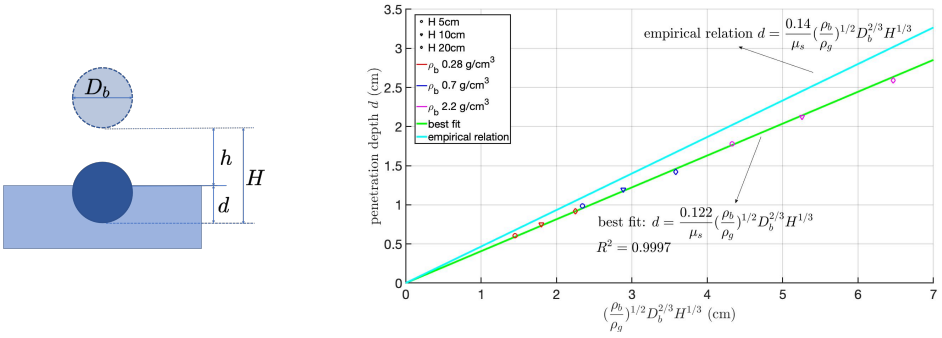
Figure 9. Diagram and results of direct shear test.

4.2.2. Cratering Test

A set of cratering tests was recreated in Chrono::GPU using the experimental setup documented in [68]. A sphere of diameter D_b and density ρ_b was dropped onto a bed of loose packing granular material at different heights, H, as shown in Figure 10a. The penetration depth of the projectile was measured and compared against the empirical relationship inferred from the experiment:

$$d = \frac{0.14}{\mu_s}\left(\frac{\rho_b}{\rho_g}\right)^{1/2} D_b^{2/3} H^{1/3}, \tag{16}$$

where ρ_g is the bulk density of the granular material. A total of nine tests were performed, with different combinations of the projectile density, $\rho_b = 0.28$, 0.7, 2.2 g/cm^3 and drop heights of $H = 5$, 10, 20 cm. All simulations used the same granular medium composed of 115,964 particles of radius 0.1 cm, density 2.5 g/cm^3, and sliding friction coefficient $\mu_s = 0.3$. It is shown experimentally in [69] that the penetration of the projectile is independent of the grain size. The relation between depth d and the scaled total drop height is illustrated in Figure 10b. Here, each simulation result is denoted by a marker, with different colors for different ball densities and different shapes for different drop heights. The green line is a linear fit for the numerical results, with a slope of $0.122/\mu_s$, compared with the empirical line of slope $0.14/\mu_s$, plotted in blue. A similar result was also achieved in [70], where a non-smooth contact model was used to simulate the granular material using Chrono::Multicore.



(**a**) Initial and final projectile positions

(**b**) Penetration depth vs. scaled total drop height

Figure 10. Diagram and results of cratering test.

4.2.3. Low-Velocity Cratering Test

To further validate the collision behavior of impact tests, quantities including the peak acceleration of the projectile, penetration depth, and collision time were evaluated and compared against experimental and numerical results [71,72]. Note that the simulation herein used monodisperse particles, whereas polydisperse ones were employed in [72]. A projectile of mass 1 kg was dropped into a cylinder of settled granular material. A preliminary test demonstrated that when the depth of the granular material exceeds 14 cm, it has a negligible influence on the collision behavior; see Figure 11. This observation was also confirmed in [73].

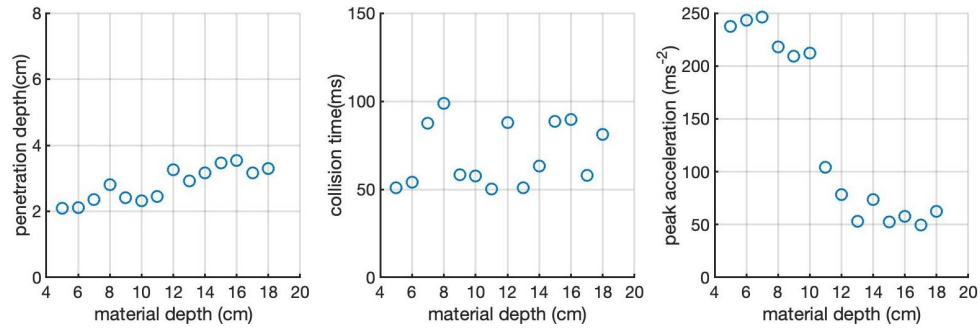


Figure 11. Effect of granular material depth on collision behavior.

In Figure 12, the rolling friction coefficient μ_r was varied to investigate the collision behavior. Increasing the rolling friction decreases the penetration and collision time, but it has an insignificant influence on peak acceleration. Overall, peak acceleration and penetration depth increase with increasing impact velocity, and the collision time decreases.

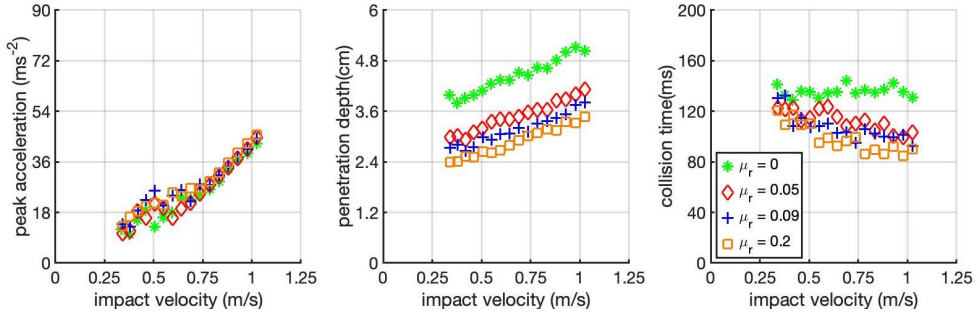


Figure 12. Effect of rolling friction coefficient μ_r on collision behavior.

4.2.4. Rotating Drum

The rotating drum has become an important benchmark test, and numerous numerical and experimental studies have investigated the mixing, segregation and flow regimes of the granular material [33,74–76]. Depending on the range of Froude numbers,

$$F_r = \omega^2 R_{drum}/g, \tag{17}$$

where ω and R_{drum} are the speed and radius of the drum, and g is the gravitational acceleration, particles can transition through six flow regime: slipping, slumping, rolling, cascading, cataracting, and centrifuging [77,78]. Our simulation mimics the setup performed in [76,79], where a 60 mm diameter drum of depth 5 mm was half-filled with particles of diameter 0.54 mm and rotated at a constant velocity. Figure 13 depicts various steady-state flow behaviors with increasing Froude numbers. In each snapshot, the color of the particles represents the normalized absolute velocity, $|v|/(\omega R_{drum})$. When $F_r = 1 \times 10^{-4}$ and 1×10^{-3}, a uniform flow of particles is created at the top thin layer, while the particles in the large lower layer are transported upwards by the drum wall, indicating a rolling regime. When $F_r = 0.01$, a cascading regime sets in, where the bed surface arches in an S shape. At $F_r = 0.5$, the flow enters the cataracting regime, wherein particles stick to the wall before being flung back to the bottom. As an extreme case of cataracting, at $F_r = 1.5$, centrifuging takes place, wherein particles close to the wall form a uniform layer that spins with the cylinder. The observed flow patterns match the predicted motion and transition behaviors described in [77–79], reproducing a similar pattern to that reported in [76].

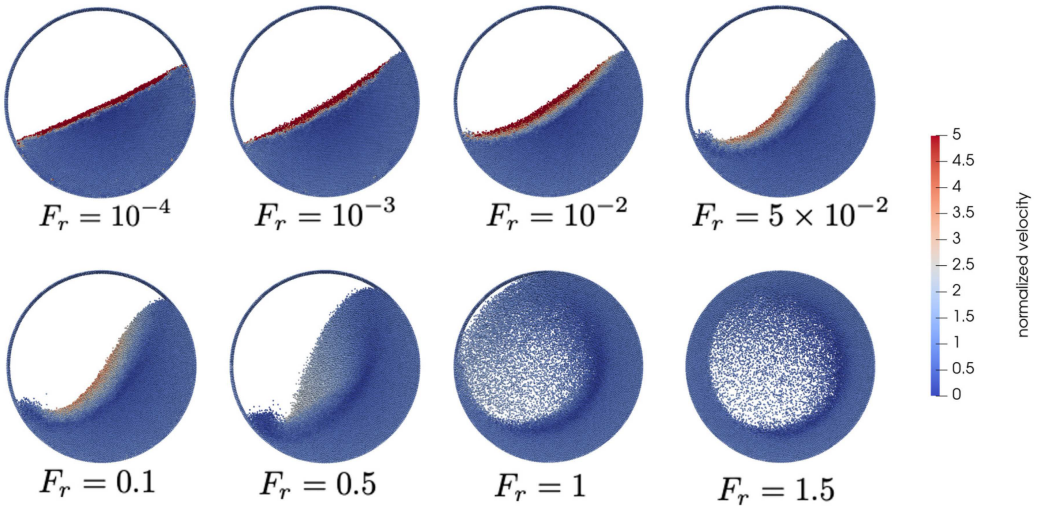


Figure 13. Steady-state flow patterns with increasing Froude number. Particles are colored by normalized velocity magnitude, $\|v\|/\omega R_{drum}$.

5. Scaling Analysis

The purpose of this Chrono::GPU scaling analysis is to provide an idea of what can be expected in terms of its simulation performance. The test scenario is based on a bladed mixer interacting with granular material. The mixer model, represented with triangular meshes, is shown in Figure 14a. During the simulation, the mixer blades rotate at a constant angular velocity of 2π rad/s. Figure 14b shows the initial positions of the granular material, which is confined within a cylindrical region of radius 0.5 m (boundary invisible) and released when the simulation starts. This test is selected because it involves intensive particle–particle and particle–mesh interactions, as illustrated in Figure 14c. Other simulation-related parameters for this test are given in Table 4 (the "Mixer" column). Note

that the stiffness $\hat{k}$ and damping coefficient $\hat{\gamma}$ reported in Table 4 are user-defined and can be tied to k and γ in Equations (3) and (4) via

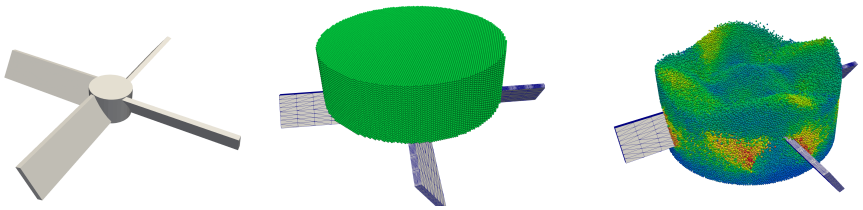
$$k = \sqrt{\frac{\delta_n}{2R}}\hat{k}, \tag{18}$$

$$\gamma = \sqrt{\frac{\delta_n}{2R}} m_{eff}\hat{\gamma}\,, \tag{19}$$

where R is the particle radius and m_{eff} is the effective particle mass.

Table 4. Simulation parameters for mixer scaling analysis and rover dynamics.

Parameter	Mixer	Curiosity
Particle radius (m)	Varies	4.5×10^{-3}
Total particle number	Varies	12,704,030
Particle density (kg/m^3)	2.8×10^3	2.8×10^3
Step size (s)	1×10^{-5}	2.5×10^{-5}
Simulation duration (s)	3	∼35
Normal force stiffness $\hat{k}_n$ (N/m)	1×10^5	1×10^5
Normal force damping coefficient $\hat{\gamma}_n$ ($\cdot$/s)	1×10^4	5×10^4
Tangential force stiffness $\hat{k}_t$ (N/m)	1×10^5	1×10^5
Tangential force damping coefficient $\hat{\gamma}_t$ ($\cdot$/s)	1×10^4	5×10^3
Friction coefficient	0.5	0.75



(**a**) Mixer model (**b**) Initial particle locations (**c**) A screenshot of the mixing process

Figure 14. Visualized mixer scaling test, which involves intensive particle–particle and particle–mesh interactions.

The radii of particles are adjusted in this test to control the total number of particles. The mesh used to represent the mixer blades is not changed between simulations; the triangle count for the mesh is 2892. The analysis is carried out on one NVIDIA Ampere A100 GPU and then one Volta V100 GPU; the timing results are reported Figure 15.

Chrono::GPU with full contact history scales up linearly to 123 million particles on Ampere A100. Past this point, the device memory runs out, leading to a notable slowdown as the solver starts paging memory in and out of the GPU at run time. A100 also offers more than a two-fold increase in speed compared with V100. The rule of thumb is that on A100 with the current Chrono::GPU implementation, a fully resolved DEM simulation, with mesh interaction and relatively fine step sizes, takes well below 1 h per 1 million particles and 1 s of simulation time (more precisely, 1×10^5 simulation steps).

Figure 16 illustrates how the more important functions in Chrono::GPU compare to each other in terms of computational effort. These functions pertain to assigning particles and triangles to their corresponding subdomains (SphereBroad and TriangleBroad; see Section 3.2 for the purpose of these steps), particle–particle contact detection (SphereContactPairs), particle–particle force calculation (SphereForce), particle–triangle force calculation (TriangleSphereForce), numerical integration (Integrate), and updating contact history (UpdateFriction; see Section 2 for history-based friction model). For small problem sizes, SphereBroad and TriangleBroad take a larger portion of the total runtime. This

means arranging the data to use GPU architecture incurs noticeable overheads for smaller problems. Although those overheads are far from nullifying Chrono::GPU's advantage over CPU-based DEM, as the problem size grows larger, those data arrangement overheads become less impactful, and the main time consumer shifts to SphereContactPairs and SphereForce. Indeed, contact detection and normal/tangential force calculation are typically time-consuming processes in DEM and are the focal points for the future performance optimization of Chrono::GPU. Integrate and UpdateFriction, on the other hand, take consistently small portions of the total runtime.

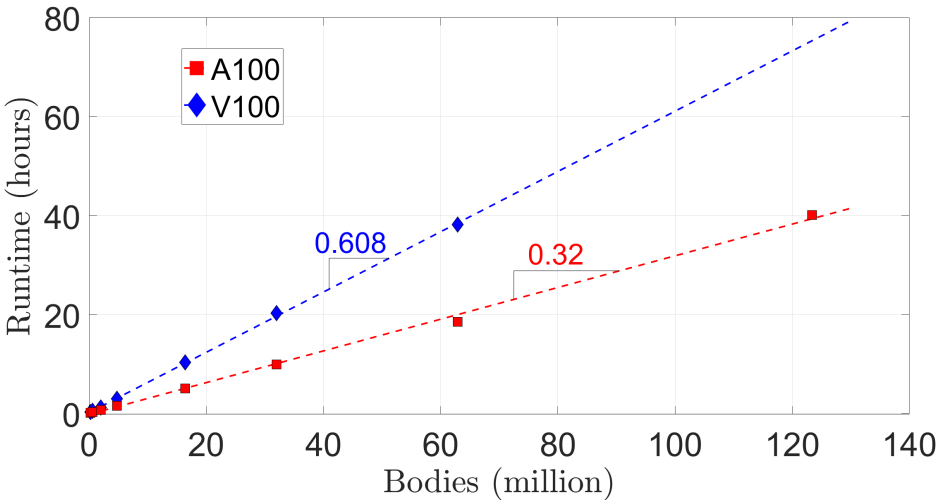


Figure 15. The scaling result of the 3 second mixer simulation at time step size 1×10^{-5} s, on NVIDIA A100 and V100. The linear scaling holds for up to 123 million particles (NVIDIA A100).

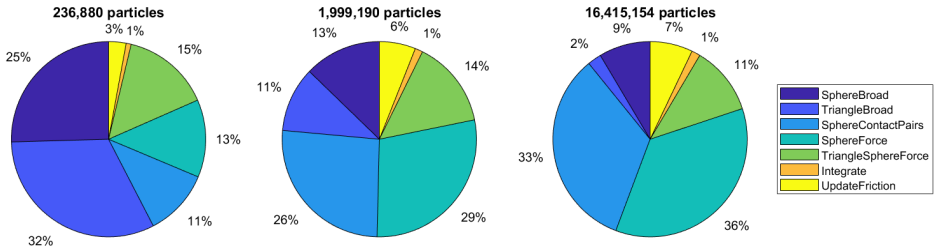


Figure 16. The runtime for each of the main subroutines in terms of percentages. On average, mixer simulations of these resolutions take 0.0031 s, 0.0096 s, and 0.065 s to finish a time step on A100, respectively.

6. Demonstration of Technology: Rover Mobility Simulations

A rover vehicle operating on DEM granular terrain is used to demonstrate the potential of Chrono::GPU. The rover model represents the Curiosity Mars Rover, which was launched by NASA on 26 November 2011 [80]. The geometry of the rover is consistent with NASA's official website [81]. A 45° front view can be found in Figure 17. This Chrono model features full support of a six-wheel Mars rover with a Rocker–Bogie suspension system. The Rocker–Bogie mechanism implementation details can be found in Figure 18, where black arrows indicate revolute joint constraints and constraint rotational directions. No shock absorber mechanism is modeled at these joints. The suspension prevents the rover from uncontrolled yaw and body motion. It also prevents both sides of the rover's wheels being lifted up when one side of the rover hits an obstacle, meaning that traction is maintained.

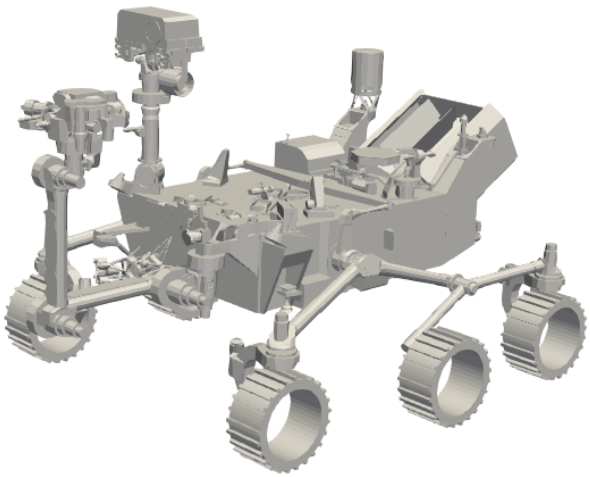


Figure 17. Curiosity rover, 45° front view.

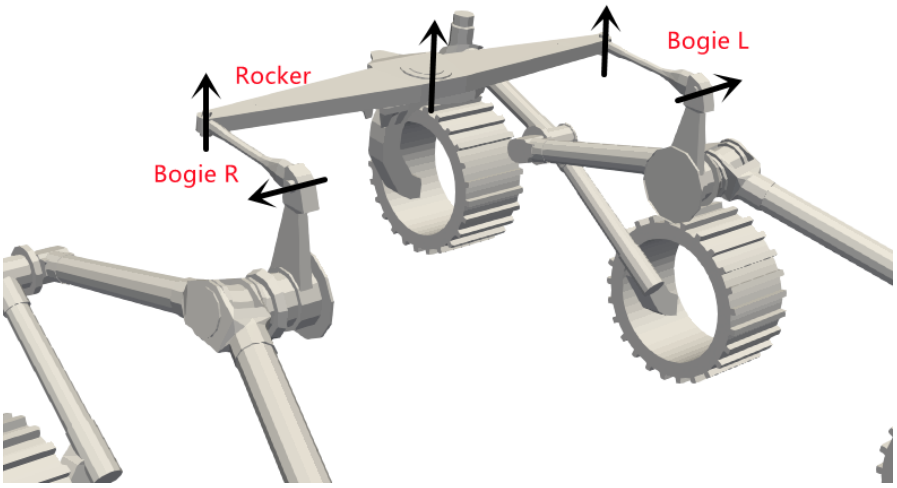


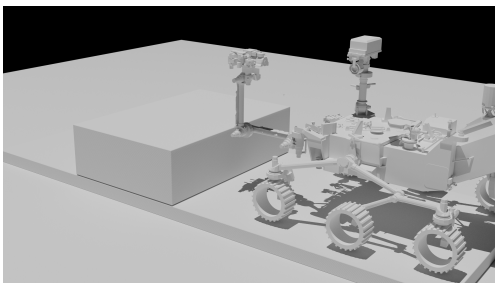
Figure 18. Details of Rocker–Bogie suspension mechanism design.

The model in this simulation uses a simplified wheel geometry, where most wheel features are omitted and the grooves are exaggerated. This choice allowed for a wheel geometry modeled with fewer triangles that burdens the simulation less while still showcasing complex geometry handling. Inspecting the effect of other, potentially more refined wheel geometries, while totally possible, will lead to longer simulation times. The wheel has a diameter of 0.5 m and a thickness of 0.05 m. The grouser on the simplified wheel geometry has a height of 0.02 m and a thickness of 0.02 m. The wheels are subject to a constant angular velocity of π rad/s in this simulation. The mass of the rover was set to 463 kg; more simulation parameters are provided in Table 4 (the "Curiosity" column). Each wheel's geometry is specified using 703 triangles.

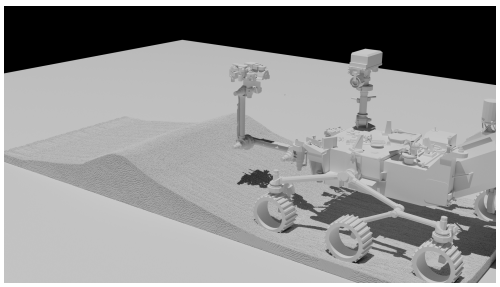
The frictional contact forces between particles and between particles and wheels are computed with Chrono::GPU and then fed into the rover mechanical model managed by the Chrono core module for vehicle-level simulation. The rest of the rover model does not participate in the interaction with the terrain; it is managed only by the Chrono core module as rigid bodies. In this example, we mimic a scenario where the rover climbs a granular heap. We instantiate an extra block of granular material on top of the underlying granular "bed" (see Figure 19a), so that after the terrain settles, a small heap forms. A "Grid" sampler is used to generate the initial positions of the particles. In total, 12,704,030 DEM particles participate in the simulation. An animation of this simulation is available online [82].

One notable aspect is that the wheels are slipping while the rover is climbing, and the material under the wheels is flowing down the slope. Figure 19 highlights key moments of this simulation.

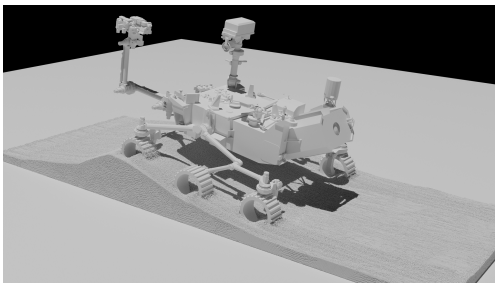
Further insights of this climbing maneuver can be gained from Figure 20, which reports the reaction forces on the joints that connect the wheels and the vehicle body. The magnitude of these reaction forces on the front-left, middle-left, and rear-left joints are plotted together with their respective positions (Z coordinates). The plot shows that with this Curiosity model, the rear wheel joint is subject to the largest reaction force as the rear wheel provides the most traction. This is followed by the front wheel, and then the middle wheel. Furthermore, from the correlation between the wheel Z coordinates and joint forces, it becomes apparent that all wheels experience increased traction when entering the climbing phase.



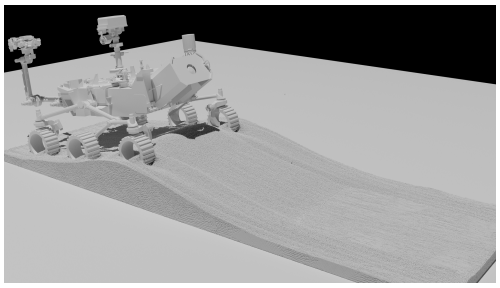
(**a**) Initial profile of simulation, before terrain settling.



(**b**) After the settling phase, the start of simulation.

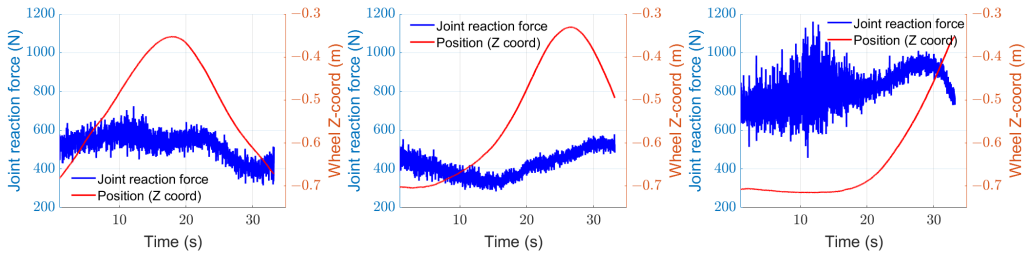


(**c**) Rover climbing the granular heap, front wheel reaching top.



(**d**) Rover climbing the granular heap, rear wheel reaching top.

Figure 19. Simulation of Curiosity rover climbing a granular heap.



(**a**) The front wheel position and corresponding joint force.

(**b**) The middle wheel position and corresponding joint force.

(**c**) The rear wheel position and corresponding joint force.

Figure 20. The evolution of wheel joint reaction forces during the course of rover climbing a heap of granular material.

7. Conclusions

Chrono::GPU, an open-source module that forms part of the Project Chrono platform, is capable of simulating large granular systems using the Discrete Element Method. Both

Chrono and Chrono::GPU are available on GitHub and released under a permissive BSD3 license for unrestricted use and distribution. By leveraging GPU computing, the simulation time scales linearly with the problem size until it reaches GPU memory capacity. A wide variety of validation tests, from static and quasi-static to dynamic problems, have been carried out to gauge the physical soundness of the solver. To the best of our knowledge, there is no other publicly available open source dynamics engine that outperforms Chrono in its GPU support for the simulation of scenarios that involve the interplay between granular material and complex mechanical systems such as rovers and wheeled and tracked vehicles. The main limitation of the current implementation is its handling of only monodisperse granular material. An updated version of the solver described herein is currently being developed to address this shortcoming using the approach described in [83]. The updated simulator will also leverage multi-GPU computation to reduce run times.

Author Contributions: Conceptualization, D.N.; methodology, D.N. and R.S.; software, R.Z., L.F., C.V.H. and D.N.; validation, L.F. and R.Z.; formal analysis, L.F. and R.Z.; investigation, R.Z., L.F. and D.N.; resources, C.V.H. and D.N.; writing—original draft preparation, L.F. and R.Z.; writing—review and editing, R.S. and D.N.; visualization, L.F. and R.Z.; supervision, C.V.H., R.S. and D.N.; project administration, D.N.; funding acquisition, D.N. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded in part by US Army Research Office grants W911NF1910431 and W911NF1810476 and National Science Foundation grant CISE1835674.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Chrono::GPU is an open-source software, available on https://github.com/projectchrono/chrono, accessed on 7 October 2021; all tests were run using feature/gpu branch.

Acknowledgments: We would like to thank Cecily Sunday for sharing experimental data and valuable advice on the low impact velocity test and rotating drum test.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Cundall, P.A.; Strack, O.D. A discrete numerical model for granular assemblies. *Geotechnique* **1979**, *29*, 47–65. [CrossRef]
2. Pöschel, T.; Schwager, T. *Computational Granular Dynamics: Models and Algorithms*; Springer: Berlin/Heidelberg, Germany, 2005.
3. Lemieux, M.; Léonard, G.; Doucet, J.; Leclaire, L.A.; Viens, F.; Chaouki, J.; Bertrand, F. Large-scale numerical investigation of solids mixing in a V-blender using the discrete element method. *Powder Technol.* **2008**, *181*, 205–216. [CrossRef]
4. Apostolou, K.; Hrymak, A. Discrete element simulation of liquid-particle flows. *Comput. Chem. Eng.* **2008**, *32*, 841–856. [CrossRef]
5. Tang, C.L.; Hu, J.C.; Lin, M.L.; Angelier, J.; Lu, C.Y.; Chan, Y.C.; Chu, H.T. The Tsaoling landslide triggered by the Chi-Chi earthquake, Taiwan: Insights from a discrete element simulation. *Eng. Geol.* **2009**, *106*, 1–19. [CrossRef]
6. Salciarini, D.; Tamagnini, C.; Conversini, P. Discrete element modeling of debris-avalanche impact on earthfill barriers. *Phys. Chem. Earth, Parts A/B/C* **2010**, *35*, 172–181. [CrossRef]
7. Sánchez, P.; Scheeres, D.J. Simulating asteroid rubble piles with a self-gravitating soft-sphere distinct element method model. *Astrophys. J.* **2011**, *727*, 120. [CrossRef]
8. Amritkar, A.; Deb, S.; Tafti, D. Efficient parallel CFD-DEM simulations using OpenMP. *J. Comput. Phys.* **2014**, *256*, 501–519. [CrossRef]
9. Yan, B.; Regueiro, R.A. A comprehensive study of MPI parallelism in three-dimensional discrete element method (DEM) simulation of complex-shaped granular particles. *Comput. Part. Mech.* **2018**, *5*, 553–577. [CrossRef]
10. Checkaraou, A.W.M.; Rousset, A.; Besseron, X.; Varrette, S.; Peters, B. Hybrid mpi+ openmp implementation of extended discrete element method. In Proceedings of the 2018 30th International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD), Lyon, France, 24–27 September 2018; pp. 450–457.
11. Xu, J.; Qi, H.; Fang, X.; Lu, L.; Ge, W.; Wang, X.; Xu, M.; Chen, F.; He, X.; Li, J. Quasi-real-time simulation of rotating drum using discrete element method with parallel GPU computing. *Particuology* **2011**, *9*, 446–450. [CrossRef]
12. Govender, N.; Wilke, D.; Kok, S. Blaze-DEMGPU: Modular high performance DEM framework for the GPU architecture. *SoftwareX* **2016**, *5*, 62–66. [CrossRef]
13. Iwashita, K.; Oda, M. Rolling resistance at contacts in simulation of shear band development by DEM. *J. Eng. Mech.* **1998**, *124*, 285–292. [CrossRef]

14. Renzo, A.D.; Maio, F.P.D. Comparison of contact-force models for the simulation of collisions in DEM-based granular flow codes. *Chem. Eng. Sci.* **2004**, *59*, 525–541. [CrossRef]
15. Da Cruz, F.; Emam, S.; Prochnow, M.; Roux, J.N.; Chevoir, F. Rheophysics of dense granular materials: Discrete simulation of plane shear flows. *Phys. Rev. E* **2005**, *72*, 021309. [CrossRef] [PubMed]
16. Rycroft, C.H.; Grest, G.S.; Landry, J.W.; Bazant, M.Z. Analysis of granular flow in a pebble-bed nuclear reactor. *Phys. Rev. E* **2006**, *74*, 021306. [CrossRef] [PubMed]
17. Kruggel-Emden, H.; Sturm, M.; Wirtz, S.; Scherer, V. Selection of an appropriate time integration scheme for the discrete element method (DEM). *Comput. Chem. Eng.* **2008**, *32*, 2263–2279. [CrossRef]
18. Wasfy, T.M.; Wasfy, H.M.; Peters, J.M. Coupled Multibody Dynamics and Discrete Element Modeling of Vehicle Mobility on Cohesive Granular Terrains. In Proceedings of the ASME 2014 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Buffalo, NY, USA, 17–20 August 2014; American Society of Mechanical Engineers: New York, NY, USA; p. V006T10A050.
19. Lommen, S. DEM speedup: Stiffness effects on behavior of bulk material. *Particuology* **2014**, *12*, 107–112. [CrossRef]
20. Utili, S.; Zhao, T.; Houlsby, G. 3D DEM investigation of granular column collapse: Evaluation of debris motion and its destructive power. *Eng. Geol.* **2015**, *186*, 3–16. [CrossRef]
21. Potticary, M.; Zervos, A.; Harkness, J. An investigation into the effect of particle platyness on the strength of granular material using the discrete element method. In Proceedings of the IV International Conference on Particle-based Methods—Fundamentals and Applications, Barcelona, Spain, 28–30 September 2015.
22. Michael, M.; Vogel, F.; Peters, B. DEM-FEM coupling simulations of the interactions between a tire tread and granular terrain. *Comput. Methods Appl. Mech. Eng.* **2015**, *289*, 227–248. [CrossRef]
23. Ciantia, M.; Arroyo, M.; Butlanska, J.; Gens, A. DEM modelling of cone penetration tests in a double-porosity crushable granular material. *Comput. Geotech.* **2016**, *73*, 109–127. [CrossRef]
24. Zheng, Z.; Zang, M. Numerical Simulations of the Interactions Between a Pneumatic Tire and Granular Sand by 3D DEM-FEM. In Proceedings of the 7th International Conference on Discrete Element Methods, Dalian, China, 1–4 August 2016; pp. 289–300.
25. Parteli, E.; Poschel, T. Particle-based simulation of powder application in additive manufacturing. *Powder Technol.* **2016**, *288*, 96–102. [CrossRef]
26. Kivugo, R. Tire-Soil Interaction for Off-Road Vehicle Applications. Ph.D. Thesis, Politecnico di Milano, Milan, Italy, 2017.
27. Calvetti, F.; Prisco, C.; Vairaktaris, E. DEM assessment of impact forces of dry granular masses on rigid barriers. *Acta Geotech.* **2016**, *12*, 129–144. [CrossRef]
28. Cleary, P.; Sawley, M. DEM modelling of industrial granular flows: 3D case studies and the effect of particle shape on hopper discharge. *Appl. Math. Model.* **2002**, *26*, 89–111. [CrossRef]
29. Bertrand, F.; Leclaire, L.; Levecque, G. DEM-based models for the mixing of granular materials. *Chem. Eng. Sci.* **2005**, *60*, 2517–2531. [CrossRef]
30. Bouffard, J.; Bertrand, F.; Chaouki, J.; Dumont, H. Discrete element investigation of flow patterns and segregation in a spheronizer. *Comput. Chem. Eng.* **2013**, *49*, 170–182. [CrossRef]
31. Kloss, C.; Goniva, C.; Hager, A.; Amberger, S.; Pirker, S. Models, algorithms and validation for opensource DEM and CFD–DEM. *Prog. Comput. Fluid Dyn. Int.* **2012**, *12*, 140–152. [CrossRef]
32. Gropp, W.; Lusk, E.; Skjellum, A. *Using MPI: Portable Parallel Programming with the Message-Passing Interface*, 2nd ed.; MIT Press: Cambridge, MA, USA, 1999.
33. Alizadeh, E.; Bertrand, F.; Chaouki, J. Comparison of DEM results and Lagrangian experimental data for the flow and mixing of granules in a rotating drum. *AIChE J.* **2014**, *60*, 60–75. [CrossRef]
34. Gan, J.; Zhou, Z.; Yu, A. A GPU-based DEM approach for modeling of particulate systems. *Powder Technol.* **2016**, *301*, 1172–1182. [CrossRef]
35. Hou, Q.; Dong, K.; Yu, A. DEM study of the flow of cohesive particles in a screw feeder. *Powder Technol.* **2014**, *256*, 529–539. [CrossRef]
36. He, Y.; Evans, T.; Yu, A.; Yang, R. A GPU-based DEM for modeling large scale powder compaction with wide size distributions. *Powder Technol.* **2018**, *333*, 219–228. [CrossRef]
37. Toson, P.; Siegmann, E.; Trogrlic, M.; Kureck, H.; Khinast, J.; Jajcevic, D.; Doshi, P.; Blackwood, D.; Bonnassieux, A.; Daugherity, P.D.; et al. Detailed modeling and process design of an advanced continuous powder mixer. *Int. J. Pharm.* **2018**, *552*, 288–300. [CrossRef] [PubMed]
38. Fleischmann, J.; Serban, R.; Negrut, D.; Jayakumar, P. On the importance of displacement history in soft-body contact models. *J. Comput. Nonlinear Dyn.* **2016**, *11*, 044502. [CrossRef]
39. Longmore, J.P.; Marais, P.; Kuttel, M.M. Towards realistic and interactive sand simulation: A GPU-based framework. *Powder Technol.* **2013**, *235*, 983–1000. [CrossRef]
40. Ucgul, M.; Fielke, J.; Saunders, C. Three-dimensional discrete element modeling (DEM) of tillage: Accounting for soil cohesion and adhesion. *Biosyst. Eng.* **2015**, *129*, 298–306. [CrossRef]
41. Ucgul, M.; Saunders, C.; Fielke, J. Comparison of the discrete element and finite element methods to model the interaction of soil and tool cutting edge. *Biosyst. Eng.* **2018**, *169*, 199–208. [CrossRef]

42. Recuero, A.M.; Serban, R.; Peterson, B.; Sugiyama, H.; Jayakumar, P.; Negrut, D. A high-fidelity approach for vehicle mobility simulation: Nonlinear finite element tires operating on granular material. *J. Terramech.* **2017**, *72*, 39–54. [CrossRef]
43. Zhao, C.L.; Zang, M.Y. Application of the FEM/DEM and alternately moving road method to the simulation of tire-sand interactions. *J. Terramech.* **2017**, *72*, 27–38. [CrossRef]
44. Johnson, J.B.; Kulchitsky, A.V.; Duvoy, P.; Iagnemma, K.; Senatore, C.; Arvidson, R.E.; Moore, J. Discrete element method simulations of Mars Exploration Rover wheel performance. *J. Terramech.* **2015**, *62*, 31–40. [CrossRef]
45. Furuichi, M.; Nishiura, D.; Asai, M.; Hori, T. The first real-scale DEM simulation of a sand-box experiment using 2.4 billion particles. In Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, Denver, CO, USA, 12–17 November 2017.
46. Furuichi, M.; Nishiura, D.; Kuwano, O.; Bauville, A.; Hori, T.; Sakaguchi, H. Arcuate stress state in accretionary prisms from real-scale numerical sandbox experiments. *Nat. Sci. Rep.* **2018**, *8*, 1–11. [CrossRef]
47. Nishiura, D.; Sakaguchi, H.; Yamamoto, S. Multibillion particle DEM to simulate centrifuge model tests of geomaterials. In *Physical Modelling in Geotechnics, Volume 1: Proceedings of the 9th International Conference on Physical Modelling in Geotechnics (ICPMG 2018), London, UK, 17–20 July 2018*; CRC Press: Boca Raton, FL, USA, 2018; p. 227.
48. Strohmaier, E.; Dongarra, J.; Simon, H.; Meuer, M. TOP500 Supercomputer Site. Available online: http://www.top500.org (accessed on 7 October 2021).
49. Tasora, A.; Negrut, D.; Anitescu, M. GPU-Based Parallel Computing for the Simulation of Complex Multibody Systems with Unilateral and Bilateral Constraints: An Overview. In *Multibody Dynamics*; Arczewski, K.; Blajer, W.; Fraczek, J.; Wojtyra, M., Eds.; Springer: Berlin/Heidelberg, Germany, 2011; Volume 23, pp. 283–307. [CrossRef]
50. Mindlin, R.; Deresiewicz, H. Elastic spheres in contact under varying oblique forces. *J. Appl. Mech.* **1953**, *20*, 327–344. [CrossRef]
51. Brilliantov, N.V.; Spahn, F.; Hertzsch, J.M.; Pöschel, T. Model for collisions in granular gases. *Phys. Rev. E* **1996**, *53*, 5382. [CrossRef] [PubMed]
52. Silbert, L.; Ertaş, D.; Grest, G.; Halsey, T.; Levine, D.; Plimpton, S. Granular flow down an inclined plane: Bagnold scaling and rheology. *Phys. Rev. E* **2001**, *64*, 051302. [CrossRef] [PubMed]
53. Hertz, H. Ueber die Verdunstung der Flüssigkeiten, insbesondere des Quecksilbers, im luftleeren Raume. *Ann. Phys.* **1882**, *253*, 177–193. [CrossRef]
54. Johnson, K.L. *Contact Mechanics*; Cambridge University Press: Cambridge, UK, 1987.
55. Baboulin, M.; Buttari, A.; Dongarra, J.; Kurzak, J.; Langou, J.; Langou, J.; Luszczek, P.; Tomov, S. Accelerating scientific computations with mixed precision algorithms. *Comput. Phys. Commun.* **2009**, *180*, 2526–2533. [CrossRef]
56. Hockney, R.; Eastwood, J. *Computer Simulation Using Particles*; CRC Press: Boca Raton, FL, USA, 1988.
57. NVIDIA Corporation. Compute Unified Device Architecture Toolkit Documentation. Available online: https://docs.nvidia.com/cuda (accessed on 7 October 2021).
58. Tasora, A.; Serban, R.; Mazhar, H.; Pazouki, A.; Melanz, D.; Fleischmann, J.; Taylor, M.; Sugiyama, H.; Negrut, D. Chrono: An open source multi-physics dynamics engine. In *High Performance Computing in Science and Engineering—Lecture Notes in Computer Science*; Kozubek, T., Ed.; Springer: Berlin/Heidelberg, Germany, 2016; pp. 19–49.
59. Fang, L.; Zhang, R.; Negrut, D. On the Validation of Chrono::GPU. In *Technical Report TR-2021-05, Simulation-Based Engineering Laboratory*; University of Wisconsin-Madison: Madison, WI, USA, 2021.
60. Mazhar, H.; Heyn, T.; Pazouki, A.; Melanz, D.; Seidl, A.; Bartholomew, A.; Tasora, A.; Negrut, D. Chrono: A parallel multi-physics library for rigid-body, flexible-body, and fluid dynamics. *Mech. Sci.* **2013**, *4*, 49–64. [CrossRef]
61. Pazouki, A.; Kwarta, M.; Williams, K.; Likos, W.; Serban, R.; Jayakumar, P.; Negrut, D. Compliant contact versus rigid contact: A comparison in the context of granular dynamics. *Phys. Rev. E* **2017**, *96*, 042905. [CrossRef] [PubMed]
62. Wu, C.Y.; Thornton, C.; Li, L.Y. Coefficients of restitution for elastoplastic oblique impacts. *Adv. Powder Technol.* **2003**, *14*, 435–448. [CrossRef]
63. Yu, K.; Elghannay, H.A.; Tafti, D. An impulse based model for spherical particle collisions with sliding and rolling. *Powder Technol.* **2017**, *319*, 102–116. [CrossRef]
64. Fang, L.; Negrut, D. Producing 3D friction loads by tracking the motion of the contact point on bodies in mutual contact. *Comput. Part. Mech.* **2021**, *8*, 905–929. [CrossRef]
65. Goldenberg, C.; Goldhirsch, I. Force chains, microelasticity, and macroelasticity. *Phys. Rev. Lett.* **2002**, *89*, 084302. [CrossRef] [PubMed]
66. Goldenberg, C.; Goldhirsch, I. Friction enhances elasticity in granular solids. *Nature* **2005**, *435*, 188–191. [CrossRef] [PubMed]
67. Hartl, J.; Ooi, J. Experiments and simulations of direct shear tests: porosity, contact friction and bulk friction. *Granul. Matter* **2008**, *10*, 263–271. [CrossRef]
68. Ambroso, M.A.; Santore, C.R.; Abate, A.R.; Durian, D.J. Penetration depth for shallow impact cratering. *Phys. Rev. E* **2005**, *71*, 051305. [CrossRef] [PubMed]
69. Uehara, J.S.; Ambroso, M.A.; Ojha, R.P.; Durian, D.J. Low-Speed Impact Craters in Loose Granular Media. *Phys. Rev. Lett.* **2003**, *90*, 194301. [CrossRef] [PubMed]
70. Heyn, T. On the Modeling, Simulation, and Visualization of Many-Body Dynamics Problems with Friction and Contact. Ph.D. Thesis, Department of Mechanical Engineering, University of Wisconsin–Madison, Madison, WI, USA, 2013. Available online: http://sbel.wisc.edu/documents/TobyHeynThesis_PhDfinal.pdf (accessed on 7 October 2021).

71. Murdoch, N.; Drilleau, M.; Sunday, C.; Thuillet, F.; Wilhelm, A.; Nguyen, G.; Gourinat, Y. Low-velocity impacts into granular material: Application to small-body landing. *Mon. Not. R. Astron. Soc.* **2021**, *503*, 3460–3471. [CrossRef]
72. Sunday, C.; Zhang, Y.; Thuillet, F.; Tardivel, S.; Michel, P.; Murdoch, N. The influence of gravity on granular impacts I. A DEM code performance comparison. *Astron. Astrophys.* **2021**, in press. [CrossRef]
73. Nelson, E.; Katsuragi, H.; Mayor, P.; Durian, D.J. Projectile interactions in granular impact cratering. *Phys. Rev. Lett.* **2008**, *101*, 068001. [CrossRef]
74. Dury, C.M.; Ristow, G.H. Radial segregation in a two-dimensional rotating drum. *J. Phys. I* **1997**, *7*, 737–745. [CrossRef]
75. Komossa, H.; Wirtz, S.; Scherer, V.; Herz, F.; Specht, E. Transversal bed motion in rotating drums using spherical particles: Comparison of experiments with DEM simulations. *Powder Technol.* **2014**, *264*, 96–104. [CrossRef]
76. Sunday, C.; Murdoch, N.; Tardivel, S.; Schwartz, S.R.; Michel, P. Validating n-body code Chrono for granular DEM simulations in reduced-gravity environments. *Mon. Not. R. Astron. Soc.* **2020**, *498*, 1062–1079. [CrossRef]
77. Henein, H.; Brimacombe, J.; Watkinson, A. Experimental study of transverse bed motion in rotary kilns. *Metall. Trans. B* **1983**, *14*, 191–205. [CrossRef]
78. Mellmann, J. The transverse motion of solids in rotating cylinders—Forms of motion and transition behavior. *Powder Technol.* **2001**, *118*, 251–270. [CrossRef]
79. Brucks, A.; Arndt, T.; Ottino, J.M.; Lueptow, R.M. Behavior of flowing granular materials under variable g. *Phys. Rev. E* **2007**, *75*, 032301. [CrossRef] [PubMed]
80. NASA. Mars Curiosity Rover. Available online: https://mars.nasa.gov/msl/spacecraft/rover/summary/ (accessed on 7 October 2021).
81. Kumanchik, B.; NASA; JPL-Caltech. Curiosity Clean, NASA 3d Resources. 2016. Available online: https://nasa3d.arc.nasa.gov/detail/curiosity-clean (accessed on 7 October 2021).
82. Zhang, R.; Fang, L.; Negrut, D. DEM Simulation of Curiosity Rover Climbing Up Granular Heap. Simulation-Based Engineering Laboratory, University of Wisconsin-Madison. 2021. Available online: https://uwmadison.box.com/s/dgq34vtb4vdtpik7c3v1ht7l4wxvp1g7 (accessed on 7 October 2021).
83. Price, M.; Murariu, V.; Morrison, G. Sphere clump generation and trajectory comparison for real particles. *Proc. Discret. Elem. Model.* **2007**. Available online: http://www.dip.ee.uct.ac.za/~mathew/publish/dem2007_sphereclump.pdf (accessed on 7 October 2021).