Proceedings of the ASME 2010 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference
IDETC/CIE 2010
August 15-18, 2010, Montreal, Quebec, Canada

DETC2010-28804

On the Validation of a Differential Variational Inequality Approach for the Dynamics of Granular Material

Dan Melanz
Simulation Based Engineering Laboratory
Department of Mechanical Engineering
University of Wisconsin
Madison, WI 53706
melanz@wisc.edu

Martin Tupy
Simulation Based Engineering Laboratory
Department of Mechanical Engineering
University of Wisconsin
Madison, WI 53706
tupy@wisc.edu

Bryce Smith
Dept. Mech. Engineering
University of Wisconsin
Madison, WI 53706
bbsmith3@wisc.edu

Kevin Turner
Dept. Mech. Engineering
University of Wisconsin,
Madison, WI 53706
kturner@engr.wisc.edu

Dan Negrut
Dept. Mech. Engineering
University of Wisconsin,
Madison, WI 53706
negrut@engr.wisc.edu

ABSTRACT

The validation of a DVI approach for the dynamics of granular material focuses on comparing the experimental and simulation results of granular flow for two tests in the Chrono::Engine simulation environment. A macro scale validation was previously carried out through examination of granular flow in PBR reactors [1]. For this work, an aluminum rig was designed and fabricated to measure the flow rate of a given amount of micro scale granular material flowing due to gravity through a slit. The flow was initiated by using a Newport UMR8.25 translational stage and Newport LTA-HL precision linear actuator to open and close the slit steadily. Once the slit was open, the weight of the granular material was transmitted to the processor via a router connected to a Cooper LFS242 Tension/Compression Cell (Serial No. 286284) and graphed over time. A model of the flow meter was created in Chrono::Engine and the results were matched to experimental runs by changing the friction coefficient between particles. After the friction coefficient of the particles was determined to be 0.15, several experimental runs with differing slit sizes were run. These flow rates were compared to the weight versus time data that Chrono::Engine output for the corresponding slit size. Runs for gap sizes of 1.5mm, 2.0mm, 2.5mm and 3.0mm were performed with 0.0624 N of granular material, which amounted to approximately 39,000 spheres with 500μm in diameter. These gap sizes corresponded to an experimental flow rate of 1.41E-2 N/s, 2.59E-2 N/s, 4.00E-2 N/s, and 4.44E-2 N/s, and a simulated flow rate of 1.40E-2 N/s, 2.62E-2 N/s, 4.05E-2 N/s, and 4.48E-2 N/s, respectively. Based on this experiment, Chrono::Engine had less than a 2% error in calculating the flow rate of the granular material through a slit. In addition to comparing flow rates, the pile repose angle from the experimental runs was compared to the simulation results. A description of the GPU execution model along with its memory spaces is provided to illustrate its potential for parallel scientific computing. The equations of motion associated with the dynamics of many rigid bodies are introduced and a solution method is presented. The solution method is designed to map well on the parallel hardware, which is demonstrated by an order of magnitude reductions in simulation time for large systems that concern the dynamics of granular material.

LARGE SCALE MULTIBODY DYNAMICS ON THE GPU

This section briefly introduces the theoretical background for mechanical systems made up of multiple rigid bodies whose time evolution is controlled by external forces, frictional contacts, bilateral constraints and motors.

The Formulation of the Equations of Motion

The state of a mechanical system with n_b rigid bodies in three dimensional space can be represented by the generalized coordinates

$$\mathbf{q} = \left[\mathbf{r}_1^T, \epsilon_1^T, \ldots, \mathbf{r}_{n_b}^T, \epsilon_{n_b}^T\right]^T \in \mathbb{R}^{7n_b}$$

and their time derivatives

$$\dot{\mathbf{q}} = \left[\dot{\mathbf{r}}_1^T, \dot{\epsilon}_1^T, \ldots, \dot{\mathbf{r}}_{n_b}^T, \dot{\epsilon}_{n_b}^T\right]^T \in \mathbb{R}^{7n_b},$$

where $\mathbf{r}_j$ is the absolute position of the center of mass of the j-th body and the quaternion ϵ_j expresses its rotation. One can also introduce the generalized velocities $\mathbf{v} = \left[\dot{\mathbf{r}}_1^T, \bar{\omega}_1^T, \ldots, \dot{\mathbf{r}}_{n_b}^T, \bar{\omega}_{n_b}^T\right]^T \in \mathbb{R}^{6n_b}$, directly related to $\dot{\mathbf{q}}$ by means of the linear mapping $\dot{\mathbf{q}} = \mathbf{L}(\mathbf{q})\mathbf{v}$ that transforms each angular velocity $\bar{\omega}_i$ (expressed in the local coordinates of the body) into the corresponding quaternion derivative $\dot{\epsilon}_i$ by means of the linear algebra formula $\dot{\epsilon}_i = \frac{1}{2}G(\epsilon_j)\bar{\omega}_i$, with

$$G(\epsilon_j) = \begin{bmatrix} +\epsilon_1 & +\epsilon_0 & -\epsilon_3 & +\epsilon_2 \\ +\epsilon_2 & +\epsilon_3 & +\epsilon_0 & -\epsilon_1 \\ +\epsilon_3 & -\epsilon_2 & +\epsilon_1 & +\epsilon_0 \end{bmatrix}.$$

Mechanical constraints, such as revolute or prismatic joints, can exist between the parts: they translate into algebraic equations that constrain the relative position of pairs of bodies. Assuming a set $\mathcal{B}$ of constraints is present in the system, they lead to the scalar equations

$$\Psi_i(\mathbf{q}, t) = 0, \quad i \in \mathcal{B}.$$

To ensure that constraints are not violated in terms of velocities, one must also satisfy the first derivative of the constraint equations, that is

$$\nabla\Psi_i^T\mathbf{v} + \frac{\partial\Psi_i}{\partial t} = 0, \quad i \in \mathcal{B}$$

with the Jacobian matrix $\nabla_q\Psi_i = [\partial\Psi_i/\partial\mathbf{q}]^T$ and $\nabla\Psi_i^T = \nabla_q\Psi_i^T\mathbf{L}(\mathbf{q})$. Note that the term $\partial\Psi_i/\partial t$ is null for all scleronomic constraints, but it might be nonzero for constraints that impose some trajectory or motion law, such as in the case of motors and actuators.

If contacts between rigid bodies must be taken into consideration, colliding shapes must be defined for each body. A collision detection algorithm must be used to provide a set of pairs of contact points for bodies whose shapes are near enough, so that a set $\mathcal{A}$ of inequalities can be used to concisely express the non-penetration condition between the volumes of the shapes:

$$\Phi_i(\mathbf{q}) \geq 0, \quad i \in \mathcal{A}$$

Note that for curved convex shapes, such as spheres and ellipsoids, there is a unique pair of contact points, that is the pair of closest points on their surfaces, but in case of faceted or non-convex shapes there might be multiple pairs of contact points, whose definition is not always trivial and whose set may be discontinuous.

Given two bodies in contact $A, B \in \{1, 2, \ldots, n_b\}$ let $\mathbf{n}_i$ be the normal at the contact pointing toward the exterior of body A, and let $\mathbf{u}_i$ and $\mathbf{w}_i$ be two vectors in the contact plane such that $\mathbf{n}_i, \mathbf{u}_i, \mathbf{w}_i \in \mathbb{R}^3$ are mutually orthogonal vectors. When a contact i is active, that is, for $\Phi_i(\mathbf{q}) = 0$, the frictional contact force acts on the system by means of multipliers $\widehat{\gamma}_{i,n} \geq 0$, $\widehat{\gamma}_{i,u}$, and $\widehat{\gamma}_{i,w}$. Specifically, the normal component of the contact force acting on body B is $\mathbf{F}_{i,N} = \widehat{\gamma}_{i,n}\mathbf{n}_i$ and the tangential component is $\mathbf{F}_{i,T} = \widehat{\gamma}_{i,u}\mathbf{u}_i + \widehat{\gamma}_{i,w}\mathbf{w}_i$ (for body A these forces have the opposite sign).

Also, according to the Coulomb friction model, in case of nonzero relative tangential speed, $\mathbf{v}_{i,T}$, the direction of the tangential contact force is aligned to $\mathbf{v}_{i,T}$ and it is proportional to the normal force as $||\mathbf{F}_{i,T}|| = \mu_{i,d}||\mathbf{F}_{i,N}||$ by means of the dynamic friction coefficient $\mu_{i,d} \in \mathbb{R}^+$. However, in case of null tangential speed, the strength of the tangential force is limited by the inequality $||\mathbf{F}_{i,T}|| \leq \mu_{i,s}||\mathbf{F}_{i,N}||$ using a static friction coefficient $\mu_{i,s} \in \mathbb{R}^+$, and its direction is one of the infinite tangents to the surface. In our model we assume that $\mu_{i,d}$ and $\mu_{i,s}$ have the same value that we will write μ_i for simplicity, so the abovementioned Coulomb model can be stated succinctly as follows:

$$\begin{aligned} \widehat{\gamma}_{i,n} &\geq 0, \quad \Phi_i(\mathbf{q}) \geq 0, \quad \Phi_i(\mathbf{q})\widehat{\gamma}_{i,n} = 0, \\ \mu_i\widehat{\gamma}_{i,n} &\geq \sqrt{\widehat{\gamma}_{i,u}^2 + \widehat{\gamma}_{i,w}^2} \\ &\quad \langle \mathbf{F}_{i,T}, \mathbf{v}_{i,T} \rangle = -\left\|\mathbf{F}_{i,T}\right\| \left\|\mathbf{v}_{i,T}\right\| \\ &\quad \left\|\mathbf{v}_{i,T}\right\| \left(\mu_i\widehat{\gamma}_{i,n} - \sqrt{\widehat{\gamma}_{i,u}^2 + \widehat{\gamma}_{i,w}^2}\right) = 0 \end{aligned}$$

Note that the condition $\widehat{\gamma}_{i,n} \geq 0, \Phi_i(\mathbf{q}) \geq 0, \Phi_i(\mathbf{q})\widehat{\gamma}_{i,n} = 0$ can also be written as a complementarity constraint: $\widehat{\gamma}_{i,n} \geq 0 \perp \Phi_i(\mathbf{q}) \geq 0$, see [2]. This model can also be interpreted as the Karush-Kuhn-Tucker first order conditions of the following equivalent maximum dissipation principle [3, 4]:

$$(\widehat{\gamma}_{i,u}, \widehat{\gamma}_{i,w}) = \underset{\sqrt{\widehat{\gamma}_{i,u}^2 + \widehat{\gamma}_{i,w}^2} \leq \mu_i\widehat{\gamma}_{i,n}}{\operatorname{argmin}} \mathbf{v}_{i,T}^T \left(\widehat{\gamma}_{i,u}\mathbf{u}_i + \widehat{\gamma}_{i,w}\mathbf{w}_i\right). \tag{1}$$

Finally, one should also consider the effect of external forces with the vector of generalized forces $\mathbf{f}(t, \mathbf{q}, \mathbf{v}) \in \mathbb{R}^{6n_b}$, that might contain gyroscopic terms, gravitational effects, forces exerted by springs or dampers, and torques applied by motors; i.e. all forces except joint reaction and frictional contact forces.

Considering the effects of both the set $\mathcal{A}$ of frictional contacts and the set $\mathcal{B}$ of bilateral constraints, the system cannot be reduced to either a set ordinary differential equations (ODEs) of the type $\dot{\mathbf{v}} = f(\mathbf{q}, \mathbf{v}, t)$, or to a set of differential-algebraic equation (DAEs). This is because the inequalities and the complementarity constraints turn the system into a differential inclusion of the type $\dot{\mathbf{v}} \in \mathcal{F}(\mathbf{q}, \mathbf{v}, t)$, where $\mathcal{F}(\cdot)$ is a set-valued multifunction [5]. In fact, the time evolution of the dynamical system is governed by the following differential variational inequality (DVI):

$$
\begin{array}{rcl}
\dot{\mathbf{q}} & = & \mathbf{L}(\mathbf{q})\mathbf{v} \\
\mathbf{M}\dot{\mathbf{v}} & = & \mathbf{f}\left(t,\mathbf{q},\mathbf{v}\right) + \sum\limits_{i\in\mathcal{B}} \widehat{\gamma}_{i,b}\nabla\Psi_i + \\
 & & + \sum\limits_{i\in\mathcal{A}} \left(\widehat{\gamma}_{i,n}\,\mathbf{D}_{i,n} + \widehat{\gamma}_{i,u}\,\mathbf{D}_{i,u} + \widehat{\gamma}_{i,w}\,\mathbf{D}_{i,w}\right) \\
i\in\mathcal{B} & : & \Psi_i(\mathbf{q},t) = 0 \\
i\in\mathcal{A} & : & \widehat{\gamma}_{i,n} \geq 0 \;\perp\; \Phi_i(\mathbf{q}) \geq 0, \qquad \text{and} \\
(\widehat{\gamma}_{i,u},\widehat{\gamma}_{i,w}) & = & \underset{\mu_i\widehat{\gamma}_{i,n}\geq\sqrt{\widehat{\gamma}_{i,u}^2+\widehat{\gamma}_{i,w}^2}}{\operatorname{argmin}} \mathbf{v}^T\left(\widehat{\gamma}_{i,u}\,\mathbf{D}_{i,u} + \widehat{\gamma}_{i,w}\,\mathbf{D}_{i,w}\right)
\end{array}
\tag{2}
$$

Here, to express the contact forces in generalized coordinates, we used the tangent space generators $D_i = [\,\mathbf{D}_{i,n},\,\mathbf{D}_{i,u},\,\mathbf{D}_{i,w}] \in \mathbb{R}^{6n_b\times 3}$ that are sparse and are defined given a pair of contacting bodies A and B as:

$$
D_i^T = \begin{bmatrix} \mathbf{0} & \ldots & -A_{i,p}^T & A_{i,p}^T A_A \tilde{\bar{\mathbf{s}}}_{i,A} & \mathbf{0} & \ldots \\ \mathbf{0} & \ldots & A_{i,p}^T & -A_{i,p}^T A_B \tilde{\bar{\mathbf{s}}}_{i,B} & \mathbf{0} & \ldots \end{bmatrix}
\tag{3}
$$

Here $A_{i,p} = [\mathbf{n}_i, \mathbf{u}_i, \mathbf{w}_i]$ is the $\mathbb{R}^{3\times 3}$ matrix of the local coordinates of the i-th contact, and the vectors $\bar{\mathbf{s}}_{i,A}$ and $\bar{\mathbf{s}}_{i,B}$ to represent the positions of the contact points expressed in body coordinates. The skew matrices $\tilde{\bar{\mathbf{s}}}_{i,A}$ and $\tilde{\bar{\mathbf{s}}}_{i,B}$ are defined as

$$
\tilde{\bar{\mathbf{s}}}_{i,A} = \begin{bmatrix} 0 & -s_{i,A_z} & +s_{i,A_y} \\ +s_{i,A_z} & 0 & -s_{i,A_x} \\ -s_{i,A_y} & +s_{i,A_x} & 0 \end{bmatrix}
$$

$$
\tilde{\bar{\mathbf{s}}}_{i,B} = \begin{bmatrix} 0 & -s_{i,B_z} & +s_{i,B_y} \\ +s_{i,B_z} & 0 & -s_{i,B_x} \\ -s_{i,B_y} & +s_{i,B_x} & 0 \end{bmatrix}
$$

The DVI in (2) can be solved by time-stepping methods. The discretization requires the solution of a complementarity problem at each time step, and it has been demonstrated that it converges to the solution to the original differential inclusion for $h \to 0$ [2, 6]. Moreover, the differential inclusion can be solved in terms of vector measures: forces can be impulsive and velocities can have discontinuities, thus supporting also the case of impacts and giving a weak solution to otherwise unsolvable situations like in the Painlevé paradox [7].

The Time Stepping Solver

Within the aforementioned measure differential inclusion approach, the unknowns are not the reaction forces and the accelerations $\dot{\mathbf{v}}$ as in usual ODEs or DAEs. Instead, given a position $\mathbf{q}^{(l)}$ and velocity $\mathbf{v}^{(l)}$ at the time step $t^{(l)}$, the unknowns are the impulses γ_s, for $s = n, u, w, b$ (that, for smooth constraints, can be interpreted as $\hat{\gamma}_n = h\gamma_n$, $\hat{\gamma}_u = h\gamma_u$, $\hat{\gamma}_w = h\gamma_w$, $\hat{\gamma}_b = h\gamma_b$) and the speeds $\mathbf{v}^{(l+1)}$ at the new time step $t^{(l+1)} = t^{(l)} + h$. These unknowns are obtained by solving the following optimization problem with equilibrium constraints [8]:

$$
\begin{array}{rcl}
\mathbf{M}(\mathbf{v}^{(l+1)} - \mathbf{v}^{(l)}) = h\mathbf{f}(t^{(l)},\mathbf{q}^{(l)},\mathbf{v}^{(l)}) + \sum\limits_{i\in\mathcal{B}} \gamma_{i,b}\nabla\Psi_i + \\
+ \sum_{i\in\mathcal{A}} \left(\gamma_{i,n}\,\mathbf{D}_{i,n} + \gamma_{i,u}\,\mathbf{D}_{i,u} + \gamma_{i,w}\,\mathbf{D}_{i,w}\right), \\
i\in\mathcal{B}: \quad \frac{1}{h}\Psi_i(\mathbf{q}^{(l)},t) + \nabla\Psi_i^T\mathbf{v}^{(l+1)} + \frac{\partial\Psi_i}{\partial t} = 0 \\
i\in\mathcal{A}: \quad 0 \leq \frac{1}{h}\Phi_i(\mathbf{q}^{(l)}) + \mathbf{D}_{i,n}^T\mathbf{v}^{(l+1)} \perp \gamma_n^i \geq 0, \\
(\gamma_{i,u},\gamma_{i,w}) = \underset{\mu_i\gamma_{i,n}\geq\sqrt{\gamma_{i,u}^2+\gamma_{i,w}^2}}{\operatorname{argmin}} \mathbf{v}^T\left(\gamma_{i,u}\,\mathbf{D}_{i,u} + \gamma_{i,w}\,\mathbf{D}_{i,w}\right) \\
\mathbf{q}^{(l+1)} = \mathbf{q}^{(l)} + h\mathbf{L}(\mathbf{q}^{(l)})\mathbf{v}^{(l+1)}.
\end{array}
$$

The $\frac{1}{h}\Phi_i(\mathbf{q}^{(l)})$ term is introduced to ensure contact stabilization, and its effect is discussed in [9]. Similarly, the term $\frac{1}{h}\Psi_i(\mathbf{q}^{(l)})$ achieves stabilization for bilateral constraints.

Several numerical methods can be used to solve (4). For instance, one can approximate the Coulomb friction cones in 3D as faceted pyramids, thus leading to a LCP whose solution is possible by using off-the-shelf pivoting methods. However, these methods usually require a large computational overhead and can be used only for a limited number of variables.

Therefore, in a previous work [10] we demonstrated that the problem can be cast as a monotone optimization problem by introducing a relaxation over the complementarity constraints, replacing $0 \leq \frac{1}{h}\Phi_i(\mathbf{q}^{(l)}) + \mathbf{D}_{i,n}^T\mathbf{v}^{(l+1)} \perp \gamma_n^i \geq 0$ with $0 \leq \frac{1}{h}\Phi_i(\mathbf{q}^{(l)}) + \mathbf{D}_{i,n}^T\mathbf{v}^{(l+1)} - \mu_i\sqrt{(\mathbf{v}^T\mathbf{D}_{i,u})^2 + (\mathbf{v}^T\mathbf{D}_{i,w})^2} \perp \gamma_n^i \geq 0$. The solution of the modified time stepping scheme approaches the solution of the original differential inclusion for $h \to 0$ just as the original scheme [9]. Most importantly, the modified scheme becomes a Cone Complementarity Problem (CCP), which can be solved efficiently by an iterative numerical method that relies on projected contractive maps. Omitting for brevity some of the details discussed in [11], the algorithm makes use of the following vectors and matrices:

$$
\begin{array}{rcll}
\gamma_{i,a} & \equiv & \{\gamma_{i,n},\gamma_{i,u},\gamma_{i,w}\}^T, & i\in\mathcal{A}, \\
\mathbf{b}_i & \equiv & \left\{\frac{1}{h}\Phi_i(\mathbf{q}^{(l)}),0,0\right\}^T, & i\in\mathcal{A}, \\
b_i & \equiv & \frac{1}{h}\Psi_i(\mathbf{q}^{(l)},t) + \frac{\partial\Psi_i}{\partial t}, & i\in\mathcal{B}
\end{array}
$$

The solution of the CCP is obtained by iterating the following expressions on r until convergence, or until r exceeds a maximum amount of iterations, starting from $\mathbf{v}^0 = \mathbf{v}^{(l)}$:

$$
\forall i\in\mathcal{A}: \quad \gamma_{i,a}^{r+1} = \Pi_{\Upsilon_i}\left[\gamma_{i,a}^r - \omega\eta_i\left(D_i^T\mathbf{v}^r + \mathbf{b}_i\right)\right] \tag{6}
$$

$$
\forall i\in\mathcal{B}: \quad \gamma_{i,b}^{r+1} = \gamma_{i,b}^r - \omega\eta_i\left(\nabla\Psi_i^T\mathbf{v}^r + b_i\right) \tag{7}
$$

$$
\begin{array}{rcl}
\mathbf{v}^{r+1} & = & \mathbf{v}^r + M^{-1}\left(\sum_{z\in\mathcal{A}} D_z\gamma_{z,a}^{r+1} + \sum_{z\in\mathcal{B}}\nabla\Psi_z\gamma_{z,b}^{r+1}\right. \\
 & + & \left. h\,\mathbf{f}(t^{(l)},\mathbf{q}^{(l)},\mathbf{v}^{(l)})\right)
\end{array}
\tag{8}
$$

Note that the superscript $(l+1)$ was omitted for brevity.

The iterative process uses the projector $\Pi_{\Upsilon_i}(\cdot)$, which is a non-expansive metric map $\Pi_{\Upsilon_i} : \mathbb{R}^3 \to \mathbb{R}^3$ acting on the triplet

of multipliers associated with the i-th contact [10]. In detail, if the multipliers fall into the friction cone

$$\Upsilon_i = \{\gamma_{i,a} \in \mathbb{R}^3 \ : \ (\gamma_{i,u}^2 + \gamma_{i,w}^2)^{1/2} \leq \mu_i \gamma_{i,n}\}$$

they are not modified; if they are in the polar cone

$$\Upsilon_i^\circ = \{\mathbf{x}_i \in \mathbb{R}^3 \ : \ \langle \mathbf{x}_i, \gamma_{i,a} \rangle \leq 0, \ \forall \gamma_{i,a} \in \Upsilon_i\}$$

they are set to zero; in the remaining cases they are projected orthogonally onto the surface of the friction cone. The over-relaxation factor ω and η_i parameters are adjusted to control the convergence. Interested readers are referred to [11] for a proof of the convergence of this method.

For improved performance, the summation of Eq. (8) can be computed only once at the beginning of the CCP iteration, while the following updates can be performed using an incremental version that avoids adding the $\mathbf{f}(t^{(l)}, \mathbf{q}^{(l)}, \mathbf{v}^{(l)})$ term all the time; in case there is no initial guess for the multipliers and $\gamma_{i,b}^0 = 0$, $\gamma_{i,a}^0 = 0$, Eq. (8) turns into:

$$\mathbf{v}^0 = \mathbf{v}^{(l)} + M^{-1} h \, \mathbf{f}(t^{(l)}, \mathbf{q}^{(l)}, \mathbf{v}^{(l)})$$

$$\mathbf{v}^{r+1} = \mathbf{v}^r + \sum \Delta \mathbf{v}_i$$

where

$$i \in \mathcal{A}: \quad \Delta \mathbf{v}_i = \sum_{i \in \mathcal{A}} M^{-1} D_i \Delta \gamma_{i,a}^{r+1}$$

$$i \in \mathcal{B}: \quad \Delta \mathbf{v}_i = \sum_{i \in \mathcal{B}} M^{-1} \nabla \Psi_i \Delta \gamma_{i,b}^{r+1}$$

In the case that only bilateral constraints are used, this method behaves like the typical fixed-point Jacobi iteration for the solution of linear problems. If one interleaves the update (8) after each time that a single *i*-th multiplier is computed in (6) or (7), the resulting scheme behaves like a Gauss-Seidel method. This variant can benefit from the use of Eq. (10) instead of Eq. (8) because it can increment only the $\Delta \mathbf{v}_i$ term corresponding to the constraint that has been just computed. Also, this immediate update of the speed vector provides better properties of convergence (especially in case of redundant constraints) but it does not fit well in a parallel computing environment because of its inherently sequential nature.

The GPU Formulation of the CCP Solver

Since the CCP iteration is a computational bottleneck of the numerical solution proposed, a great benefit will follow from an implementation that can take advantage of the parallel computing resources available on GPU boards.

In the proposed approach, the data structures on the GPU are implemented as large arrays (*buffers*) to match the execution model associated with NVIDIA's CUDA. Specifically, threads are grouped in rectangular thread blocks, and thread blocks are arranged in rectangular grids. Four main buffers are used: the contacts buffer, the constraints buffer, the reduction buffer, and the bodies buffer. Since repeated transfers of large data structures can adversely impact the performance of the entire algorithm, an attempt was made to organize the data structures in a way that minimized the number of fetch and store operations and maximized the arithmetic intensity of the kernel code. This ensures that the latency of the global memory can be hidden by the hardware multithread scheduler if the GPU code interleaves the memory access with enough arithmetic instructions.

Figure 1 shows the data structure for contacts, which contains two pointers B_A and B_B to the two touching bodies. There is no need to store the entire D_i matrix for the i-th contact because it has zero entries everywhere except for the two 12x3 blocks corresponding to the coordinates of the two bodies in contact. In detail, we store only the following 3x3 matrices:

$$\begin{array}{rclrcl} D_{i,v_A}^T & = & -A_{i,p}^T, & D_{i,\omega_A}^T & = & A_{i,p}^T A_A \tilde{\mathbf{s}}_{i,A} \\ D_{i,v_B}^T & = & A_{i,p}^T, & D_{i,\omega_B}^T & = & -A_{i,p}^T A_B \tilde{\mathbf{s}}_{i,B} \end{array}$$

Once the velocities of the two bodies $\dot{\mathbf{r}}_{A_i}$, $\bar{\omega}_{A_i}$, $\dot{\mathbf{r}}_{B_i}$ and $\bar{\omega}_{B_i}$ have been fetched, the product $D_i^T \mathbf{v}^r$ in Eq. (6) can be performed as

$$D_i^T \mathbf{v}^r = D_{i,v_A}^T \dot{\mathbf{r}}_{A_i} + D_{i,\omega_A}^T \bar{\omega}_{A_i} + D_{i,v_B}^T \dot{\mathbf{r}}_{B_i} + D_{i,\omega_B}^T \bar{\omega}_{B_i} \tag{11}$$

Since $D_{i,v_A}^T = -D_{i,v_B}^T$, there is no need to store both matrices, so in each contact data structure only a matrix $D_{i,v_{AB}}^T$ is stored, which is then used with opposite signs for each of the two bodies.

Also, the velocity update vector $\Delta \mathbf{v}_i$, needed for the sum in Eq. (10) is sparse: it can be decomposed in small 3x1 vectors. Specifically, given the masses and the inertia tensors of the two bodies m_{A_i}, m_{B_i}, J_{A_i} and J_{B_i}, the term $\Delta \mathbf{v}_i$ will be computed and stored in four parts as follows:

$$\begin{array}{rclrcl} \Delta \dot{\mathbf{r}}_{A_i} & = & m_{A_i}^{-1} D_{i,v_A} \Delta \gamma_{i,a}^{r+1}, & \Delta \bar{\omega}_{A_i} & = & J_{A_i}^{-1} D_{i,\omega_A} \Delta \gamma_{i,a}^{r+1} \\ \Delta \dot{\mathbf{r}}_{B_i} & = & m_{B_i}^{-1} D_{i,v_B} \Delta \gamma_{i,a}^{r+1}, & \Delta \bar{\omega}_{B_i} & = & J_{B_i}^{-1} D_{i,\omega_B} \Delta \gamma_{i,a}^{r+1} \end{array} \tag{12}$$

Note that those four parts of the $\Delta \mathbf{v}_i$ terms are not stored in the i-th contact or data structures of the two referenced bodies (because multiple contacts may refer the same body, hence they would overwrite the same memory position). These velocity updates are instead stored in the reduction buffer, which will be used to efficiently perform the summation in Eq. (10). This will be discussed shortly.

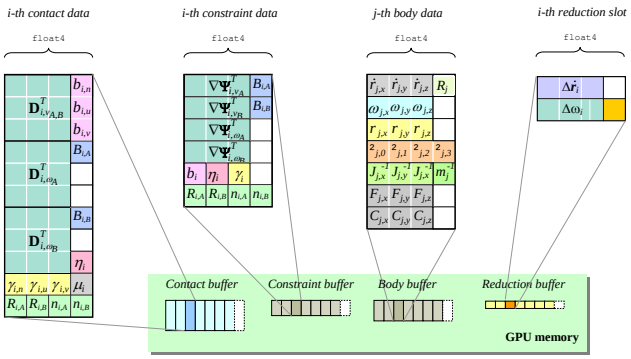

Figure 1: DATA STRUCTURES IN GPU GLOBAL MEMORY

The constraints buffer, shown in Figure 14, is based on a similar concept. Jacobians $\nabla\Psi_i$ of all scalar constraints are stored in a sparse format, each corresponding to four rows $\nabla\Psi_{i,v_A}$, $\nabla\Psi_{i,\omega_A}$, $\nabla\Psi_{i,v_B}$, $\nabla\Psi_{i,\omega_B}$. Therefore the product $\nabla\Psi_i^T\mathbf{v}^r$ in Eq. (7) can be performed as the scalar value:

$$\begin{aligned}\nabla\Psi_i^T\mathbf{v}^r &= \nabla\Psi_{i,v_A}^T\dot{\mathbf{r}}_{A_i} + \nabla\Psi_{i,\omega_A}^T\omega_{A_i} + \nabla\Psi_{i,v_B}^T\dot{\mathbf{r}}_{B_i} \\ &+ \nabla\Psi_{i,\omega_B}^T\omega_{B_i}\end{aligned} \quad (13)$$

Also, the four parts of the sparse vector $\Delta\mathbf{v}_i$ can be computed and stored as

$$\begin{aligned}\Delta\dot{\mathbf{r}}_{A_i} &= m_{A_i}^{-1}\nabla\Psi_{i,v_A}\Delta\gamma_{i,b}^{r+1}, \quad \Delta\bar{\omega}_{A_i} = J_{A_i}^{-1}\nabla\Psi_{i,\omega_A}\Delta\gamma_{i,b}^{r+1} \\ \Delta\dot{\mathbf{r}}_{B_i} &= m_{B_i}^{-1}\nabla\Psi_{i,v_B}\Delta\gamma_{i,b}^{r+1}, \quad \Delta\bar{\omega}_{B_i} = J_{B_i}^{-1}\nabla\Psi_{i,\omega_B}\Delta\gamma_{i,b}^{r+1}\end{aligned} \quad (14)$$

Figure 1 shows that each body is represented by a data structure containing the state (velocity and position), the mass moments of inertia and mass values, and the external applied force $\mathbf{F}_j$ and torque $\mathbf{C}_j$. Those data are needed to compute the CCP iteration and solve for unknowns.

When it comes to the implementation of the CCP solver on the GPU, using kernels that operate on the abovementioned data buffers, the task is not trivial because the iteration cannot be performed with a single kernel. In fact, considering the iteration over Eqs. (6), (7), and (10), one can see that Eqs. (6) and (7) fit into parallel kernels that operate, respectively, one thread per contact and one thread per bilateral constraint. Moreover, the summation in Eq. (10) cannot be easily parallelized in the same way because it may happen that two or more contacts need to add their velocity updates $\Delta\mathbf{v}_i$ to the same rigid body: this would cause a race condition where multiple threads might need to update the same memory value, something that can cause errors or indefinite/nondeterministic behaviors on the

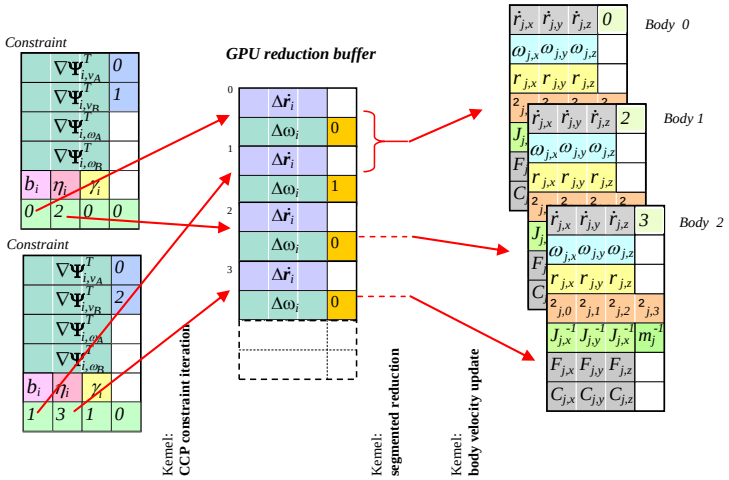

Figure 2: EXAMPLE OF REDUCTION BUFFER FOR SUMMING UP BODY VELOCITIES

GPU hardware. Therefore, in order to parallelize Eq. (10), a parallel segmented scan algorithm [12] was adopted that operates on an intermediate reduction buffer (see Figure 2); this method sums the values in the buffer using a binary-tree approach that keeps the computational load well balanced among the many processors. In the example of Figure 2, the first constraint refers to bodies 0 and 1, the second to bodies 0 and 2; multiple updates to body 0 are then accumulated with parallel a segmented reduction.

Note that several other auxiliary kernels that have minimal impact on the computation time are used to prepare pre-process data before the CCP starts, for example to compute Eq. (9). Also, to speed up the computation, matrices D_{i,v_A}^T, D_{i,ω_A}^T and D_{i,ω_B}^T are not provided by the host; instead they are computed on the GPU using the data coming from the collision detection code, that is, $\bar{\mathbf{s}}_{i,A}$, $\bar{\mathbf{s}}_{i,B}$ and $\mathbf{n}_i$.

The following pseudocode shows the sequence of the main computational stages at each time step, which for the most part are executed as parallel kernels on the GPU.

Table 1: Pseudocode for the CCP solver.

Stage	Context	Operations / kernels
1	HOST *Serial*	**Copy memory CPU→ GPU** Copy contact and body data structures from host memory to GPU buffers. Copy constraint data (residuals b_i and Jacobians) into the constraint buffer.
2	GPU *Parallel on* ***bodies***	**Force kernel** For each body, compute forces $\mathbf{f}(t^{(l)}, \mathbf{q}^{(l)}, \mathbf{v}^{(l)})$, if any. Store these forces and torques into $\mathbf{F}_j$ and $\mathbf{C}_j$.

3	GPU *Parallel on* ***contacts***	**Contact preprocessing kernel** For each contact, given contact normal and position, compute in place the matrices D^T_{i,v_A}, D^T_{i,ω_A} and D^T_{i,ω_B}, then compute η_i and the contact residual $\mathbf{b}_i = \{\frac{1}{h}\Phi_i(\mathbf{q}), 0, 0\}^T$.
4	GPU *Parallel on* ***bodies***	**CCP force kernel** For each body j, initialize body velocities: $\dot{\mathbf{r}}_j^{(l+1)} = h\, m_j^{-1}\mathbf{F}_j$ and $\bar{\omega}_j^{(l+1)} = h\, J_j^{-1}\mathbf{C}_j$.
5	GPU *Parallel on* ***contacts***	**CCP contact iteration kernel** For each contact i, do $\gamma_{i,a}^{r+1} = \Pi_{\Upsilon_i}[\gamma_{i,a}^r - \omega\eta_i\left(D_i^T\mathbf{v}^r + \mathbf{b}_i\right)]$. Note that $D_i^T\mathbf{v}^r$ is evaluated with sparse data, using Eq.(11). Store $\Delta\gamma_{i,a}^{r+1} = \gamma_{i,a}^{r+1} - \gamma_{i,a}^r$ in contact buffer. Use Eq.(12) to compute sparse updates $\Delta\dot{\mathbf{r}}$ and $\Delta\bar{\omega}$ to the velocities of the two connected bodies A and B, and store them in the $R_{i,A}$ and $R_{i,B}$ slots of the reduction buffer.
6	GPU *Parallel on* ***constraints***	**CCP constraint iteration kernel**. For each constraint i, do $\gamma_{i,b}^{r+1} = \gamma_{i,b}^r - \omega\eta_i\left(\nabla\Psi_i^T\mathbf{v}^r + b_i\right)$. Note that $\nabla\Psi_i^T\mathbf{v}^r$ is evaluated with sparse data, using Eq.(11). Store $\Delta\gamma_{i,b}^{r+1} = \gamma_{i,b}^{r+1} - \gamma_{i,b}^r$ in contact buffer. Use Eq.(14) to compute sparse updates $\Delta\dot{\mathbf{r}}$ and $\Delta\bar{\omega}$ to the velocities of the two connected bodies A and B, and store them in the $R_{i,A}$ and $R_{i,B}$ slots of the reduction buffer.
7	GPU *Parallel on* ***reduction slots***	**Segmented reduction kernel**. Sum all the $\Delta\dot{\mathbf{r}}$ and $\Delta\bar{\omega}$ terms belonging to the same body, in the reduction buffer. This may require a sequence of short kernels.
8	GPU *Parallel on* ***bodies***	**Body velocity updates kernel**. For each j body, add the cumulative velocity updates which can be fetched from the reduction buffer, using the index R_j.
9	HOST *Serial*	*Check convergence and repeat from stage 5 if convergence tolerance is not reached*
10	HOST *Serial*	**Copy memory GPU→ CPU** Copy contact multipliers from GPU buffers to host memory, if interested in reaction forces. Copy constraints multipliers from GPU buffers to host memory, if interested in reaction forces. Copy rigid body velocities from GPU buffers to host memory.

Stages 1 and 10 can be avoided if one manages to keep all the data on the GPU, by letting the collision detection engine communicate with the CCP solver directly. Even if those memory transfers are executed only at the beginning and at the end of the CCP solution process, their impact on the overall simulation time might be significant.

A SETUP OF THE SIMULATION IN CHRONO::ENGINE

In order to validate the experimental results with the DVI Approach it is necessary to build a simulation model which represents the actual setup as detailed as possible. The experimental setup consists of four major parts, the trough, the spheres, the translational stage and the load cell. Figure 3 shows these parts as they are implemented in the simulation model.

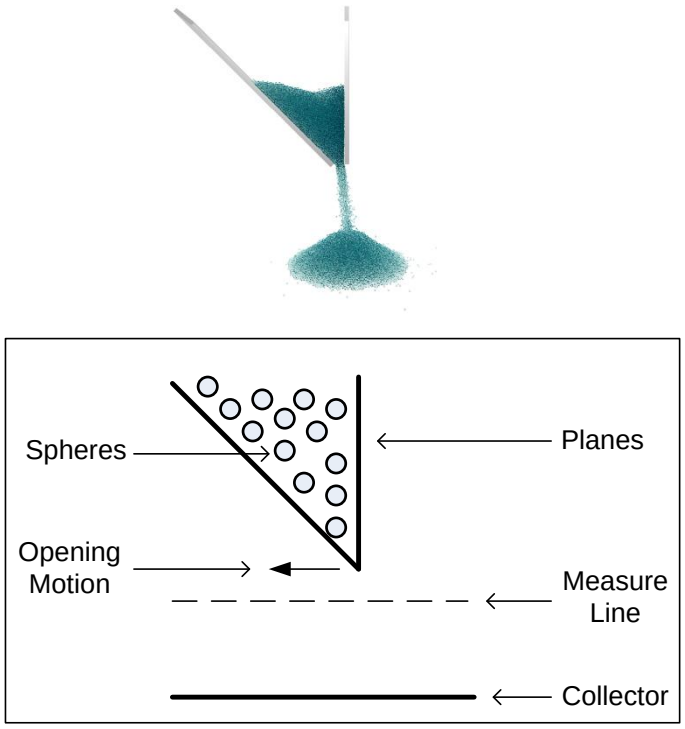

Figure 3. SIMULATION CONSTRUCTION

The glass spheres are described as perfect spheres. Each sphere has the same mass and friction coefficients as the other ones.

The trough, with its three walls and slope, is integrated as a mathematical description of planes. Each plane represents one wall or slope respectively. All planes are part of one body. This is necessary to give the planes certain properties such as friction and is also important for the collision detection. Not only the spheres need to be checked for collision between each other but each sphere also needs to be checked for collision with the planes.

In the experiment the trough is opened by a translational stage. This stage moves the slope out of the holder. In the simulation model this behavior is described as a motion of the plane representing the slope. The motion is captured from the data sheet of the translational stage. Different gap sizes can be adjusted for each simulation and the opening time varied.

The load cell measures the outflow through the gap. Because this measurement is based on the weight per time it can be implemented as a count of spheres under a certain height for each time step and be outputted into a data file for later post processing. The number of spheres multiplied by the mass and gravity provides the weight which can be compared with experimental results.

The spheres leaving the trough are falling on a plane described as the collector in the experimental setup.

In order to save computational time the simulation is split into two parts, one containing the filling process of the trough and the other one the opening and measure process. If the same design of the trough is observed multiple times, for example at different gap sizes, the filling process is always equal. Therefore it is unnecessary to compute this process for each observation and it only needs to be run once. The second part then can be used for observation of the behavior.

In the filling process the same amount of spheres will be created as used in the real experiment and randomly distributed in the trough. After the spheres have settled, the x-, y- , and z-position of each sphere is saved for the following simulation.

At the beginning of the outflow simulation the position data set of the spheres is loaded into the model and the spheres will be created at the same positions they appeared in the filling process. The material properties of the spheres can be set as desired. The positions of each sphere can be saved at each time step for further post processing observation and the outflow is also saved at any time step.

Approximately 39000 spheres were used in both simulations for the representation of the behavior of the trough at a 45° slope and a step size of 5e-4 seconds.

PROCEDURE USED TO DETERMINE THE FRICTION COEFFICIENT

The friction coefficient of a certain material is not a constant value. It can depend on various environmental factors such as humidity, surface quality, temperature etc. The friction coefficient of glass used in the experiment was an unknown in the validation process and needed to be determined before further observations could be done.

To achieve this, one experiment at a given gap size was performed and multiple simulations with the same setup and different friction coefficients were carried out. The effects of altering the friction coefficient on the flow rate can be seen in Figure 4, with the experimental data overlaid on the simulation result.

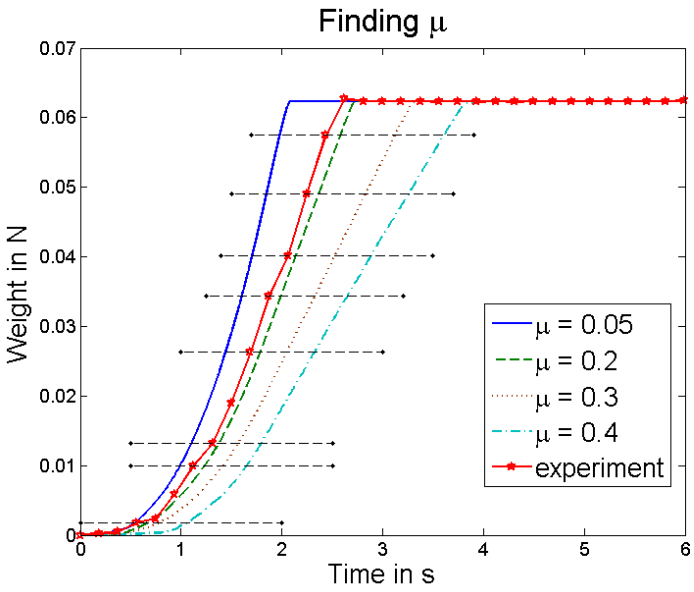

Figure 4. SELECTION OF FRICTION COEFFICIENT

This set of observation was done for a gap size of 2.5 mm. Four simulations were run with friction coefficients varying from 0.05 up to 0.4.

Each simulation reaches a certain weight at a different time. Looking at the above graph at fixed values of weight, it is reasonable that the time needed to reach a certain value depends on the friction. The black lines indicate the different values chosen for examination. For each line, a second order polynomial can be fit to the data points and be evaluated for the experimental time value. The mean of these resulting eight values is 0.1498. The friction coefficient of the granular material was chosen to be 0.15.

EQUIPMENT AND PROCEDURE

This section focuses on the different components of the equipment used for the validation experiment.

Description of Design with Iterations

Three criteria were considered essential for proper functionality of the granular flow meter:

- The granular material must be contained in the cavity and leave only through the slit.
- The walls of the cavity must have a similar friction coefficient to the granular material.
- The shape of the cavity must be easily altered.

In an effort to achieve these design goals, several versions of the flow meter were created.

Final Design. The final design closely resembles its predecessors, however the dimensions have been altered to

maximize the aluminum stock that was made available by the University of Wisconsin-Madison Student Shop. The height of the entire assembly was increased to ensure sufficient room for the load cell.

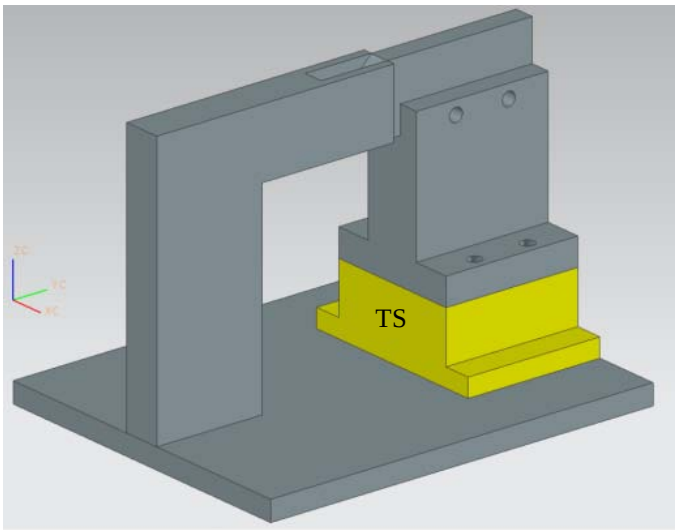

Figure 5. FINAL DESIGN OF THE EXPERIMENTAL SETUP

Assembly. The final assembly, including the translational stage, actuator, and load cell, is shown below. To complete the assembly, ¼"-20 bolts were used to fasten the components together.

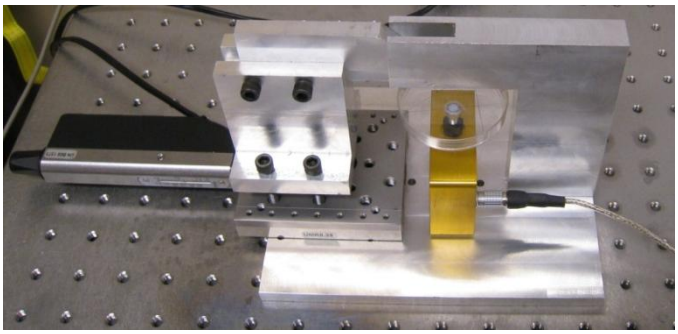

Figure 6. FINAL ASSEMBLY, INCLUDING THE TRANSLATIONAL STAGE, ACTUATOR, AND LOAD CELL

Description of Translational Stage

The Newport UMR8.25 Linear Translational Stage [13] was used to constrain the motion of the angled insert. A schematic of the UMR8.25 is shown in Figure 7**Error! Reference source not found.**.

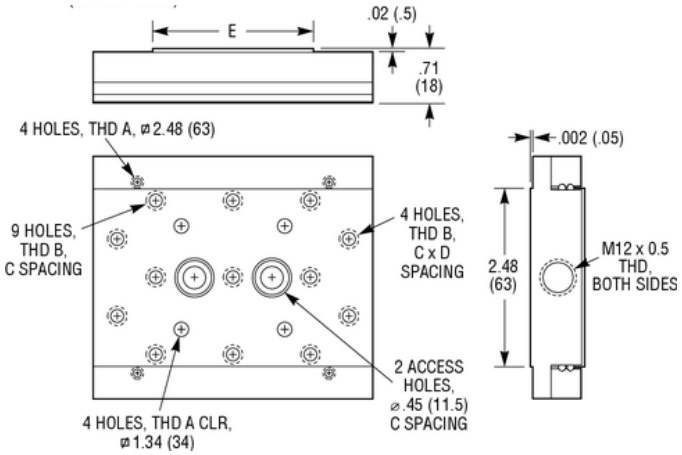

Figure 7. SCHEMATIC OF THE UMR8.25 TRANSLATIONAL STAGE

Description of Actuator

The actuator used in this experiment was the Newport LTA-HL Precision Linear Actuator. It has a 25mm range and a minimum incremental motion of 0.05 microns. The Newport LTA-HL features a manual knob that allows the user to match the actuator to the zero position of the rig. Figure 8 shows a picture of the actuator used in this experiment.

Figure 8. THE NEWPORT LTA-HL PRECISION LINEAR ACTUATOR

Description of Load Cell

The load cell used in this experiment is a Cooper LFS242 Tension/Compression Cell (Serial No. 286284) [14] with a maximum load of .25 lbs and a repeatability of ±0.05%. Although load cells typically work better in tension, the load cell was used in compression for this experiment. Several different mass displacements were tested to ensure that irregular piling did not cause an error in the reading. A Cooper DFI Infinity Digital Indicator [15] was used to send the load cell signal through a router and to the processor.

Schematic of Experimental Setup and Data Transmission

Data sent from the load cell was interpreted using a script written in MATLAB. The script took an initial reading from the load cell and considered that value to be zero. A representation of the data flow can be seen below:

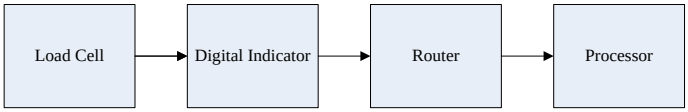

Figure 9. FLOW CHART OF THE DATA TRANSMISSION. WEIGHT IS DETERMINED BY THE LOAD CELL AND EVENTUALLY SENT TO THE PROCESSOR TO BE INTERPRETED BY MATLAB

An analog signal is generated by the load cell which is converted into a digital output by the indicator. The digital signal is sent to the router which is then sent to the processor to be analyzed by MATLAB. The signal, that is, sand weight as a function of time, is read at 6 Hz.

SUMMARY OF EXPERIMENTAL RESULTS

Due to the ease of which the actuator could alter the gap size of the flow meter, several experimental runs were taken for varying gap sizes. Since the actuator measures distance with a unit called steps, it was necessary to measure the desired gap size using a digital caliper. A fixed weight of 0.0624 N of granular material was used for each test with an uncertainty of 0.0004 N due to measurement error. Based on the results of these experiments, an allowable "bandwidth" could be created with which the simulation results could be compared to.

Results of 3mm Gap

Shown below are the experimental results with a gap size of 3mm. Nine experimental runs were completed.

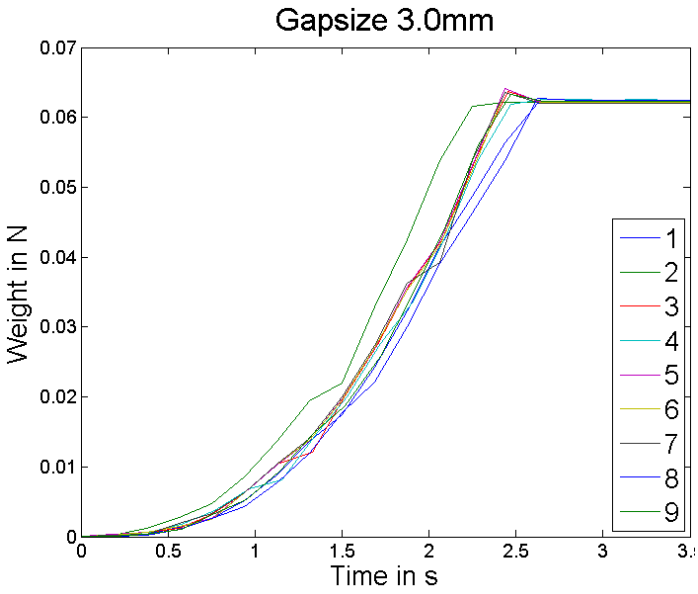

Figure 10. WEIGHT VERSUS TIME EXPERIMENTAL DATA FOR GRANULAR FLOW WITH A 3MM GAP

Results of 2.5mm Gap

Shown below are the experimental results with a gap size of 2.5mm. Eight experimental runs were completed.

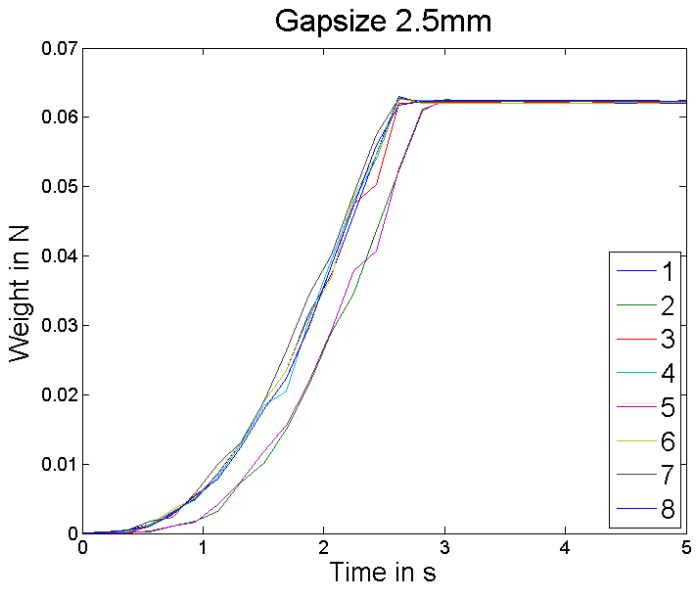

Figure 11. WEIGHT VERSUS TIME EXPERIMENTAL DATA FOR GRANULAR FLOW WITH A 2.5MM GAP

Results of 2mm Gap

Shown below are the experimental results with a gap size of 2mm. Nine experimental runs were completed.

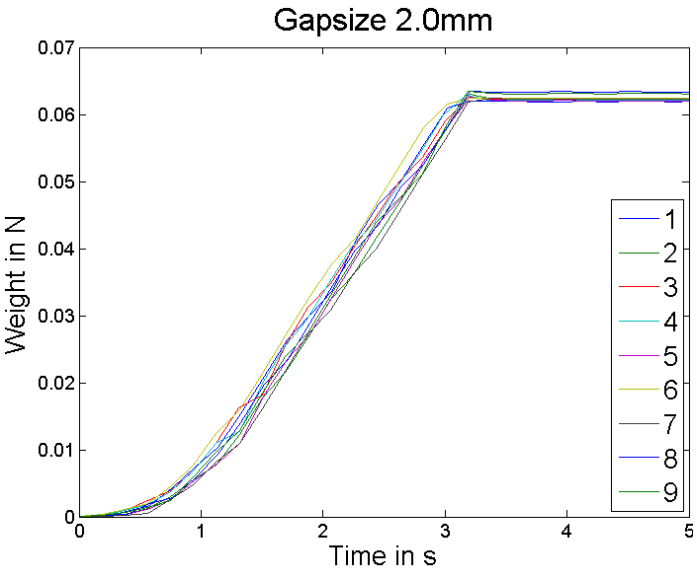

Figure 12. WEIGHT VERSUS TIME EXPERIMENTAL DATA FOR GRANULAR FLOW WITH A 2MM GAP

Results of 1.5mm Gap

Shown below are the experimental results with a gap size of 1.5mm. Nine experimental runs were completed.

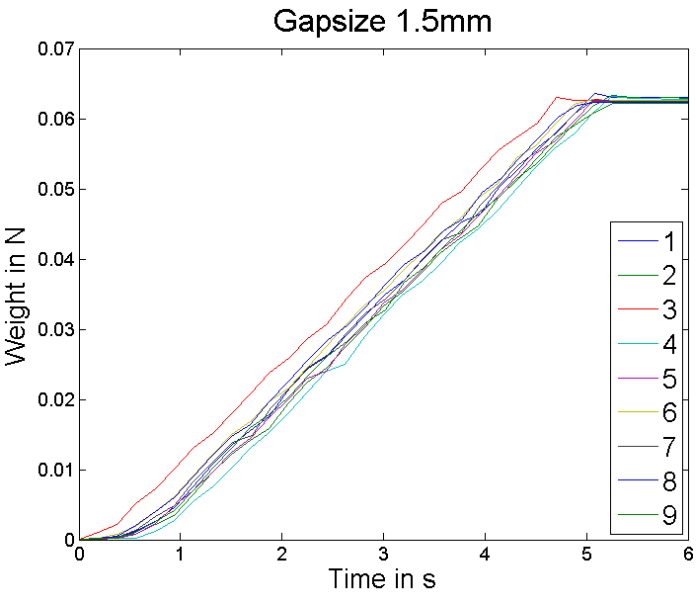

Figure 13. WEIGHT VERSUS TIME EXPERIMENTAL DATA FOR GRANULAR FLOW WITH A 1.5MM GAP

COMPARISON OF THE EXPERIMENTAL AND SIMULATION RESULTS

Several experimental runs were taken for varying gap sizes. A fixed weight of 0.0624 N of granular material was used for each test with an uncertainty of 0.0004 N due to measurement error. Based on the results of these experiments, a range of values was obtained that was later used in validation.

Results of 3mm Gap

Shown below are the experimental results with a gap size of 3mm. Nine experimental runs were completed. The average slope of the experimental runs was 1.41E-2 [N/s] and the slope of the simulation was 1.40E-2 [N/s].

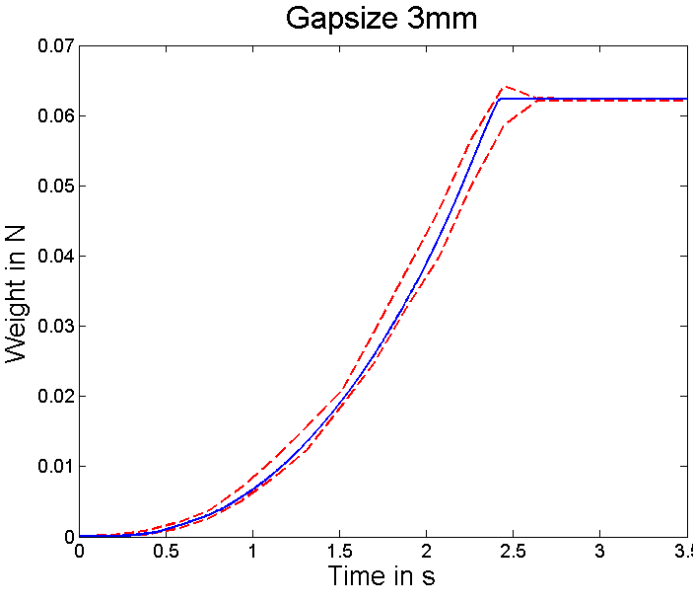

Figure 14. WEIGHT VERSUS TIME EXPERIMENTAL AND SIMULATION DATA FOR GRANULAR FLOW WITH 3MM GAP

Results of 2.5mm Gap

Shown below are the experimental results with a gap size of 2.5mm. Eight experimental runs were completed. The average slope of the experimental runs was 2.59E-2 [N/s] and the slope of the simulation was 2.62E-2 [N/s].

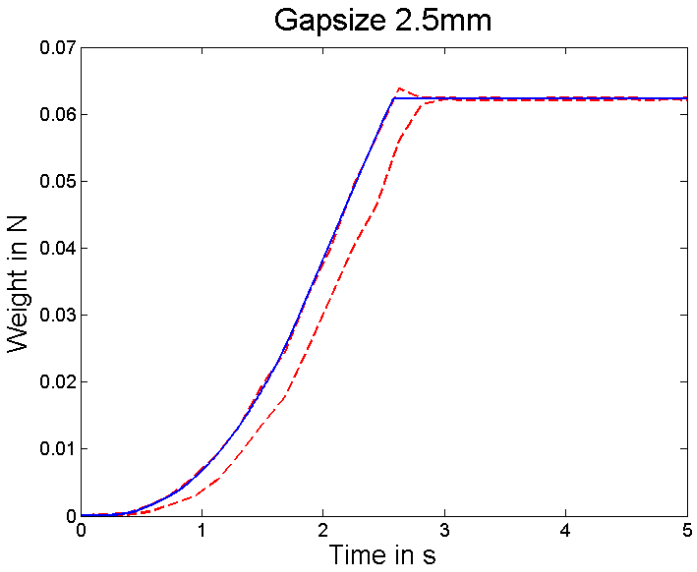

Figure 15. WEIGHT VERSUS TIME EXPERIMENTAL AND SIMULATION DATA FOR GRANULAR FLOW WITH 2.5MM GAP

Results of 2mm Gap

Shown below are the experimental results with a gap size of 2mm. Nine experimental runs were completed. The average slope of the experimental runs was 4.00E-2 [N/s] and the slope of the simulation was 4.05E-2 [N/s].

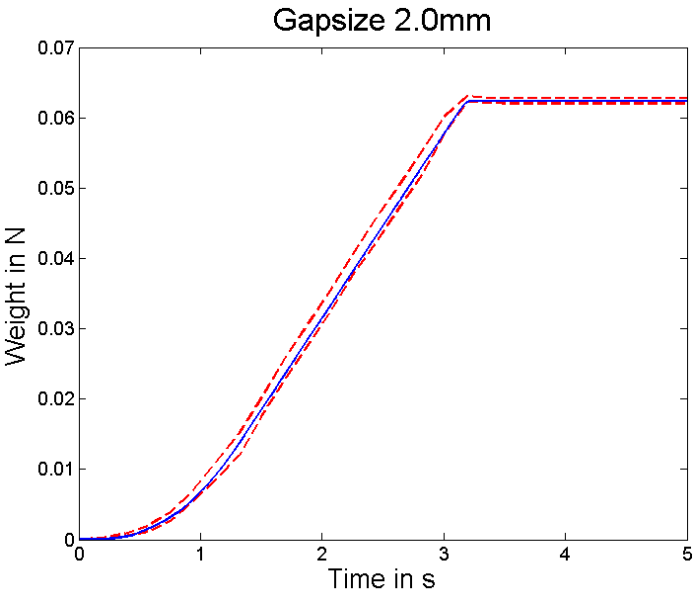

Figure 16. WEIGHT VERSUS TIME EXPERIMENTAL AND SIMULATION DATA FOR GRANULAR FLOW WITH 2MM GAP

Results of 1.5mm Gap

Shown below are the experimental results with a gap size of 1.5mm. Nine experimental runs were completed. The average slope of the experimental runs was 4.44E-2 [N/s] and the slope of the simulation was 4.48E-2 [N/s].

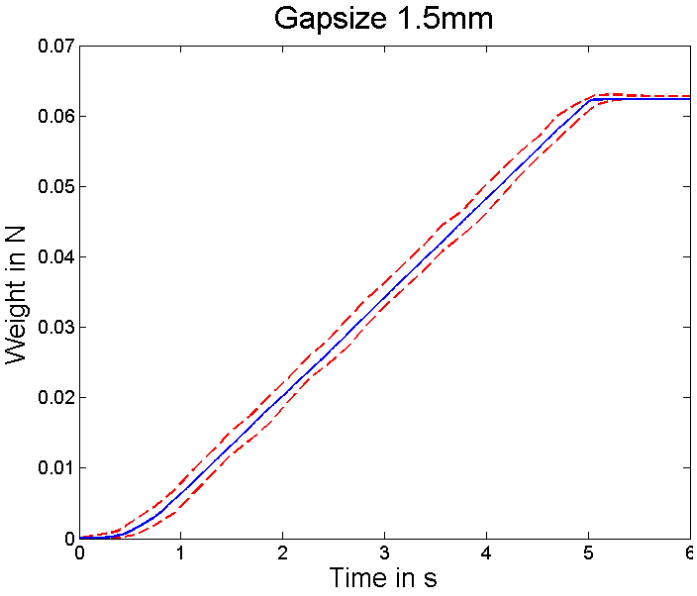

Figure 17. WEIGHT VERSUS TIME EXPERIMENTAL AND SIMULATION DATA FOR GRANULAR FLOW WITH 1.5MM GAP

VALIDATION OF CHRONO::ENGINE IN REGARDS TO ANGLE OF REPOSE

Granular material poured onto a flat surface forms a pile angle in its rest configuration. This angle is called the angle of repose. The repose angle depends on the shape of the particles, cohesion and friction. Both the Chrono::Engine model and the experiment should show the same shape of the pile meaning the angle of repose needs to be the same. In our experimental setup as well as in the simulation model the shape of the particles is spherical with a diameter of 500µm and cohesion is assumed to be zero. The Mohr-Coulomb criterion indicates that the angle of repose is approximately equal to the friction angle. In order to validate that the simulation model behaves in the same way as the experiment the friction angle needs to be determined from the forming pile in the experiment and subsequently used in the Chrono::Engine model. Figure 18 shows the conical pile formed in the experiment. The friction coefficient is determined through the measurement of the angle φ between the ground and the pile surface. For spherical particles the angle of repose is typically between 10°-20°[16]. From experimental measurements, it was obtained that φ = 19.5°.

Figure 18. PHOTO OF THE GLASS SPHERE PILE

$$\mu_{static} \approx tan^{-1}(\varphi)$$

The static friction coefficient was determined to be 0.35, a value subsequently used in the simulation in Chrono::Engine.

Figure 19 below shows the pile built up in the simulation model with exactly the same shape as in the experiment. The angle measurement shows that the angle of repose is the same for the simulation and the experiment.

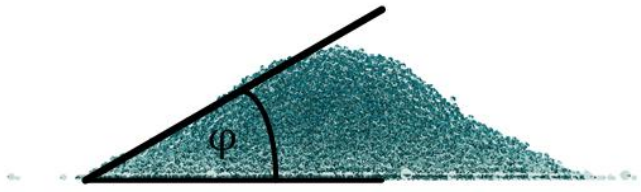

Figure 19. PHOTO OF THE SIMULATED PILE

FUTURE DIRECTIONS OF WORK

The work performed during this experiment leads to several directions of study. Of these directions, the easiest to pursue with the current experimental setup would be testing different cavity shapes and particle sizes. By changing the material properties of the walls (such as using plastic instead of aluminum) the effects of boundary conditions on the flow rate could be judged. Additionally, the streamlines of the granular flow could be mapped by using different colored sand. Finally, to capture yet another property of the system, a cone penetrometer could be used to measure the force required to penetrate a container of sand with a cone. This experiment would require an entirely different experimental and simulation setup, as well as a larger number of particles.

PROJECT CONCLUSIONS

Based on an error of less than 2%, the simulated results match the experimental results well. Despite this, there are several factors that could be improved upon. Namely, the design of the rig did not exactly match the simulated trough. The rig was machined from aluminum, whereas the trough in the simulation had the same coefficient of friction as the granular material. Likewise, the simulation was conducted in essentially vacuum, ignoring any aerodynamic forces, while the experiment was performed at ambient conditions. Lastly, the effects of humidity were not taken into account by the experiment. With that said, the trends of the flow rates are encouraging because the simulation engine is able to predict the nonlinear behavior that occurs near a 3mm gap.

ACKNOWLEDGEMENTS

The authors would like to acknowledge Alessandro Tasora and Mihei Anitescu for their work in GPU-based parallel computing for the simulation of complex multibody systems.

REFERENCES

1. Tasora, A., *Experimental validation of a new method for simulating granular flows in PBR reactors*. 2009, Parma: Dipartimento di Ingegneria Industriale.
2. Stewart, D.E. and J.C. Trinkle, *An implicit time-stepping scheme for rigid-body dynamics with inelastic collisions and Coulomb friction.* International Journal for Numerical Methods in Engineering, 1996. **39**: p. 2673-2691.
3. Moreau, J., *Unilateral contact and dry friction in finite freedom dynamics.* Nonsmooth mechanics and Applications, 1988: p. 1-82.
4. Monteiro-Marques, M., *Differential inclusions in nonsmooth mechanical problems: Shocks and dry friction.* Progress in Nonlinear Differential Equations and Their Applications, 1993. **9**.
5. Pfeiffer, F., M. Foerg, and H. Ulbrich, *Numerical aspects of non-smooth multibody dynamics.* Computer Methods in Applied Mechanics and Engineering, 2006. **195**(50-51): p. 6891-6908.
6. Stewart, D.E., *Rigid-body dynamics with friction and impact.* SIAM Review, 2000. **42**(1): p. 3-39.
7. Stewart, D., *Convergence of a Time Stepping Scheme for Rigid Body Dynamics and Resolution of Painlevé's Problem.* Archive for Rational Mechanics and Analysis, 1998. **145**(3): p. 215-260.
8. Anitescu, M., *Optimization-based simulation of nonsmooth dynamics.* Mathematical Programming, 2006. **105**(1): p. 113-143.
9. Anitescu, M. and G.D. Hart, *A constraint-stabilized time-stepping approach for rigid multibody dynamics with joints, contact and friction.* International Journal for Numerical Methods in Engineering, 2004. **60**(14): p. 2335-2371.
10. Tasora, A. and M. Anitescu, *A Fast NCP Solver for Large Rigid-Body Problems with Contacts, Friction, and Joints.* Multibody Dynamics: Computational Methods and Applications, 2008: p. 45.
11. Anitescu, M. and A. Tasora, *An iterative approach for cone complementarity problems for nonsmooth*

dynamics. Computational Optimization and Applications, 2009,accepted.
12. Mark Harris, Shubhabrata Sengupta, and J.D. Owens, *Parallel Prefix Sum (Scan) with CUDA*, in *GPU Gems 3*, H. Nguyen, Editor. 2008, Addison-Wesley. p. 851-876.
13. *UMR Series Precision Double-Row Ball Bearing Linear Stages*. 2010 [cited 2009 December 21]; Available from: http://www.newport.com.
14. *LFS 242 - Tension/Compression Cell*. 2007, Cooper Instruments and Systems: Warrenton, Virginia, United States of America.
15. *DFI Infinity - Digital Indicator/Controller*. 2007, Cooper Instruments and Systems: Warrenton, Virginia, United States of America.
16. Herrmann, H., *On the shape of a sandpile.* NATO ASI Series E Applied Sciences-Advanced Study Institute, 1998. **350**: p. 319-338.