Enabling High Performance Computational Dynamics in a Heterogeneous Hardware Ecosystem

Assistant Prof. **Dan Negrut**
Simulation-Based Engineering Lab
Department of Mechanical Engineering
University of Wisconsin – Madison

Toward High-Performance Computing Support
for the Simulation and Planning of Robot Contact Tasks
June 27, 2011

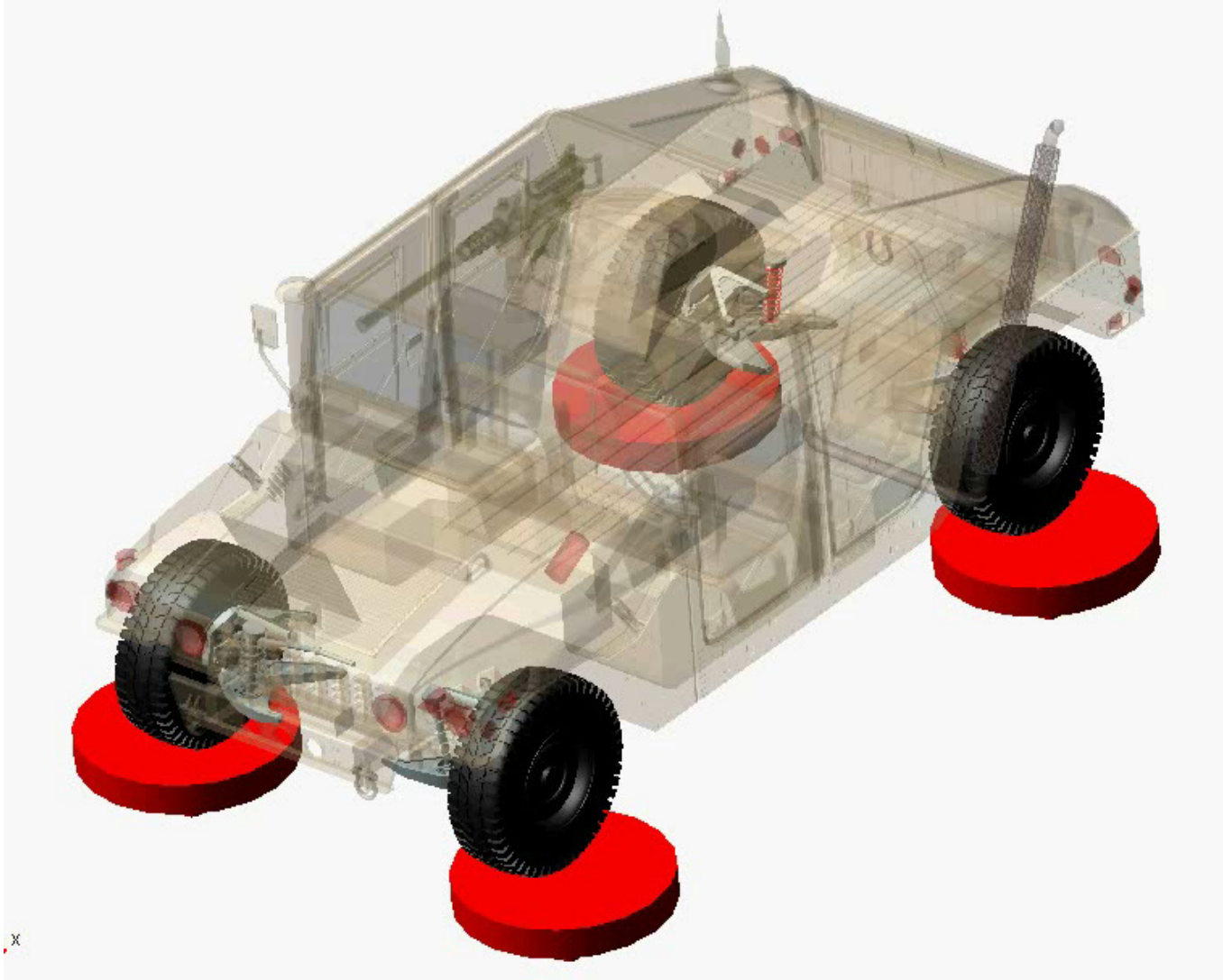
Acknowledgements

- People:
  - Alessandro Tasora – University of Parma, Italy
  - Mihai Anitescu – Argonne National Lab
  - Lab Students:
    - Hammad Mazhar
    - Toby Heyn
    - Andrew Seidl
    - Arman Pazouki
    - Hamid Ansari
    - Justin Madsen
    - Naresh Khude
    - Makarand Datar
    - Dan Melanz
    - Markus Schmid
- Financial support
  - National Science Foundation, Young Investigator Career Award
  - FunctionBay, S. Korea
  - NVIDIA
  - Microsoft
  - Caterpillar

Talk Overview

- Overview of the engineering problems of interest
- Large-scale Multibody Dynamics
  - Problem formulation, solution method, and parallel implementation
- Overview of Heterogeneous Computing Template (HCT)
- Numerical Experiments
- Validation efforts
- Conclusions

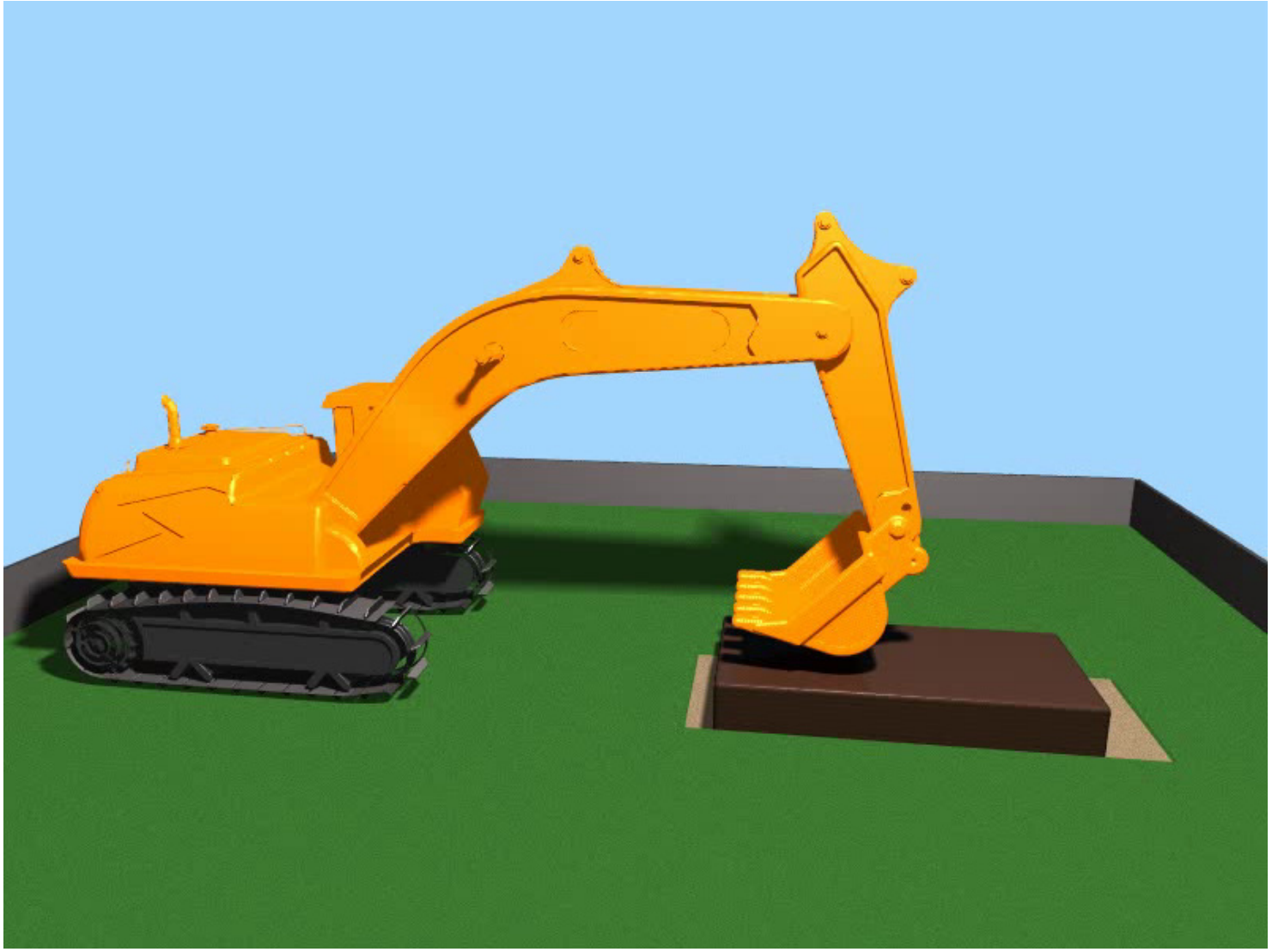
Computational Multibody Dynamics



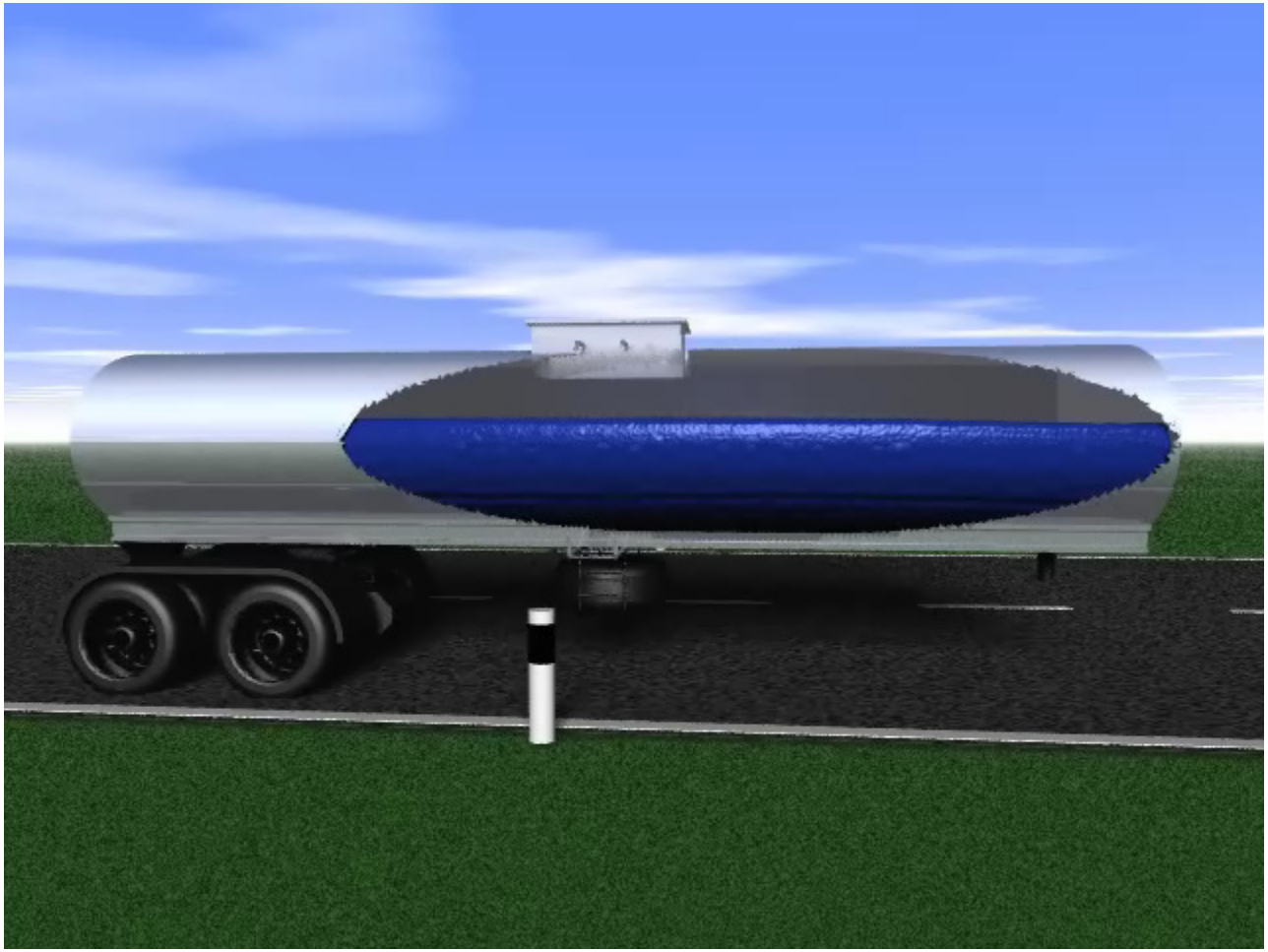
Simulation generated in ADAMS

Multi-Physics…

Fluid-Solid Interaction: Navier-Stokes + Newton-Euler.

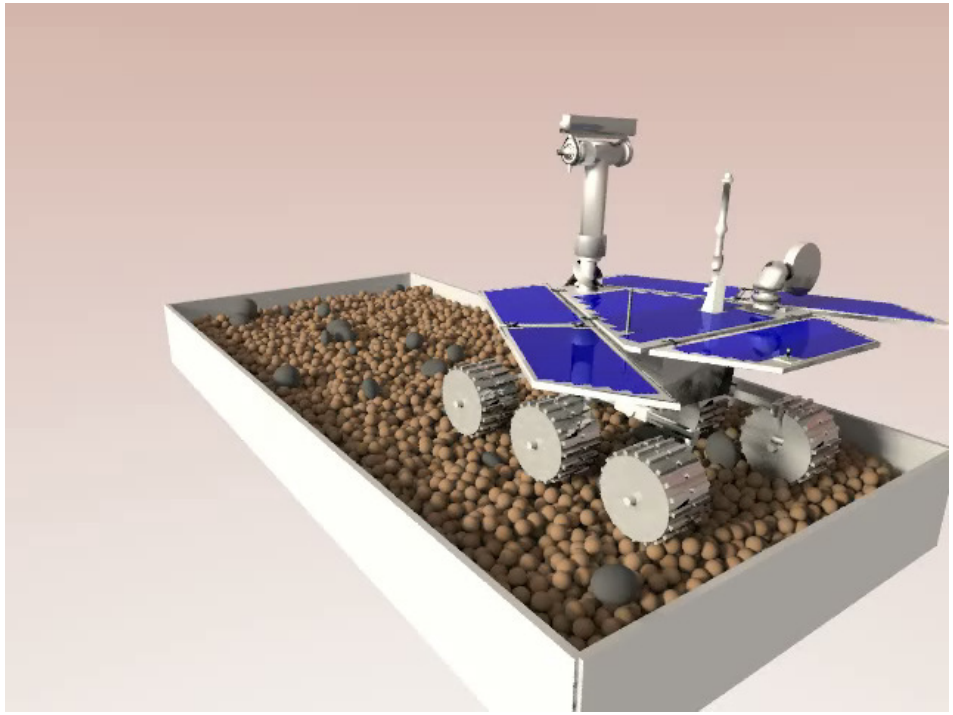


Computational Dynamics

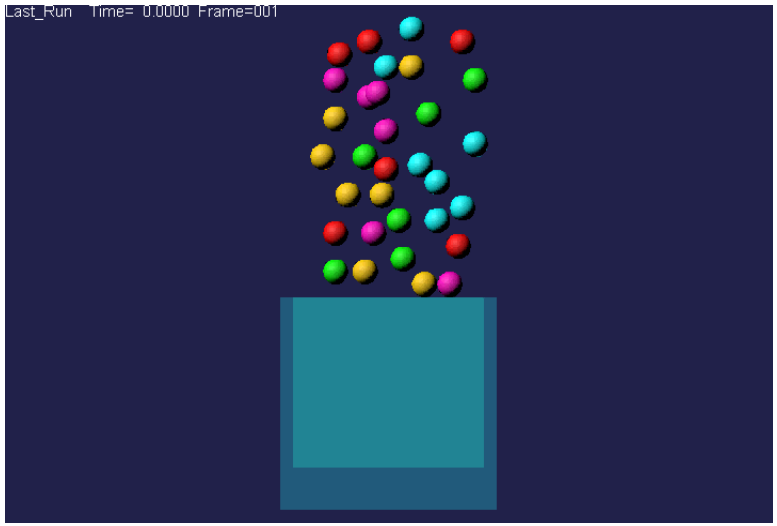


Rover Mobility on Granular Terrain

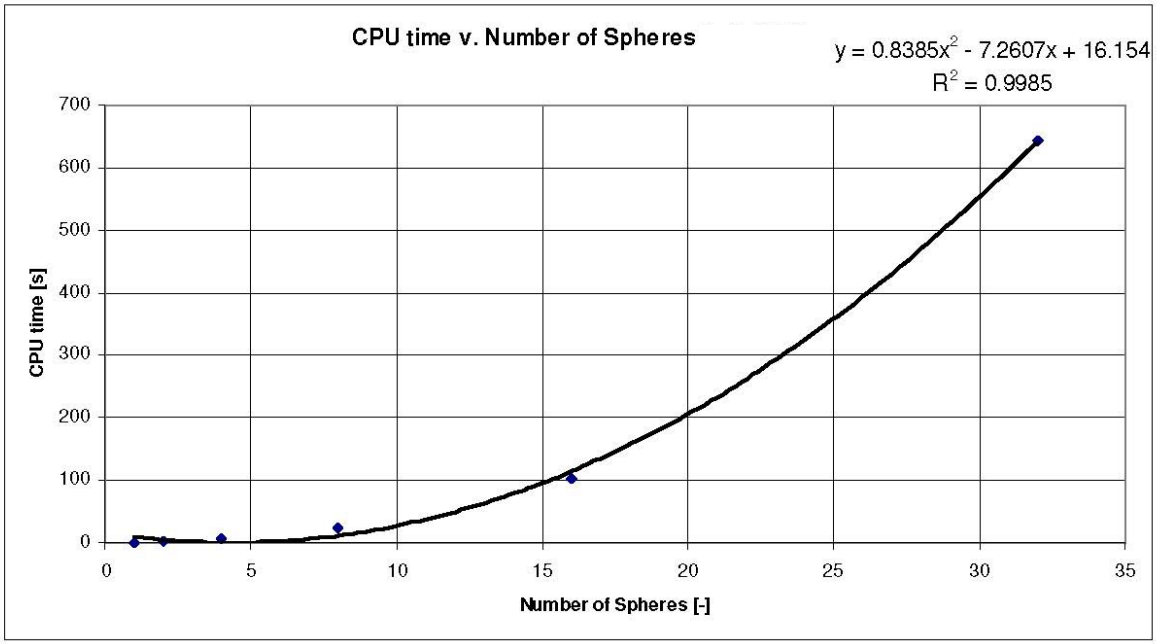
- Wheeled/tracked vehicle mobility on granular terrain
- Also interested in scooping and loading granular material



Frictional Contact Simulation
[Commercial Solution]



- Model Parameters:
  - Spheres: 60 mm diameter and mass 0.882 kg
  - Forces: smoothing with stiffness of 1E5, force exponent of 2.2, damping coefficient of 10.0, and a penetration depth of 0.1
  - Simulation length: 3 seconds



Frictional Contact: Two Different Approaches Considered

- Discrete Element Method (DEM) - draws on a “smoothing” (penalty) approach
  - Lots of heuristics
  - Slow
  - General purpose
  - Used in ADAMS

- DVI-based (Differential Variational Inequalities)
  - A set of differential equations combined with inequality constraints
  - Fast (stable for significantly larger integration step-sizes)
  - Less general purpose
  - Used widely in computer games

The Modeling Component

Equations of Motion: Multibody Dynamics

Generalized Positions

Generalized Mass Matrix

Velocity Transformation Matrix

Generalized Velocities

Kinematic Differential Equations

$$\dot{\mathbf{q}} = \mathbf{T}(\mathbf{q})\mathbf{v}$$

Force Balance Equations

$$\mathbf{M}(\mathbf{q})\dot{\mathbf{v}} = \underbrace{\mathbf{f}(t,\mathbf{q},\mathbf{v})}_{\text{Applied Force}} - \overbrace{\mathbf{g}_{\mathbf{q}}^{\mathrm{T}}(\mathbf{q},t)\boldsymbol{\lambda}}^{\text{Reaction Force}} + \overbrace{\sum_{i=1}^{N_c}(\gamma_n^i \mathbf{D}_n^{T,i} + \gamma_u^i \mathbf{D}_u^{T,i} + \gamma_w^i \mathbf{D}_w^{T,i})}^{\text{Frictional Contact Force}}$$

Holonomic Kinematic Constraints

$$\mathbf{g}(\mathbf{q},t) = \mathbf{0}$$

Contact Complementarity Conditions

$$0 \le \Phi^i(\mathbf{q},t) \quad \perp \quad \overbrace{\gamma_n^i \ge 0}^{\text{Contact Impulse, for Contact "}i\text{"}} \qquad i = 1,2,\ldots,\underbrace{N_c}_{\text{Total Number of Contacts}}$$

Coulomb Friction Model

$$(\gamma_u^i, \gamma_w^i) = \underset{\mu^i \gamma_n^i \ge \sqrt{(\gamma_u^i)^2 + (\gamma_w^i)^2}}{\arg\min} \underbrace{(\gamma_u^i \mathbf{v}^T \mathbf{D}_u^i + \gamma_w^i \mathbf{v}^T \mathbf{D}_w^i)}_{\text{Friction Dissipation Energy}}$$

Gap Function, for Contact "i"

Friction Impulse Components, for Contact "i"

Traditional Discretization Scheme

positions

time step index

$$\mathbf{q}^{(l+1)} = \mathbf{q}^{(l)} + h\mathbf{L}(\mathbf{q}^{(l)})\mathbf{v}^{(l+1)}$$

Mass Mat.

speeds

Applied Forces

Reaction impulses

$$\mathbf{M}(\mathbf{v}^{(l+1)} - \mathbf{v}^{l}) = h\mathbf{f}(t^{(l)}, \mathbf{q}^{(l)}, \mathbf{v}^{(l)}) + \sum_{i\in\mathcal{A}(q^{(l)},\delta)} \left(\gamma_{i,n}\,\mathbf{D}_{i,n} + \gamma_{i,u}\,\mathbf{D}_{i,u} + \gamma_{i,w}\,\mathbf{D}_{i,w}\right)$$

$$i \in \mathcal{A}(q^{(l)},\delta): \quad 0 \leq \frac{1}{h}\Phi_i(\mathbf{q}^{(l)}) + \mathbf{D}_{i,n}^T\mathbf{v}^{(l+1)} \perp \gamma_n^i \geq 0,$$

Complementarity Condition

$$(\gamma_{i,u}, \gamma_{i,w}) = \underset{\mu_i\gamma_{i,n} \geq \sqrt{\gamma_{i,u}^2 + \gamma_{i,w}^2}}{\operatorname{argmin}} \mathbf{v}^T\left(\gamma_{i,u}\,\mathbf{D}_{i,u} + \gamma_{i,w}\,\mathbf{D}_{i,w}\right).$$

Coulomb 3D fricion model

Stabilization term

(Stewart & Trinkle, 1996)

Relaxed Discretization Scheme Used

$$\mathbf{q}^{(l+1)} = \mathbf{q}^{(l)} + h\mathbf{L}(\mathbf{q}^{(l)})\mathbf{v}^{(l+1)}$$

$$\mathbf{M}(\mathbf{v}^{(l+1)} - \mathbf{v}^{l}) = h\mathbf{f}(t^{(l)}, \mathbf{q}^{(l)}, \mathbf{v}^{(l)}) + \sum_{i \in \mathcal{A}(q^{(l)},\delta)} \left(\gamma_{i,n}\,\mathbf{D}_{i,n} + \gamma_{i,u}\,\mathbf{D}_{i,u} + \gamma_{i,w}\,\mathbf{D}_{i,w}\right)$$

$$i \in \mathcal{A}(q^{(l)}, \delta): \quad 0 \leq \frac{1}{h}\Phi_i(\mathbf{q}^{(l)}) + \mathbf{D}_{i,n}^T\mathbf{v}^{(l+1)} - \boxed{\mu^i\sqrt{(\mathbf{v}^T\,\mathbf{D}_{i,u})^2 + \mathbf{v}^T\,\mathbf{D}_{i,w})^2}} \perp \gamma_n^i \geq 0,$$

Relaxation Term

$$(\gamma_{i,u}, \gamma_{i,w}) = \underset{\mu_i\gamma_{i,n} \geq \sqrt{\gamma_{i,u}^2 + \gamma_{i,w}^2}}{\operatorname{argmin}} \mathbf{v}^T\left(\gamma_{i,u}\,\mathbf{D}_{i,u} + \gamma_{i,w}\,\mathbf{D}_{i,w}\right).$$

(Anitescu & Tasora, 2008)

The Cone Complementarity Problem (CCP)

- First order optimality conditions lead to Cone Complementarity Problem

- Introduce the convex hypercone...

$$\Upsilon = \left(\bigoplus_{i \in \mathcal{A}(\mathbf{q}^l, \epsilon)} \mathcal{FC}^i \right)$$

... and its polar hypercone:

$$\Upsilon^\circ = \left(\bigoplus_{i \in \mathcal{A}(\mathbf{q}^l, \epsilon)} \mathcal{FC}^{i\circ} \right)$$

$\mathcal{FC}^i \in \mathbb{R}^3$ represents friction cone associated with i^{th} contact

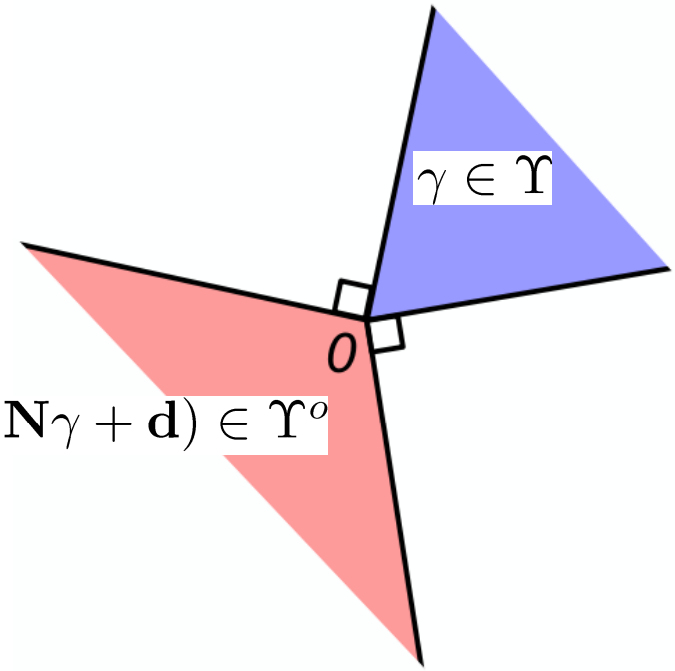
CCP assumes following form: Find γ such that

$$\gamma \in \Upsilon \perp -(\mathbf{N}\gamma + \mathbf{d}) \in \Upsilon^\circ$$

Putting Things in Perspective…

- Problem solved at each time step:
  (advancing simulation from t_l to t_{l+1})

$$
\begin{aligned}
\gamma \in \Upsilon \quad \perp \quad & -(\mathbf{N}\gamma + \mathbf{d}) \in \Upsilon^o \\
\mathbf{v}^{(l+1)} &= \mathbf{M}^{-1}\left(\tilde{\mathbf{k}} + \mathbf{D}\gamma\right) \\
\mathbf{q}^{(l+1)} &= \mathbf{q}^{(l)} + h\mathbf{L}(\mathbf{q}^{(l)})\mathbf{v}^{(l+1)}
\end{aligned}
$$



- Three key points led to above algorithm:
  - Friction model posed as an optimization problem
  - Working with velocity and impulses rather than acceleration and forces
  - Contact complementarity expression altered to lead to CCP

Implementation

- Method outlined implemented using two loops
  - Outer loop – runs the time stepping
  - Inner loop – CCP Algorithm (solves CCP problem at each time step)

Granular Dynamics: How Parallel Computing is Leveraged

1. Parallel Collision Detection
2. (Body parallel) Force kernel
3. (Contact parallel) Contact preprocessing kernel
4. (Contact parallel) CCP contact kernel
5. (Constraint parallel) CCP constraint kernel
6. (Reduction-slot parallel) Velocity reduction kernel
7. (Body parallel) Body velocity update kernel
8. (Body parallel) Time integration kernel

Inner Loop

Outer Loop

Inner Loop (CCP Algorithm)

1. For each contact i, evaluate $\eta_i = 3/\text{Trace}(\mathbf{D}_i^T \mathbf{M}^{-1} \mathbf{D}_i)$.

2. If some initial guess γ^* is available for multipliers, then set $\gamma^0 = \gamma^*$, otherwise $\gamma^0 = \mathbf{0}$.

3. Initialize velocities: $\mathbf{v}^0 = \sum_i \mathbf{M}^{-1} \mathbf{D}_i \gamma_i^0 + \mathbf{M}^{-1} \tilde{\mathbf{k}}$.

4. For each contact i, compute changes in multipliers for contact constraints:

$$\gamma_i^{r+1} = \lambda\, \Pi_{\Upsilon_i} \left(\gamma_i^r - \omega \eta_i \left(\mathbf{D}_i^T \mathbf{v}^r + \mathbf{b}_i \right) \right) + (1-\lambda) \gamma_i^r \; ;$$

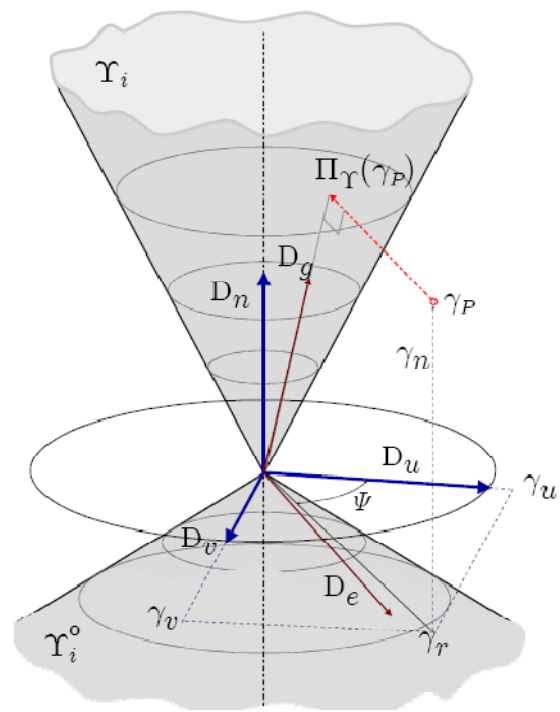
$$\Delta \gamma_i^{r+1} = \gamma_i^{r+1} - \gamma_i^r \; ;$$

$$\Delta \mathbf{v}_i = \mathbf{M}^{-1} \mathbf{D}_i \Delta \gamma_i^{r+1}.$$

5. Apply updates to the velocity vector:

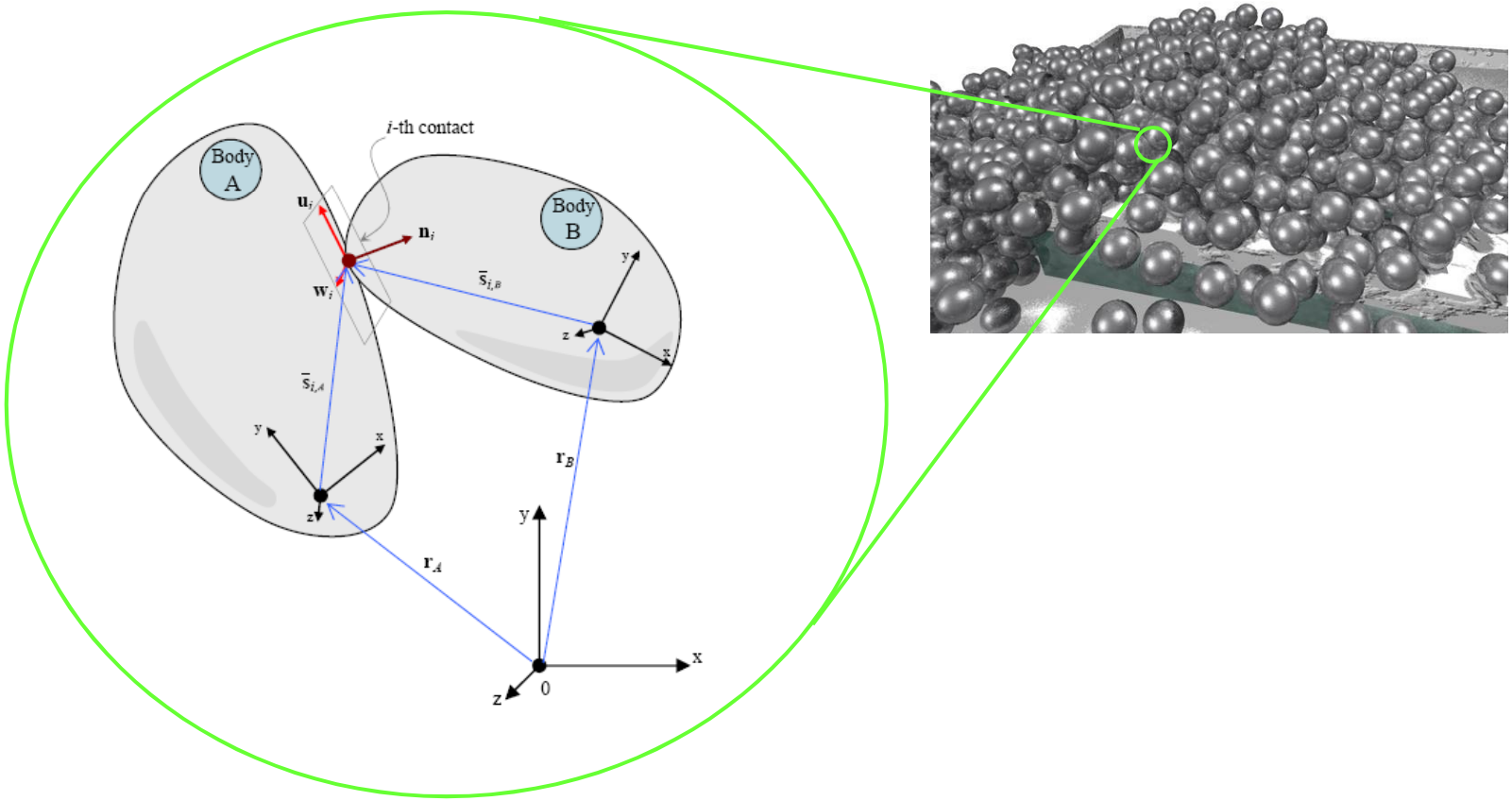
$$\mathbf{v}^{r+1} = \mathbf{v}^r + \sum_i \Delta \mathbf{v}_i$$

6. $r := r + 1$. Repeat from 4 until convergence or $r > r_{max}$

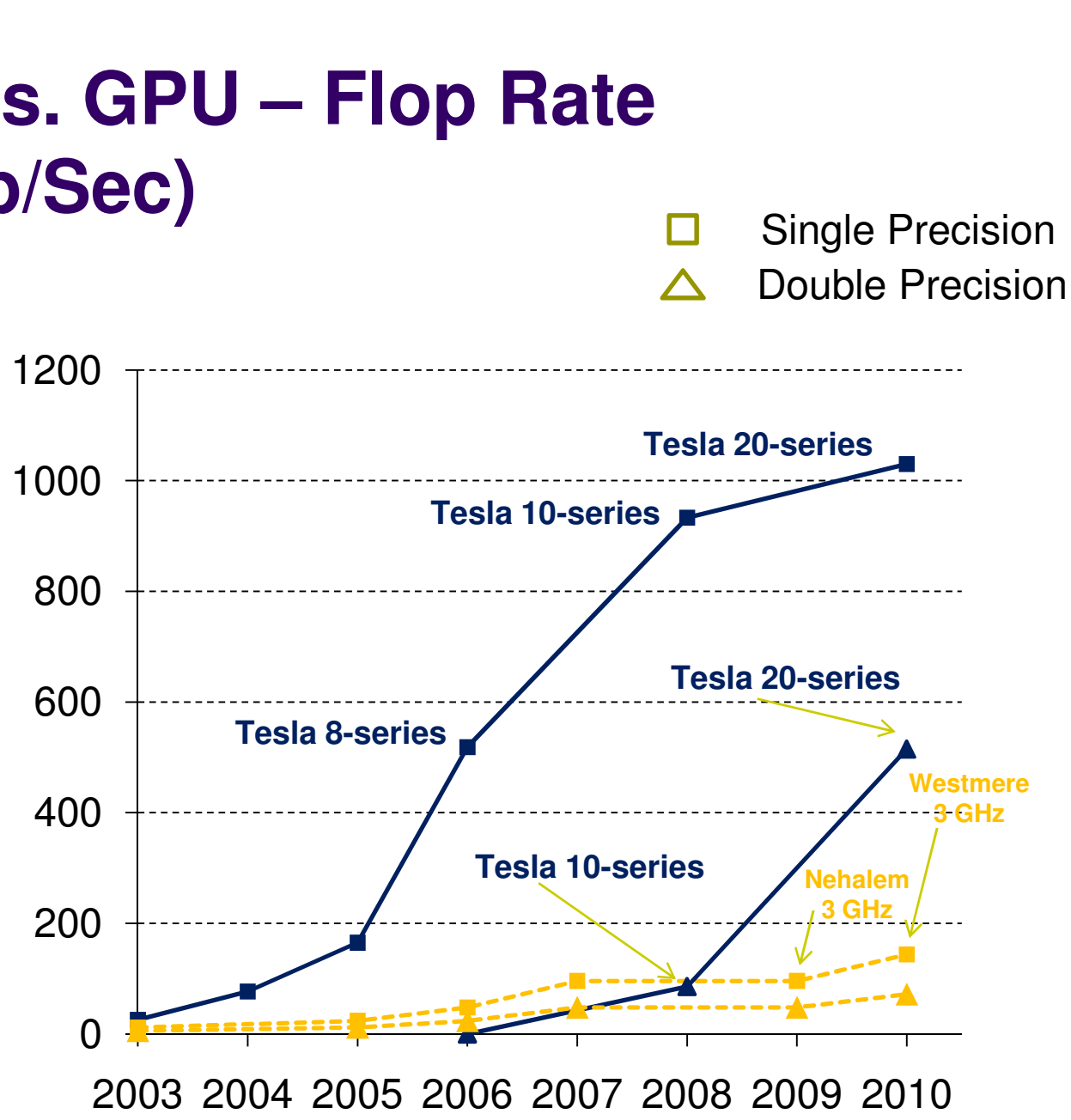


Large Scale Granular Dynamics

- Numerical solution can leverage parallel computing

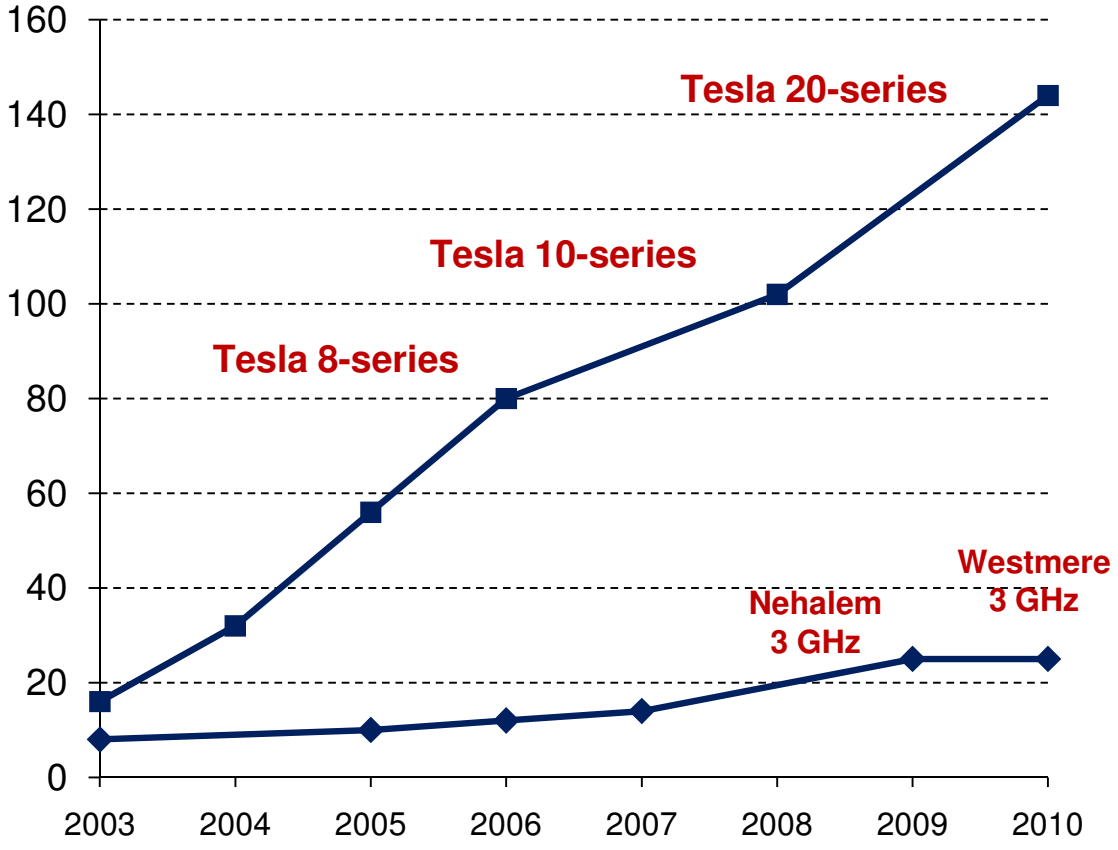


CPU vs. GPU – Flop Rate (GFlop/Sec)

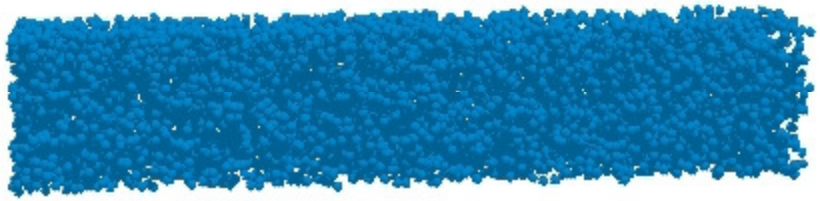


CPU vs. GPU– Memory Bandwidth

[GB/sec]

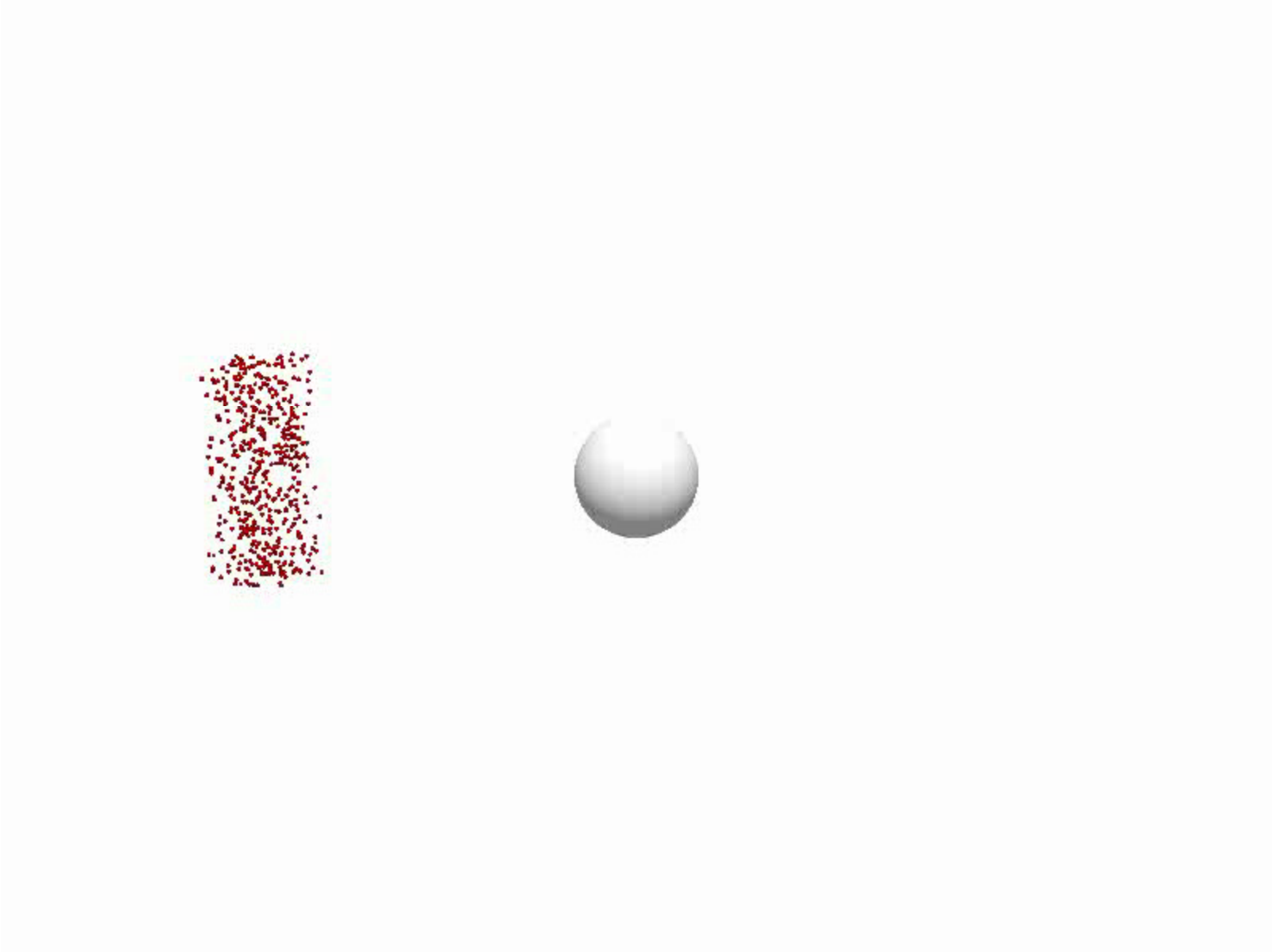


Mixing 40,000 Spheres on the GPU



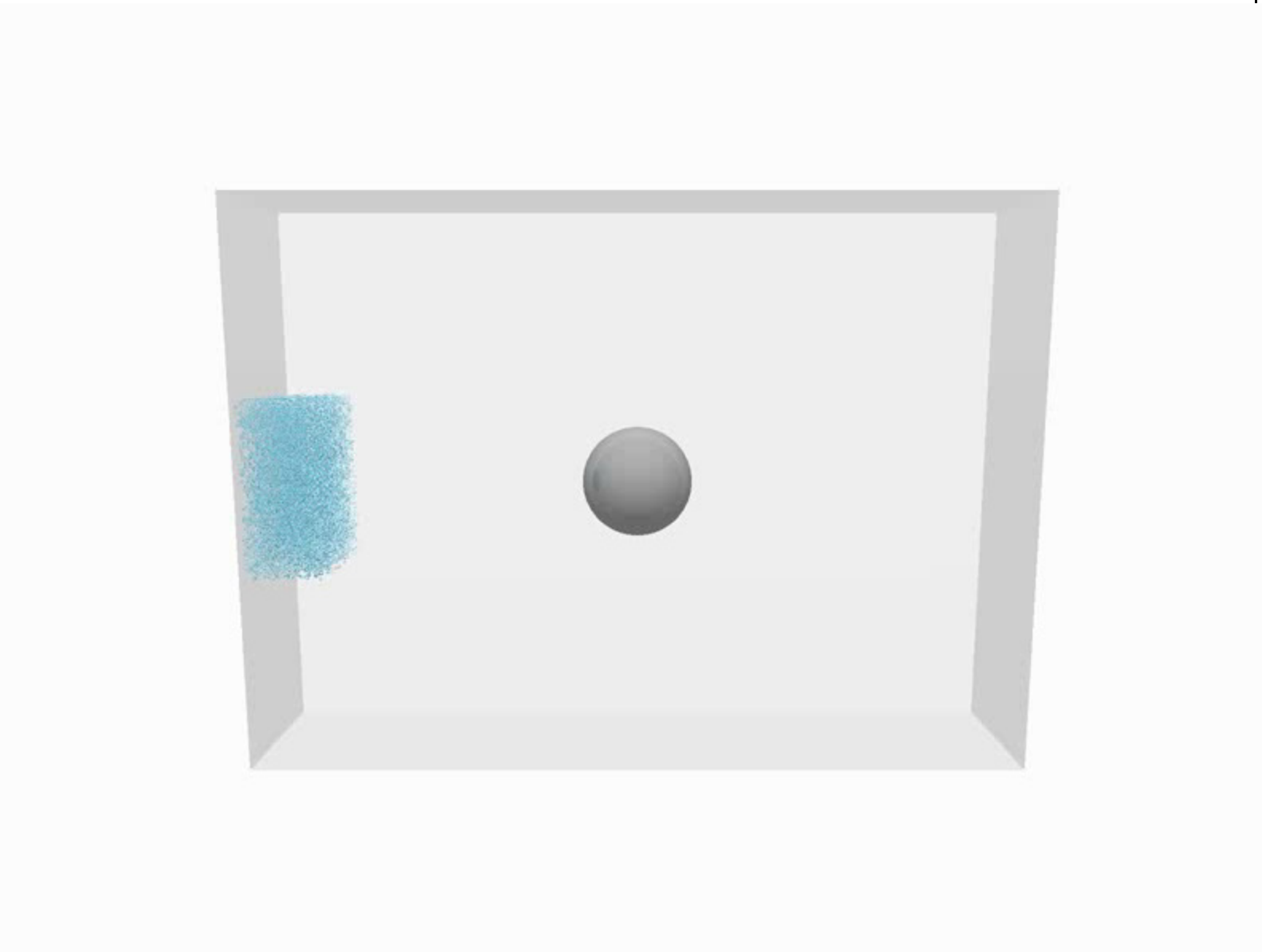
300K Spheres in Tank

[parallel on the GPU]



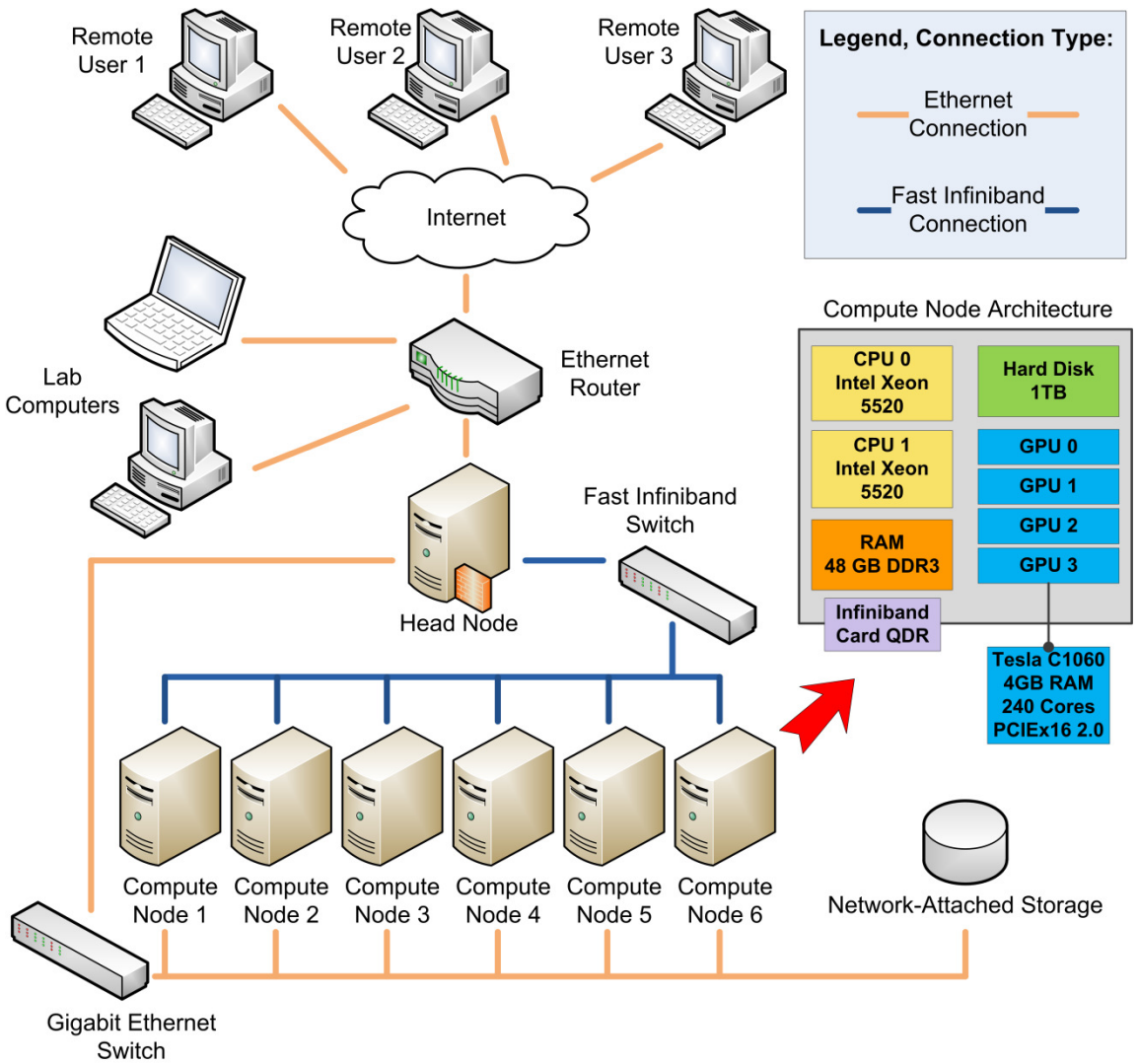
1.1 Million Rigid Spheres

[parallel on the GPU]



A Heterogeneous Computing Template for Computational Dynamics

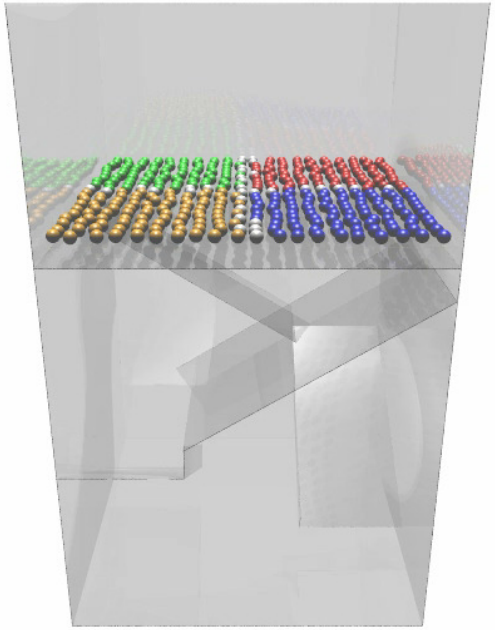
Heterogeneous Cluster



Second fastest cluster at University of Wisconsin-Madison

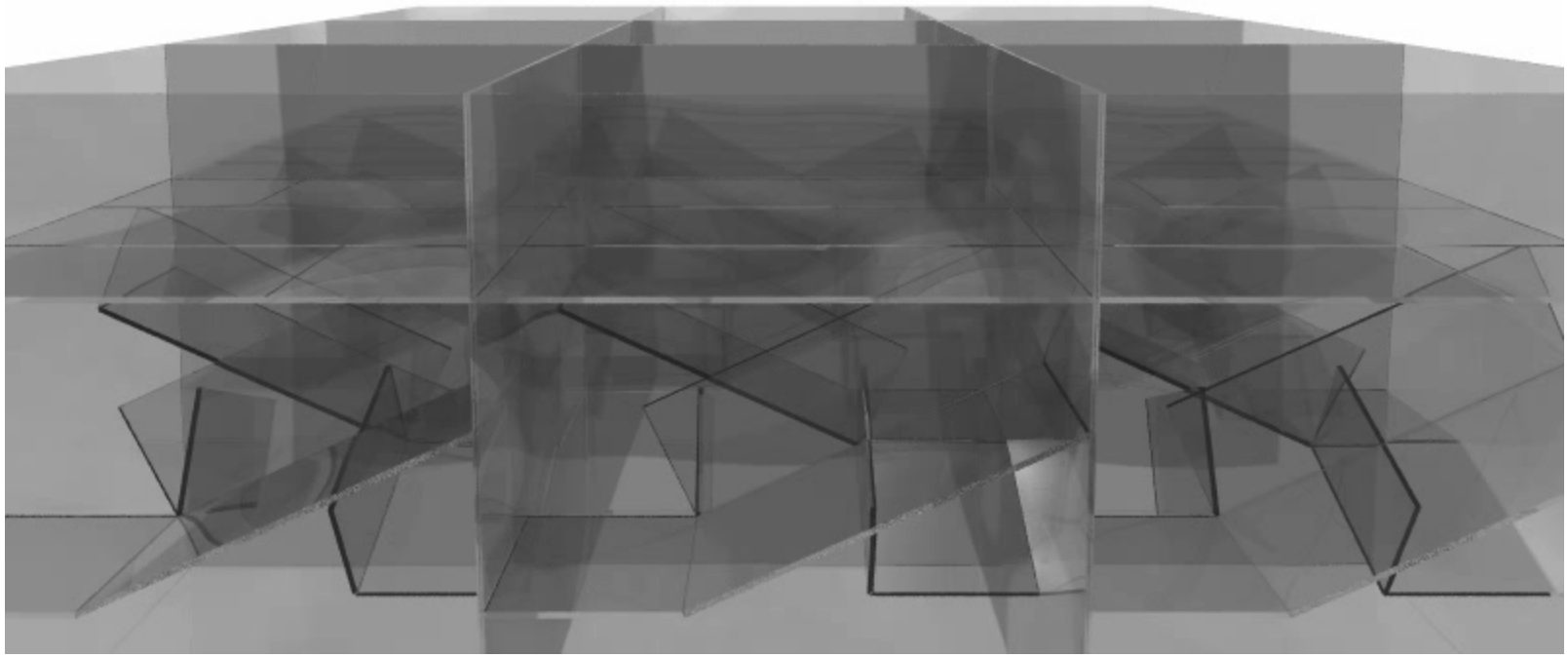
Computation Using Multiple CPUs

[DEM solution]



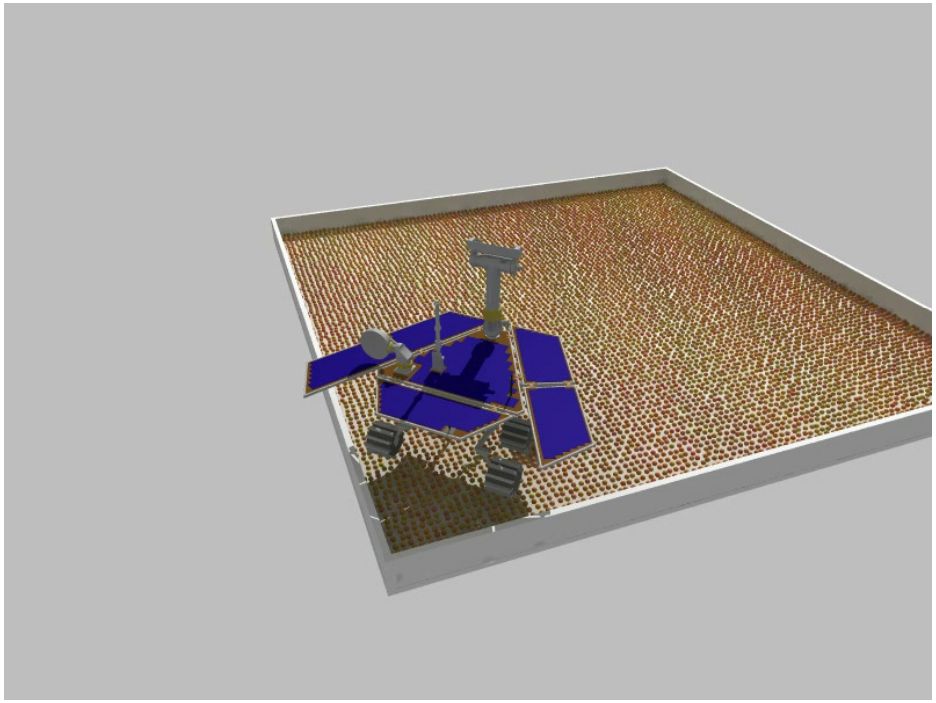
Computation Using Multiple CPUs

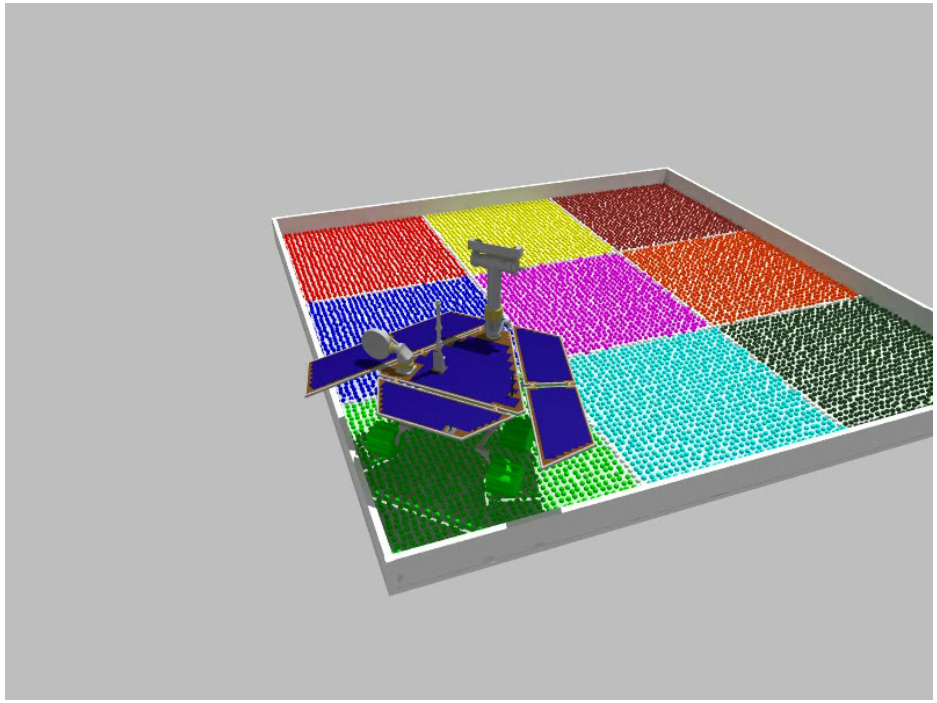
[DEM solution]



Computation Using Multiple CPUs

[DEM solution]

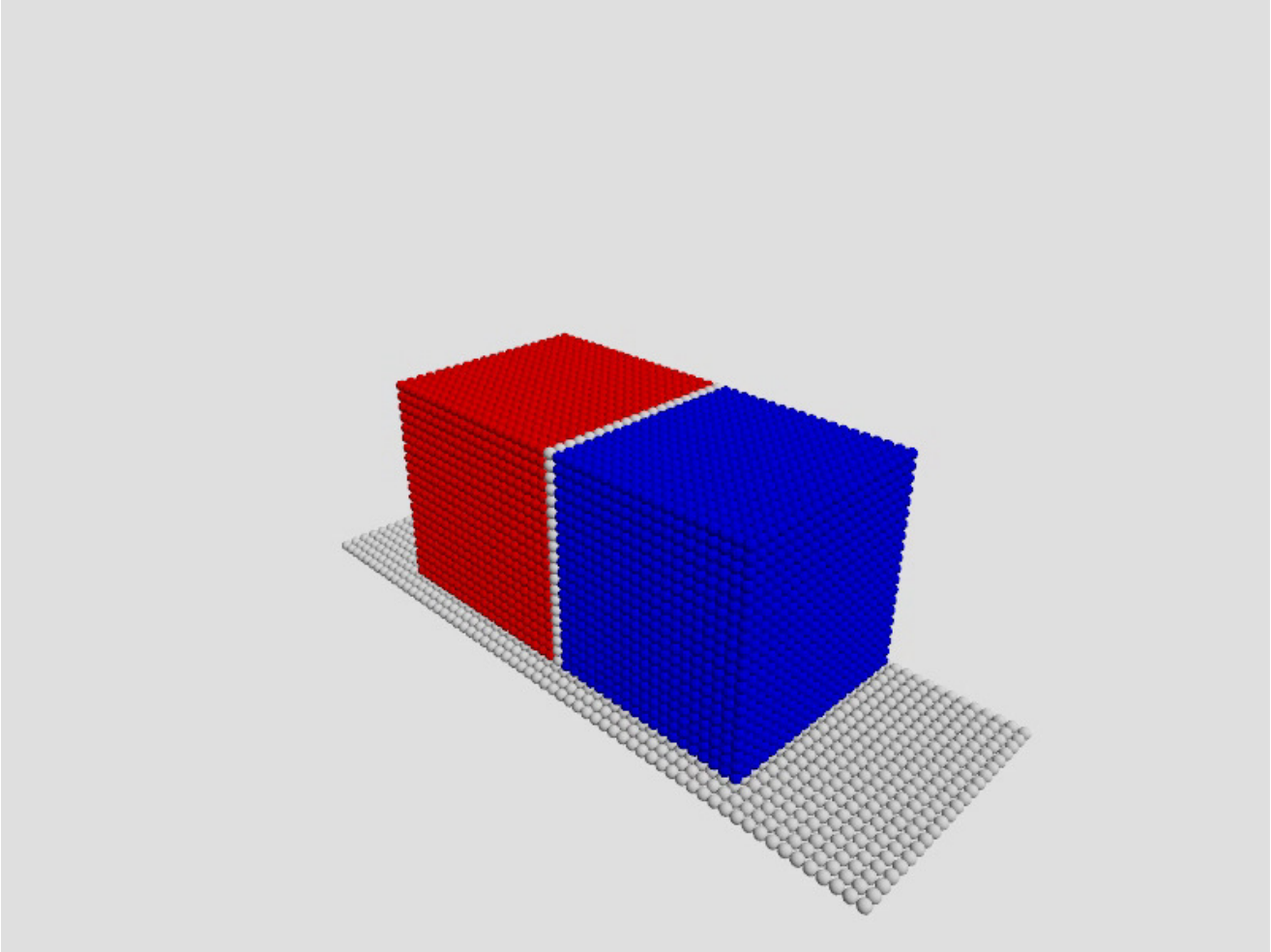




True Heterogeneous Computing

- Use 1 GPU per MPI process
- Each process uses its GPU to perform the collision detection task during each time step
  - As before, CPU takes care of communication between sub-domains
  - State data is copied to GPU, CD is performed, and collision data is copied back
  - GPU is used as an accelerator/co-processor

Demonstration



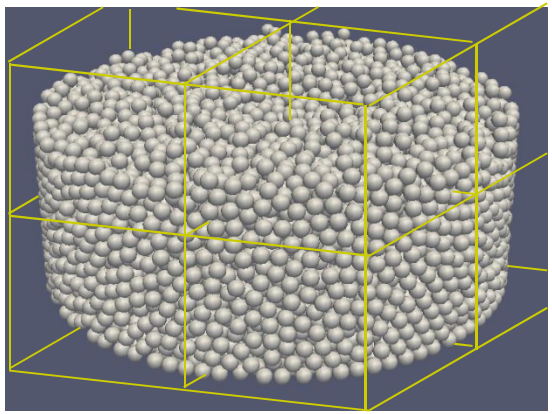
16,000 bodies
2 sub-domains

CPU:

CPU+GPU: 9.43 hrs

Heterogeneous Computing Template Five Major Components

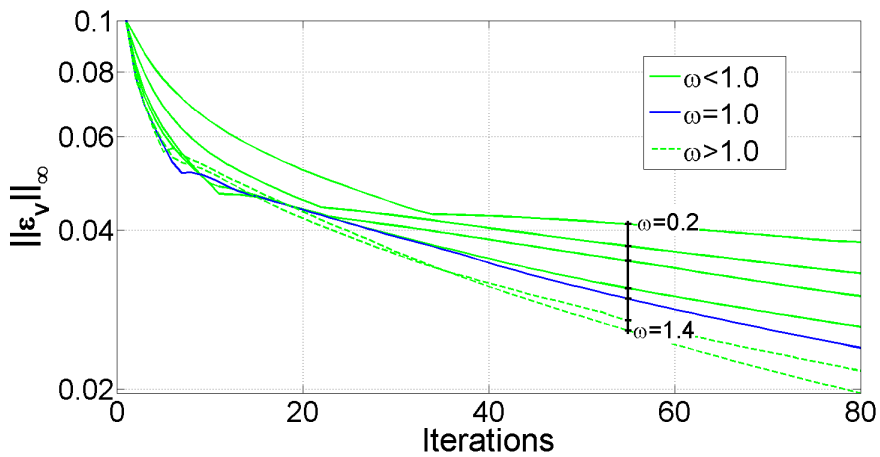
- Computational Dynamics requires
  - Domain decomposition
  - Proximity computation
  - Inter-domain data exchange
  - Numerical algorithm support
  - Post-processing (visualization)

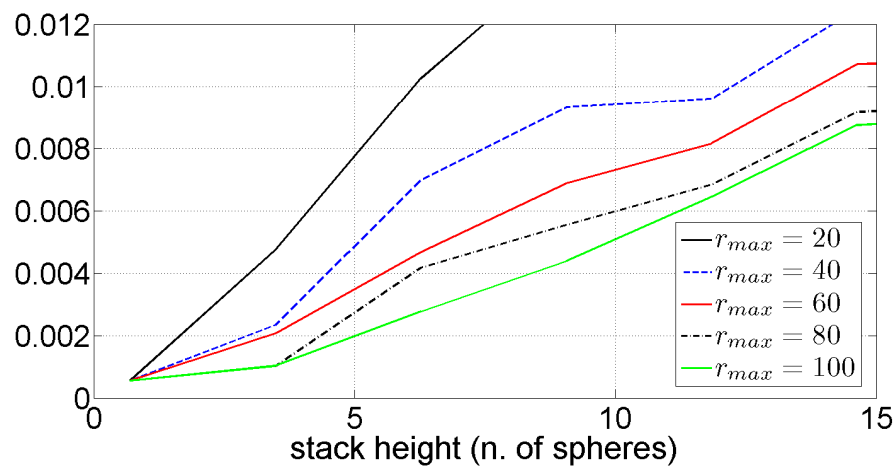


- HCT represents the library support and associated API that capture this five component abstraction

Typical Simulation Results…

- LEFT: Infinity norm of the residual vs. iteration index in the CCP solution
  - Convergence rate (slope of curve) becomes smaller as the iteration index increases.





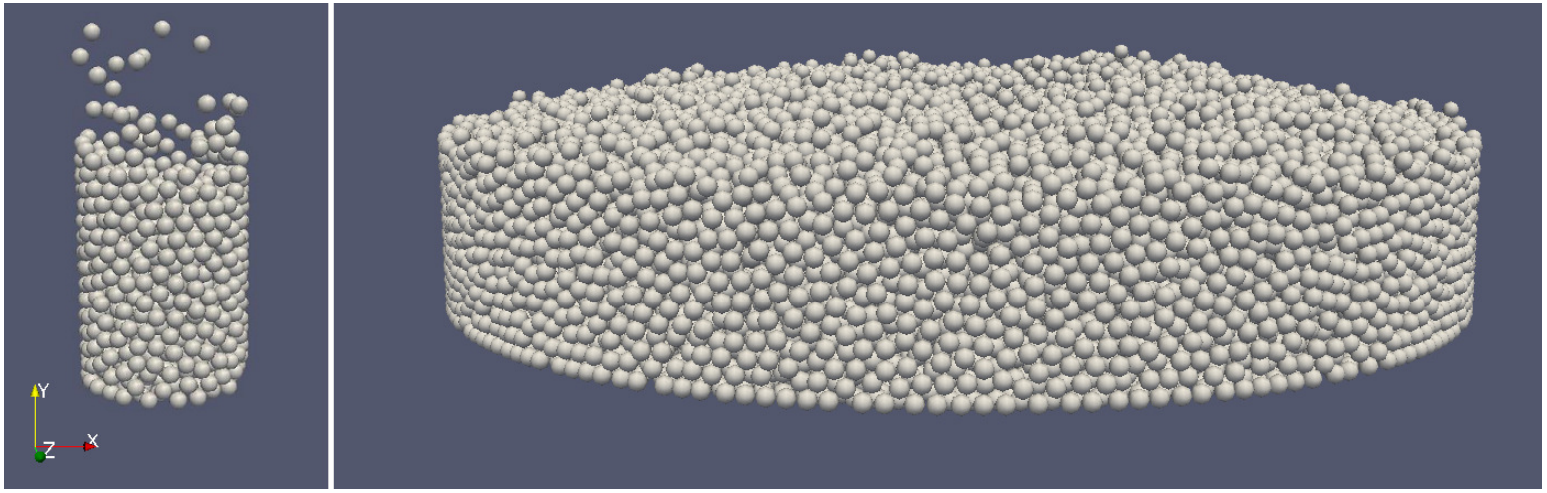
- RIGHT: Infinity norm of the CCP residual after r_{max} iterations as function of granular material depth (number of spheres stacked on each other).

Searching for Better Methods

- Frictionless case (bound constraints in place)
  - Gauss-Jacobi (CE)
  - Projected conjugate gradient (ProjCG)
  - Gradient projected conjugate gradient (GPCG)
  - Gradient projected MINRES (GPMINRES)

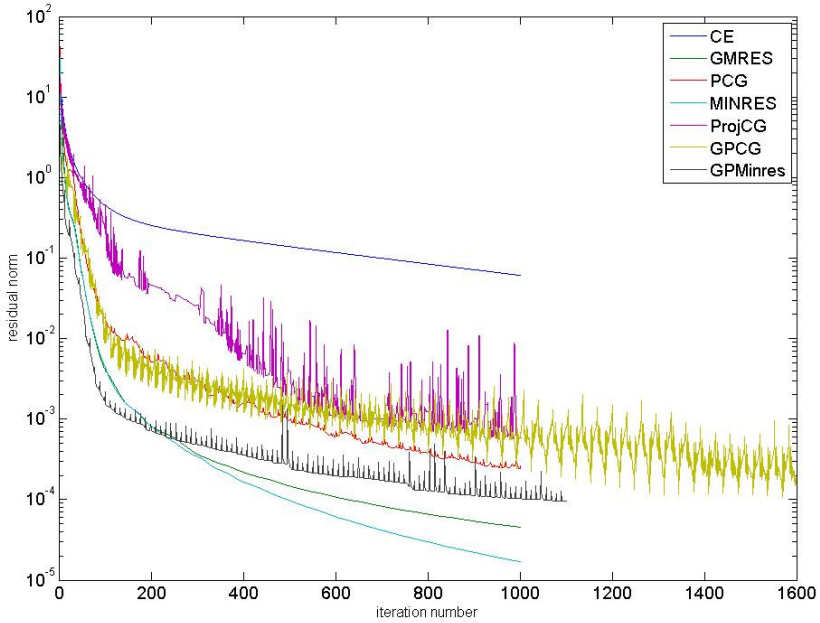
- Friction case (cone constraints - ongoing)
  - Newton's Method for large bound-constrained problems
    - Uses re-parameterization to handle friction cones (replace with bound constraints)

Numerical Experiments



- Test Problem: 40,000 bodies $\Rightarrow$ 157,520 contacts
- Frictionless

Test Problem (MATLAB)



Method	Iterations	Final Residual Norm	γ_{min}	γ_{max}	Time [sec]
CE	1000	6.11×10^{-2}	0.0	2.0598	1849.5
ProjCG	1002	5.6344×10^{-4}	0.0	2.2286	1235.6
GPCG	1600	1.0675×10^{-4}	0.0	2.6349	382.3644
GPMinres	1100	9.5239×10^{-5}	0.0	2.3090	238.0744
PCG	1000	2.4053×10^{-4}	-1.1116	2.5254	27.9686
GMRES	1000	4.5315×10^{-5}	-1.1635	2.5227	736.3007
MINRES	1000	1.6979×10^{-5}	-1.1316	2.5253	41.5790

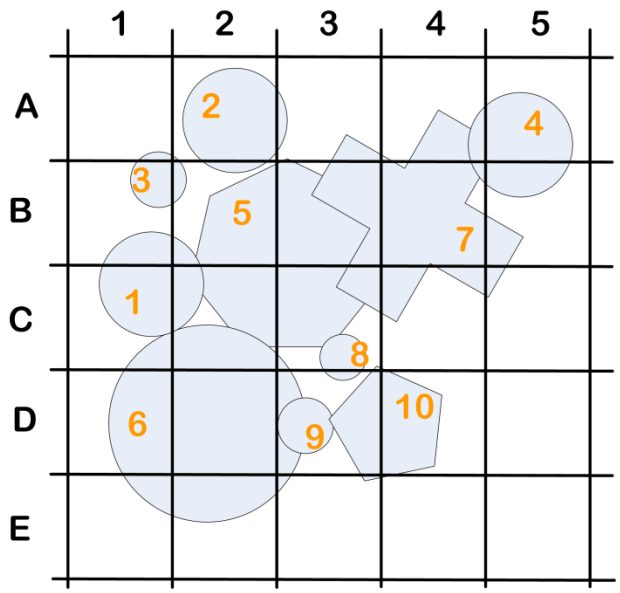
Proximity Computation

GPU Collision Detection (CD)

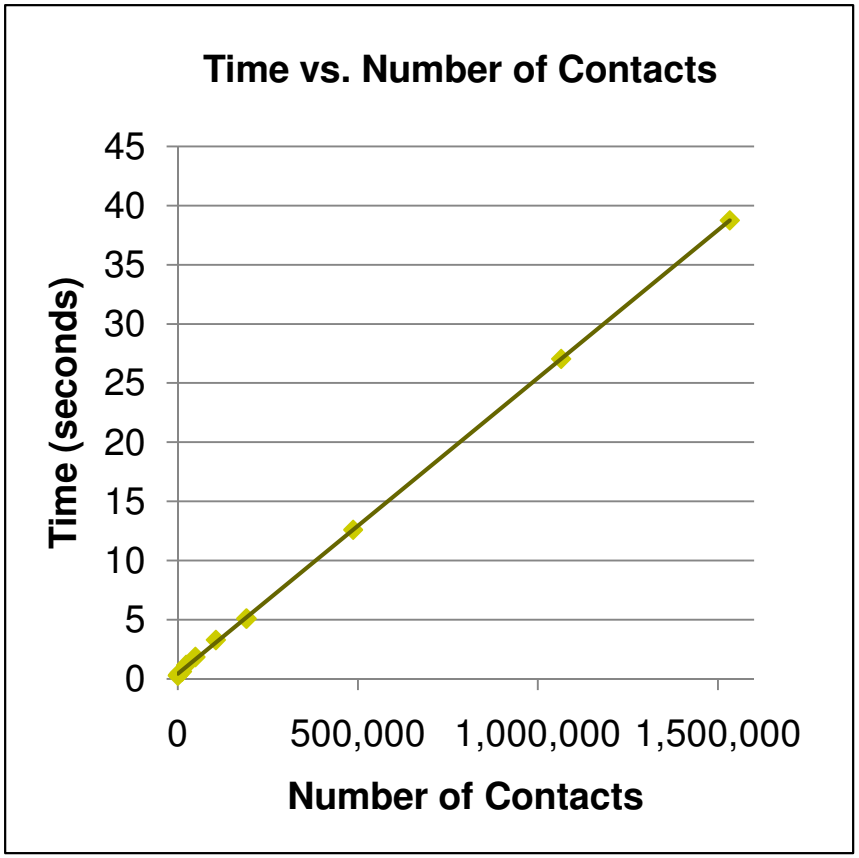
- 30,000 feet perspective:
  - Carry out spatial partitioning of the volume occupied by the bodies
    - Place bodies in bins (cubes, for instance)
  - Follow up by brute force search for all bodies touching each bin
    - Embarrassingly parallel

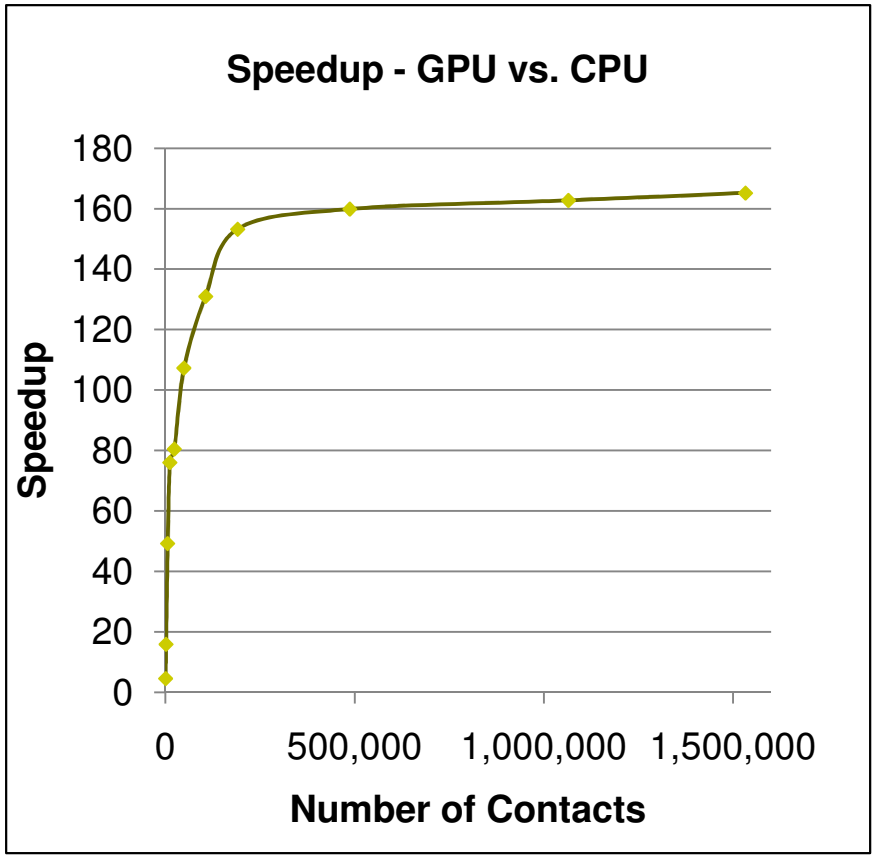
Basic Idea:
Search for Contacts in Different Bins in Parallel

- Example: 2D collision detection, bins are squares



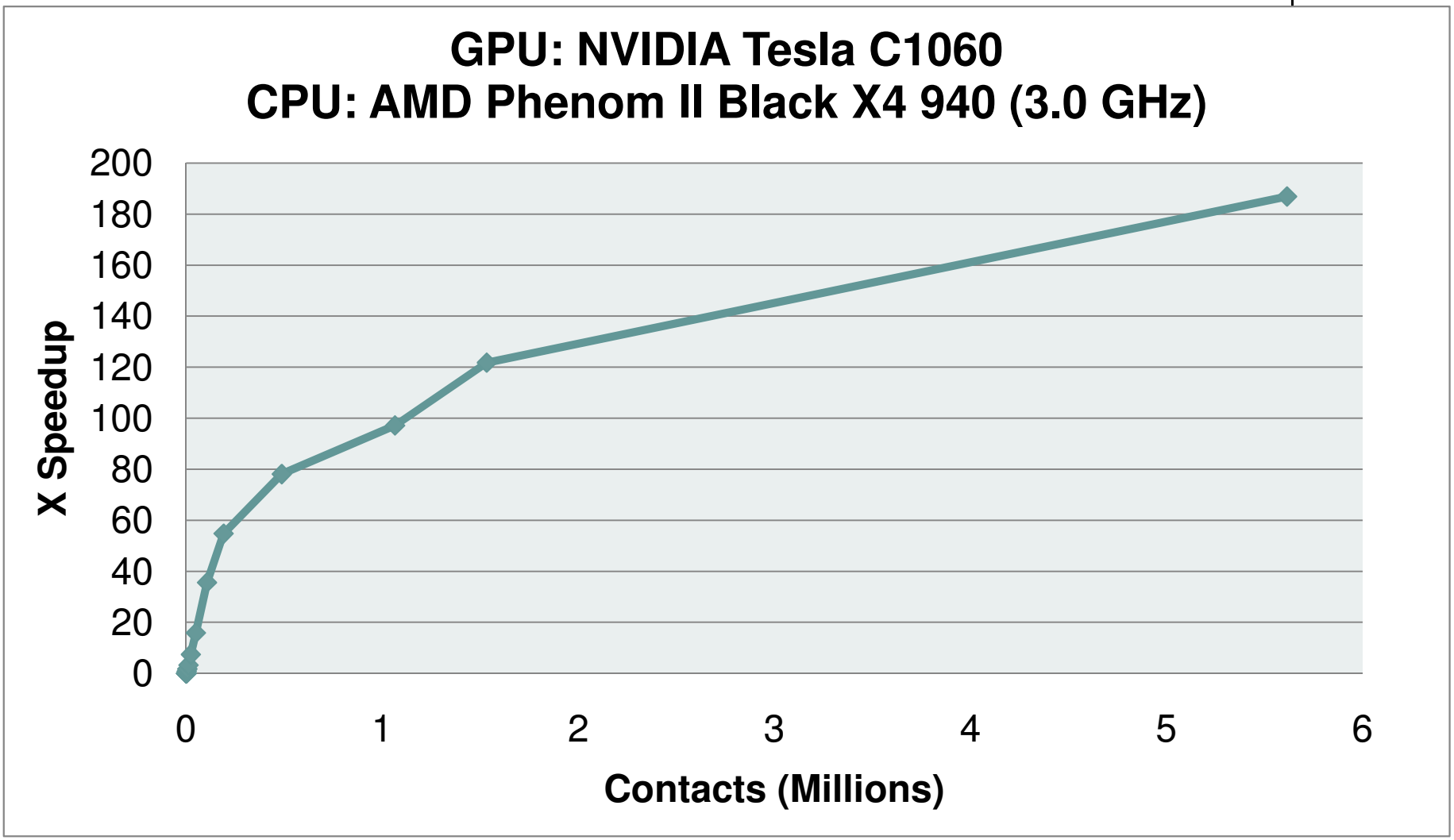
Ellipsoid-Ellipsoid CD: Results



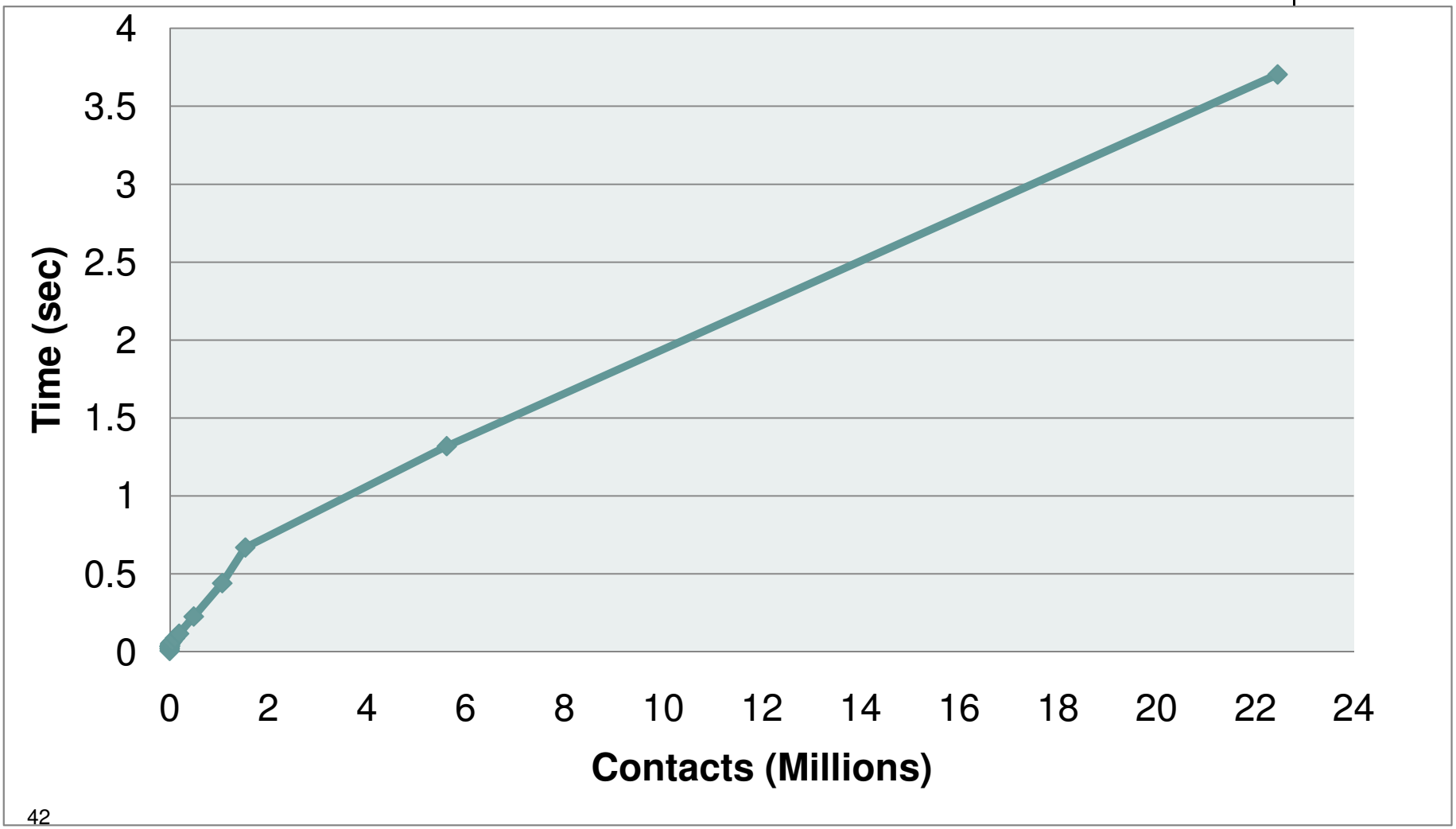


Speedup - GPU vs. CPU (Bullet library)

[results reported are for spheres]

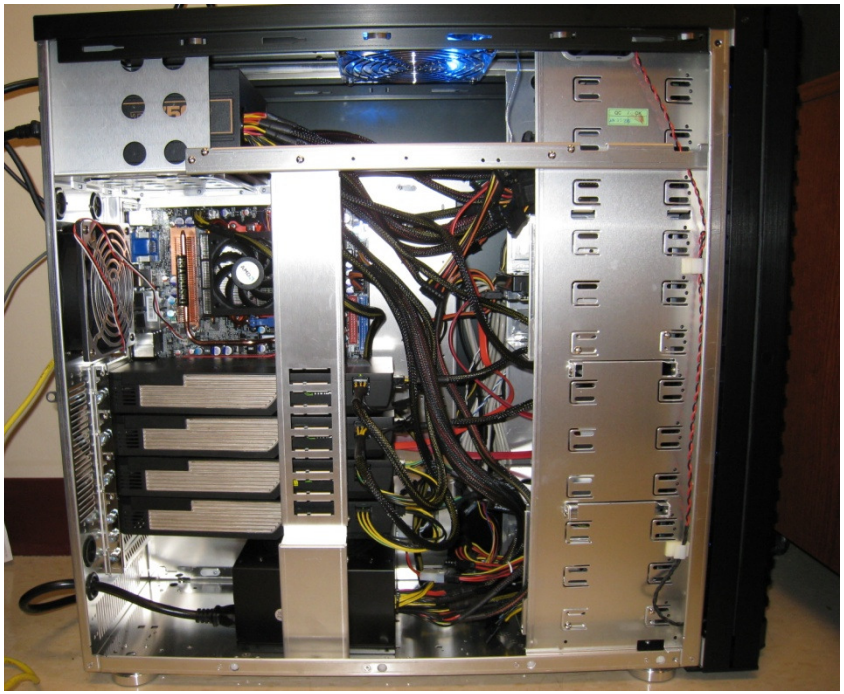


Parallel Implementation: Number of Contacts vs. Detection Time [results reported are for spheres]



Multiple-GPU Collision Detection

Assembled Quad GPU Machine



Processor: AMD Phenom II X4 940 Black

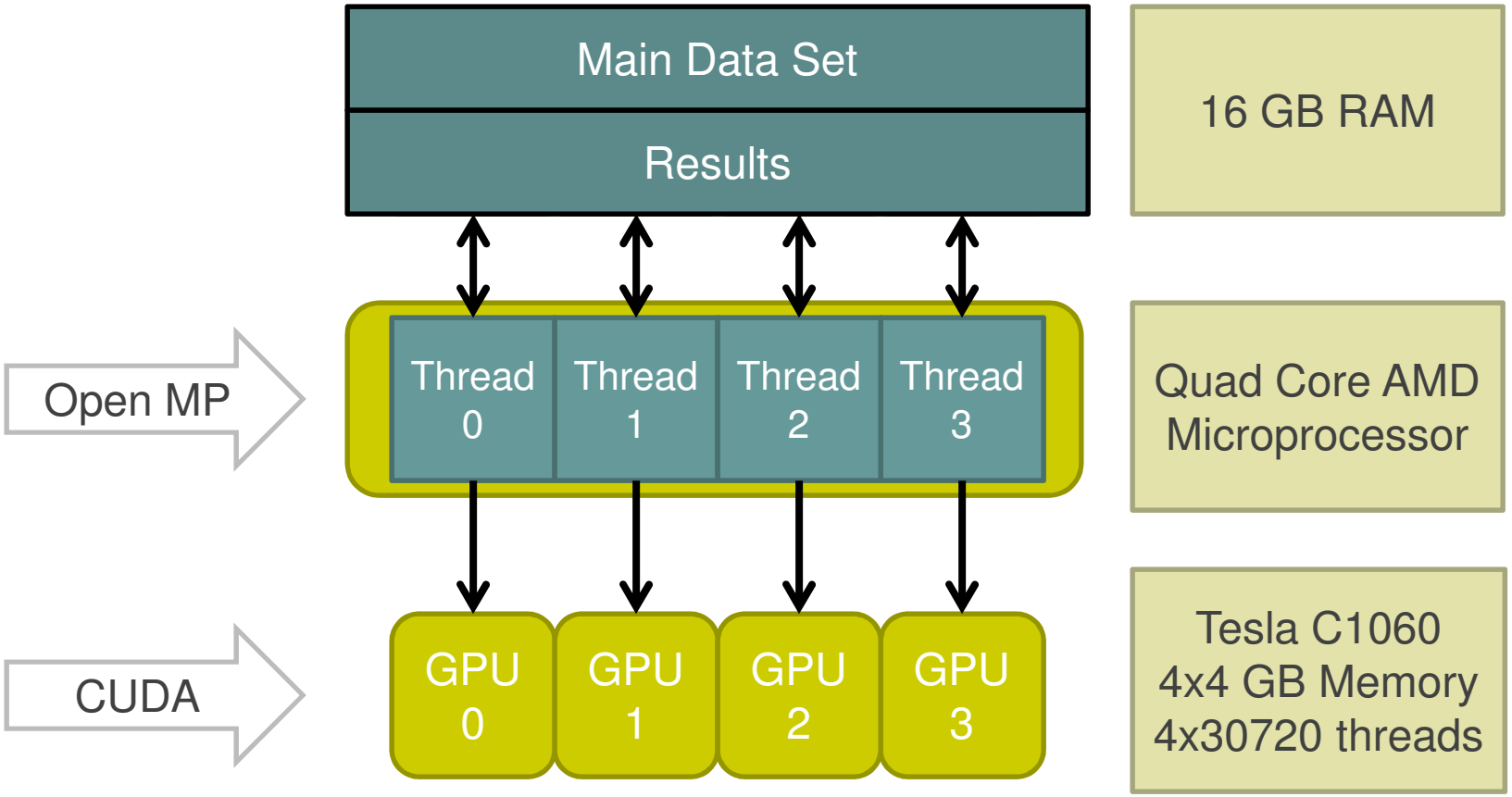
Memory: 16GB DDR2

Graphics: 4x NVIDIA Tesla C1060

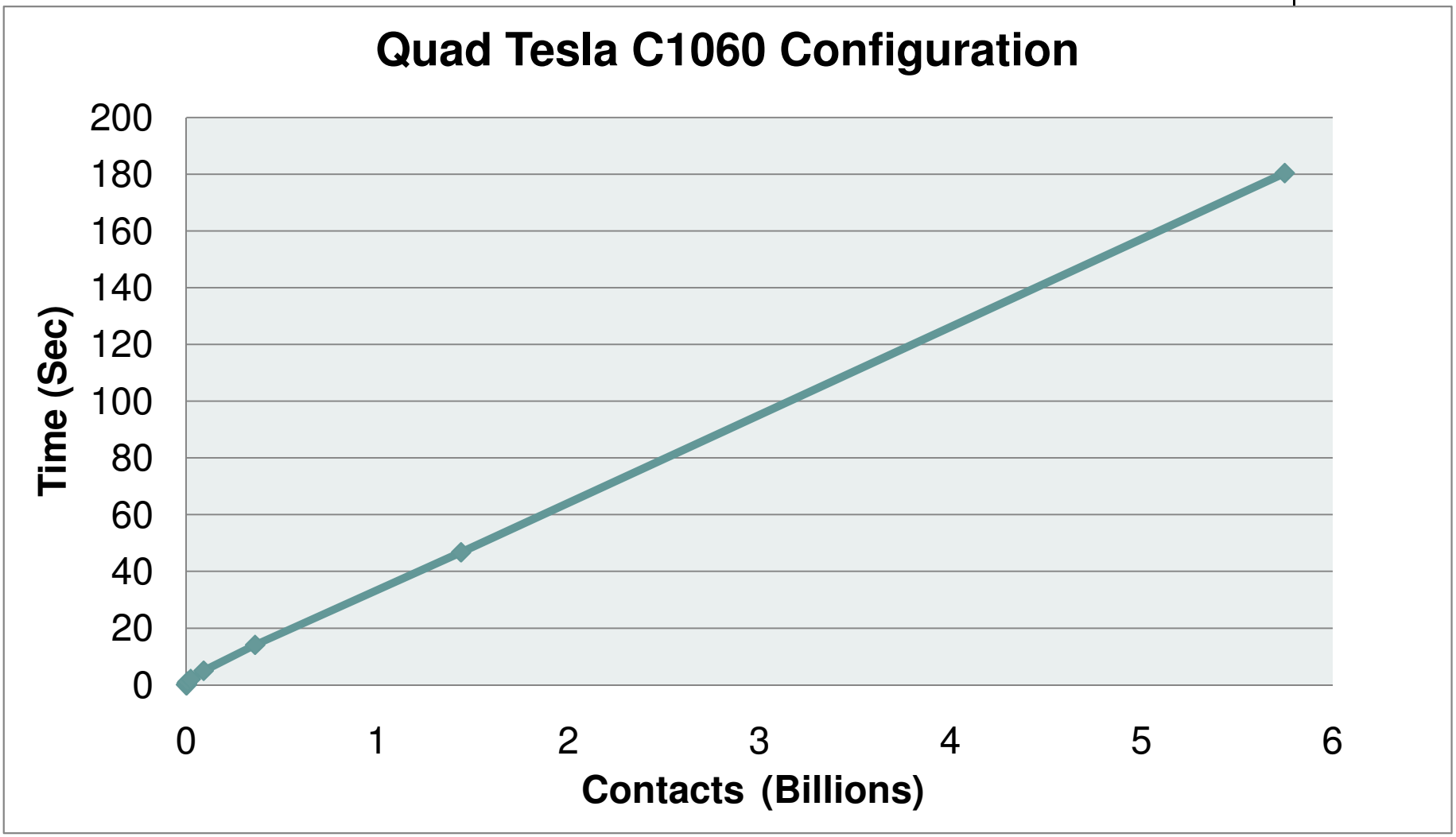
Power supply 1: 1000W

Power supply 2: 750W

SW/HW Setup


Main Data Set
Results
16 GB RAM
Open MP
Thread 0
Thread 1
Thread 2
Thread 3
Quad Core AMD Microprocessor
CUDA
GPU 0
GPU 1
GPU 2
GPU 3
Tesla C1060
4x4 GB Memory
4x30720 threads

Results – Contacts vs. Time



Conclusions

- Work aimed at enabling high-fidelity discrete models using a physics-based approach
- Approach draws on unique CPU + GPU parallel approach, which leverages a Heterogeneous Computing Template
- Accomplishments to date
  - Billion body parallel collision detection
  - Parallel solution of cone complementarity problem, about 12 million unknowns
  - Early validation results encouraging
- Aiming at billion bodies simulations

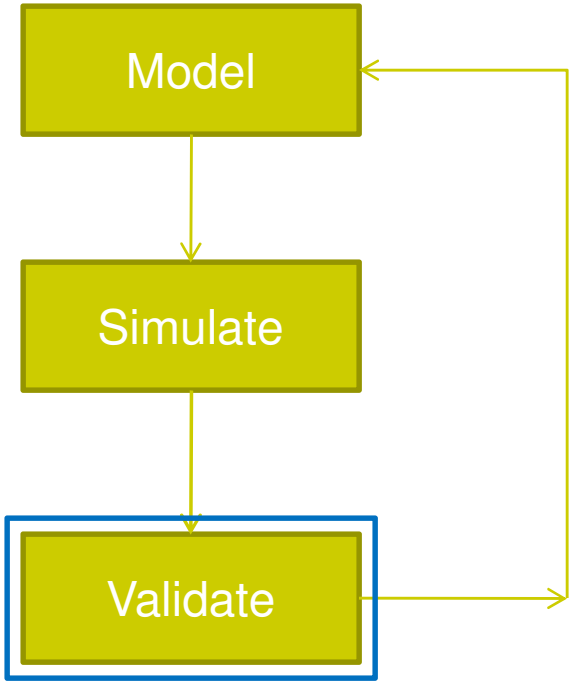
Ongoing/Future Work

- Massively parallel linear algebra for solution of CCP problem
- More general collision detection code
- Multiphysics:
  - Fluid-solid interaction
  - Electrostatics

Thank You.

Validation.

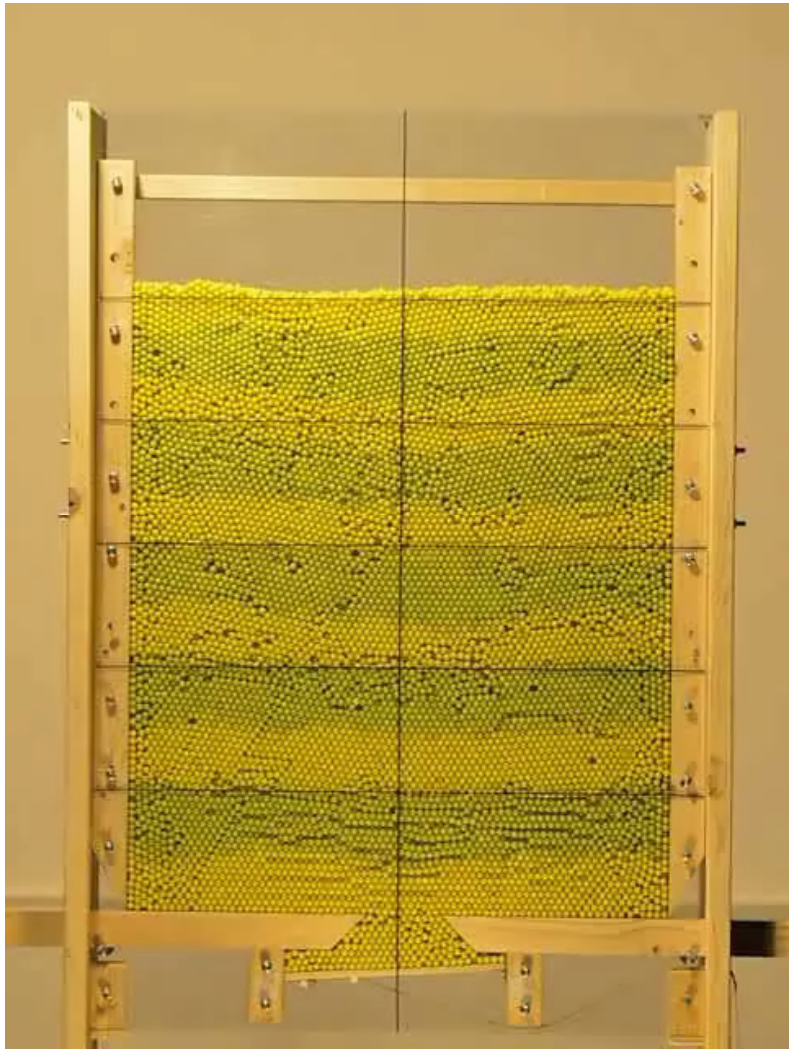
Simulation is doomed to succeed.
(Rod Brooks, roboticist)



- Validation at “microscale” – University of Wisconsin-Madison
  - Work in progress

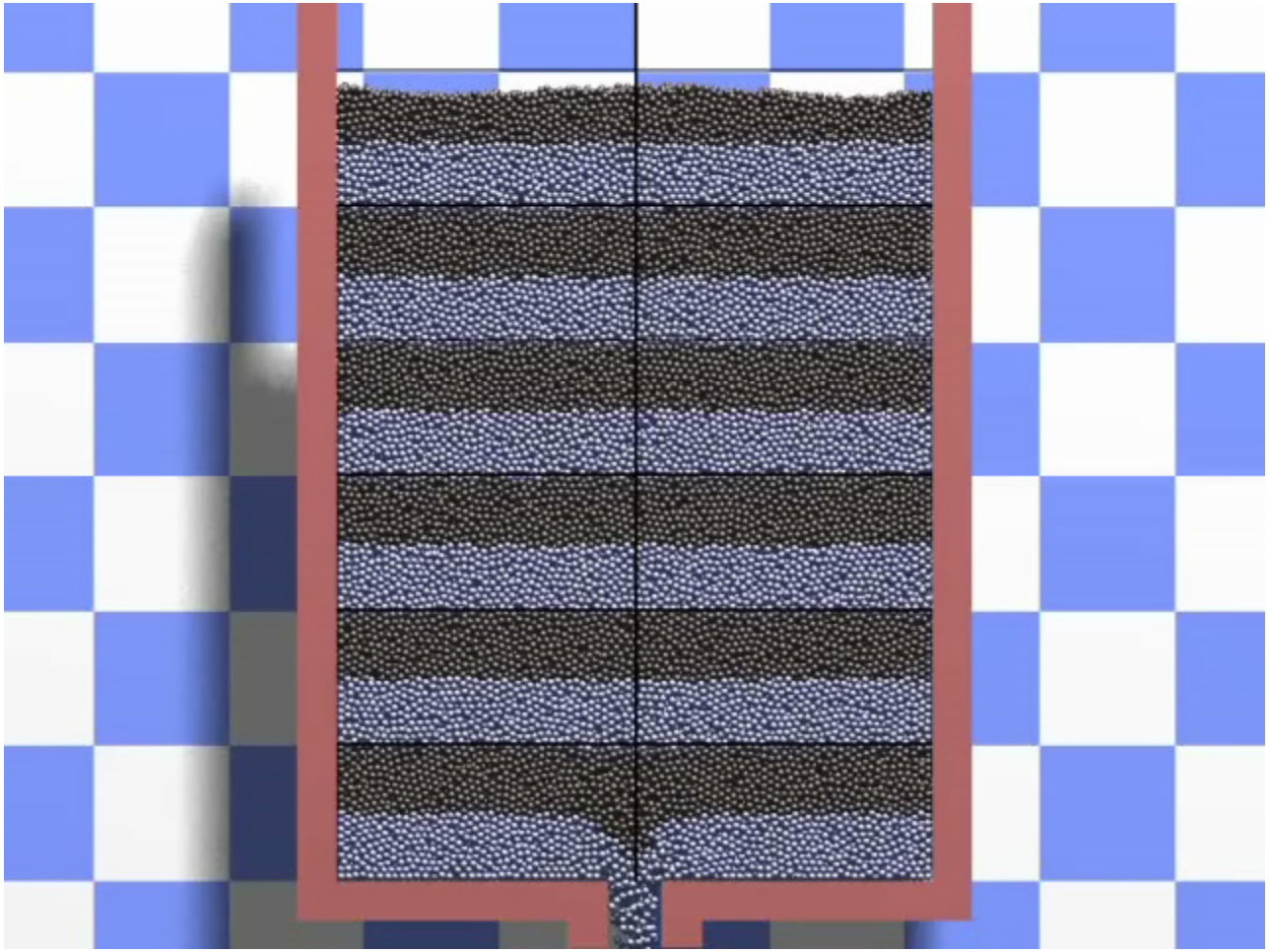
- Validation at “macroscale” – University of Parma, Italy

Flat Hopper Tests



Video recording from a test (a case that starts from high crystallization)

Flat Hopper Tests

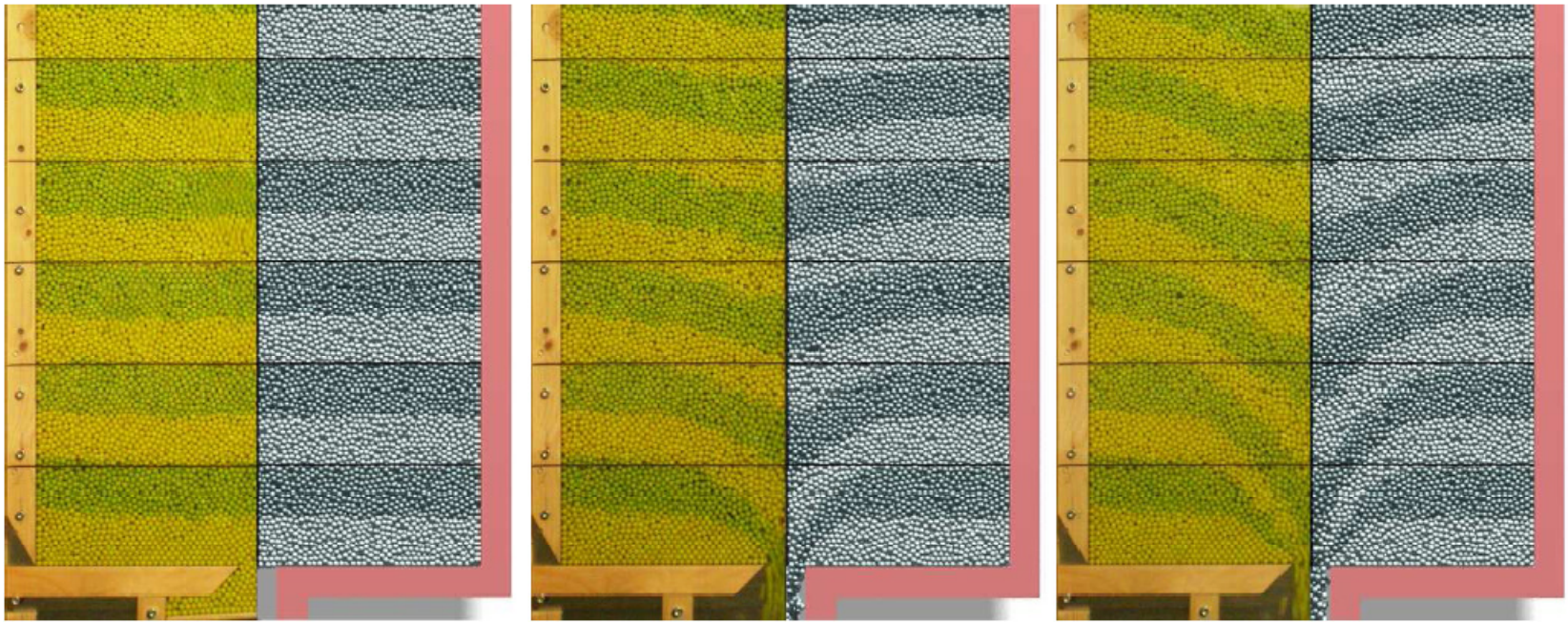


3D rendering from a simulation (4x slower than real-time)

Flat Hopper Tests

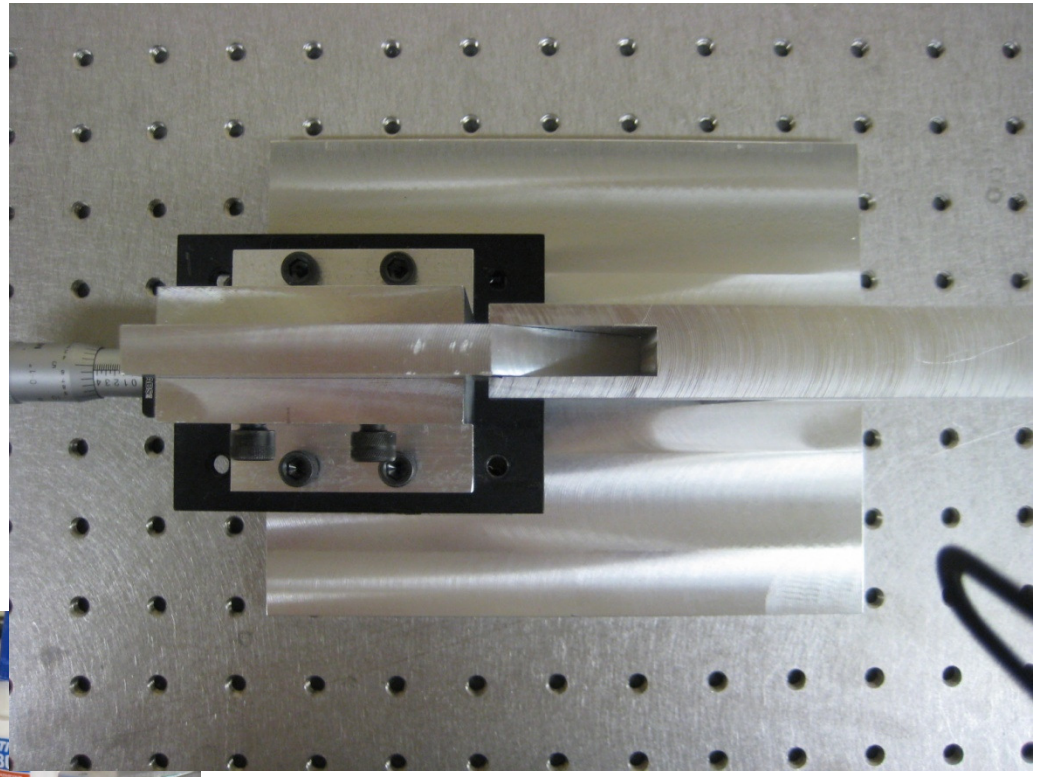
- Comparison experimental - simulated

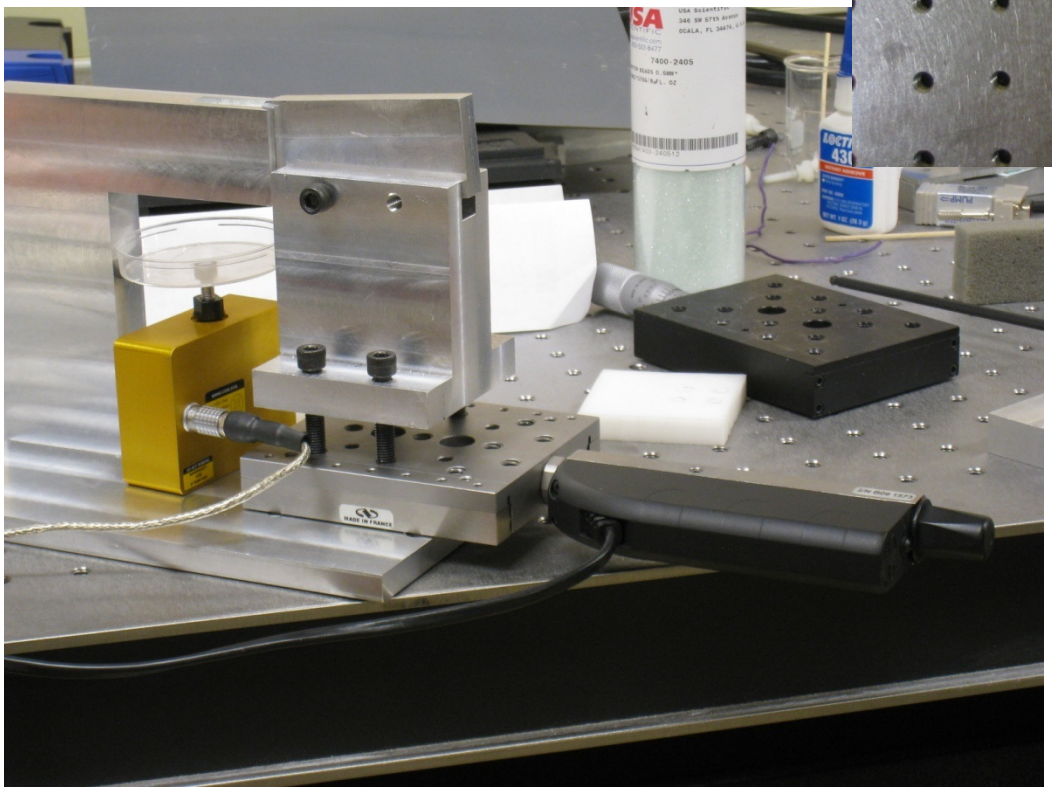
Experimental Simulated

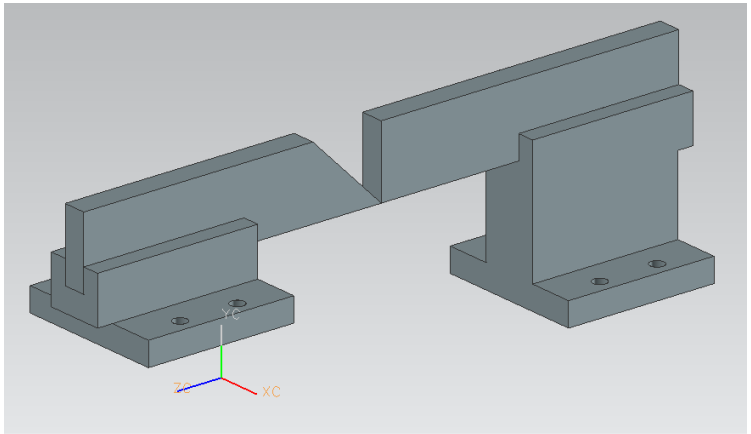


Validation at Microscale

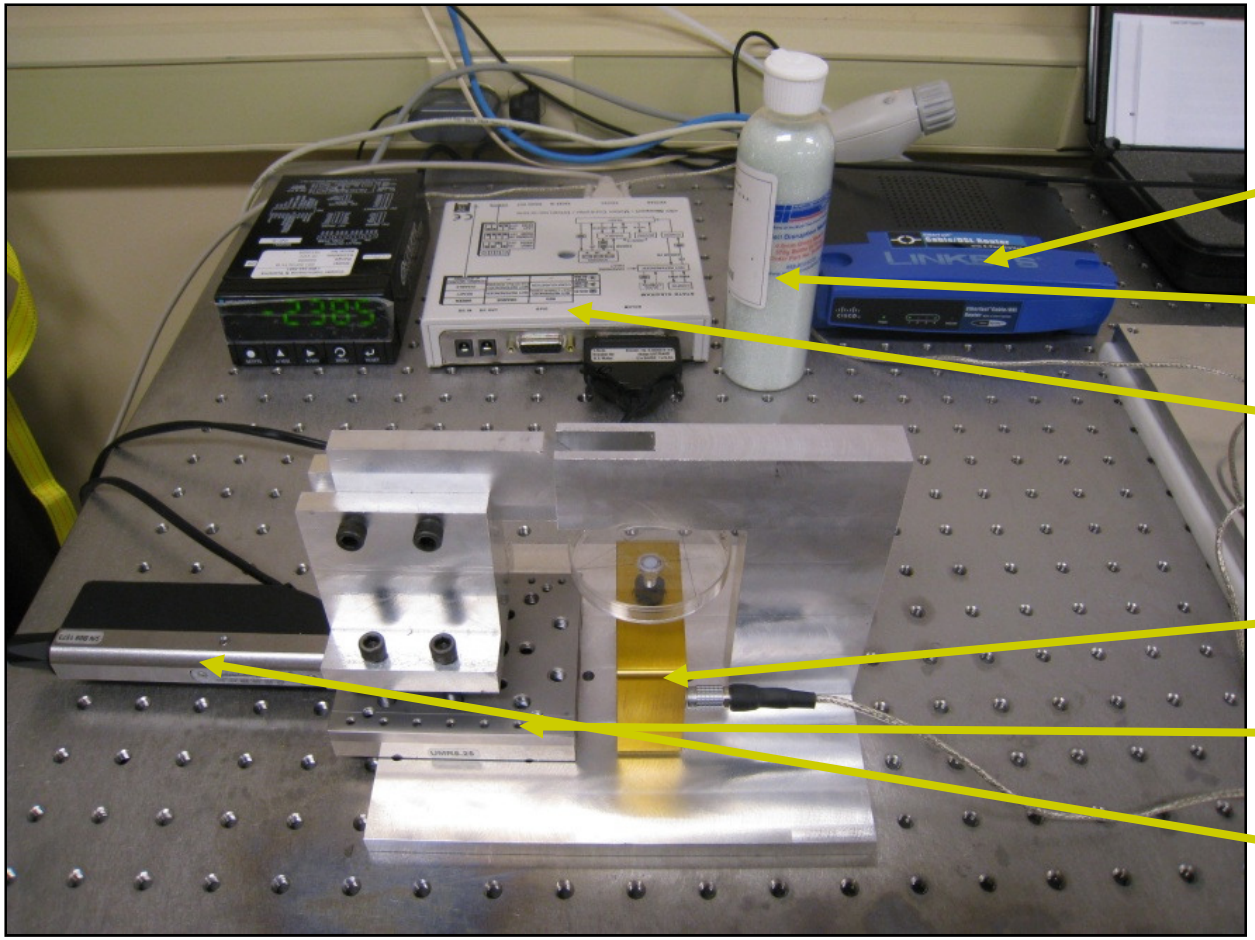
- Sand flow rate measurements
- Approx. 40K bodies
- Glass beads
- Diameter: 100-500 microns



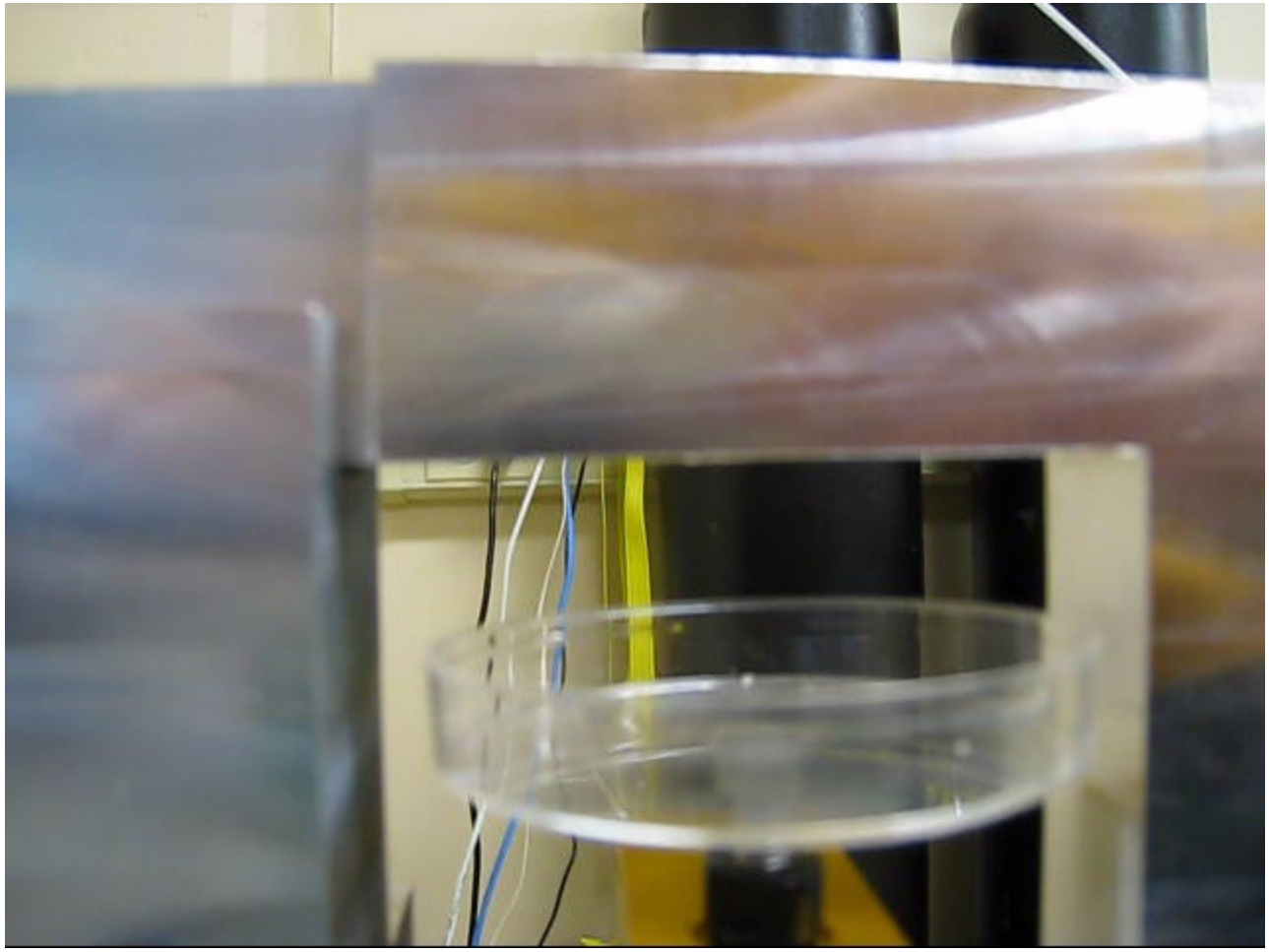




Experimental Setup

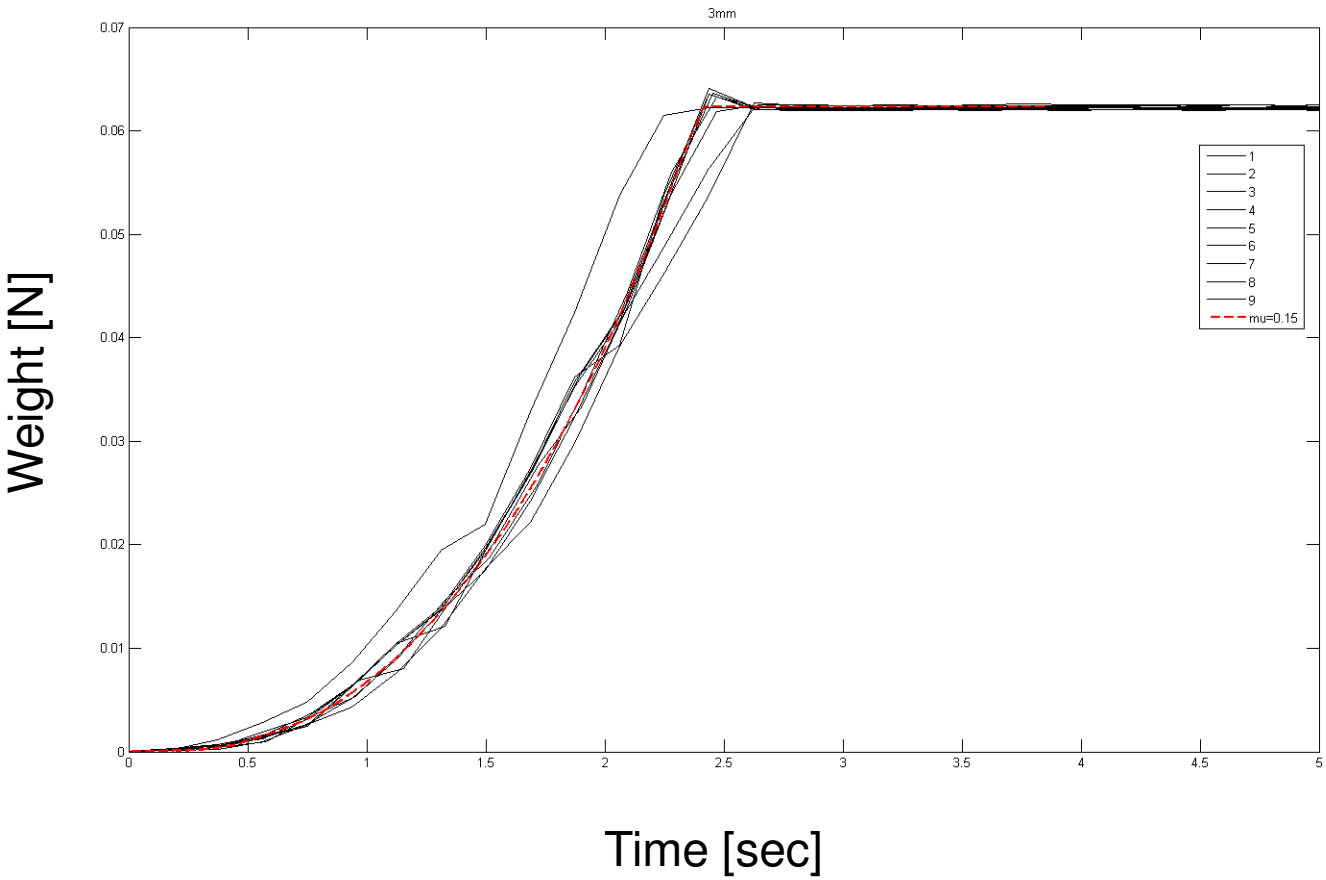


Flow Measurement, 500 micron Spheres

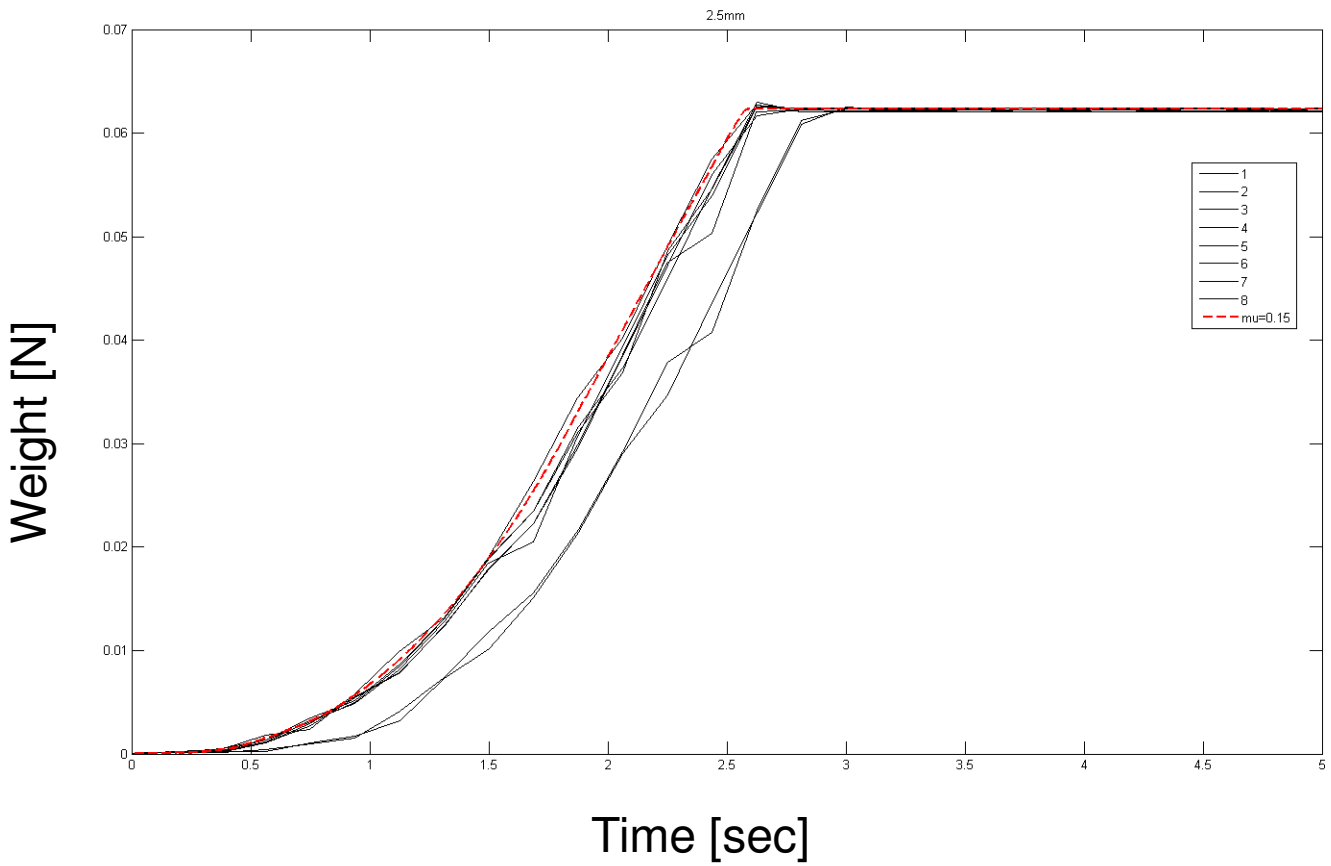


Flow Simulation, 500 micron Spheres

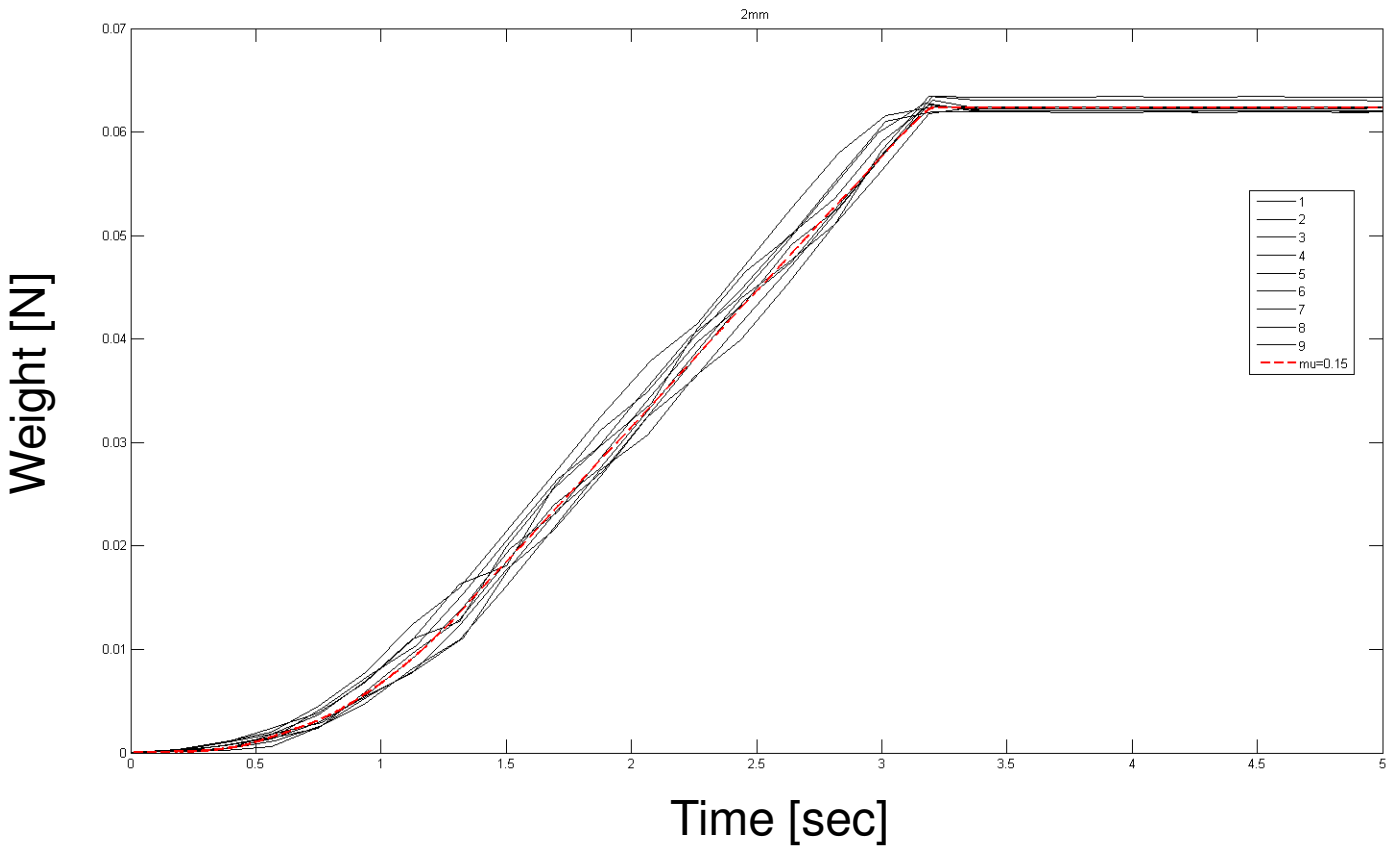
Flow Measurement Results, 3mm Gap Size



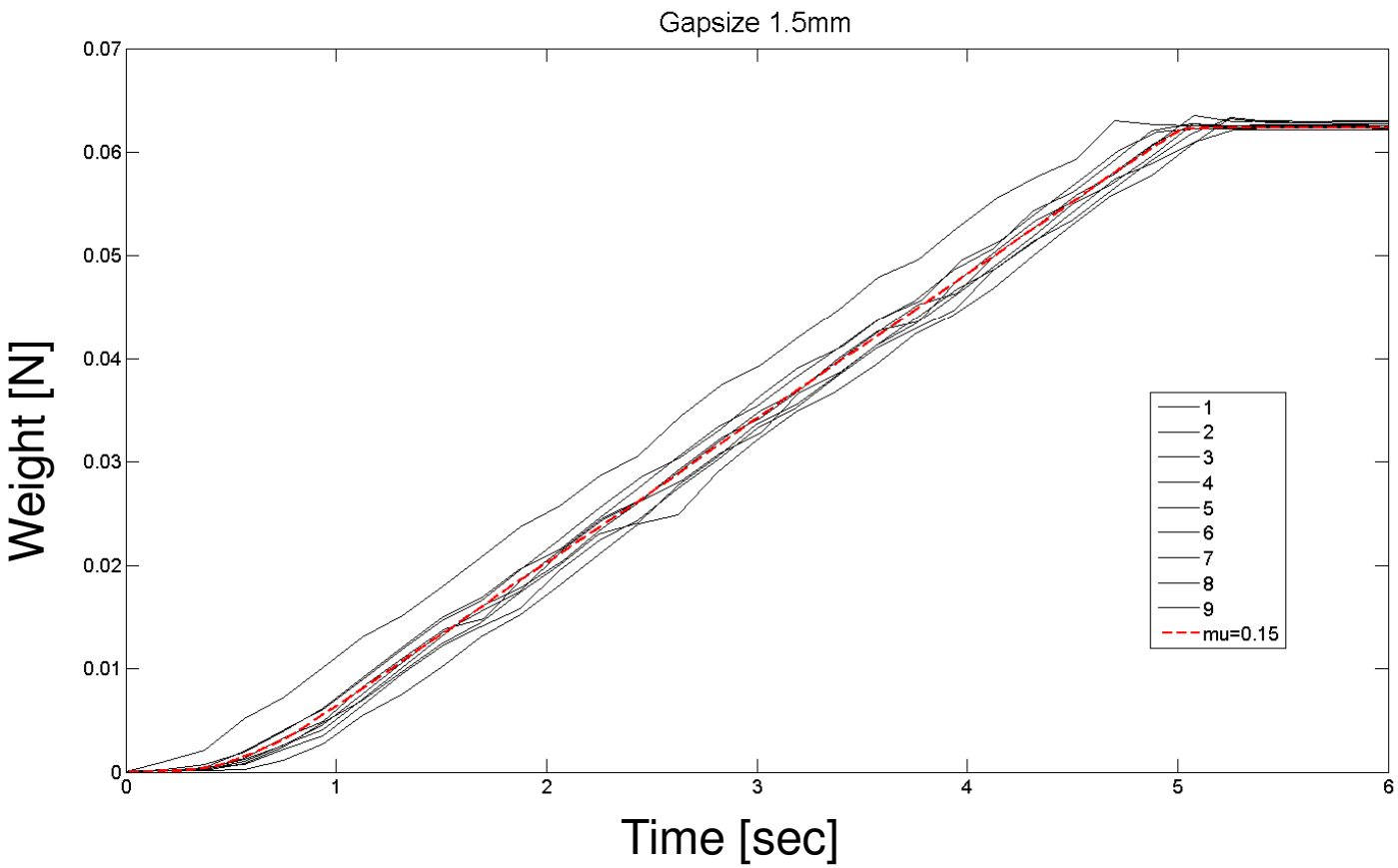
Flow Measurement Results, 2.5mm Gap Size



Flow Measurement Results, 2mm Gap Size



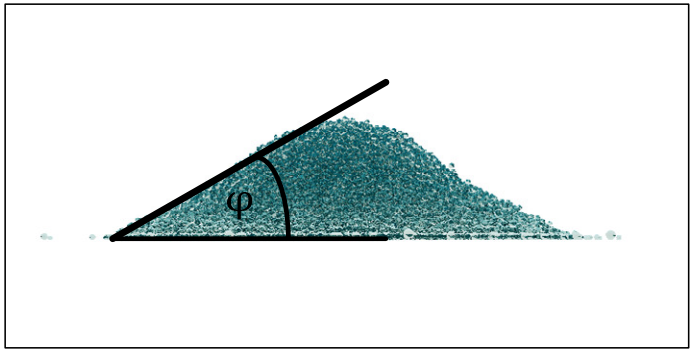
Flow Measurement Results, 1.5mm Gap Size



Validation Experiment: Repose Angle

- Experiment

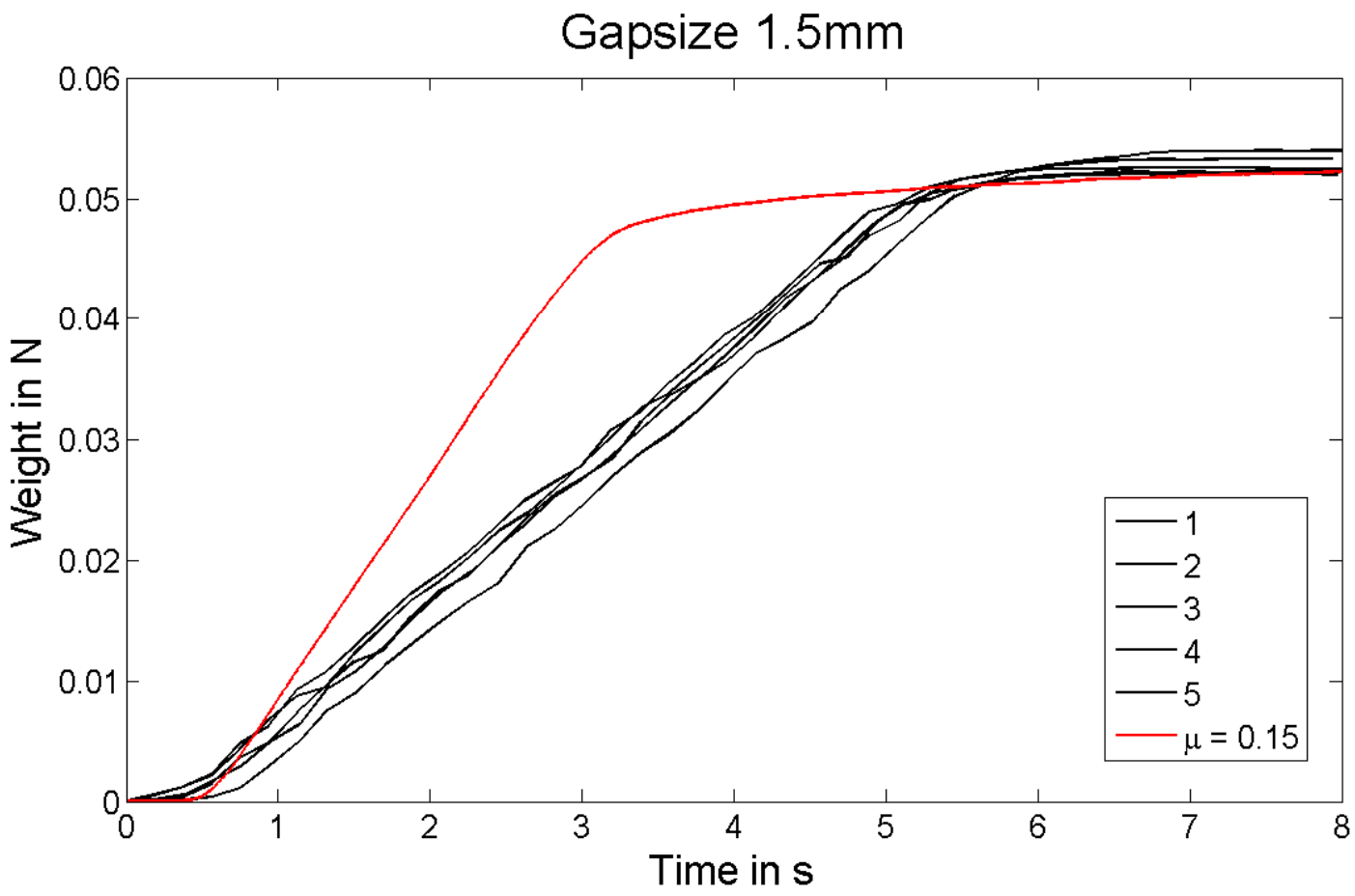
- Simulation



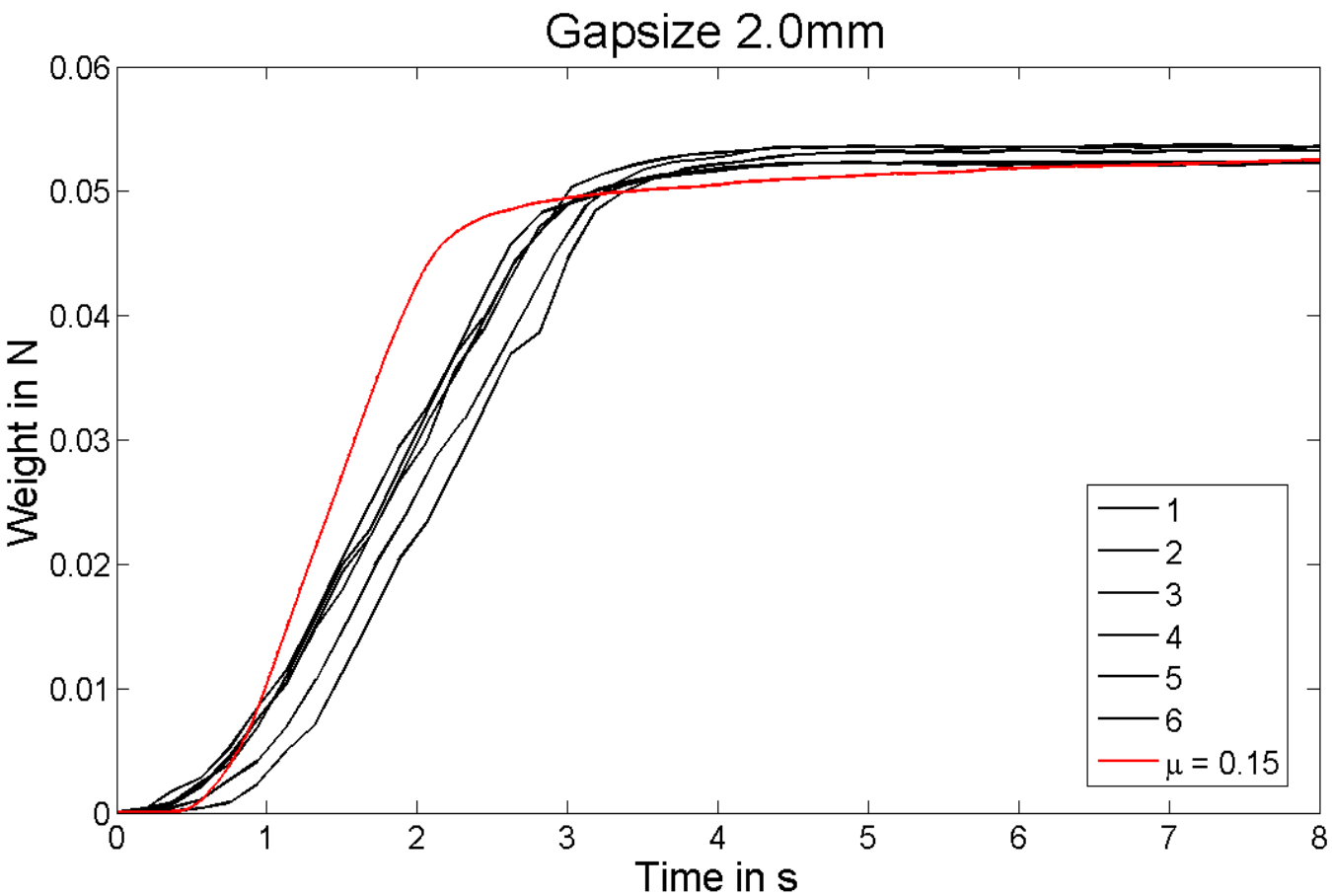
$\phi = 19.5^\circ \qquad \text{for} \quad \mu = 0.39$

Validation Experiment Flow and Stagnation

Validation, Flow and Stagnation

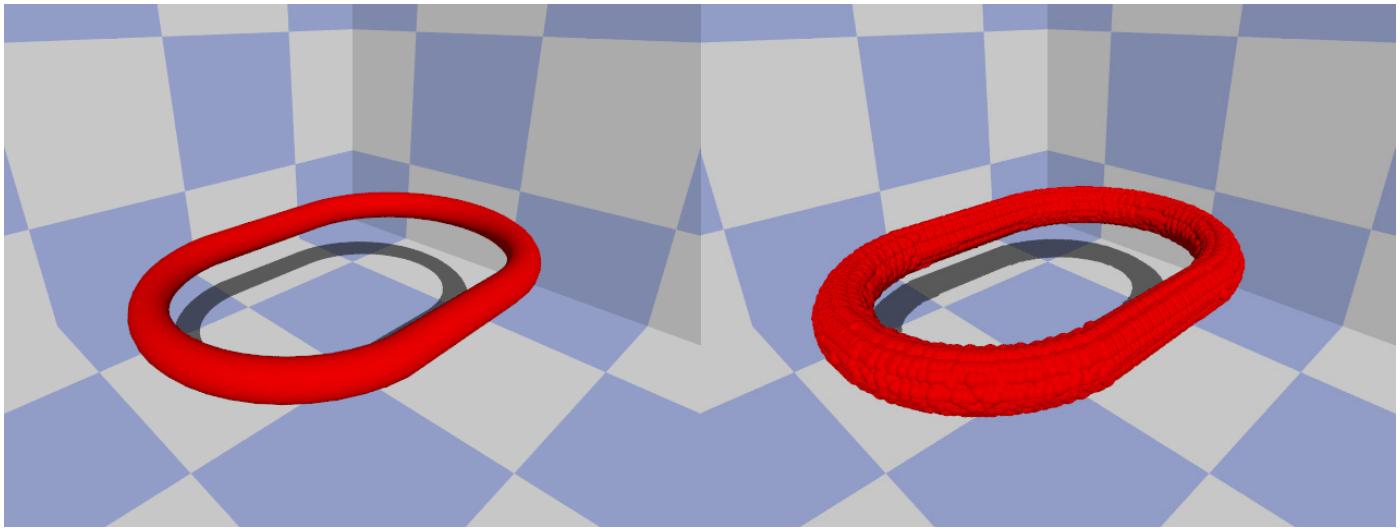


Validation, Flow and Stagnation



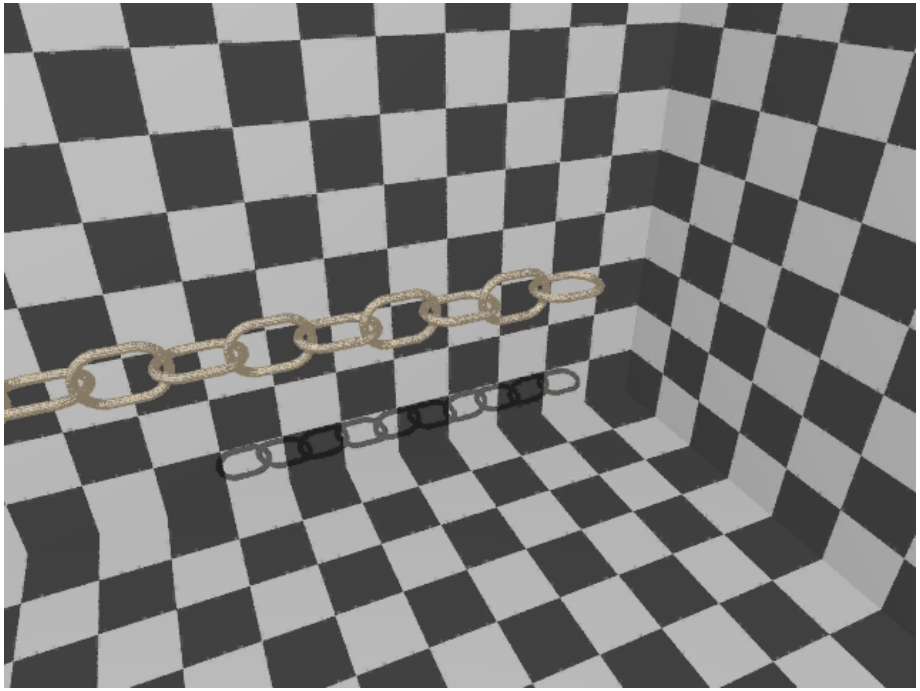
Spherical Decomposition

- Represent complex geometry as a union of spheres
  - Fast parallel collision detection on GPU
  - Allows non-convex geometry
  - NOTE: Used only for CD, the dynamics is performed using original geometry

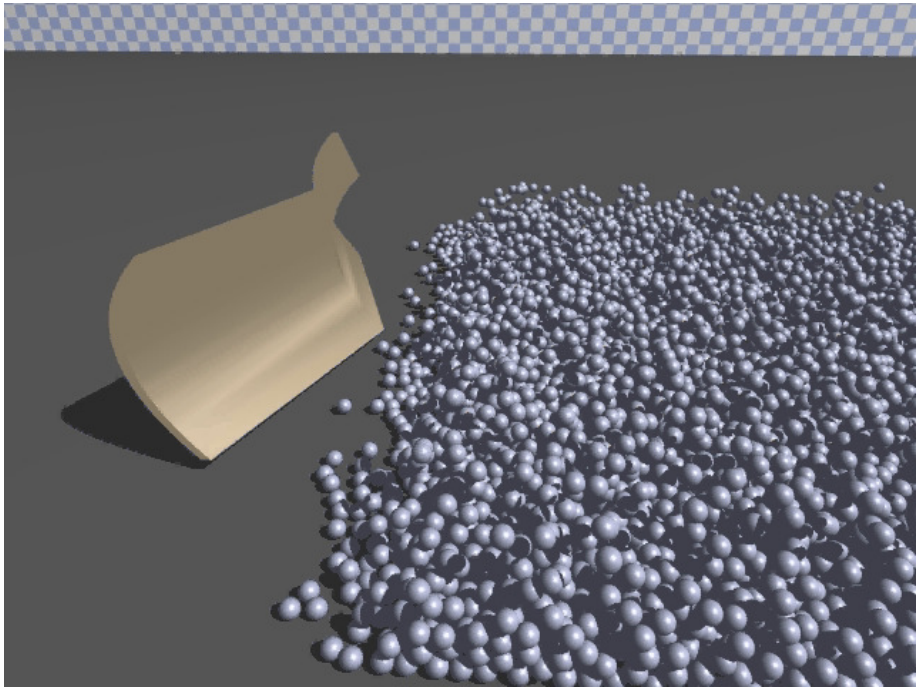


Examples…

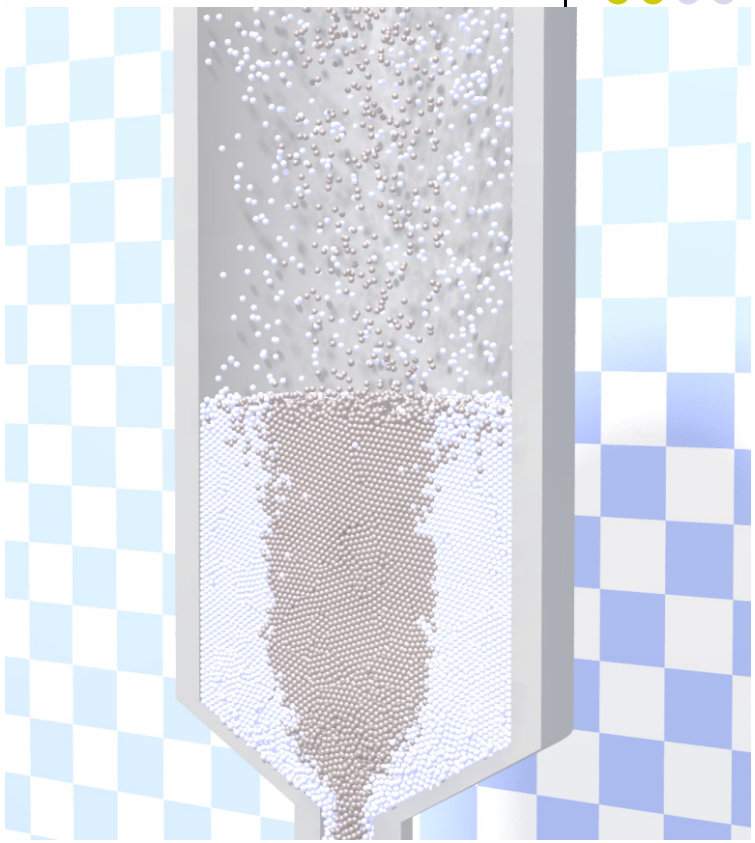
- Chain model
  - 10 links
  - 7,797 spheres per link



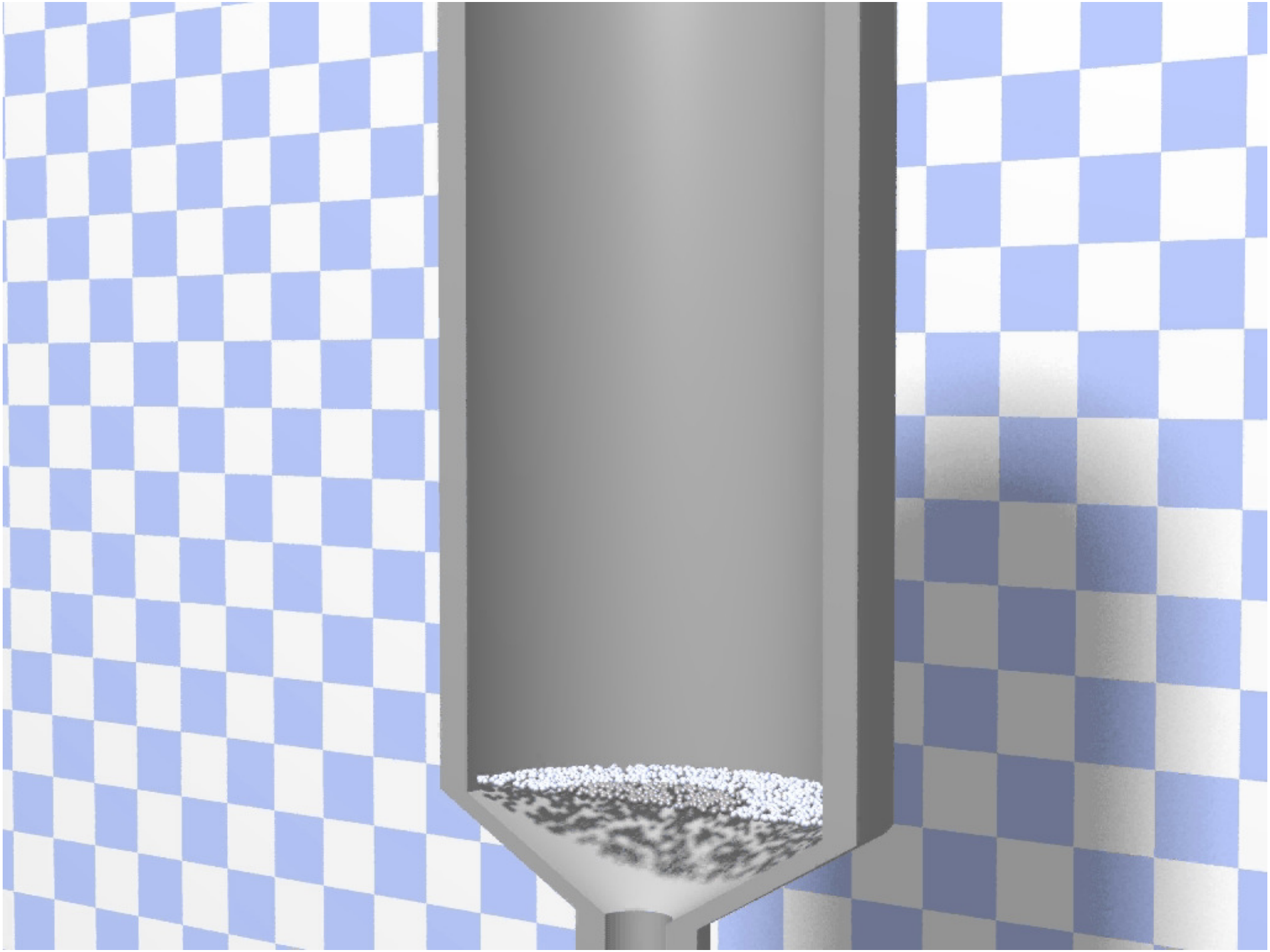
- Plow model
  - 31,791 spheres in plow blade model
  - 15,000 spheres representing terrain



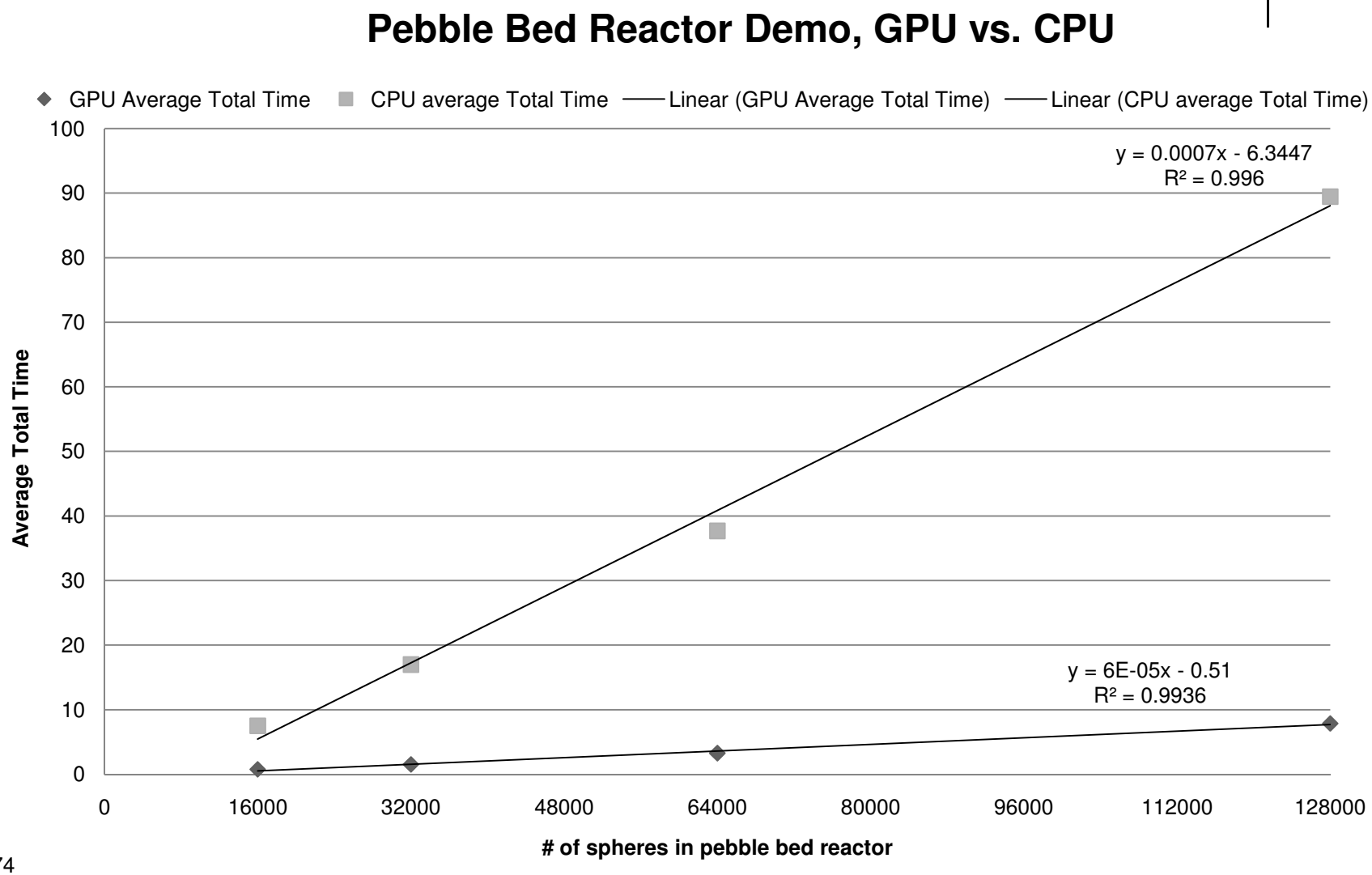
Numerical Results: Pebble Bed Nuclear Reactor



- Two types of tests were run
- On the GPU
  - CD: NVIDIA 8800 GT
  - CCP: NVIDIA Tesla C870
- On the CPU
  - Single threaded
  - Quad Core Intel Xeon E5430
  - 2.66 GHz
- The reactor contains spheres which flow out the bottom the nozzle and are recycled to the top of the reactor
- Performed simulations with 16K, 32K, 64K, and 128K bodies

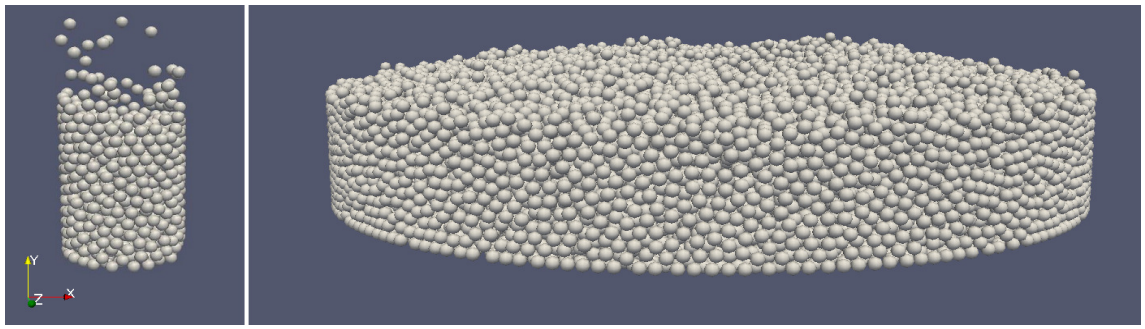


Results: Average Duration for Taking One Integration Step [Δt=0.01 s]

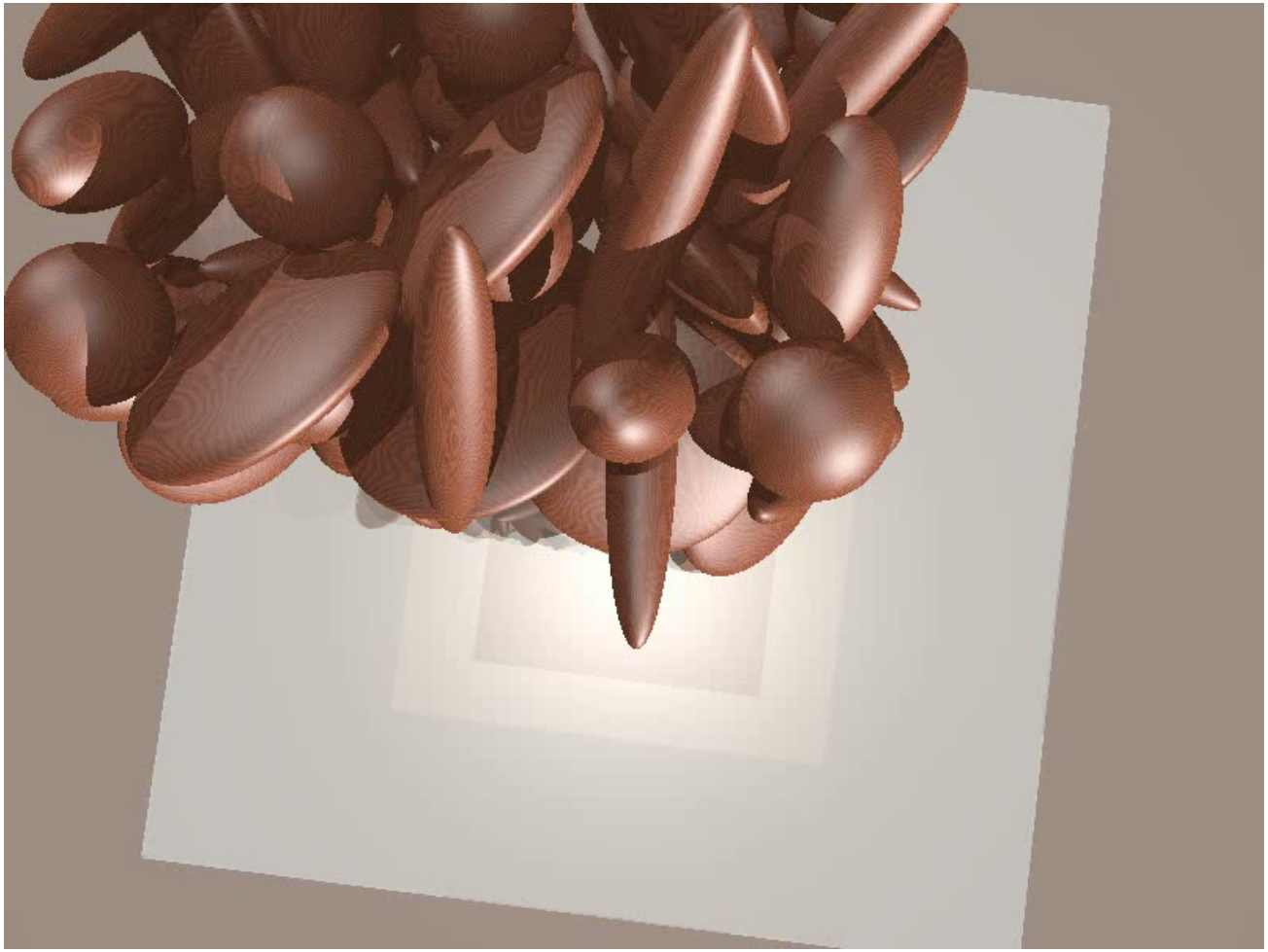


Algorithmic Challenges: Gauss-Jacobi Doesn't Scale Well

- Convergence stalls for certain classes of problems
  - Very large problems
  - Problems where I have many body stacked on each other
    - Inherent problem with Gauss-Jacobi, pertains the propagation of information
- Granular dynamics problem have an intrinsic degree of redundancy



Ellipsoid-Ellipsoid CD: Visualization



Example: Ellipsoid-Ellipsoid CD

$$\mathbf{d} = \mathbf{P}_1 - \mathbf{P}_2 = (\frac{1}{2\lambda_1}\mathbf{M}_1 + \frac{1}{2\lambda_2}\mathbf{M}_2)\mathbf{c} + (\mathbf{b}_1 - \mathbf{b}_2)$$

$$\frac{\partial \mathbf{d}}{\partial \alpha_i} = \frac{\partial \mathbf{P}_1}{\partial \alpha_i} - \frac{\partial \mathbf{P}_2}{\partial \alpha_i} \quad , \quad \frac{\partial^2 \mathbf{d}}{\partial \alpha_i \partial \alpha_j} = \frac{\partial^2 \mathbf{P}_1}{\partial \alpha_i \partial \alpha_j} - \frac{\partial^2 \mathbf{P}_2}{\partial \alpha_i \partial \alpha_j}$$

$$\boxed{\frac{\partial \mathbf{P}}{\partial \alpha_i} = (\frac{1}{2\lambda}\mathbf{M} - \frac{1}{8\lambda^3}\mathbf{M}\mathbf{c}\mathbf{c}^T\mathbf{M})\frac{\partial \mathbf{c}}{\partial \alpha_i}}$$

$$\boxed{\begin{aligned}\frac{\partial^2 \mathbf{P}}{\partial \alpha_i \partial \alpha_j} &= (-\frac{1}{8\lambda^3}\mathbf{M} + \frac{3}{32\lambda^5}\mathbf{M}\mathbf{c}\mathbf{c}^T\mathbf{M})\mathbf{c}^T\mathbf{M}\frac{\partial \mathbf{c}}{\partial \alpha_j}\frac{\partial \mathbf{c}}{\partial \alpha_i} \\ &\quad -\frac{1}{8\lambda^3}[(\mathbf{c}^T\mathbf{M}\frac{\partial \mathbf{c}}{\partial \alpha_i})\mathbf{M} + \mathbf{M}\mathbf{c}(\frac{\partial \mathbf{c}}{\partial \alpha_i})^T\mathbf{M}]\frac{\partial \mathbf{c}}{\partial \alpha_j} \\ &\quad +(\frac{1}{2\lambda}\mathbf{M} - \frac{1}{8\lambda^3}\mathbf{M}\mathbf{c}\mathbf{c}^T\mathbf{M})\frac{\partial^2 \mathbf{c}}{\partial \alpha_i \partial \alpha_j}\end{aligned}}$$

$$\varepsilon : \frac{x^2}{r_1^2} + \frac{y^2}{r_2^2} + \frac{z^2}{r_3^2} = 1$$

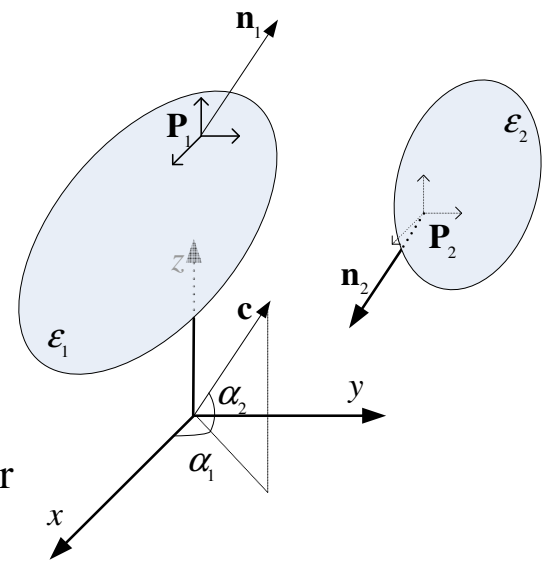
$\mathbf{A}$: Rotation Matrix

$$\mathbf{M} = \mathbf{A}\mathbf{R}^2\mathbf{A}^T$$

$$\mathbf{R} = diag(r_1, r_2, r_3)$$

$\mathbf{b}$: Translation of ellipsoids center

$$\lambda^2 = \frac{1}{4}\mathbf{n}^T\mathbf{M}\mathbf{n}$$

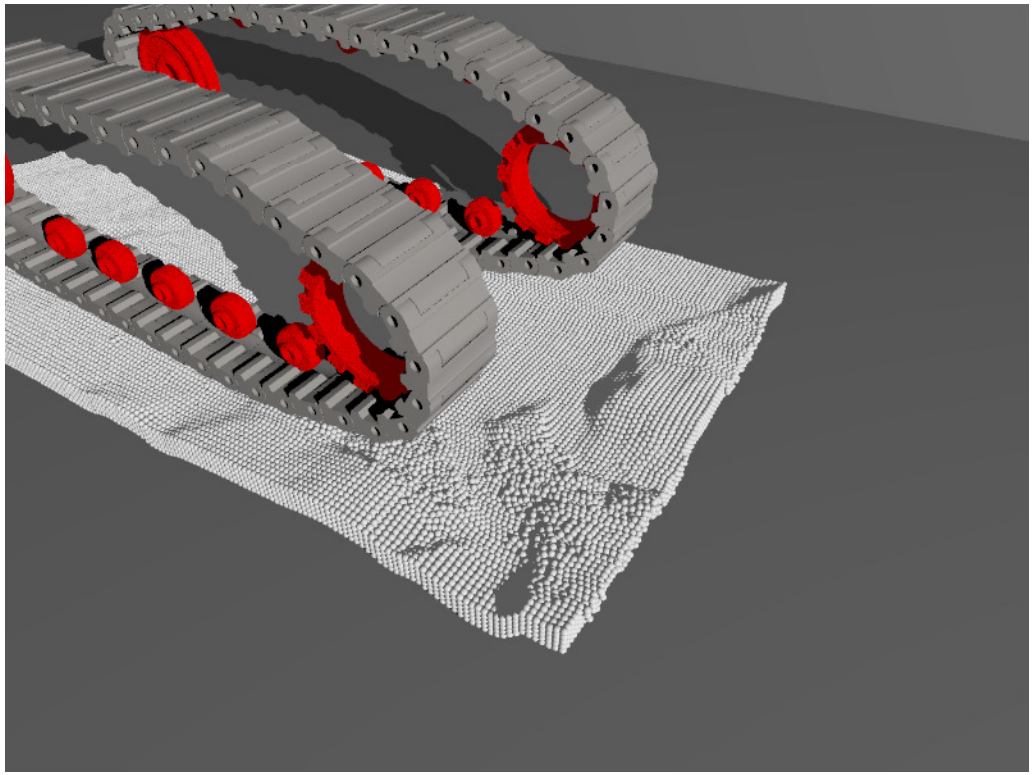


$$\mathbf{d} = \mathbf{P}_1 - \mathbf{P}_2$$

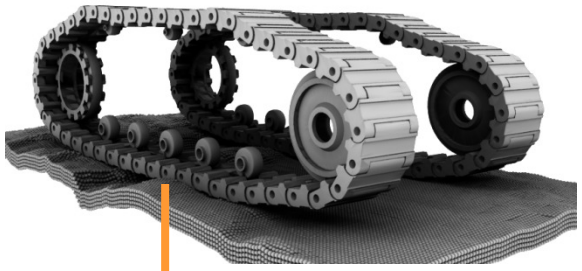
$$\min_{\alpha_1, \alpha_2} \| d(\alpha_1, \alpha_2) \|^2$$

Going Back to Practical Problem of Interest

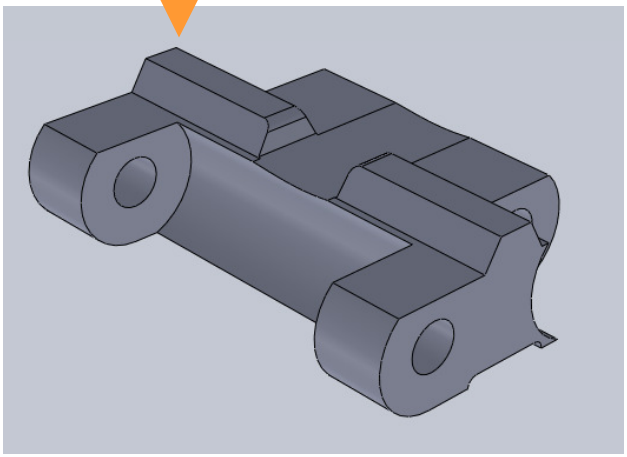
- Tracked Vehicle on Granular Terrain…



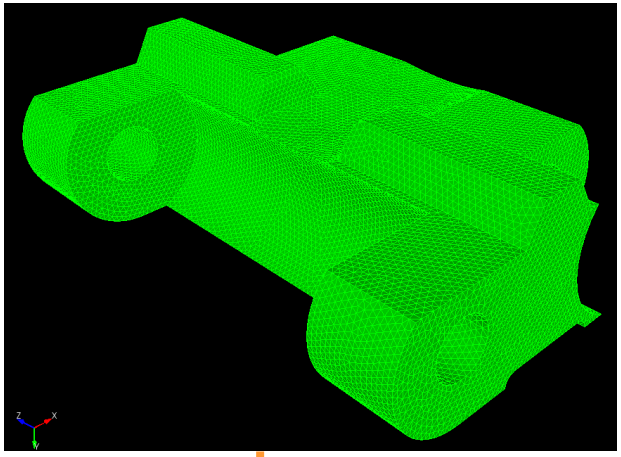
Spherical Decomposition Steps



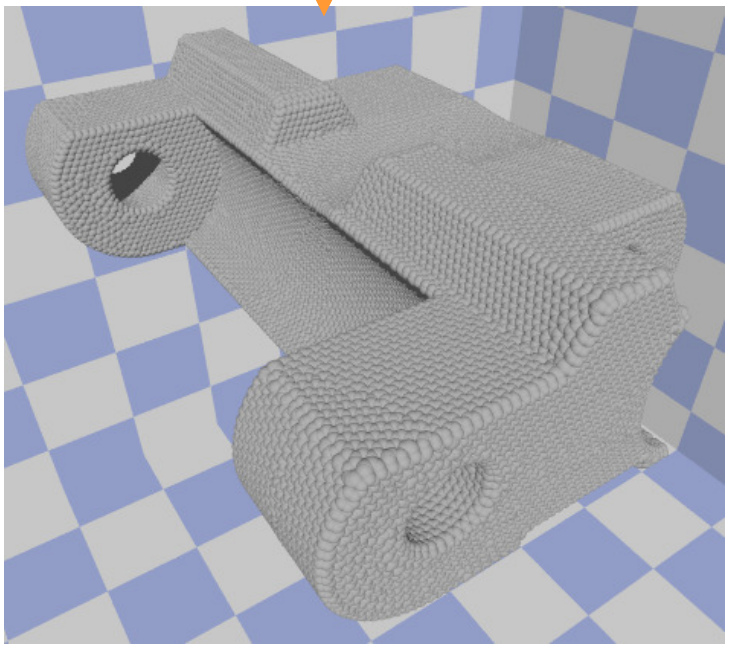
1: Take shoe element



2: Cubit

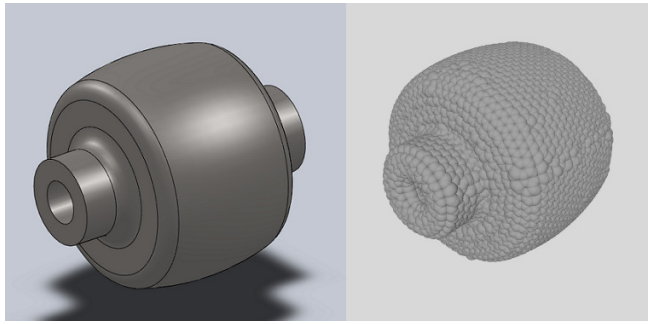


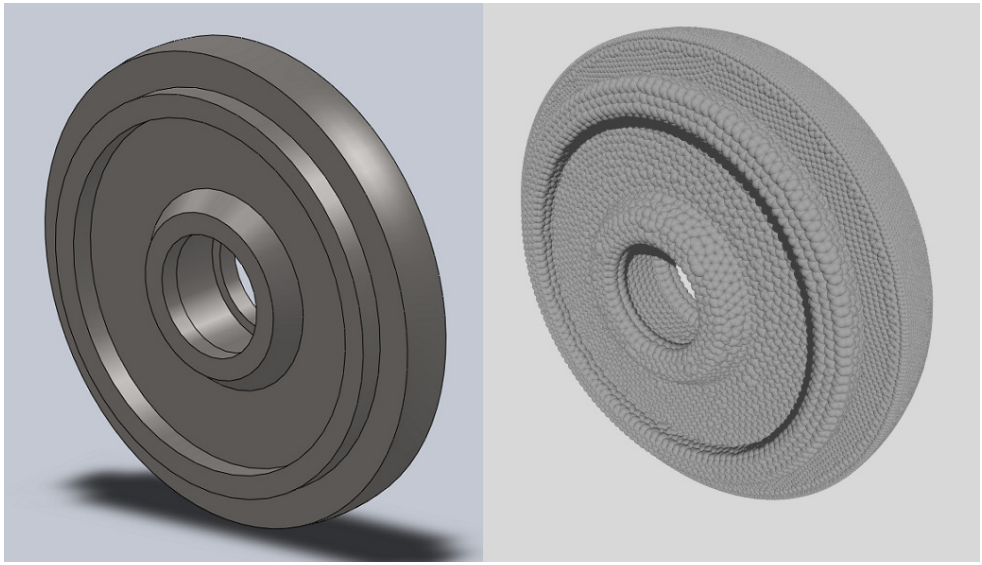
3: Spherical Padding

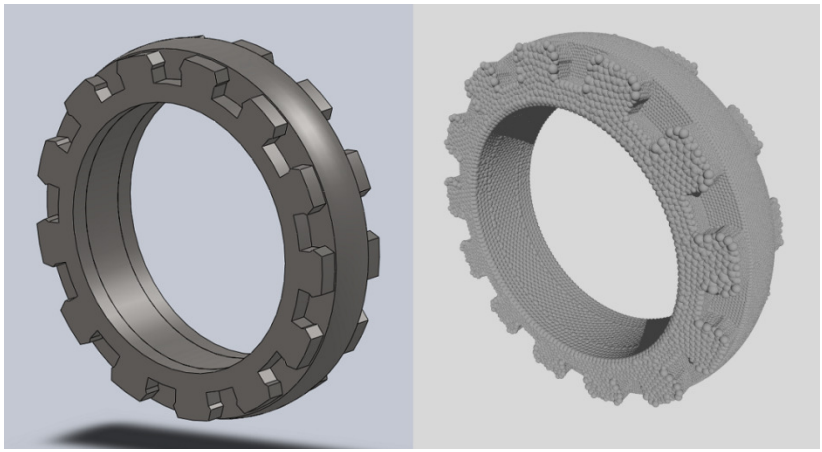


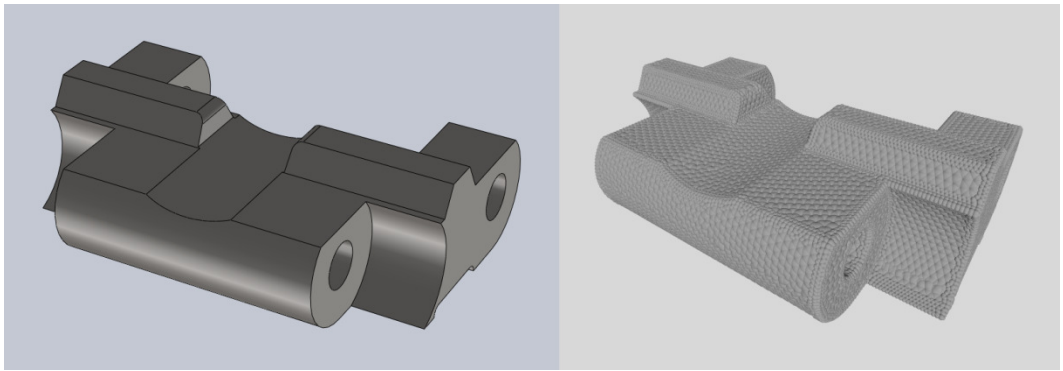
Track Components

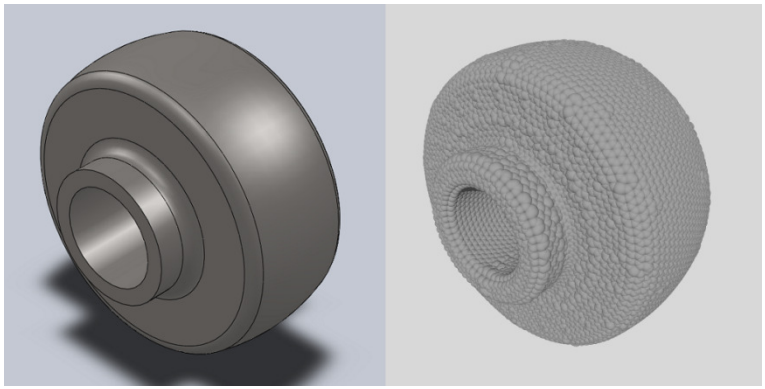
1,594,908 spheres per track





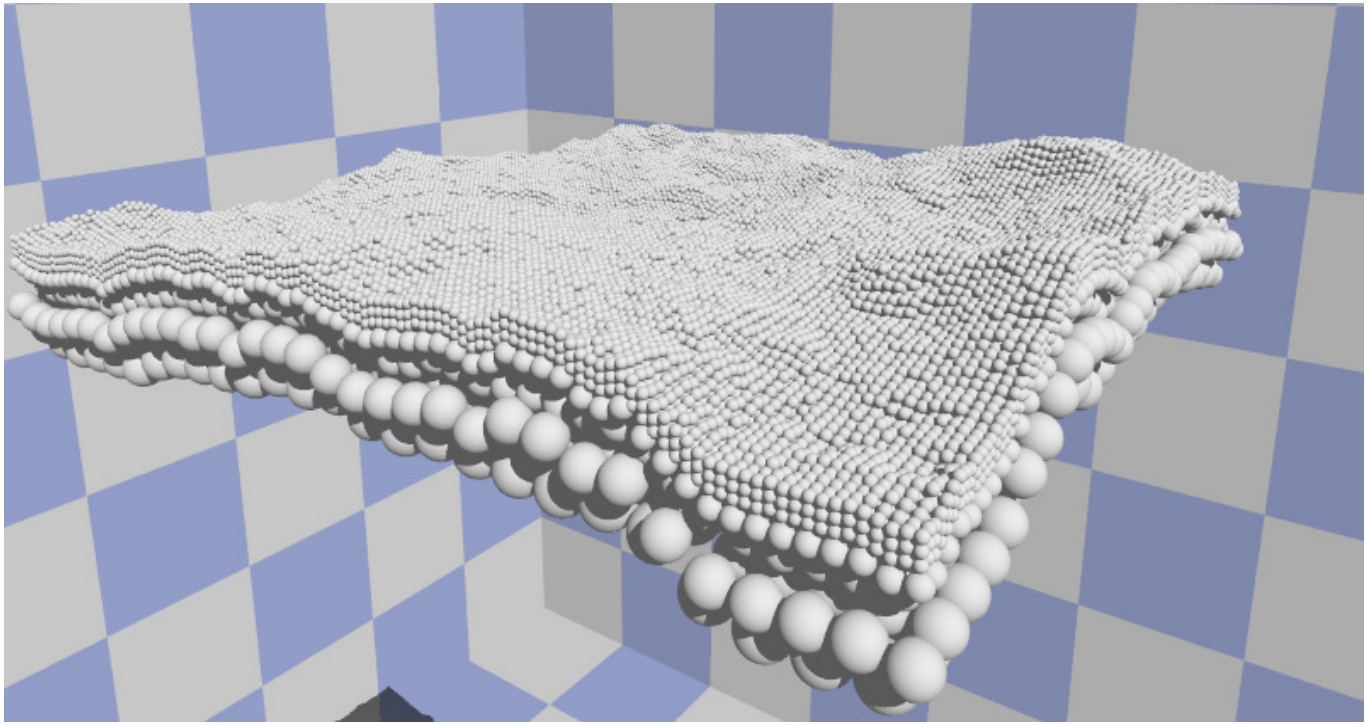




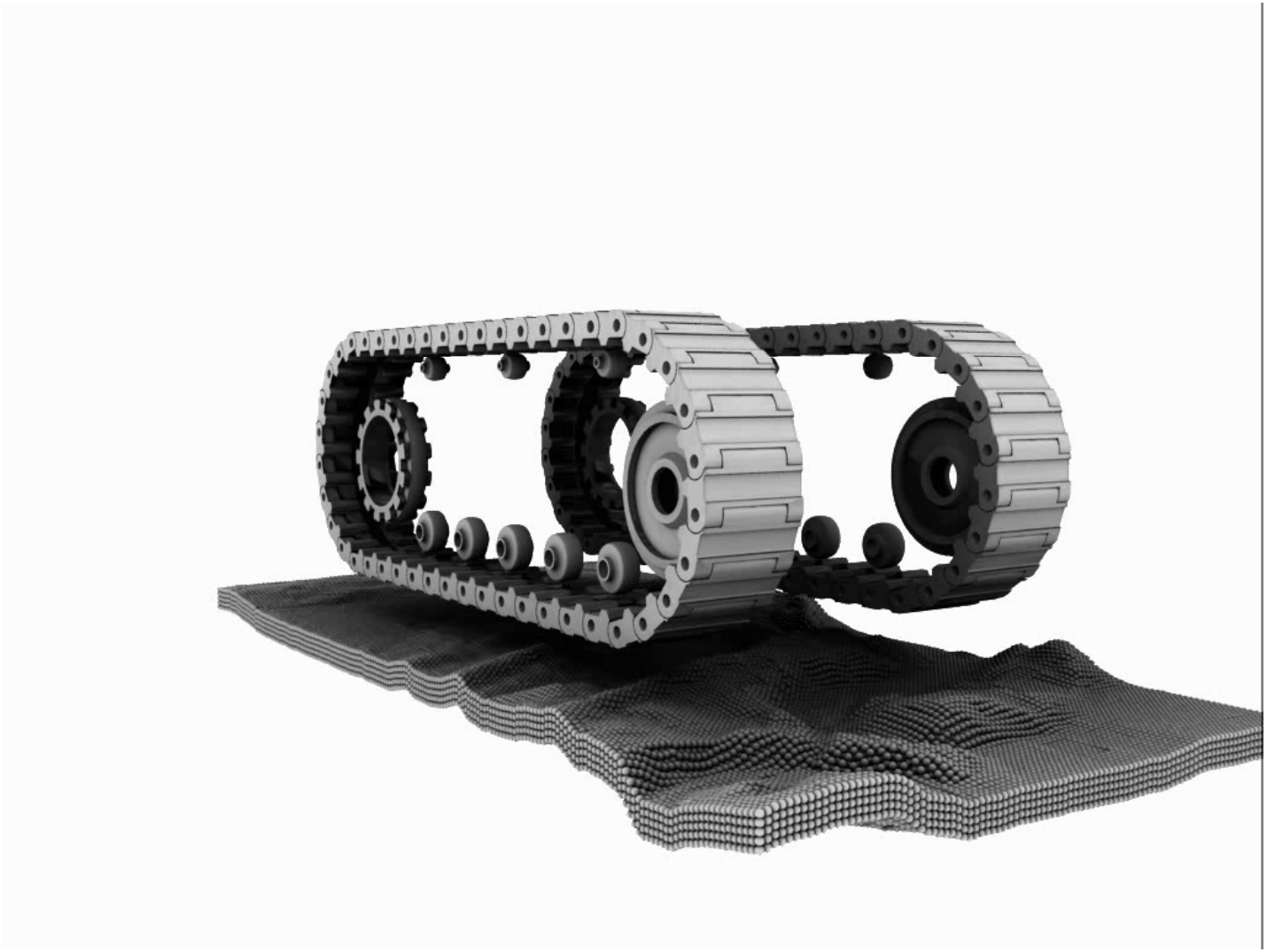


Granular Terrain Model

- Represent terrain as collection of discrete particles
- Match terrain surface profile
- Capture changing granularity with depth



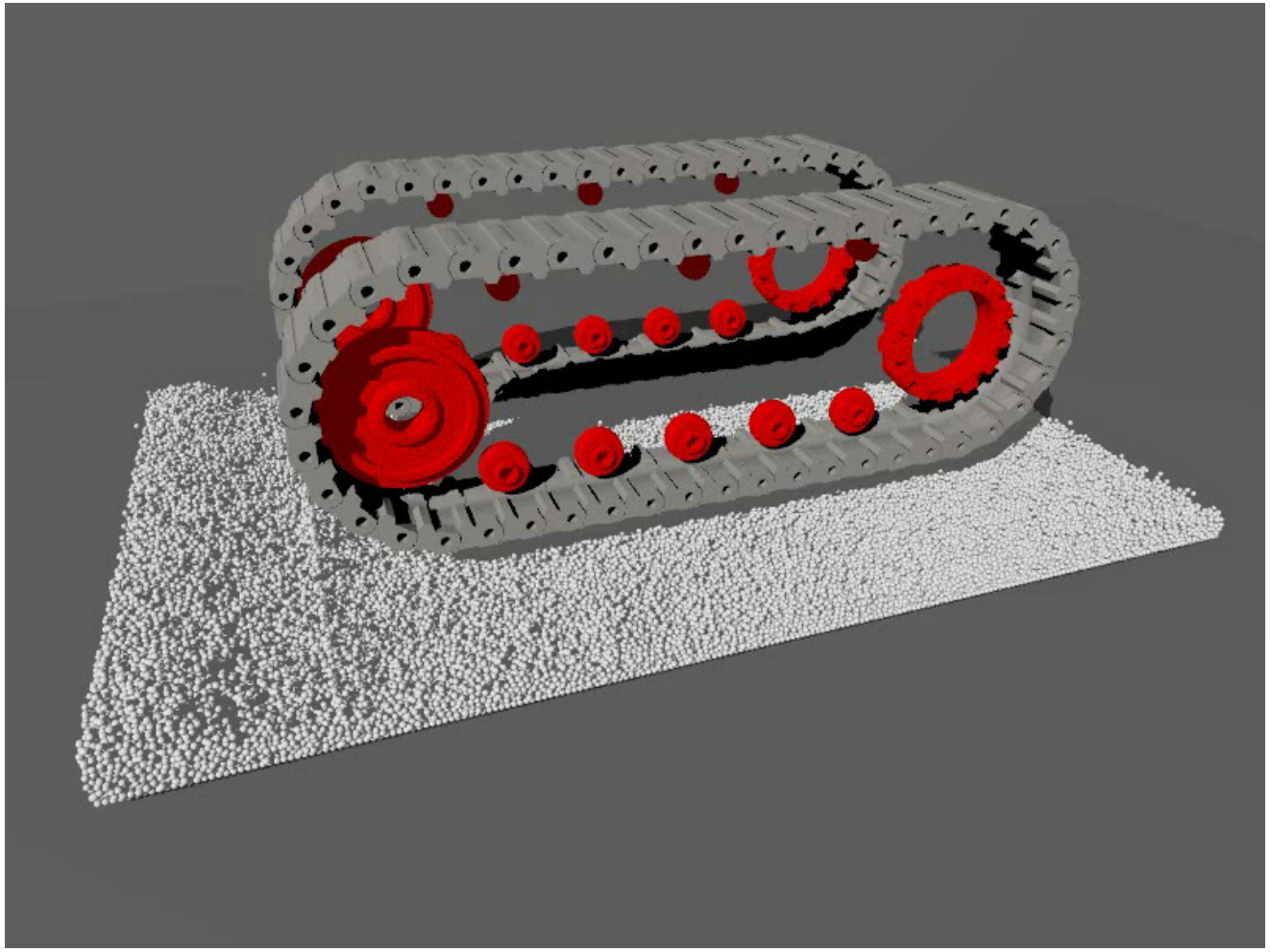
Track Simulation 1



Parameters:

- Driving speed: 1.0 rad/sec
- Length: 12 seconds
- Time step: 0.005 sec
- Computation time: 18.5 hours
- Particle radius: .027273 m
- Terrain: 284,715 particles

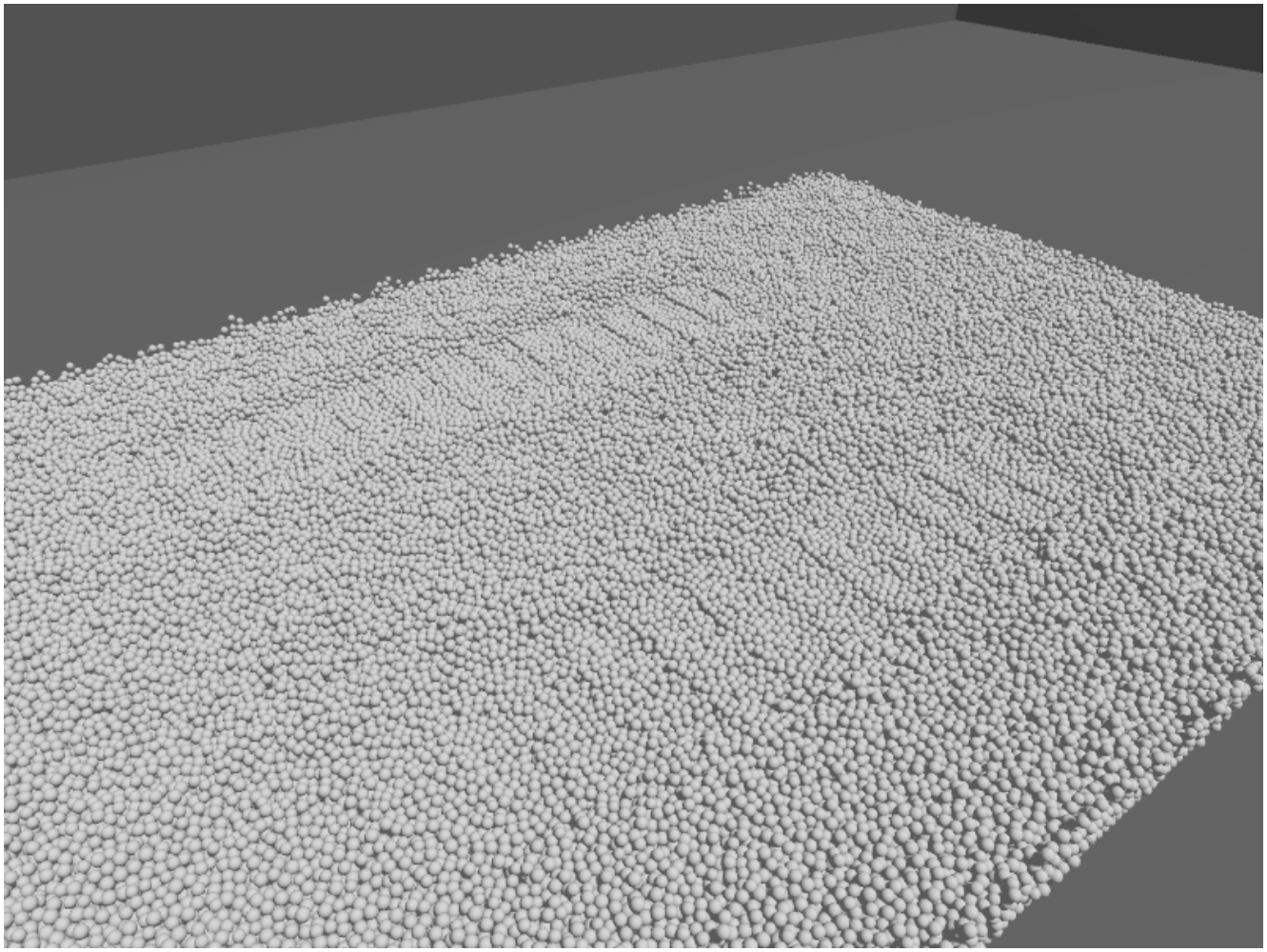
Track Simulation 2



Parameters:

- Driving speed: 1.0 rad/sec
- Length: 10 seconds
- Time step: 0.005 sec
- Computation time: 17.8 hours
- Particle radius: .025±.0025 m
- Terrain: 467,100 particles

Results: Track ‘Footprint’



Results: Positions

Track Simulation 1