High Performance Computational Dynamics in a Heterogeneous Hardware Ecosystem

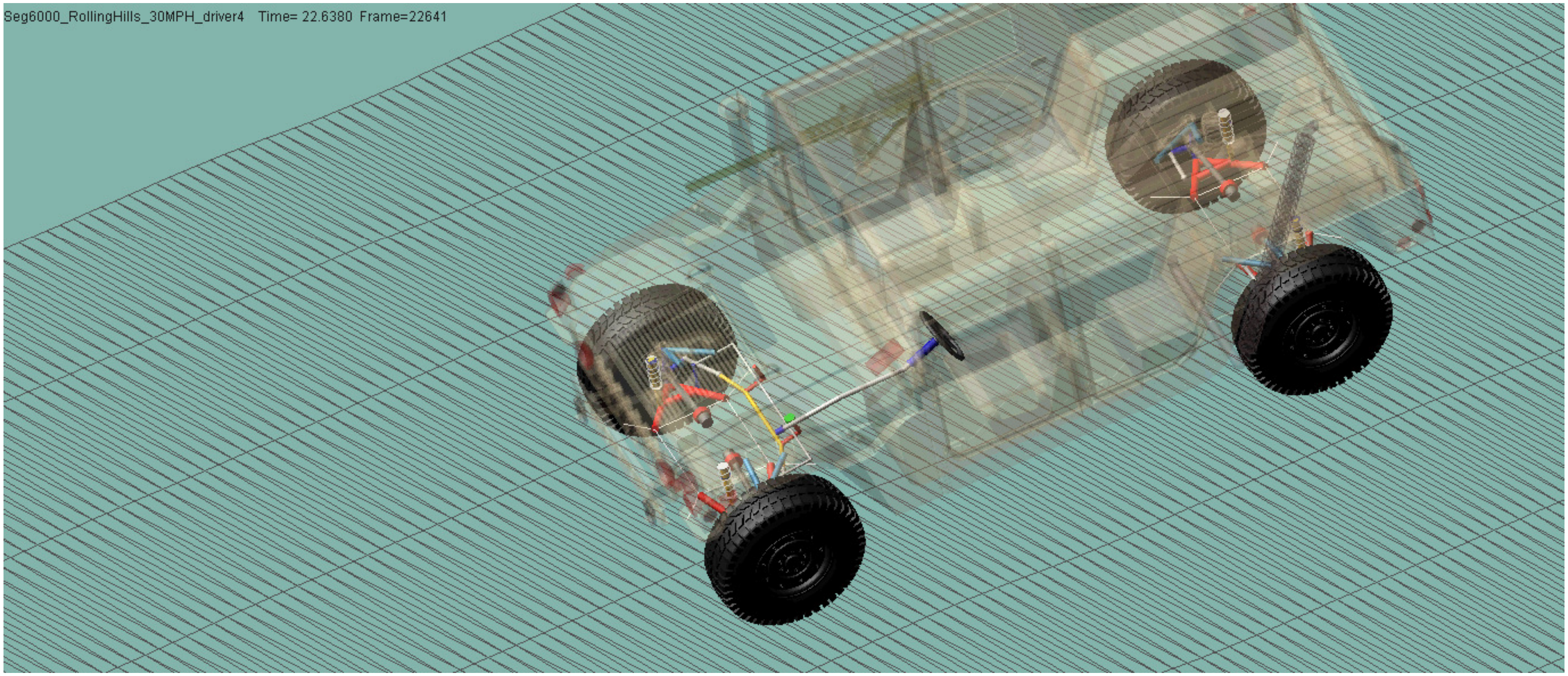
Associate Prof. **Dan Negrut**
NVIDIA CUDA Fellow
Simulation-Based Engineering Lab
Department of Mechanical Engineering
University of Wisconsin – Madison

Beijing
December 13,2011

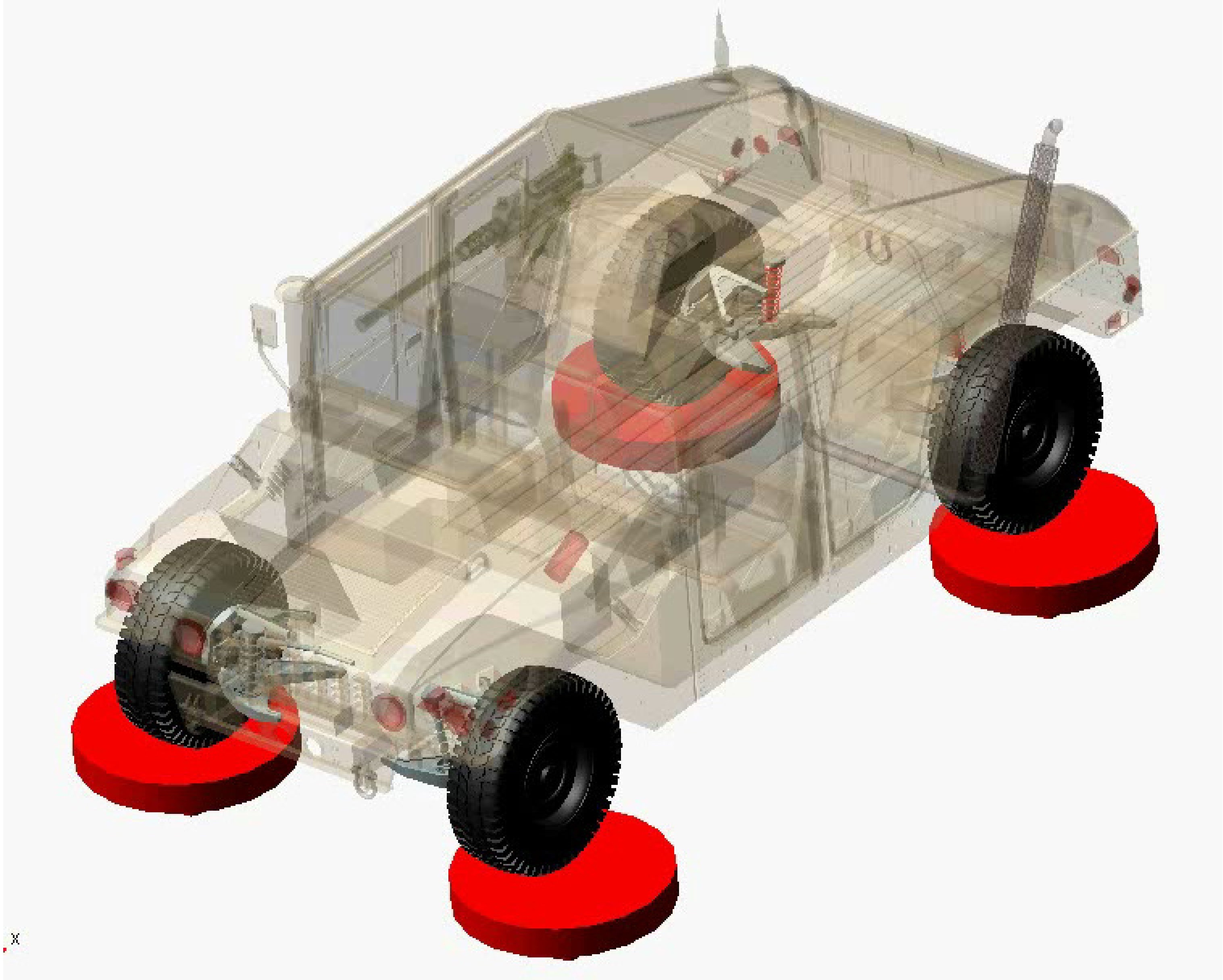
Acknowledgements

- People:
  - Alessandro Tasora – University of Parma, Italy
  - Mihai Anitescu – Argonne National Lab, USA
  - Lab Students:
    - Aaron Bartholomew
    - Makarand Datar
    - Toby Heyn
    - Naresh Khude
    - Justin Madsen
    - Hammad Mazhar
    - Dan Melanz
    - Spencer O'Rourke
    - Arman Pazouki
    - Andrew Seidl
    - Rebecca Shotwell
- Financial support
  - National Science Foundation, Young Investigator Career Award
  - Army Research Office (ARO)
  - US Army TARDEC
  - FunctionBay, S. Korea
  - NVIDIA
  - Caterpillar
  - NASA (Jet Propulsion Lab)

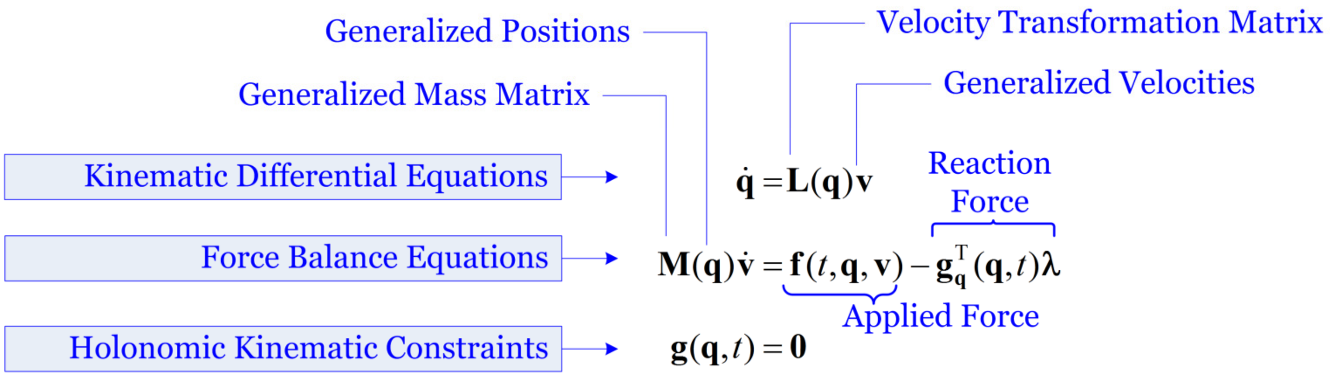
Example, Computational Dynamics
[simulated in commercial package]


Seg6000_RollingHills_30MPH_driver4   Time= 22.6380 Frame=22641

Example, Computational Dynamics
[simulated in commercial package]



Classical Computational Dynamics, Constrained Equations of Motion



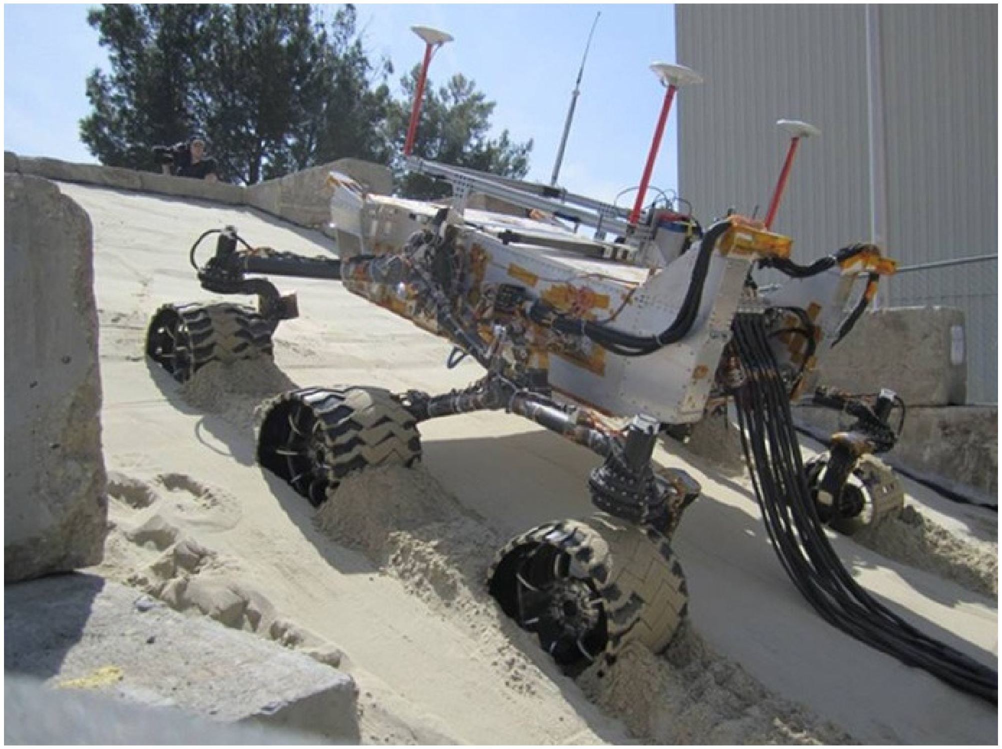
Kinematic Differential Equations → $\dot{\mathbf{q}} = \mathbf{L}(\mathbf{q})\mathbf{v}$

Force Balance Equations → $\mathbf{M}(\mathbf{q})\dot{\mathbf{v}} = \underbrace{\mathbf{f}(t,\mathbf{q},\mathbf{v})}_{\text{Applied Force}} - \overbrace{\mathbf{g}_{\mathbf{q}}^{\mathrm{T}}(\mathbf{q},t)\boldsymbol{\lambda}}^{\text{Reaction Force}}$

Holonomic Kinematic Constraints → $\mathbf{g}(\mathbf{q},t) = \mathbf{0}$

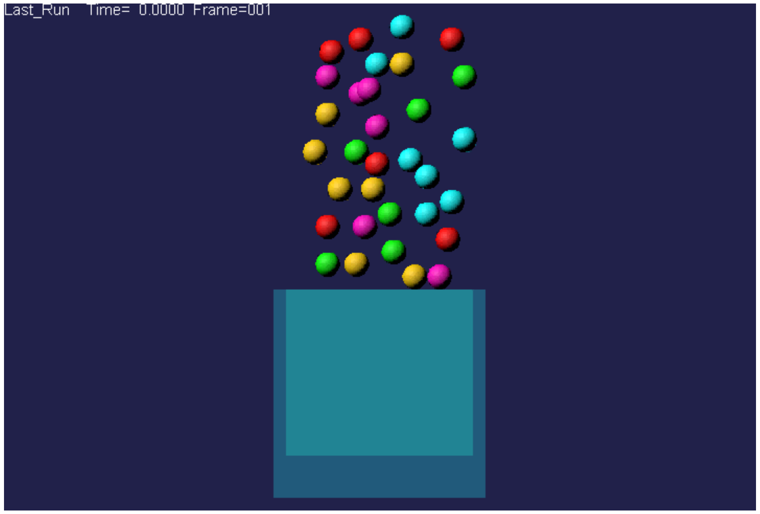
An Engineering Application…

- How is the Rover moving along on a slope with granular material?
- What wheel geometry is more effective?

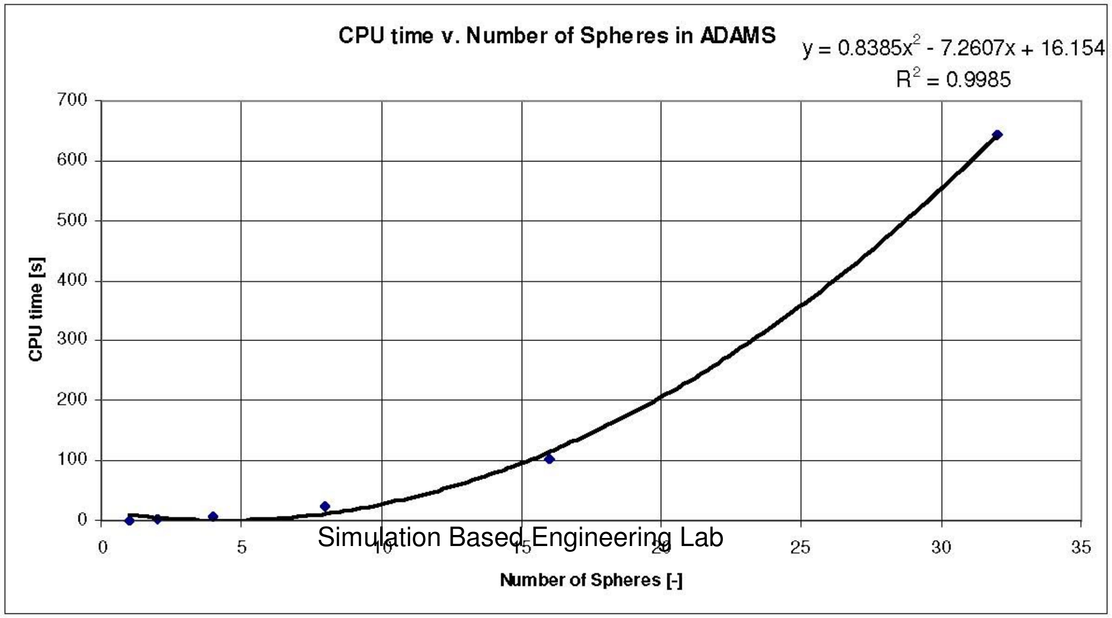


Frictional Contact Simulation

[Commercial Solution]

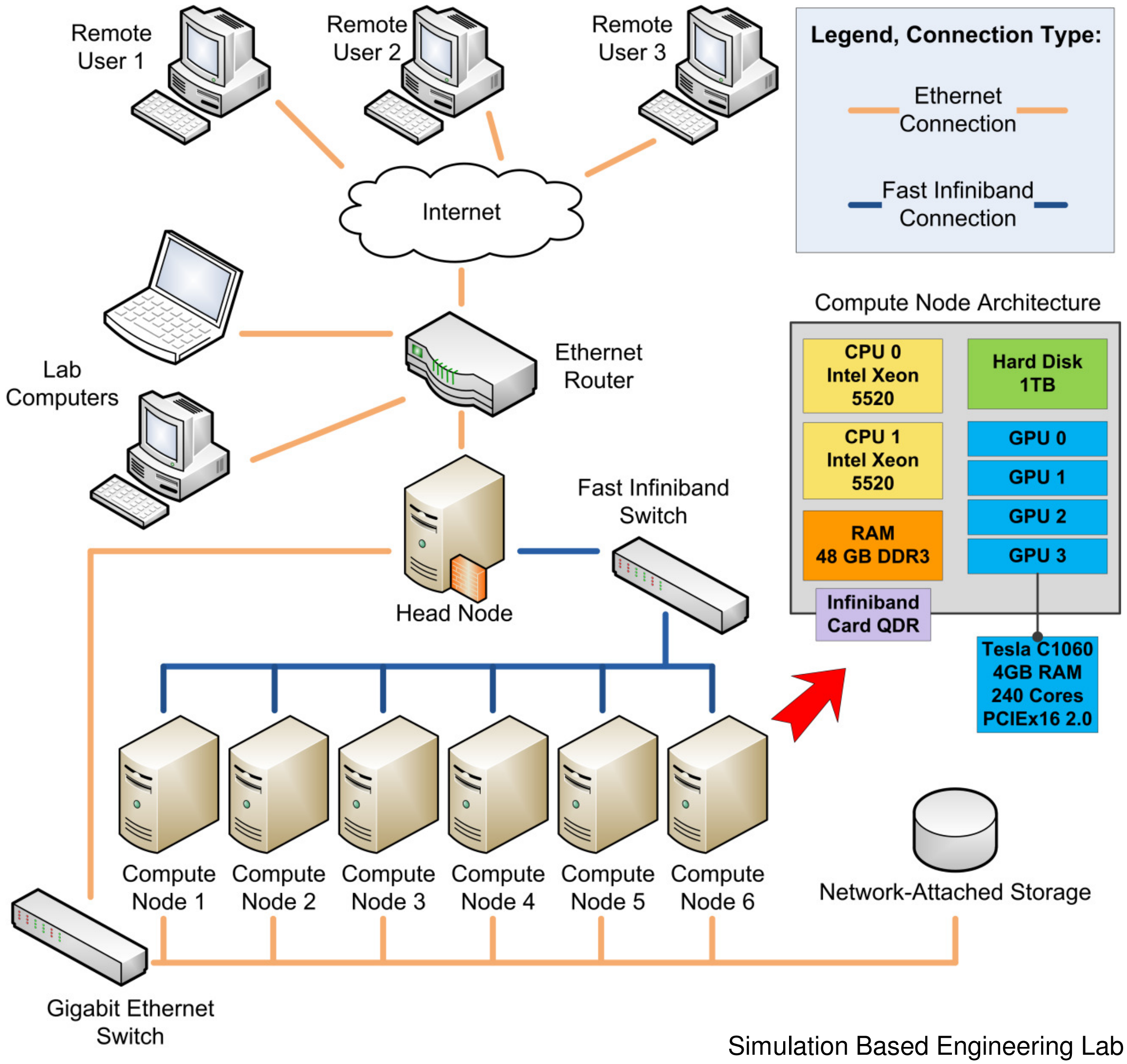


- Model Parameters:
  - Spheres: 60 mm diameter and mass 0.882 kg
  - Forces: smoothing with stiffness of 1E5, force exponent of 2.2, damping coefficient of 10.0, and a penetration depth of 0.1
  - Simulation length: 3 seconds



Simulating large problems remains a challenge…

Heterogeneous Cluster



Lab's Heterogeneous Computing Cluster

- More than 20,000 GPU scalar processors
- More than 150 CPU cores
- Mellanox Infiniband Interconnect, 40Gb/sec
- About 0.7 TB of RAM
- More than 20 Tflops
- Can manage at each time about 1,000,000 parallel GPU threads
- …
- Third fastest cluster at UW-Madison

The issues is not hardware availability. Rather, it is producing modeling and solution techniques that can leverage this hardware

Heterogeneous Computing Template (HCT):

A Software Infrastructure for Large Scale Physics-Based Simulation

- Underlying theme of our lab's effort
  - Develop a Heterogeneous Computing Template (HCT) that leverages emerging hardware architectures and suitable algorithms to solve large engineering problems

- Targeted "emerging hardware architectures" :
  - Clusters of CPUs and GPUs (accelerators)
    - More than 100 CPU cores, tens of GPU cards, tens of thousands of GPU cores

- Targeted "large engineering problems"
  - Granular dynamics, compliant elements, soil modeling, tire/terrain modeling, FSI, etc.

HCT: Five Major Components

- Computational Dynamics requires
  - Advanced modeling techniques
  - Strong algorithmic (applied math) support
  - Proximity computation
  - Domain decomposition & Inter-domain data exchange
  - Post-processing (visualization)

- HCT represents the library support, the associated API, and the embedded tools that support this five component abstraction

- Multi-Physics targeted Computational Dynamics requires
  - **Advanced modeling techniques**
  - Strong algorithmic (applied math) support
  - Proximity computation
  - Domain decomposition & Inter-domain data exchange
  - Post-processing (visualization)

HCT:
Support for Advanced Modeling Techniques

- Modeling: what does it mean?
  - The process of formulating a set of governing differential equations that captures the multi-physics associated with the engineering problem of interest

- Modeling Issues:
  - Modeling approaches are sometimes completely new or have seen little previous usage
  - Multi-physics: multiple spatial and temporal scales, difficult to solve

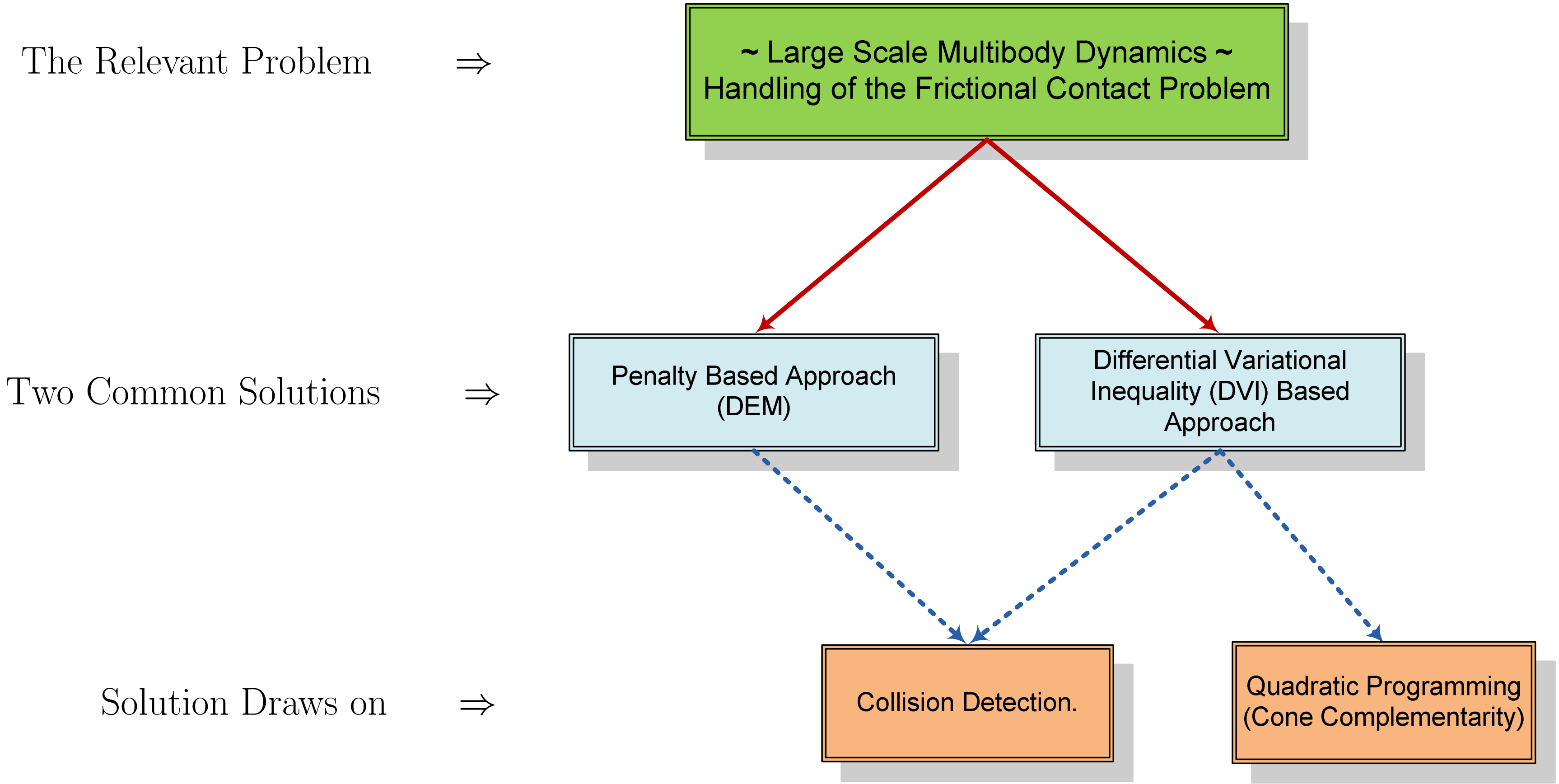
- Modeling can get you a head start
  - Good modeling places you at an advantage when it comes to simulating hard problems

Modeling Example:
Handling Frictional Contact Phenomena

- Two broadly used approaches for handling frictional contact:
  - Soft-body approaches
    - Called a "DEM approach", draws on penalty method
  - Hard-body approaches
    - Called a "DVI approach", draws on Lagrange Multiplier method

Modeling Example: Handling Frictional Contact Phenomena

[Cntd.]



The Relevant Problem ⇒

~ Large Scale Multibody Dynamics ~
Handling of the Frictional Contact Problem

Two Common Solutions ⇒

Penalty Based Approach (DEM)

Differential Variational Inequality (DVI) Based Approach

Solution Draws on ⇒

Collision Detection.

Quadratic Programming (Cone Complementarity)

The DVI Framework…

- The hard-body approach: two rigid bodies in contact shall move so that their boundaries are not overlapping

- There is a complementarity condition that captures this requirement

$$0 \leq \Phi(\mathbf{q}, t) \quad \perp \quad \gamma_n \geq 0$$

The DVI Framework…

[Cntd.]

- There is also friction between bodies (acts in the tangent plane):

$$\mathbf{F}_f = \gamma_u \mathbf{t}_u + \gamma_w \mathbf{t}_w$$

- Coulomb friction model states that the following conditions hold

$$\begin{array}{rcl} 0 \le \mu^2 \gamma_n^2 - (\gamma_u^2 + \gamma_w^2) & \perp & ||\mathbf{v}_T|| \ge 0 \\ \langle \mathbf{v}_T, \mathbf{F}_f \rangle & = & -||\mathbf{v}_T|| \cdot ||\mathbf{F}_f|| \end{array}$$

The DVI Framework…

[Cntd.]

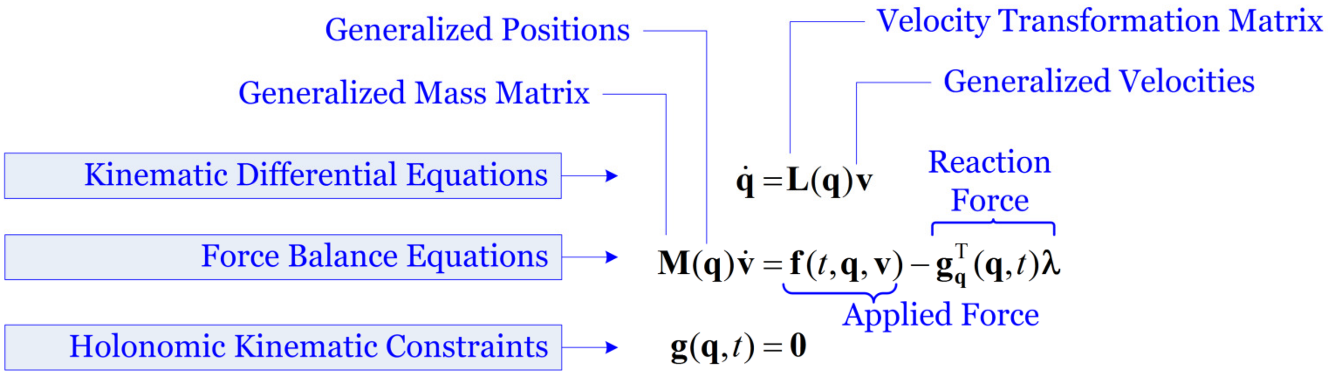
- An equivalent way of stating the Coulomb friction model is

$$(\gamma_u^*, \gamma_w^*) = \underset{\gamma_u^2+\gamma_w^2-\mu^2\gamma_n^2 \le 0}{\arg\min} \left[\mathbf{v}^T \left(\gamma_u \mathbf{t}_u + \gamma_w \mathbf{t}_w\right)\right]$$

  - Recall that $$\mathbf{F}_f = \gamma_u^* \mathbf{t}_u + \gamma_w^* \mathbf{t}_w$$

- In other words, the friction force should be such that the relative motion between the two bodies maximizes the amount of power dissipated

Multi-Body Dynamics



Generalized Positions

Generalized Mass Matrix

Velocity Transformation Matrix

Generalized Velocities

Kinematic Differential Equations

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

Force Balance Equations

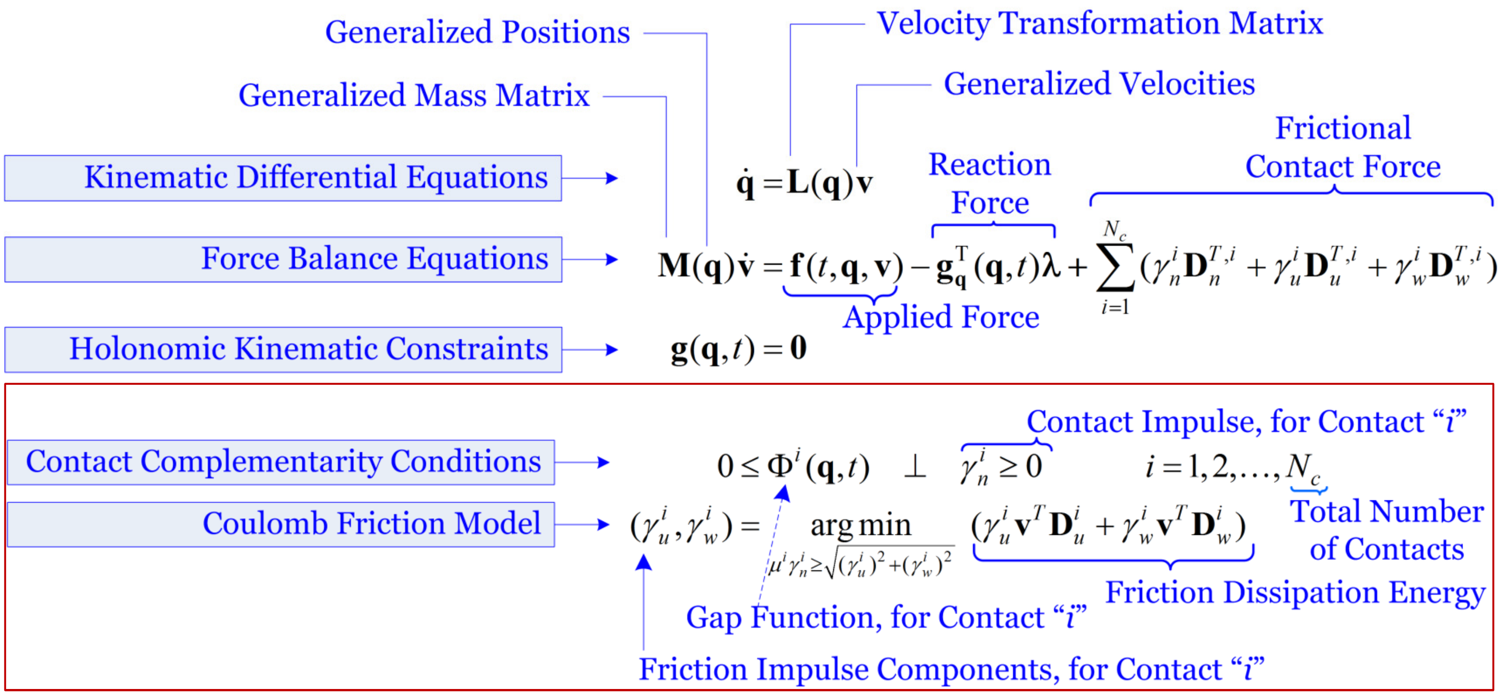
$$\mathbf{M}(\mathbf{q})\dot{\mathbf{v}} = \underbrace{\mathbf{f}(t,\mathbf{q},\mathbf{v})}_{\text{Applied Force}} - \overbrace{\mathbf{g}_{\mathbf{q}}^{T}(\mathbf{q},t)\boldsymbol{\lambda}}^{\text{Reaction Force}}$$

Holonomic Kinematic Constraints

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

Many-Body Dynamics

[with Friction and Contact]



Generalized Positions

Generalized Mass Matrix

Velocity Transformation Matrix

Generalized Velocities

Kinematic Differential Equations →

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

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 →

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

Coulomb Friction Model →

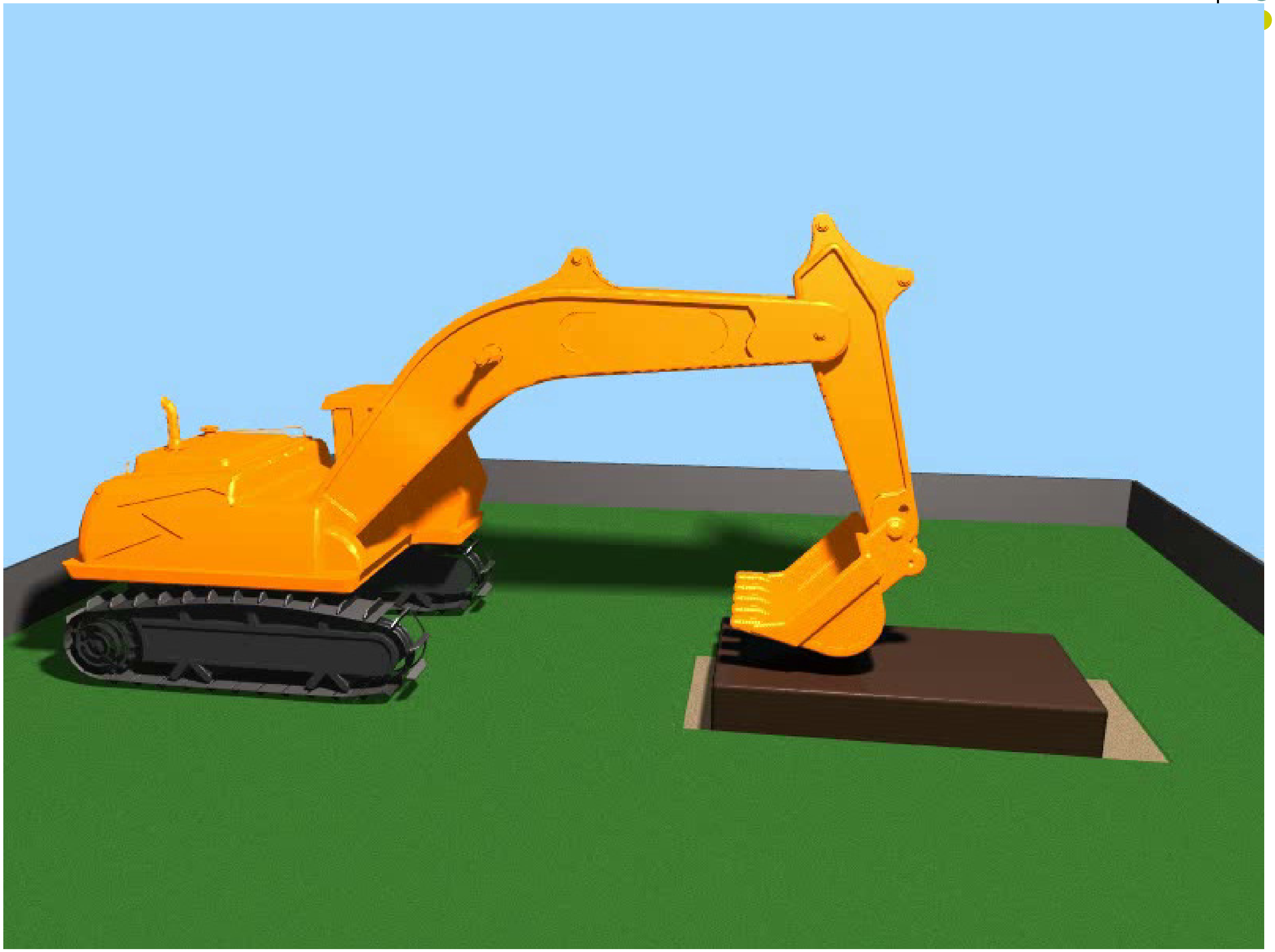
$$(\gamma_u^i,\gamma_w^i) = \underset{\mu^i\gamma_n^i \ge \sqrt{(\gamma_u^i)^2+(\gamma_w^i)^2}}{\arg\min} \underbrace{(\gamma_u^i \mathbf{v}^T \mathbf{D}_u^i + \gamma_w^i \mathbf{v}^T \mathbf{D}_w^i)}_{\text{Friction Dissipation Energy}}$$

Gap Function, for Contact “*i*”

Friction Impulse Components, for Contact “*i*”

Multi-Physics…

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



Fluid-Solid Interaction Example

- Separating living/dead cells

Multi-Physics:
Multi-Body Dynamics & Fluid Dynamics

Generalized Positions

Generalized Mass Matrix

Velocity Transformation Matrix

Generalized Velocities

Kinematic Differential Equations →

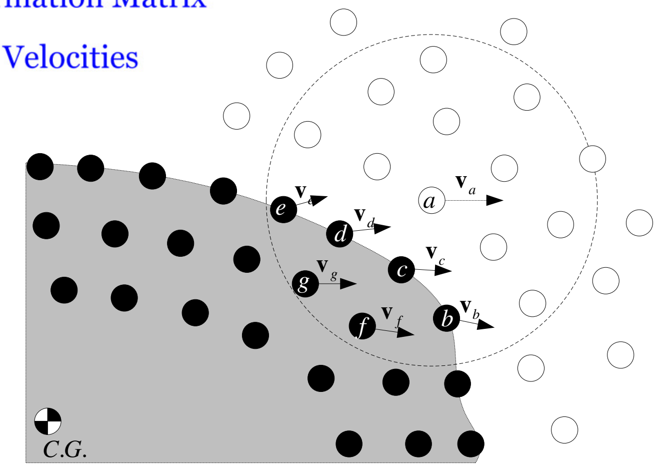
$$\dot{\mathbf{q}} = \mathbf{L}(\mathbf{q})\mathbf{v}$$

Force Balance Equations →

$$\mathbf{M}(\mathbf{q})\dot{\mathbf{v}} = \underbrace{\mathbf{f}(t,\mathbf{q},\mathbf{v})}_{\text{Applied Force}} - \overbrace{\mathbf{g}_{\mathbf{q}}^{\mathrm{T}}(\mathbf{q},t)\boldsymbol{\lambda}}^{\text{Reaction Force}}$$

Holonomic Kinematic Constraints →

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

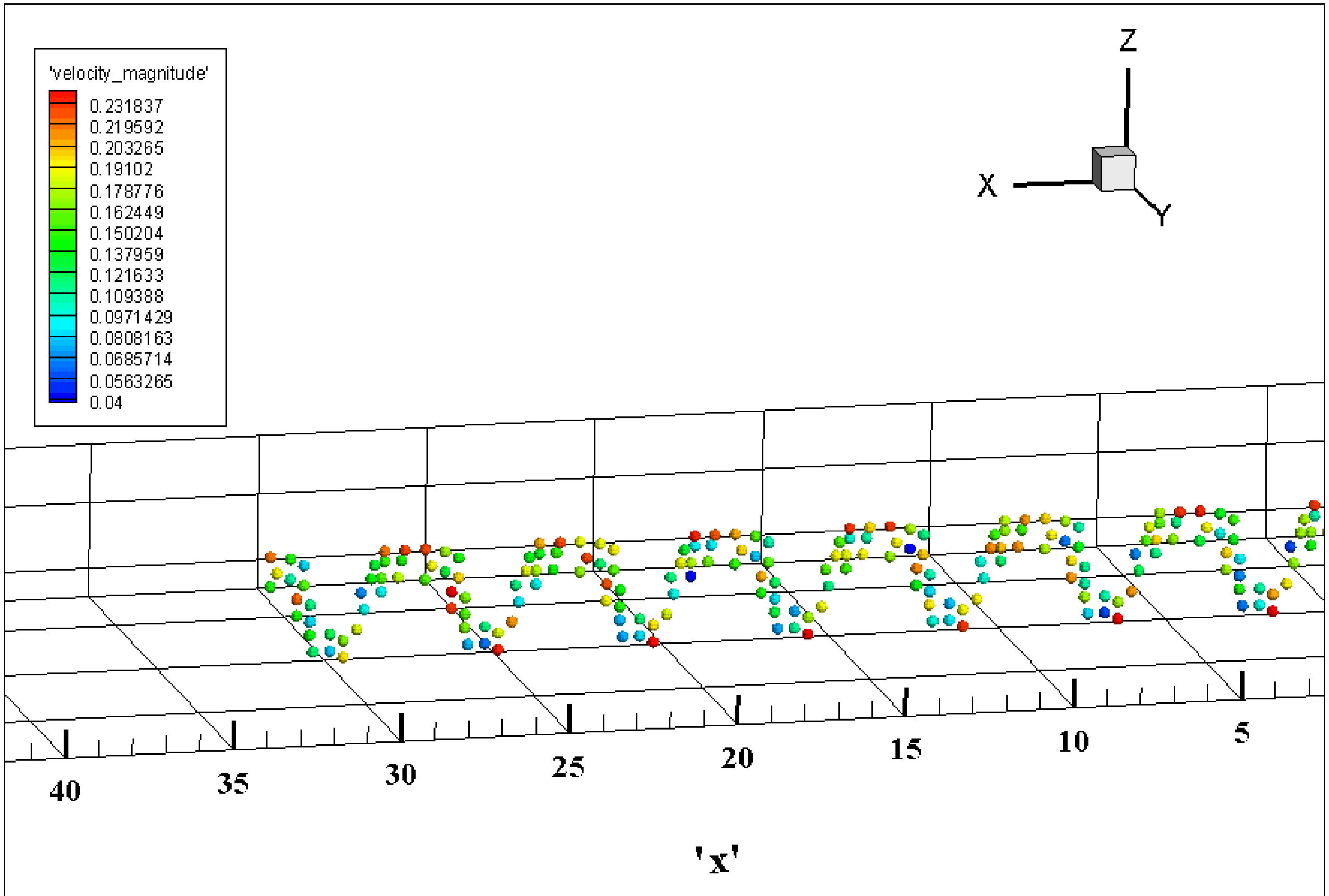


Conservation of mass: $\dfrac{d\rho}{dt} = -\rho\dfrac{\partial v^\beta}{\partial x^\beta} \Rightarrow \dfrac{d\rho_i}{dt} \approx \sum_j m_j \mathbf{v}_{ij}\cdot\nabla_i W_{ij}$

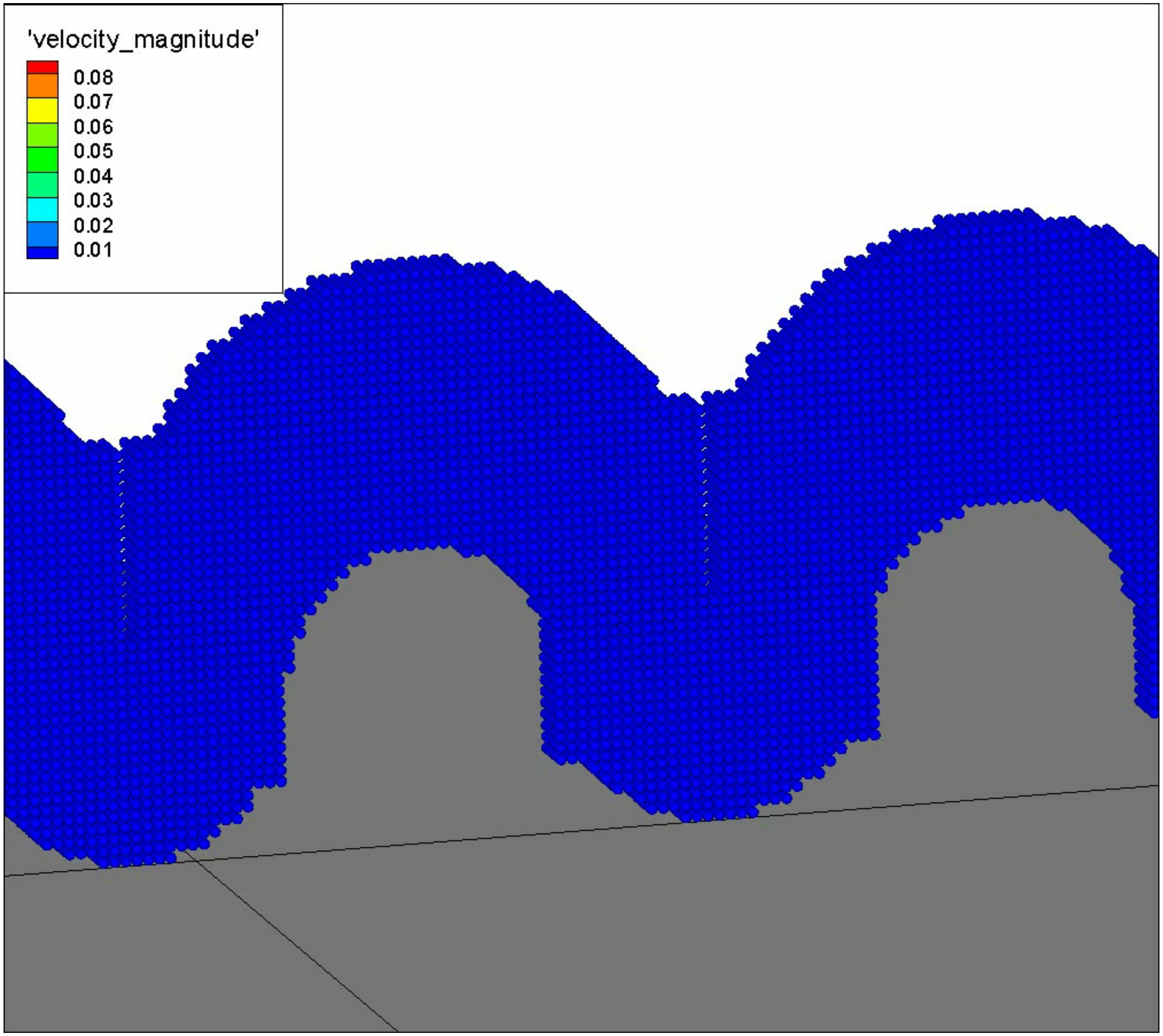
Conservation of momentum: $\dfrac{dv^\alpha}{dt} = \dfrac{1}{\rho}\dfrac{\partial\sigma^{\alpha\beta}}{\partial x^\beta} + \dfrac{f^\alpha}{\rho} \Rightarrow \dfrac{d\mathbf{v}_i}{dt} \approx -\sum_j m_j\left(\dfrac{p_j}{\rho_j^2} + \dfrac{p_i}{\rho_i^2} + \Pi_{ij}\right)\nabla_i W_{ij} + \dfrac{\mathbf{f}}{m_i}$

Conservation of energy: $\dfrac{du}{dt} = \dfrac{\sigma^{\alpha\beta}}{\rho}\dfrac{\partial v^\alpha}{\partial x^\beta} \Rightarrow \dfrac{du_i}{dt} \approx \dfrac{1}{2}\sum_j m_j\left(\dfrac{p_j}{\rho_j^2} + \dfrac{p_i}{\rho_i^2} + \Pi_{ij}\right)\mathbf{v}_{ij}\cdot\nabla_i W_{ij}$

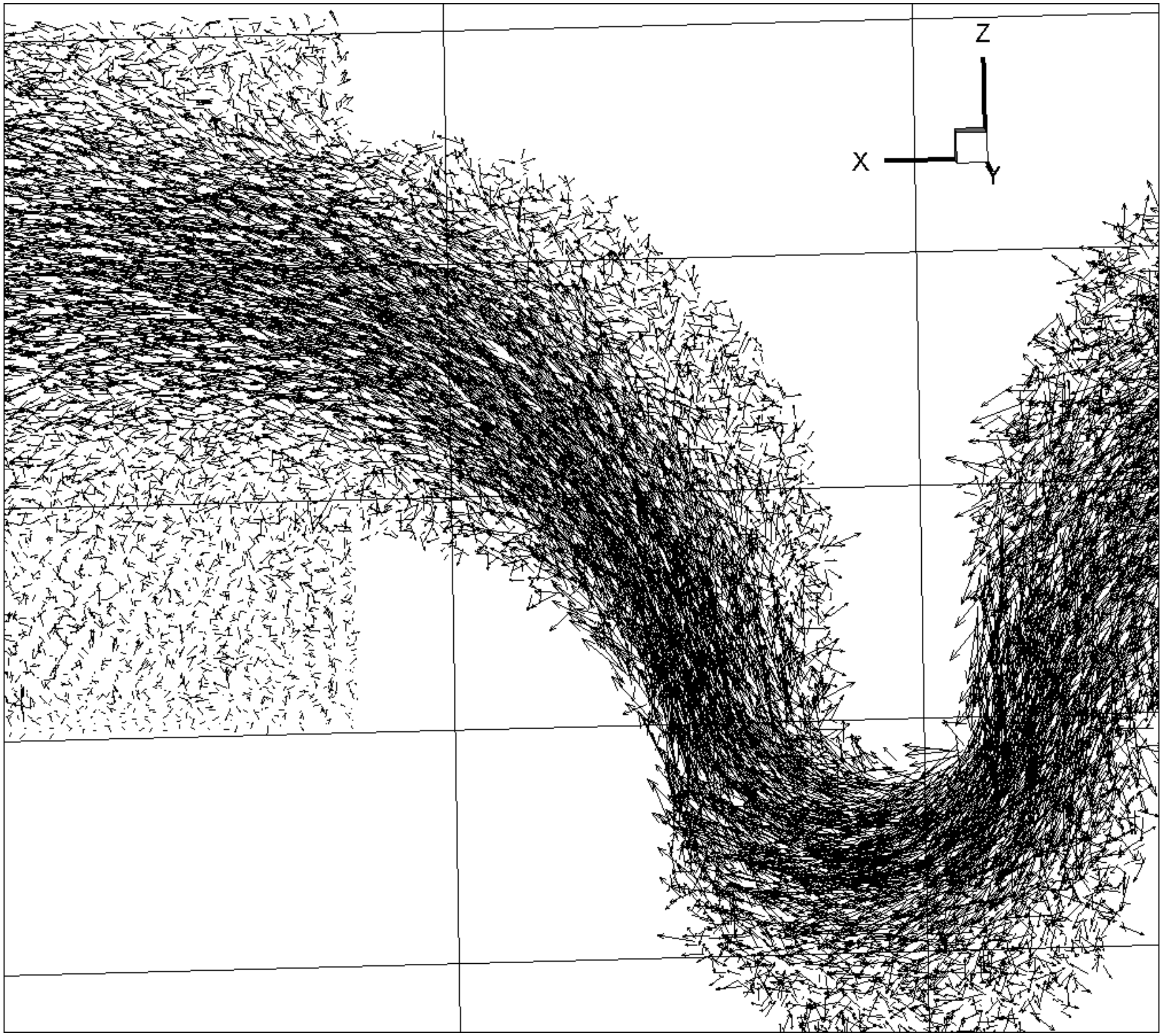
Simulation results: velocity magnitude



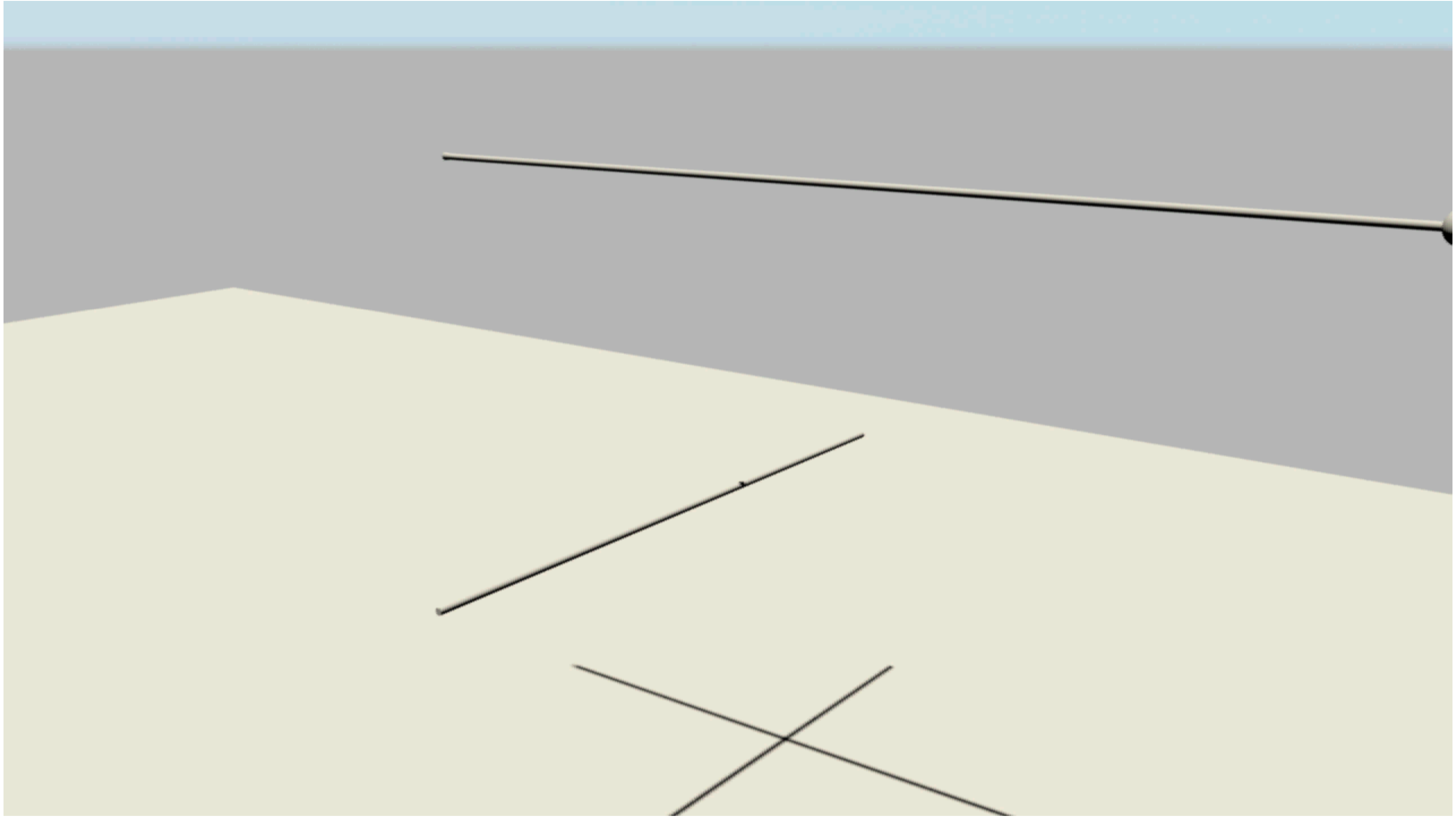
Simulation results: velocity field



Simulation results: Velocity Field



Dealing with Compliant Bodies



Modeling, Dynamics of Systems with Compliant Elements

- Finite Element node coordinates

$$\mathbf{e}^k = \left[(\mathbf{r}^k)^T, \left(\frac{\partial \mathbf{r}^k}{\partial x} \right)^T, \left(\frac{\partial \mathbf{r}^k}{\partial y} \right)^T, \left(\frac{\partial \mathbf{r}^k}{\partial z} \right)^T \right]^T$$

- The global position vector of an arbitrary point on the beam centerline is

$$\mathbf{r}(x, \mathbf{e}) = \mathbf{S}(x)\mathbf{e}$$

- The shape function matrix for this element is defined as

$$\mathbf{S} = [S_1\mathbf{I} \;\; S_2\mathbf{I} \;\; S_3\mathbf{I} \;\; S_4\mathbf{I} \;] \in \mathbb{R}^{3\times 12}$$

$$s_1 = 1 - 3\xi^2 + 2\xi^3, \quad s_2 = l(\xi - 2\xi^2 + \xi^3)$$
$$s_3 = 3\xi^2 - 2\xi^3, \quad s_4 = l(-\xi^2 + \xi^3)$$

$$\xi = x/l$$

Element Mass Matrix and Element Elastic Force

- M is the symmetric consistent mass matrix of element defined as

$$\mathbf{M} = A\int_{0}^{l} \rho \mathbf{S}^T \mathbf{S}\, dx$$

A - c/s area, ρ - density, l - element length

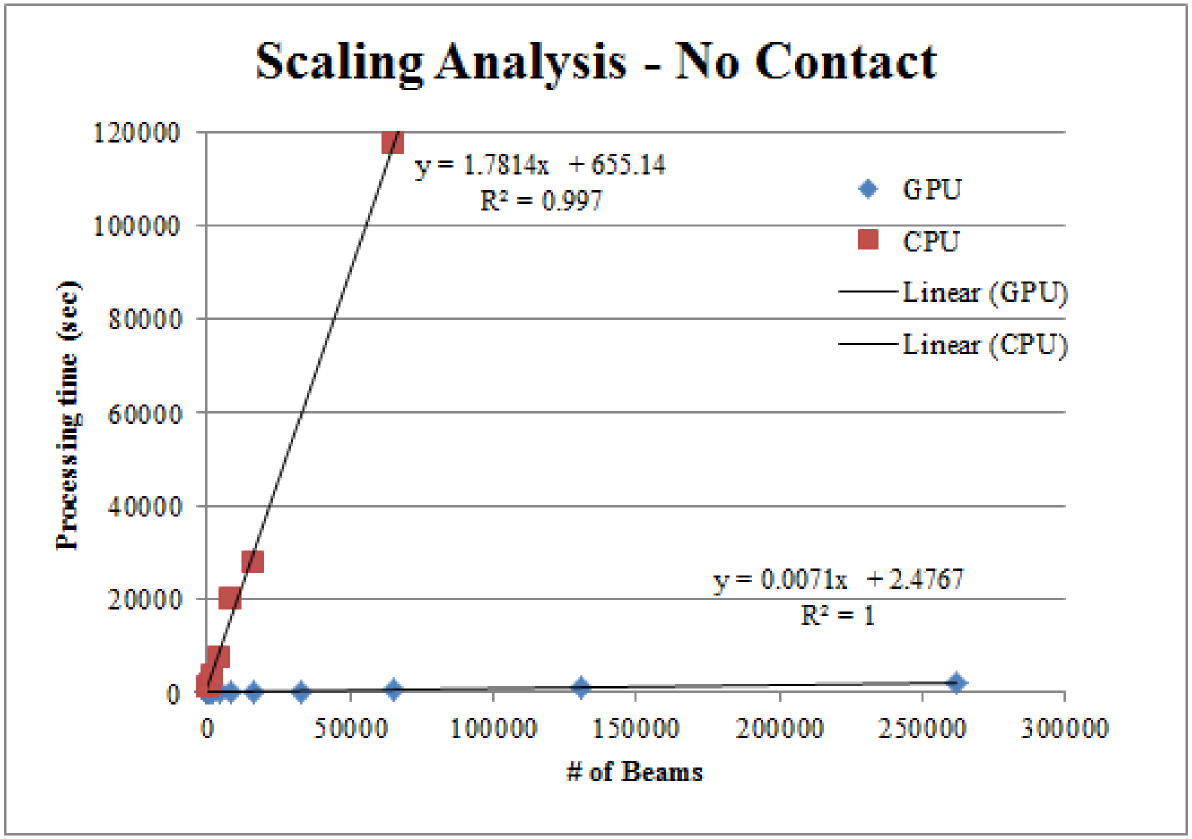
- The vector of the element elastic forces is determined using the strain energy as

$$\mathbf{Q}_s = \left(\frac{\partial U}{\partial \mathbf{e}}\right)^T = \int_{0}^{l} EA(\varepsilon_{11})\left(\frac{\partial \varepsilon_{11}}{\partial \mathbf{e}}\right)^T dx + \int_{0}^{l} EI(\kappa)\left(\frac{\partial \kappa}{\partial \mathbf{e}}\right)^T dx$$

ε_{11} - axial strain κ - magnitude of curvature vector

CPU vs. GPU Scaling Analysis

[results up to 120,000 deformable beams]

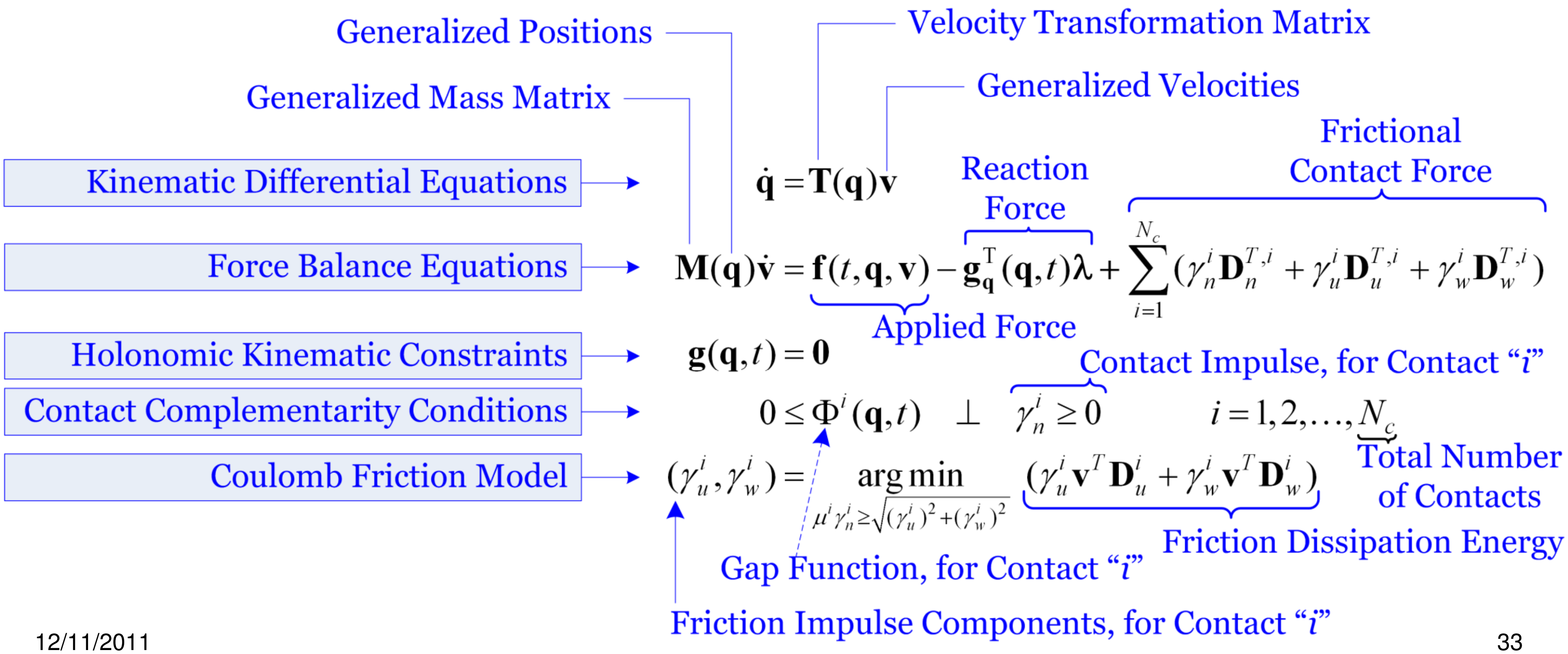


- Intel Nehalem Xeon E5520 2.26GHz processor with an NVIDIA Tesla C2070 graphics cards

- Multi-Physics targeted Computational Dynamics requires
  - Advanced modeling techniques
  - **Strong algorithmic (applied math) support**
  - Proximity computation
  - Domain decomposition & Inter-domain data exchange
  - Post-processing (visualization)

HCT: Novel modeling techniques

- Main issue: I should be able to solve the equations of motion effectively in a heterogeneous hardware environment



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

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

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

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

Complementarity Condition

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

Stabilization term

Coulomb 3D fricion model

(Stewart, 1998)

Relaxed Discretization Scheme Used

$$
\begin{aligned}
\mathbf{q}^{(l+1)} &= \mathbf{q}^{(l)} + h\mathbf{L}(\mathbf{q}^{(l)})\mathbf{v}^{(l+1)} \\
\mathbf{M}(\mathbf{v}^{(l+1)} - \mathbf{v}^{l}) &= h\mathbf{f}(t^{(l)}, \mathbf{q}^{(l)}, \mathbf{v}^{(l)}) + \textstyle\sum_{i \in \mathcal{A}(q^{(l)},\delta)} \left(\gamma_{i,n}\,\mathbf{D}_{i,n} + \gamma_{i,u}\,\mathbf{D}_{i,u} + \gamma_{i,w}\,\mathbf{D}_{i,w}\right) \\
i \in \mathcal{A}(q^{(l)},\delta) : \quad 0 &\le \tfrac{1}{h}\Phi_i(\mathbf{q}^{(l)}) + \mathbf{D}_{i,n}^{T}\mathbf{v}^{(l+1)} - \boxed{\mu^{i}\sqrt{(\mathbf{v}^{T}\,\mathbf{D}_{i,u})^2 + \mathbf{v}^{T}\,\mathbf{D}_{i,w})^2}} \perp \gamma_n^i \ge 0, \\
(\gamma_{i,u}, \gamma_{i,w}) &= \underset{\mu_i \gamma_{i,n} \ge \sqrt{\gamma_{i,u}^2 + \gamma_{i,w}^2}}{\operatorname{argmin}} \quad \mathbf{v}^{T}\left(\gamma_{i,u}\,\mathbf{D}_{i,u} + \gamma_{i,w}\,\mathbf{D}_{i,w}\right).
\end{aligned}
$$

Relaxation Term

(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

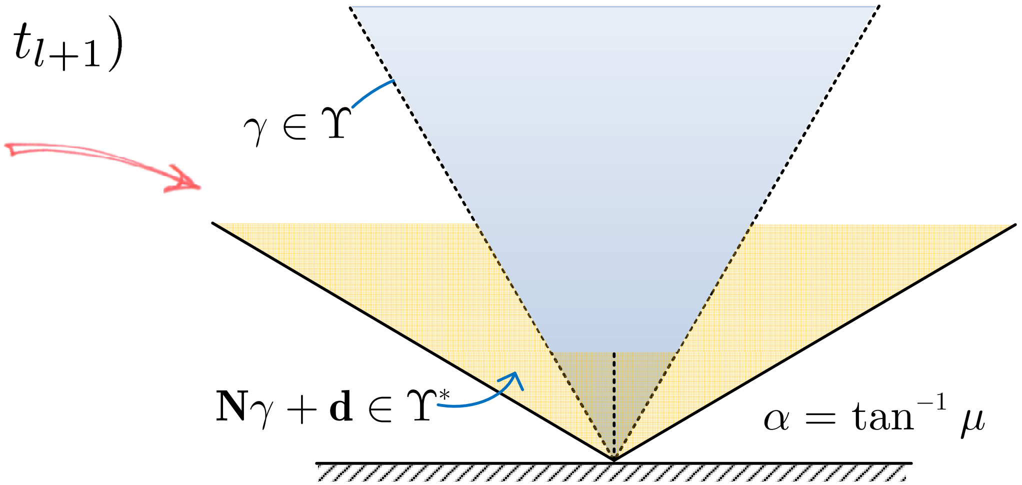
CCP assumes following form: Find γ such that

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

Putting Things in Perspective…

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

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



- Four key points led to above algorithm:
  - Coulomb Friction posed as an optimization problem
  - Working with velocity and impulses rather than acceleration and forces
  - Working with constraint equations (unilateral and bilateral) at the velocity level
  - Contact complementarity expression relaxed to lead to CCP

The Quadratic Programming Angle…

- The relaxed EOM represent a cone-complementarity problem (CCP)
- The CCP captures the first-order optimality condition for a quadratic optimization problem with conic constraints:

$$\begin{cases} \min \mathbf{q}(\gamma) = \frac{1}{2}\gamma^{\mathbf{T}}\mathbf{N}\gamma + \mathbf{d}^{\mathbf{T}}\gamma \\ \text{subject to} \quad \gamma_i \in \Upsilon_i \ \text{ for } \ i = 1, 2, \ldots, N_c \end{cases}$$

- Notation used:

$$\gamma \equiv [\gamma_1^T, \gamma_2^T, \ldots, \gamma_{N_c}^T]^T \in \mathbb{R}^{3 \times N_c} \qquad \text{and} \qquad \Upsilon_i : (\gamma_{u,i}^2 + \gamma_{w,i}^2) - \mu_i^2 \gamma_{n,i}^2 \leq 0$$

CCP Solution 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:

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

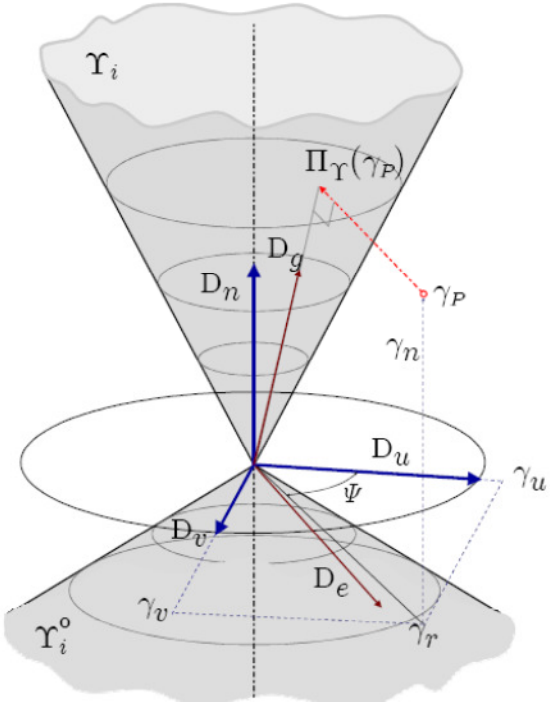
$$\Delta\gamma_i^{r+1} = \gamma_i^{r+1} - \gamma_i^r\ ;$$

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

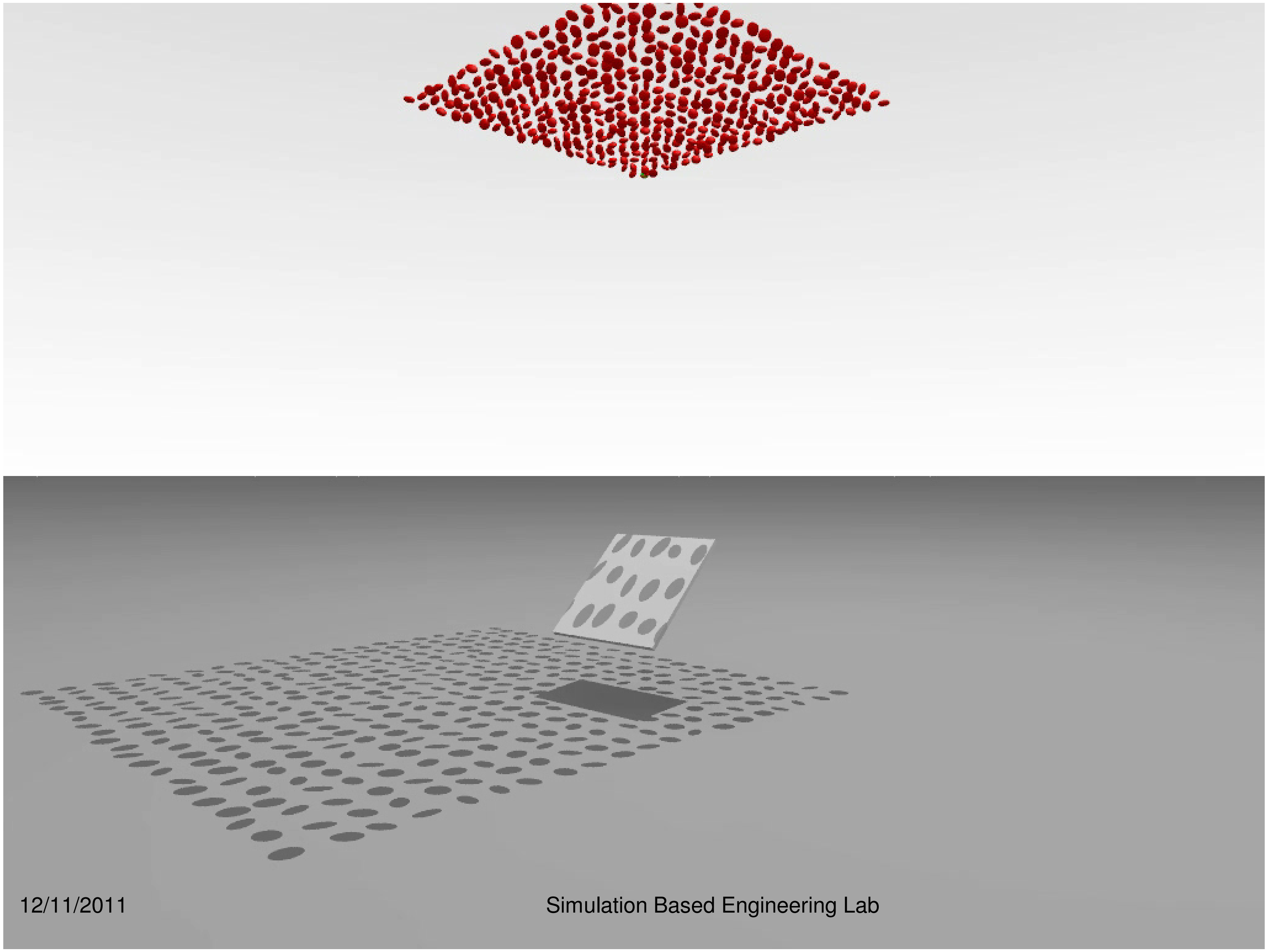
5. Apply updates to the velocity vector:

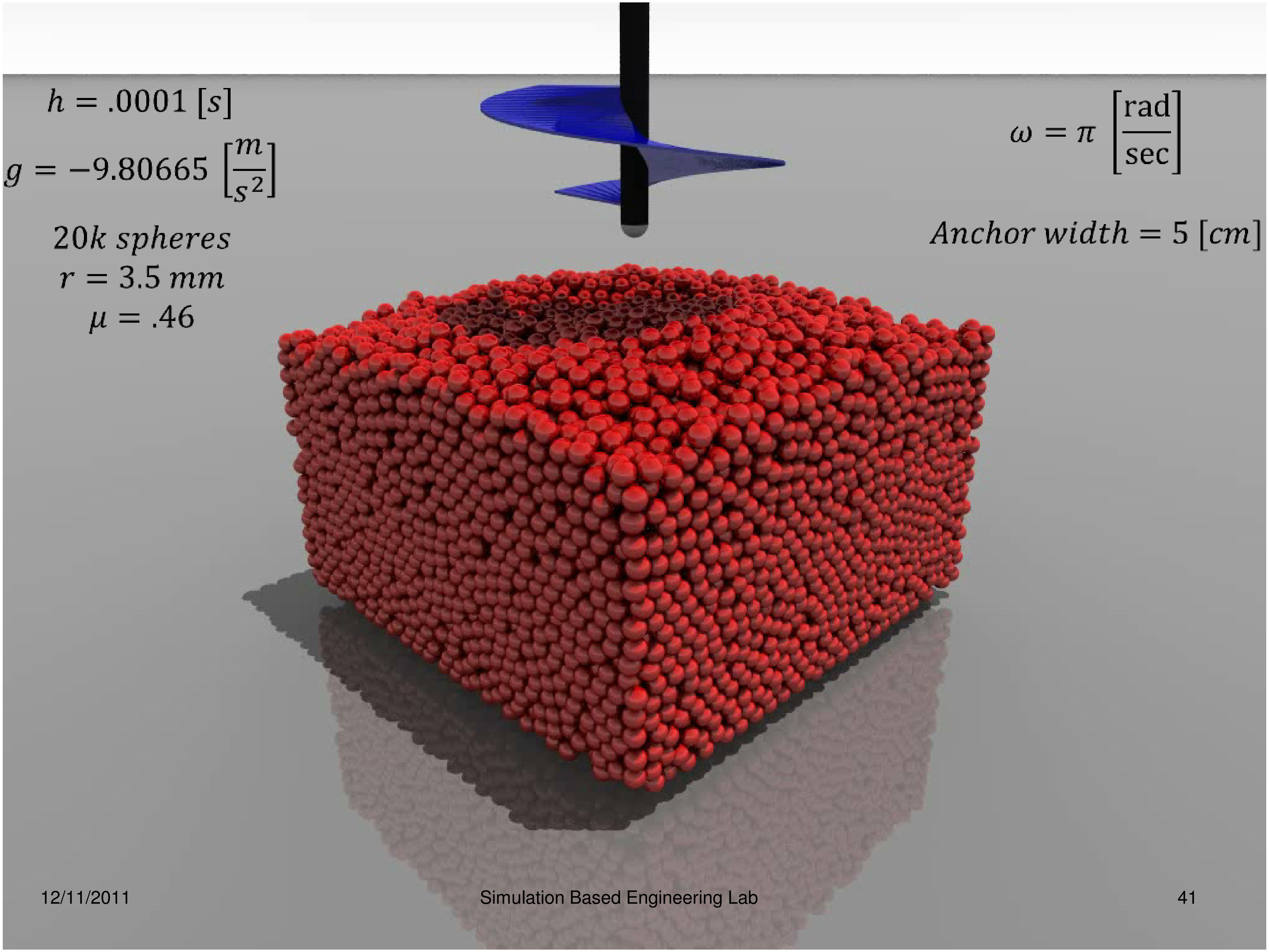
$$\mathbf{v}^{r+1} = \mathbf{v}^r + \sum_i \Delta\mathbf{v}_i$$

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



Mixing 50,000 M&Ms on the GPU





$h = .0001\ [s]$

$g = -9.80665\ \left[\frac{m}{s^2}\right]$

$20k\ spheres$

$r = 3.5\ mm$

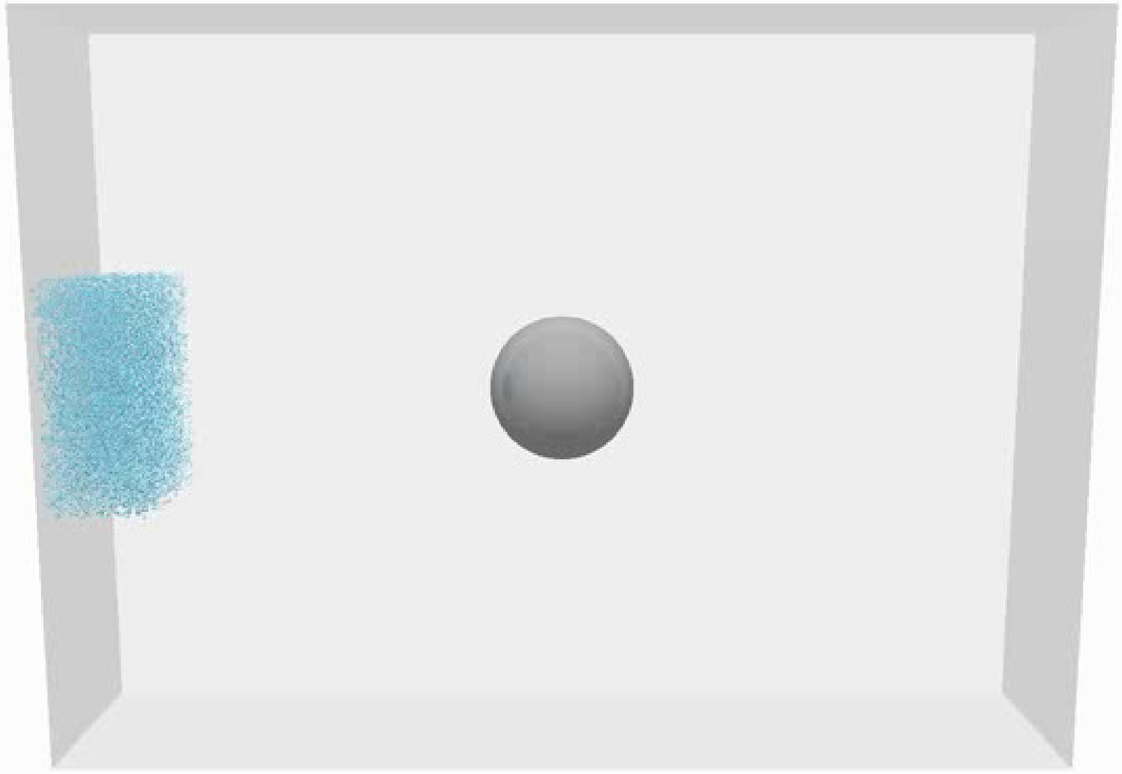
$\mu = .46$

$\omega = \pi\ \left[\frac{\text{rad}}{\text{sec}}\right]$

$Anchor\ width = 5\ [cm]$

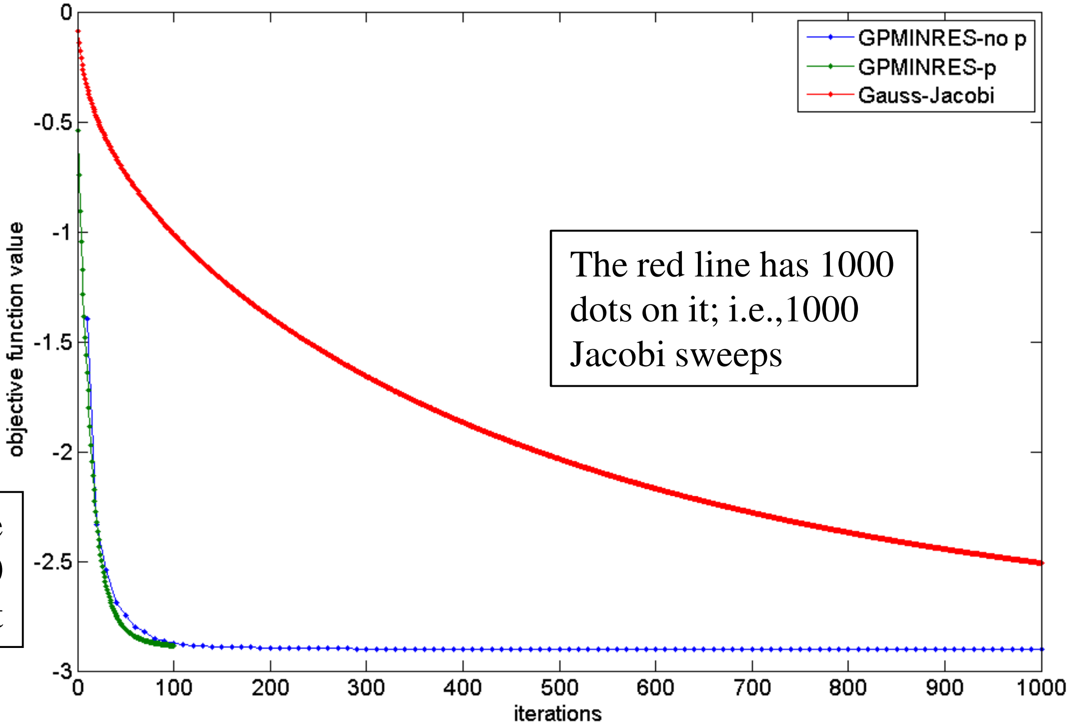
1 Million Rigid Spheres

[parallel on the GPU]



Objective Function Value

[1K bodies, 3525 contacts]



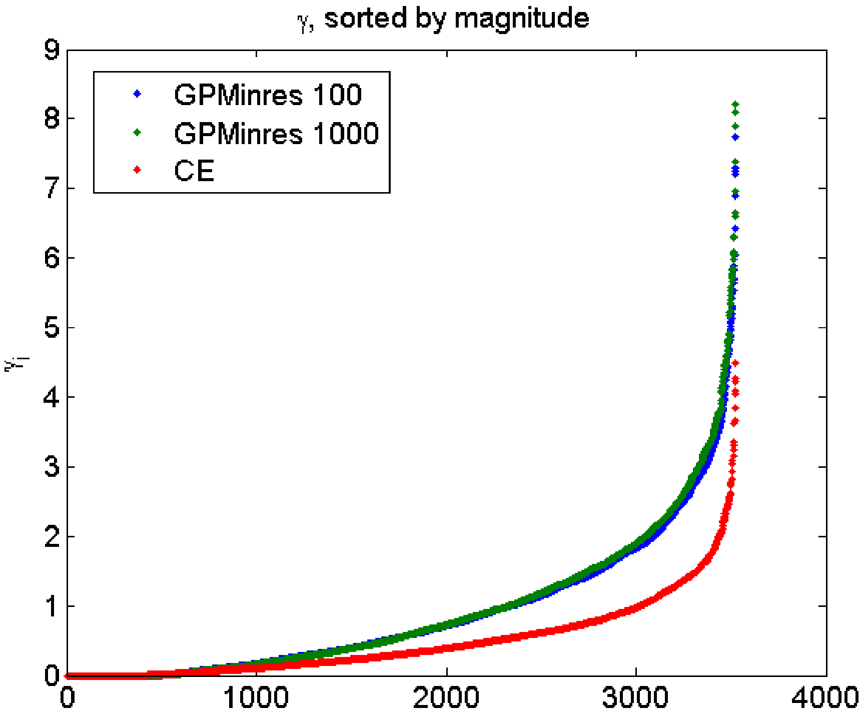
The red line has 1000 dots on it; i.e.,1000 Jacobi sweeps

The green & blue lines have 100 dots on them; i.e.,100 changes of active set

Method	Iterations	Final Objective Function Value	γ_{min}	γ_{max}	Computation Time [sec]
GPMINRES-no p	1000 MinRes Its. [within 100 changes of active set]	-2.9035	0.0	7.7487	6.7002
GPMINRES-no p (not plotted above)	10000 MinRes Its. [within 1000 changes of active set]	-2.9045	0.0	8.2002	61.0698
GPMINRES-p	100 MinRes Its. [within 100 changes of active set]	-2.8854	0.0	6.8551	1675
Jacobi	1000	-2.5077	0.0	4.4961	3.6643

Magnitudes of x_k components

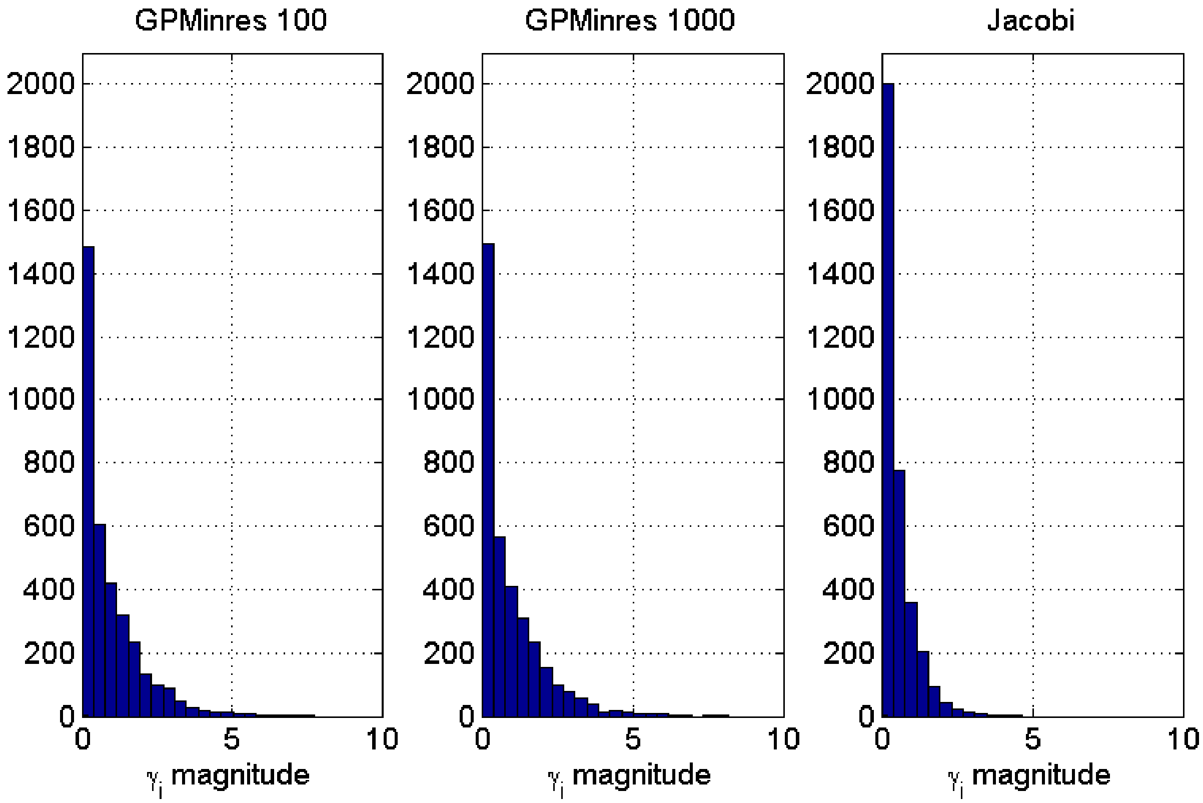
[1K bodies, 3525 contacts]



- Here, the solution vector x_k is sorted by size and plotted.
- The blue dots represent the solution after 100 active sets (1,000 total MinRes iterations).
- The green dots represent the solution after 1000 active sets (10,000 total MinRes iterations).
- The red dots correspond to Jacobi after 1000 sweeps.
- The solution is ‘sharper’ when performing more iterations.

Magnitudes of x_k components

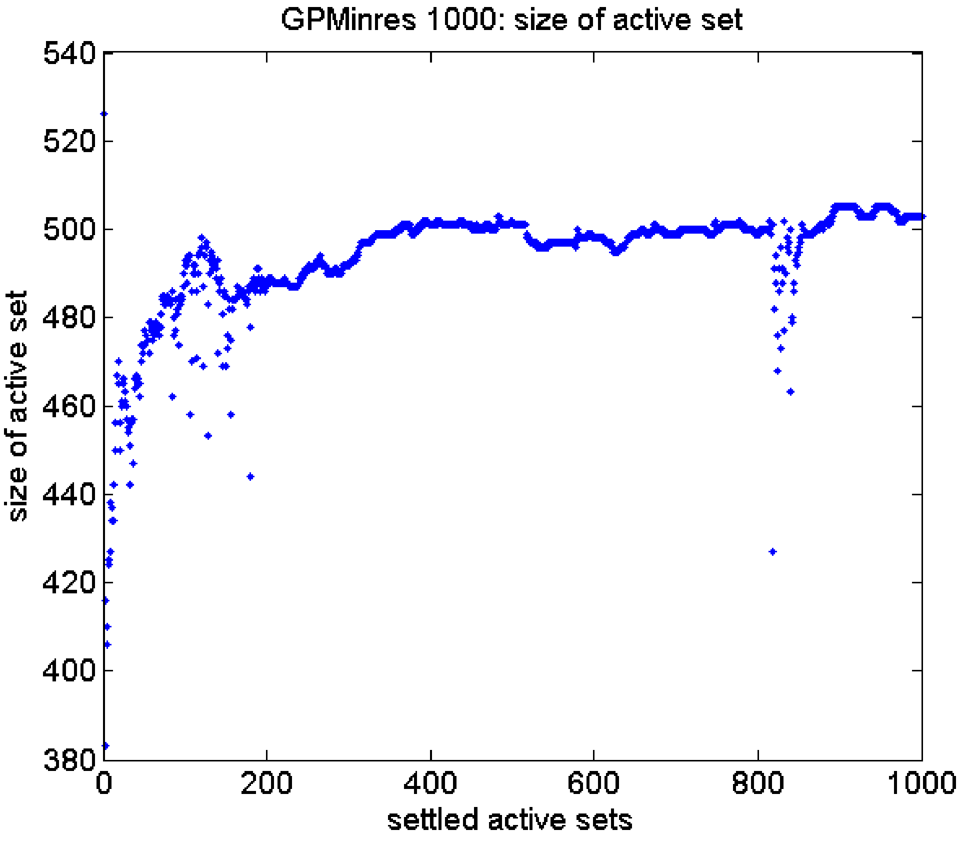
[1K bodies, 3525 contacts]



This is basically the same data as the previous slide, this time plotted in histograms. Again, the results from 100 and 1000 active sets are quite similar.

History of Active Set Size

[1K bodies, 3525 contacts]

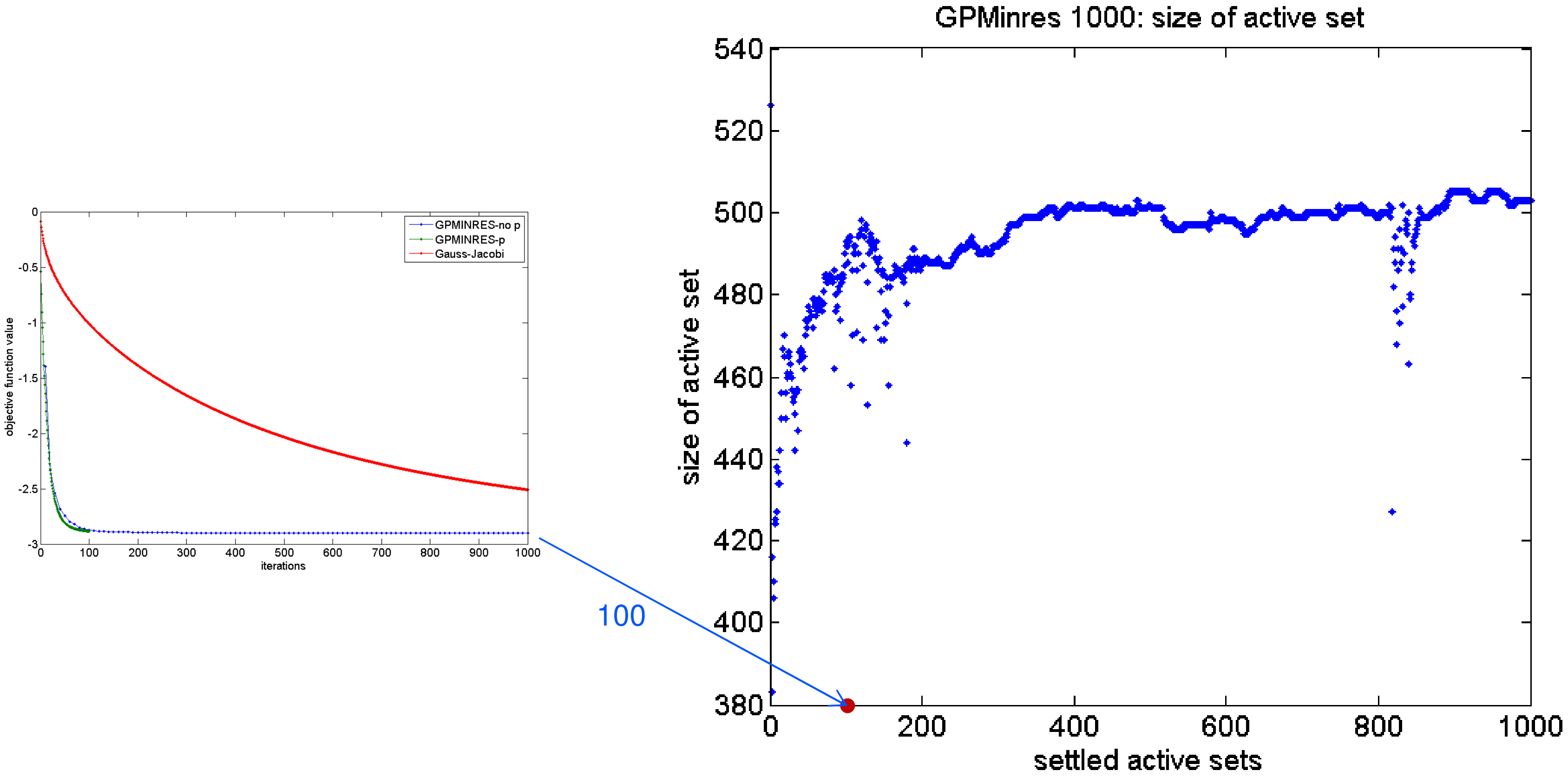


- Plot shows the size of the active set each time the inner unconstrained sub-problem is solved.

- For some undetermined reason, the active set briefly becomes unsettled at about 850 active sets.

History of Active Set Size

[1K bodies, 3525 contacts]

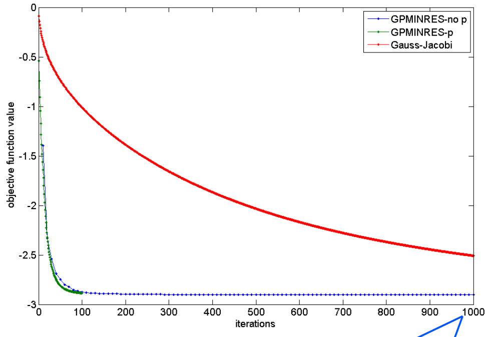


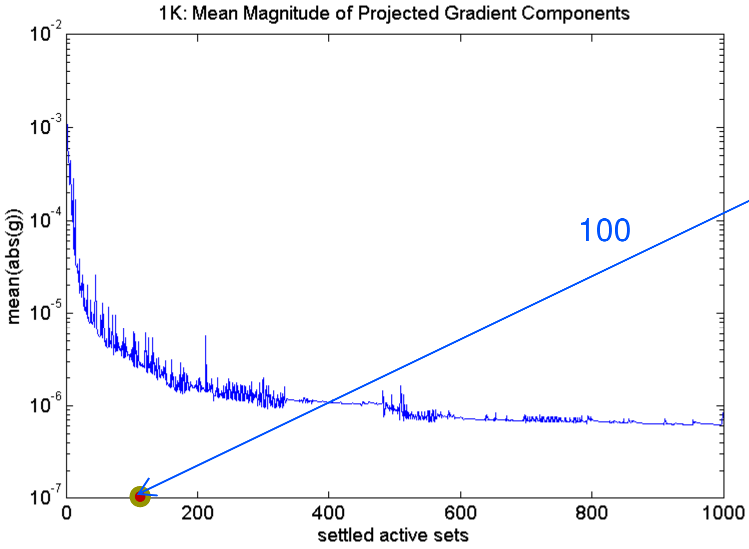
- Note that the value of the objective function after 100 active sets reported a couple of slides back comes at a time when the active set is relatively unsettled

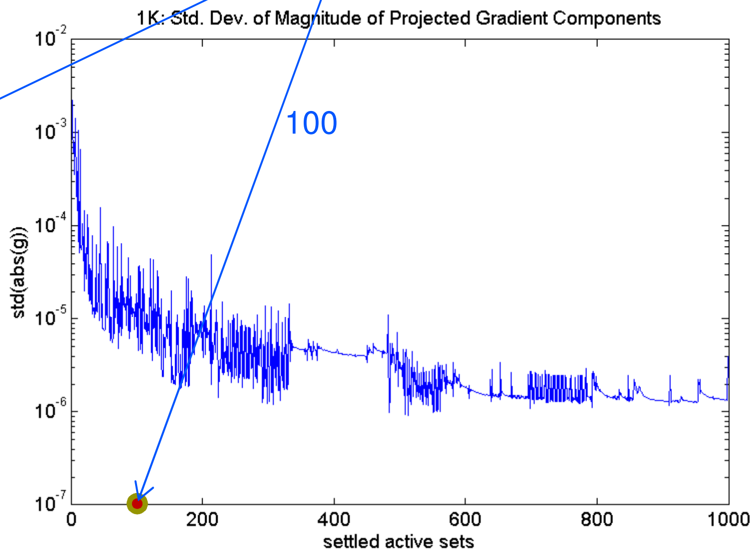
- However, it is not drastically different than the value of the cost function after 1000 active set changes.

Magnitude of Projected Gradient

[1K bodies, 3525 contacts]




1K: Mean Magnitude of Projected Gradient Components


1K: Std. Dev. of Magnitude of Projected Gradient Components

- Stopping criteria should be based on magnitude of projected gradient:

$$||\nabla_\Omega q(\gamma)|| \leq \tau$$

- Projected gradient defined as

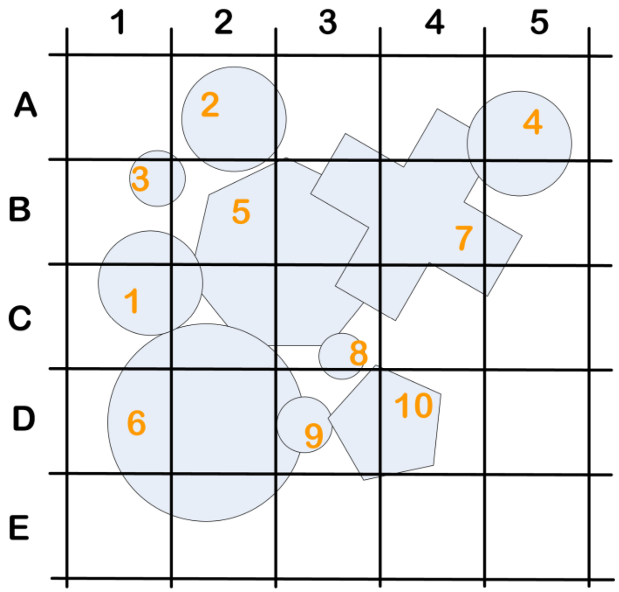
$$[\nabla_\Omega q(\gamma)]_i = \begin{cases} \partial_i q(\gamma) & \text{if} \quad \gamma_i > 0 \\ \min(\partial_i q(\gamma), 0) & \text{if} \quad \gamma_i = 0 \end{cases}$$

- Multi-Physics targeted Computational Dynamics requires
  - Advanced modeling techniques
  - Strong algorithmic (applied math) support
  - **Proximity computation**
  - Domain decomposition & Inter-domain data exchange
  - Post-processing (visualization)

- Collision Detection is hard

CD: Binning

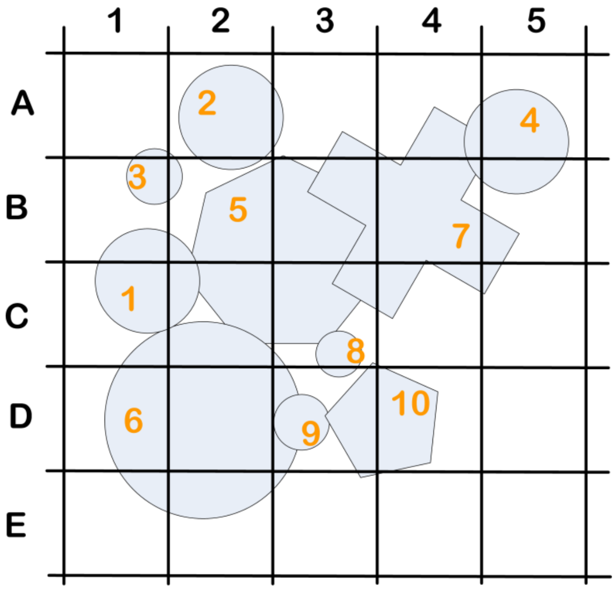
- Example: 2D collision detection, bins are squares

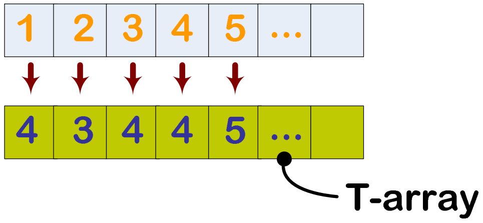


- Body 4 touches bins A4, A5, B4, B5
- Body 7 touches bins A3, A4, A5, B3, B4, B5, C3, C4, C5
- In proposed algorithm, bodies 4 and 7 will be checked for collision by three threads (associated with bin A4, A5, B4)

Stage 1 (Body Parallel)

- Purpose: find the number of bins touched by each body
- Store results in the "**T**", array of N integers
- Key observation: it's easy to bin bodies



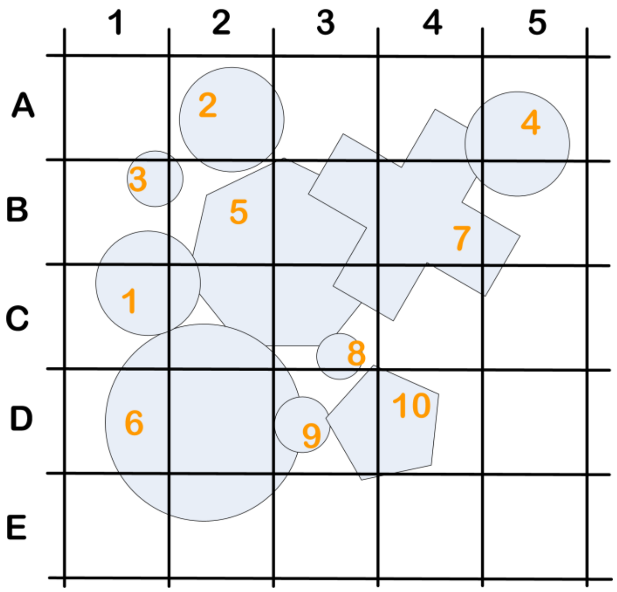


Stage 2: Parallel Inclusive Scan

- Run a parallel inclusive scan on the array **T**
  - The last element is the total number of bin touches, including the last body

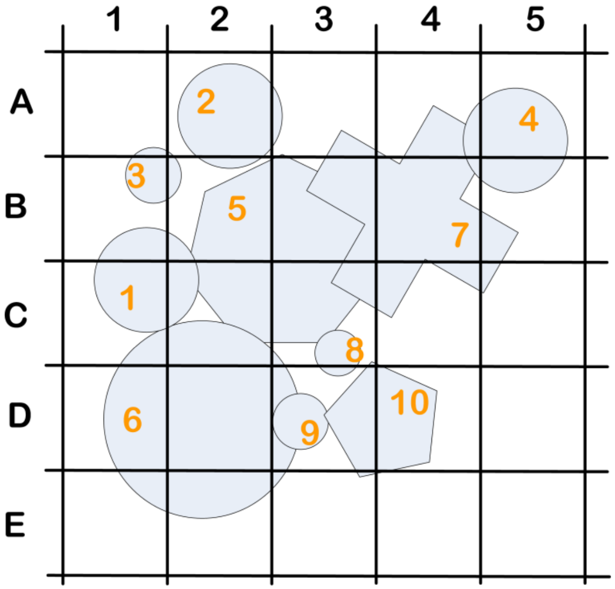
- Complexity of Stage: O(N) – **thrust** library

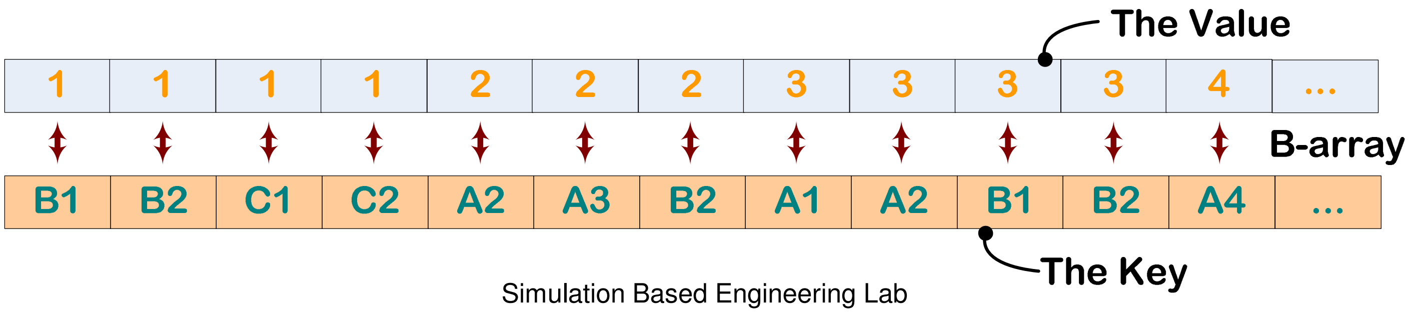
- Purpose: determine the number of entries M needed to store the indices of all the bins touched by each body in the problem



Stage 3: Determine bin-to-body association

- Stage executed in parallel on a per-body basis
- Allocate an array **B** of M pairs of integers.
  - The key (first entry of the pair), is the bin index
  - The value (second entry of pair) is the body that touches that bin





The Value

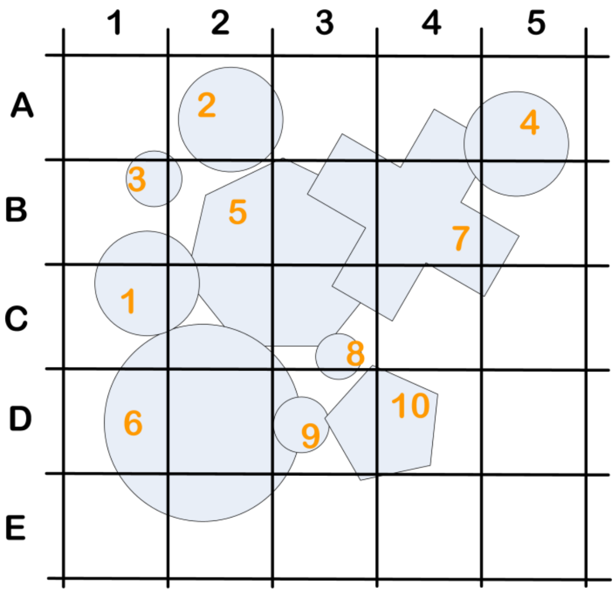
1	1	1	1	2	2	2	3	3	3	3	4	…
B1	B2	C1	C2	A2	A3	B2	A1	A2	B1	B2	A4	…

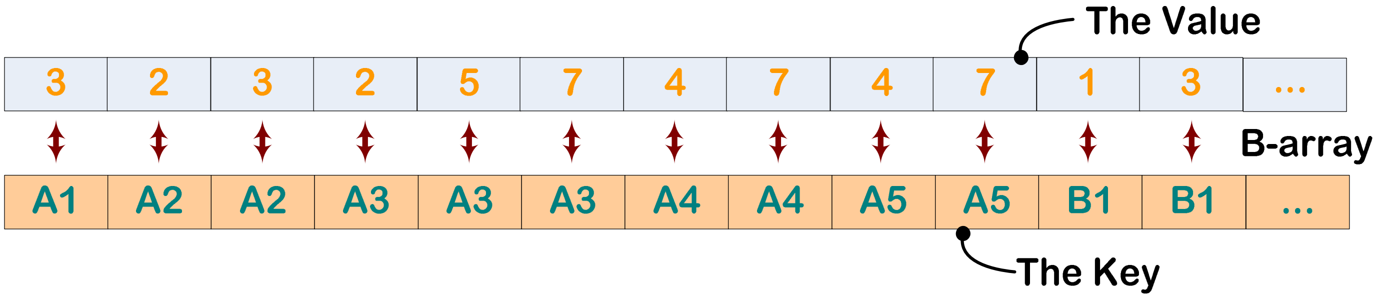
B-array

The Key

Stage 4: Radix Sort

- In parallel, run radix sort to order the B array according to key values
- Work load: O(N)
  - Relies on thrust library



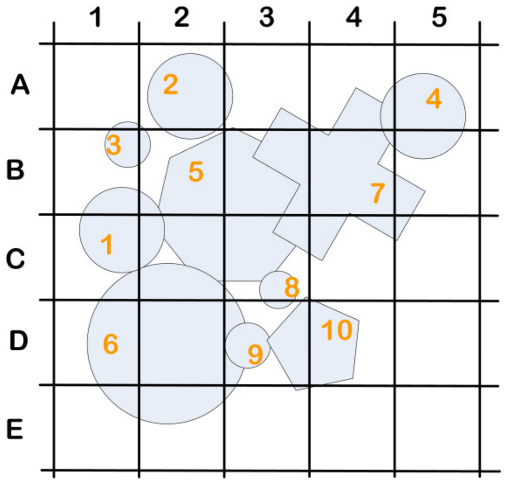


3	2	3	2	5	7	4	7	4	7	1	3	…
A1	A2	A2	A3	A3	A3	A4	A4	A5	A5	B1	B1	…

Stage 5: Find Bin Starting Index

- Host allocates on device an array of length N_b of pairs of unsigned integers

- Run in parallel, on a per bin basis:
  - Load in parallel in shared memory chunks of the **B** array and find the location where each bin starts
  - Store it in entry *k* of **C**, as the key associated with this pair
  - Key of bins with one or no bodies is set to maximum unsigned int value of 0xffffffff



0	1	2	3	4	5	6	7	8	9	10	11	
3	2	3	2	5	7	4	7	4	7	1	3	…
A1	A2	A2	A3	A3	A3	A4	A4	A5	A5	B1	B1	…

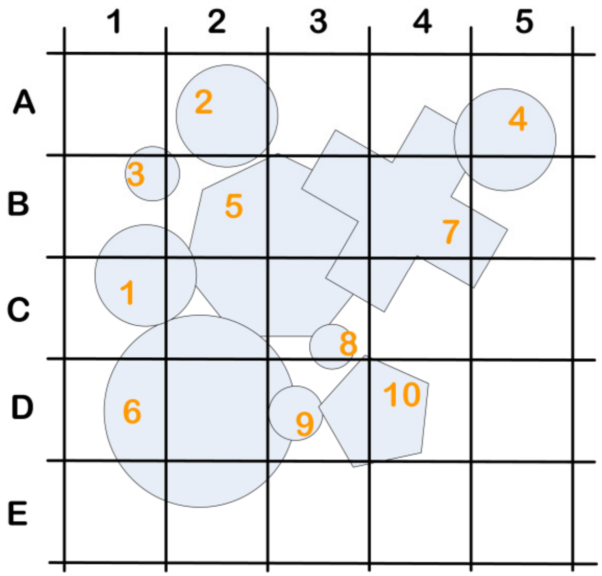
B-array

C-array

The Value	0	1	3	6	8	10	…
The Key	0xfff	2	3	2	2	2	…
	A1	A2	A3	A4	A5	B1	…

Stage 6: Sort C for Pruning

- Do a parallel radix sort on the array C based on the key
- Purpose: move unused bins to the end of array
- Effort: $O(N_b)$



C-array

The Value	0	1	3	6	8	10	...
The Key	0xfff	2	3	2	2	2	...
	A1	A2	A3	A4	A5	B1	…

Becomes this

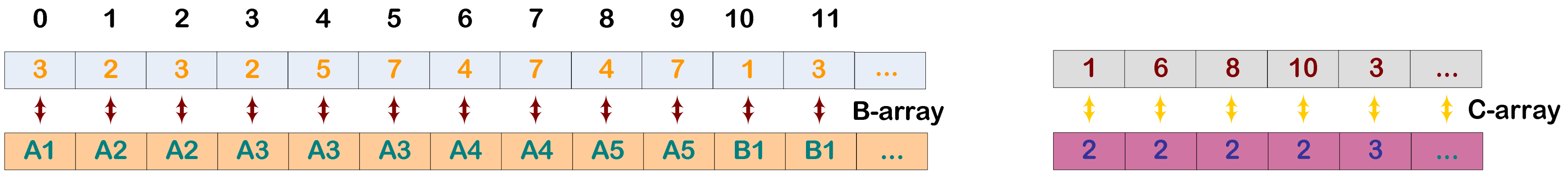
C-array

The Value	1	6	8	10	3	...	
The Key	2	2	2	2	3	...	

…

Stage 7: Investigate Collisions in each Bin

- Carried out in parallel, one thread per bin



0	1	2	3	4	5	6	7	8	9	10	11	
3	2	3	2	5	7	4	7	4	7	1	3	…
A1	A2	A2	A3	A3	A3	A4	A4	A5	A5	B1	B1	…

B-array

1	6	8	10	3	…
2	2	2	2	3	…

C-array

- To store information generated during this stage, host needs to allocate an unsigned integer array **D** of length N_b
  - Array **D** stores the number of actual contacts occurring in each bin
  - **D** is in sync with (linked to) **C**, which in turn is in sync with (linked to) **B**

- Parallelism: one thread per bin
  - Thread k reads the pair key-value in entry k of array C
  - Thread k reads does rehearsal for brute force collision detection
  - Outcome: the number *s* of active collisions taking place in a bin
    - Value *s* stored in kth entry of the **D** array

Stage 7, details…

- In order to carry out this stage you need to keep in mind how C is organized, which is a reflection of how B is organized

B-array

0	1	2	3	4	5	6	7	8	9	10	11	
3	2	3	2	5	7	4	7	4	7	1	3	…
A1	A2	A2	A3	A3	A3	A4	A4	A5	A5	B1	B1	…

Bin vs. Body Touching This Bin

Shows up 2 since there are two bodies (4 & 7) in bin with offset 6 (A4)

Bin offset in B and number of bodies touching that bin

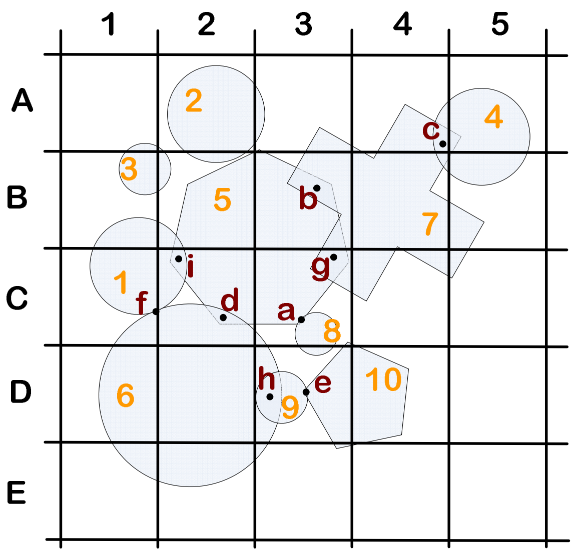
C-array

1	6	8	10	3	…
2	2	2	2	3	…

- The drill: thread 0 relies on info at **C**[0], thread 1 relies on info at **C**[1], etc.
- Let's see what thread 2 (goes with **C**[2]) does:
  - Read the first 2 bodies that start at offset 6 in B.
    - These bodies are 4 and 7, and as **B** indicates, they touch bin A4
    - Bodies 4 and 7 turn out to have 1 contact in A4, which means that entry 2 of **D** needs to reflect this

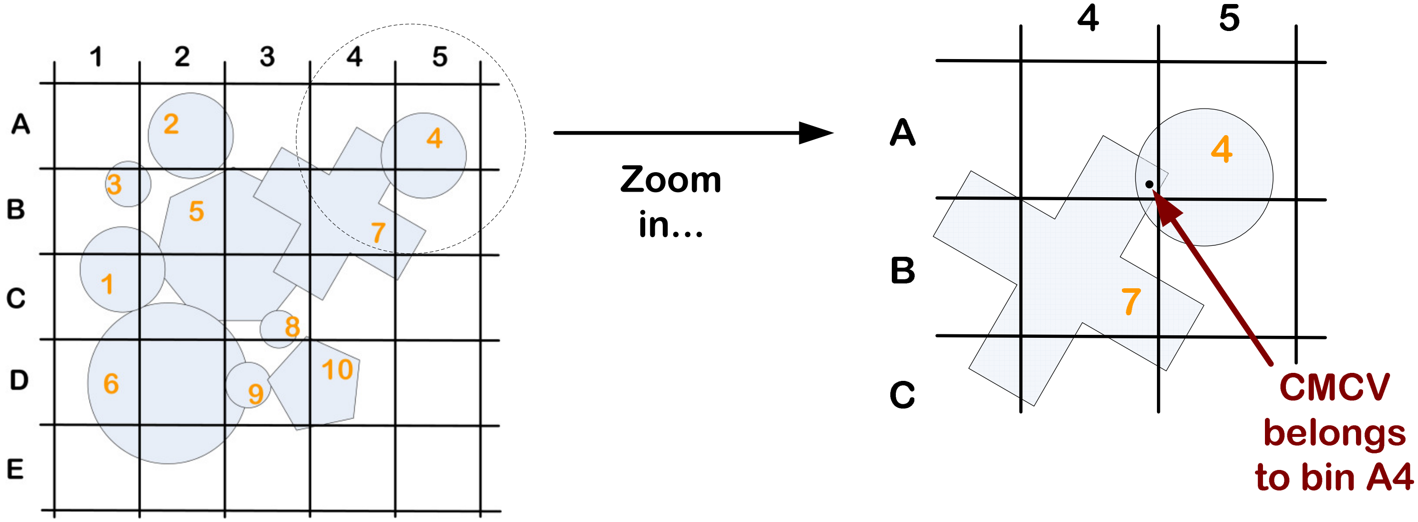
D-array (Length: N_b)

0	1	2	3	4	…
0	1	0	0	0	…
(A2)	(A4)	(A5)	(B1)	(A3)	…



Stage 7, details

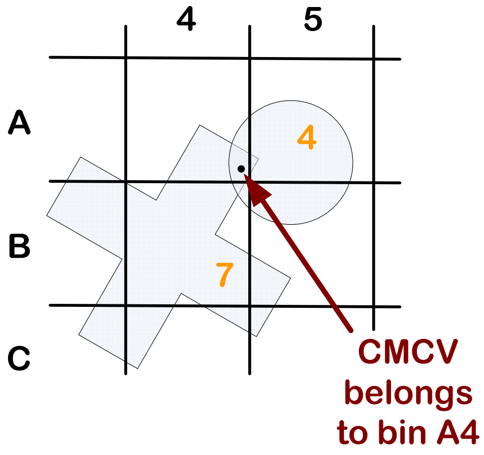
- Brute Force CD rehearsal
  - Carried out to understand the memory requirements associated with collisions in each bin
    - Finds out the total number of contacts owned by a bin
  - Key question: which bin does a contact belong to?
    - Answer: It belongs to bin containing the CM of the Contact Volume (CMCV)



Stage 7, Comments

- Two bodies can have multiple contacts, handled ok by the method

- Easy to define the CMCV for two spheres, two ellipsoids, and a couple of other simple geometries
  - In general finding CMCV might be tricky
    - Notice picture below, CM of 4 is in A5, CM of 7 is in B4 and CMCV is in A4
  - Finding the CMCV is the subject of the so called "narrow phase collision detection"
    - It'll be simple in our case since we are going to work with simple geometry primitives



Stage 8: Inclusive Prefix Scan

- Save to the side the number of contacts in the last bin (last entry of **D**) d_{last}
  - Last entry of **D** will get overwritten

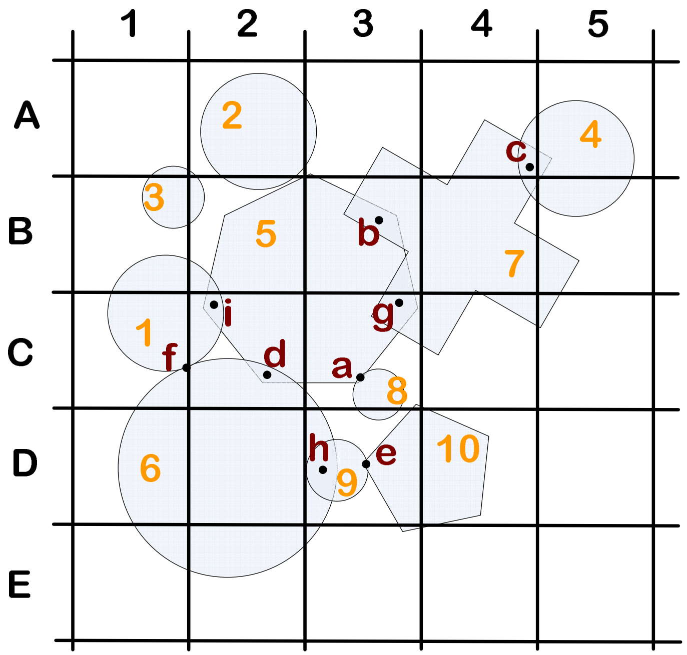
0	1	2	3	4	...
0	1	0	0	0	...
(A2)	(A4)	(A5)	(B1)	(A3)	...

D-array (Length: N_b)

- Run parallel exclusive prefix scan on **D**:

0	1	2	3	4	...
0	0	1	1	1	...
(A2)	(A4)	(A5)	(B1)	(A3)	...

D-array, after exclusive prefix scan



- Total number of actual collisions:

$$N_c = \mathbf{D}[N_b] + d_{last}$$

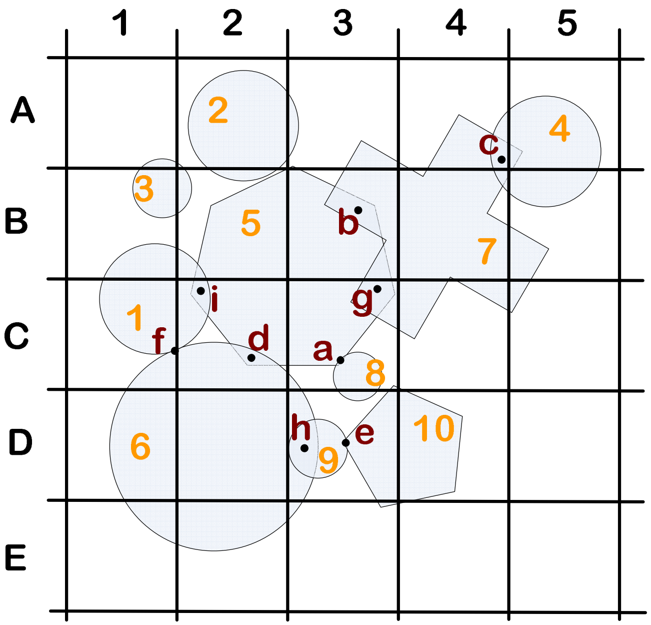
Stage 9: Populate Array E

- From the host, allocate on the device memory for array **E**
  - Array **E** stores the required collision information: normal, two tangents, etc.
  - Number of entries in the array: N_c (see previous slide)

- In parallel, on a per bin basis (one thread/bin):
  - Populate the **E** array with required info

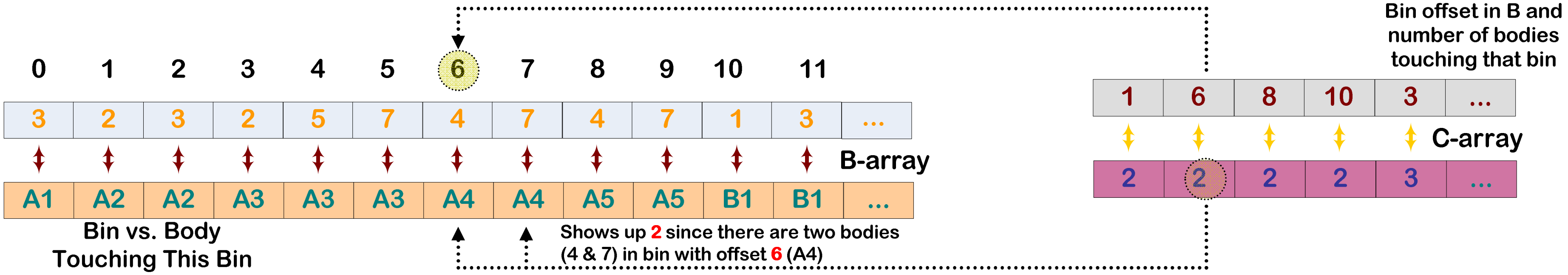
- Not discussed in greater detail, this is just like Stage 7, but now you have to generate actual collision info (stage 7 was the rehearsal)

- Thread for A4 will generate the info for contact “c”
- Thread for C2 will generate the info for “i” and “d”
- Etc.



Stage 9, details

- **B**, **C**, **D** required to populate array **E** with collision information



Bin offset in B and number of bodies touching that bin

0	1	2	3	4	5	6	7	8	9	10	11	
3	2	3	2	5	7	4	7	4	7	1	3	…
A1	A2	A2	A3	A3	A3	A4	A4	A5	A5	B1	B1	…

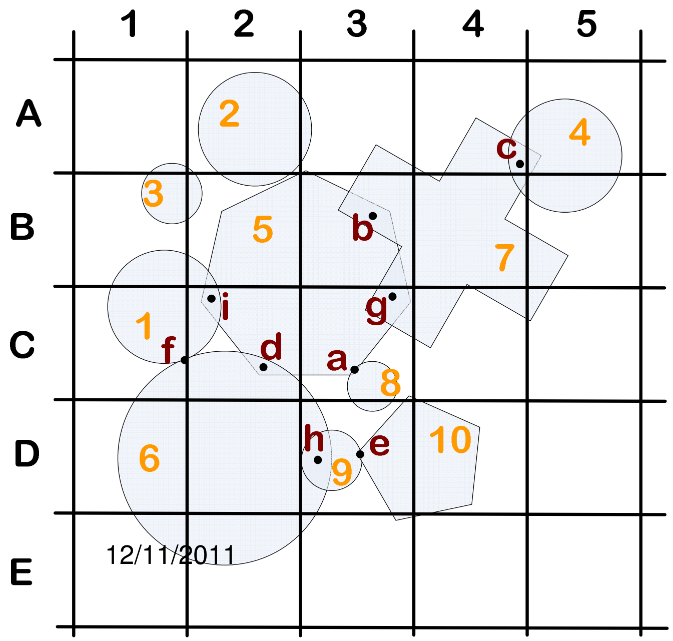
B-array

Bin vs. Body Touching This Bin

1	6	8	10	3	…
2	2	2	2	3	…

C-array

Shows up 2 since there are two bodies (4 & 7) in bin with offset 6 (A4)



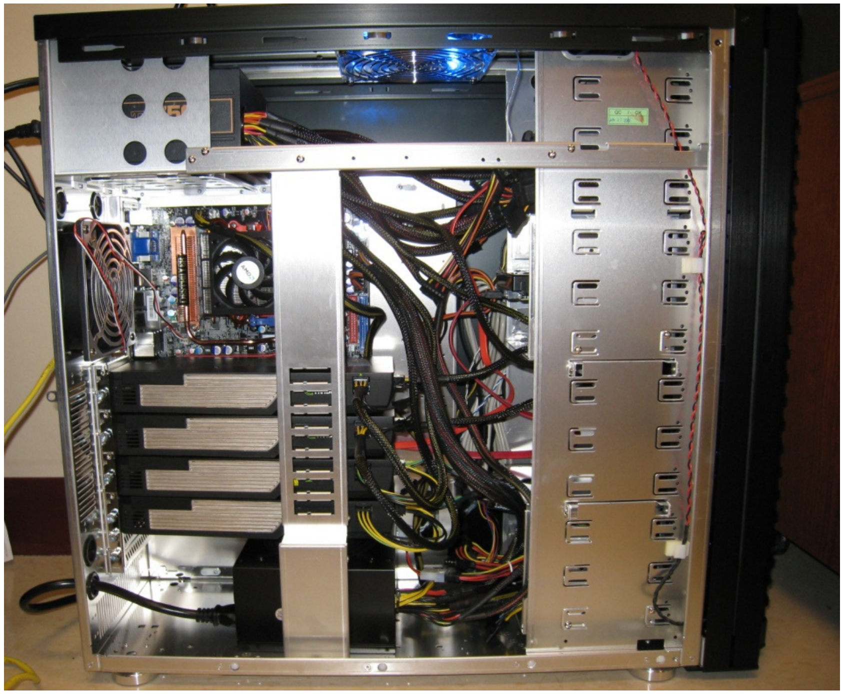
0	1	2	3	4	…
0	0	1	1	1	…
(A2)	(A4)	(A5)	(B1)	(A3)	…

D-array, after exclusive prefix scan

- **C** and **B** are needed to compute the collision information
- **D** is needed to understand where the collision information will be stored in **E**

Multiple-GPU Collision Detection

Assembled Quad GPU Machine



Processor: AMD Phenom II X4 940 Black

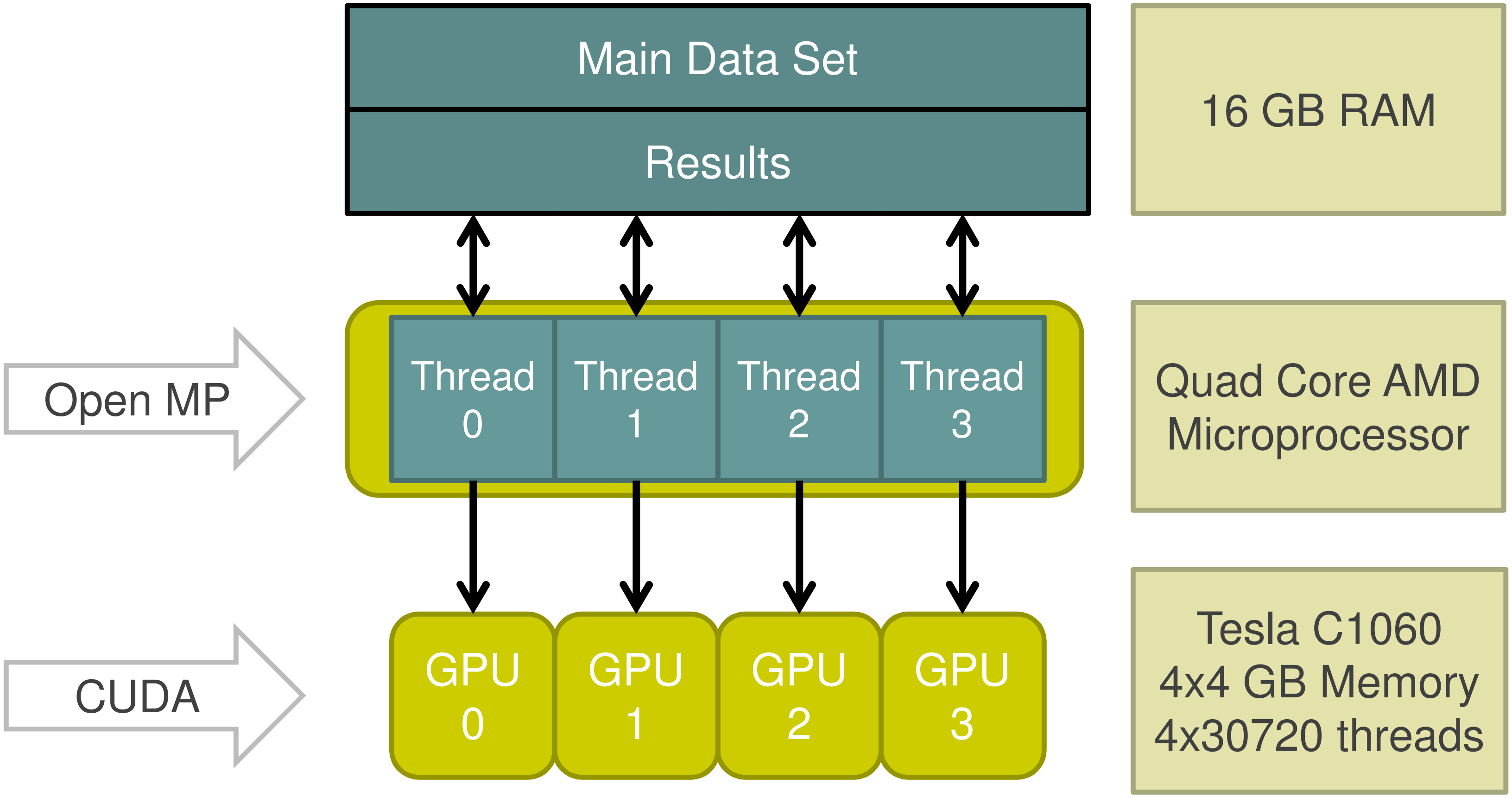
Memory: 16GB DDR2

Graphics: 4x NVIDIA Tesla C1060

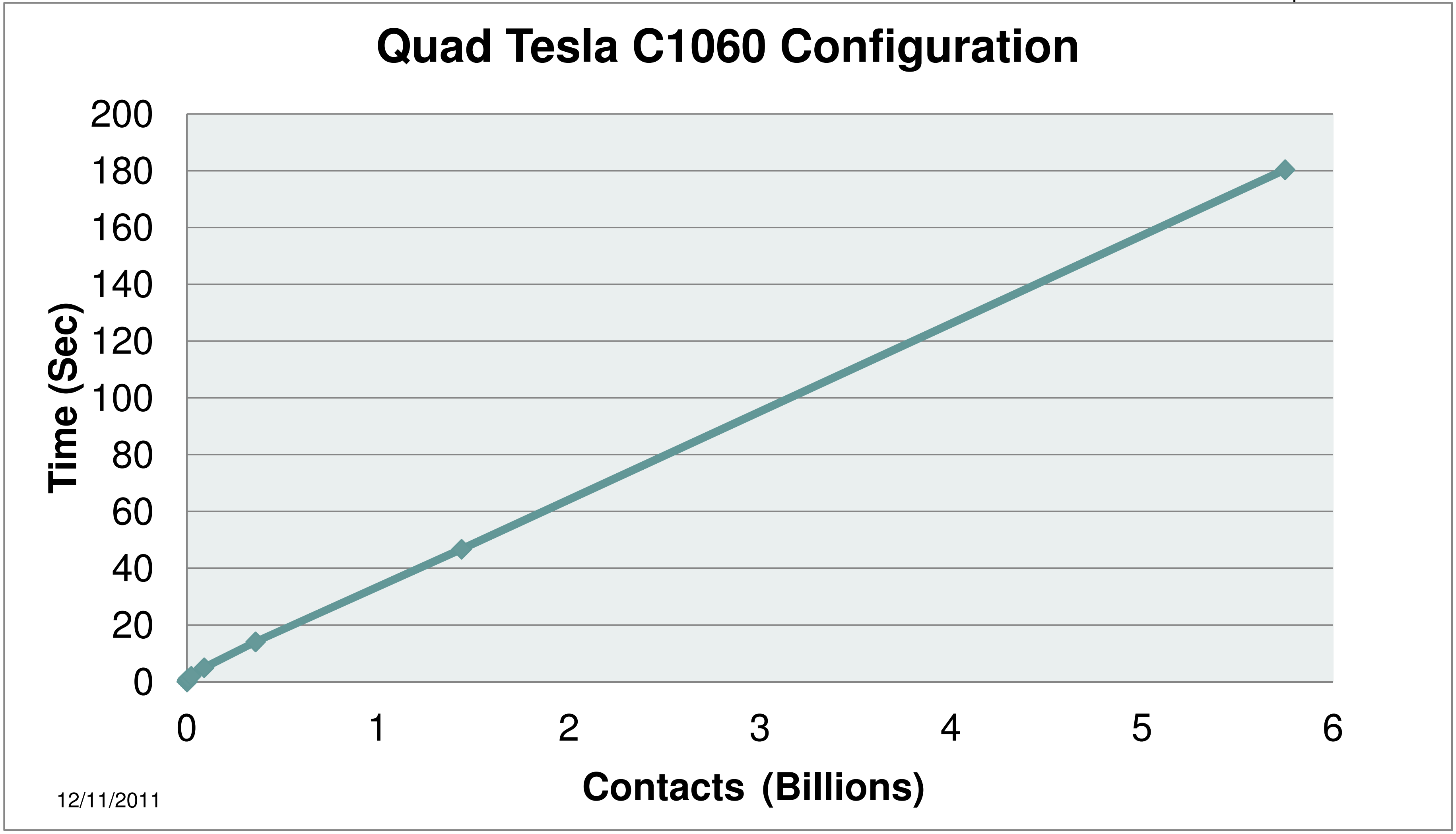
Power supply 1: 1000W

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

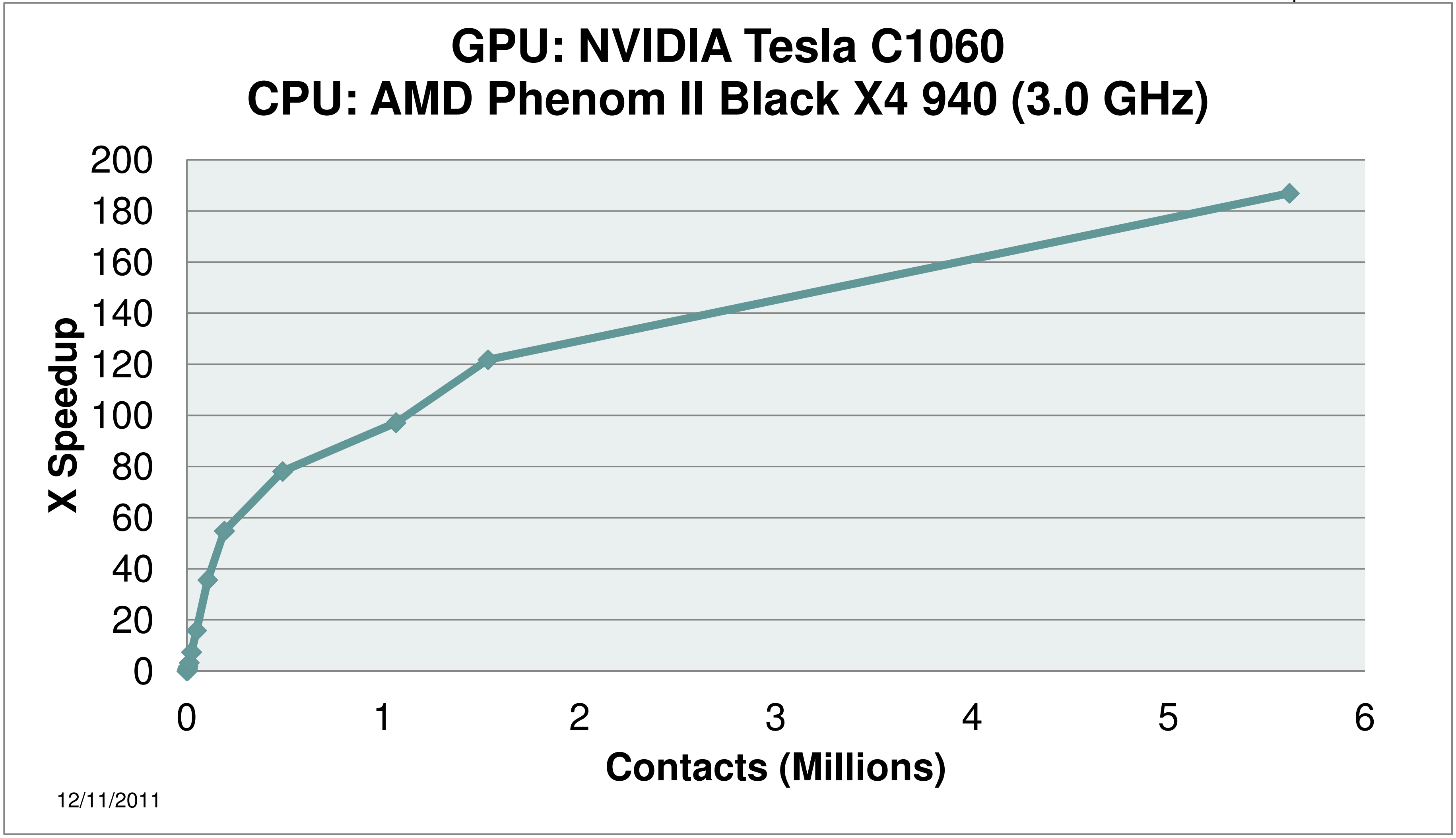
Results – Contacts vs. Time



12/11/2011

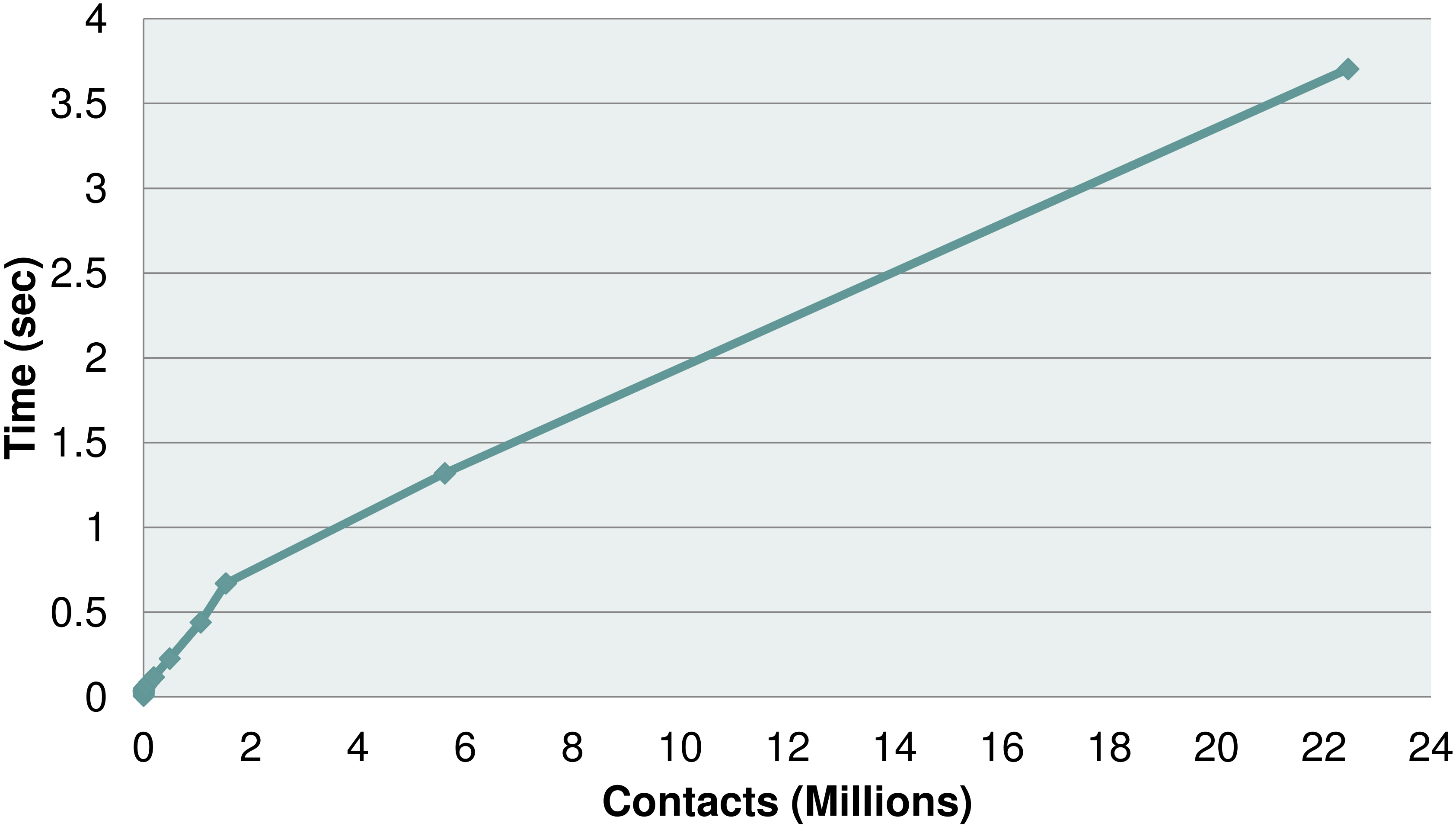
Speedup - GPU vs. CPU (Bullet library)

[results reported are for spheres]

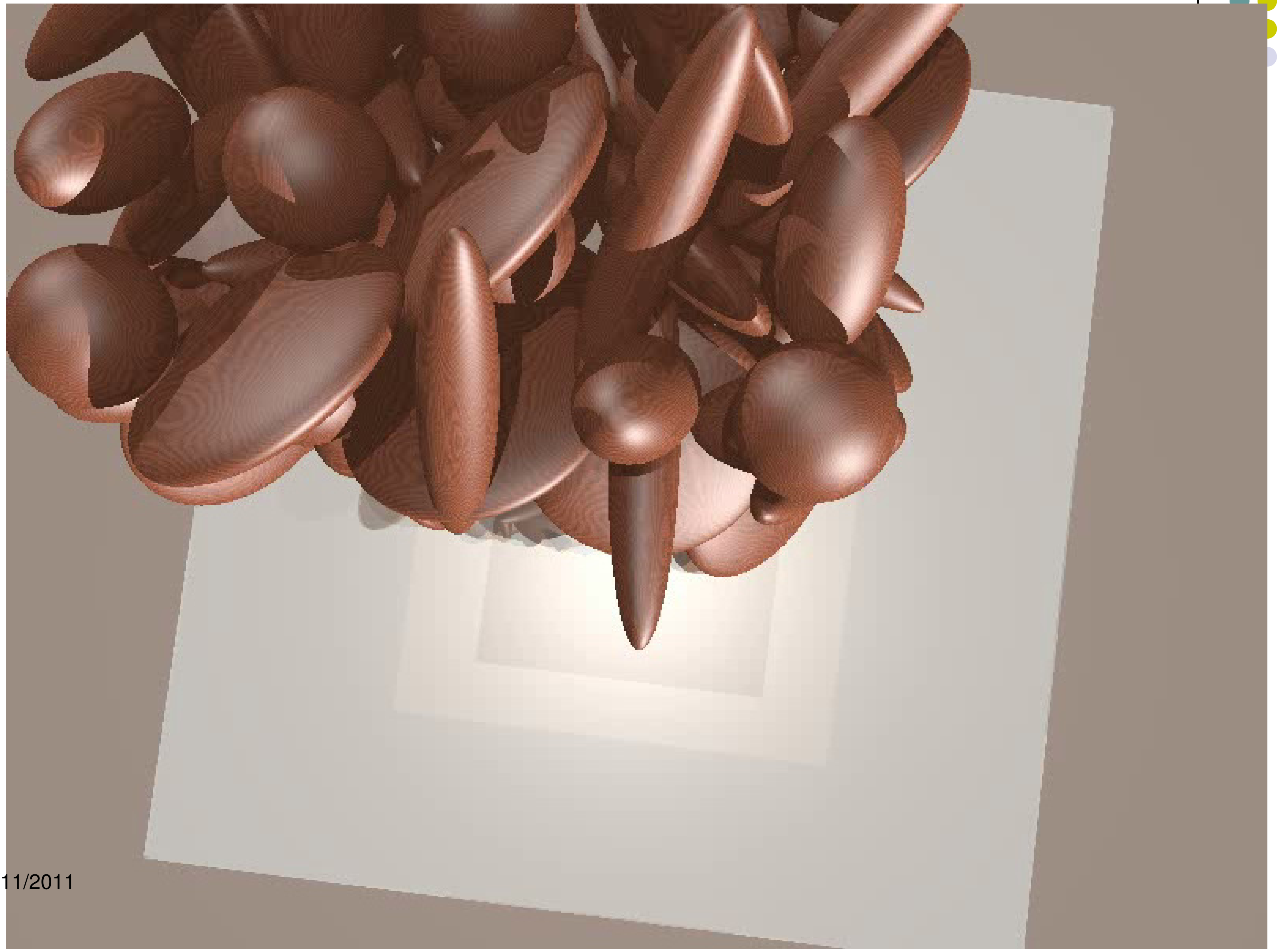


12/11/2011

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



Ellipsoid-Ellipsoid CD: Visualization



Example: Ellipsoid-Ellipsoid CD

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

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

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

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

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

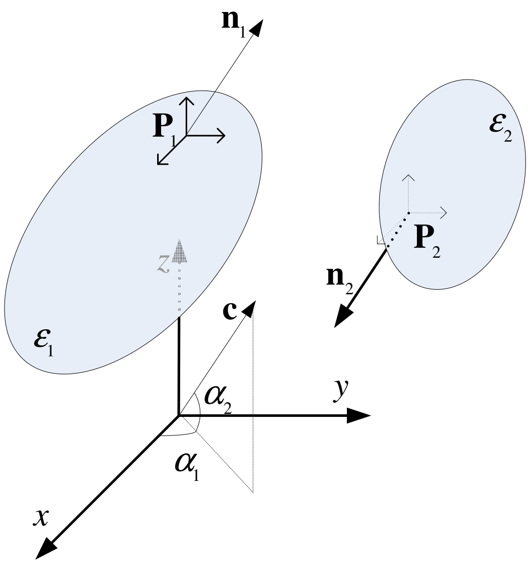
$\mathbf{A}$: Rotation Matrix

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

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

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

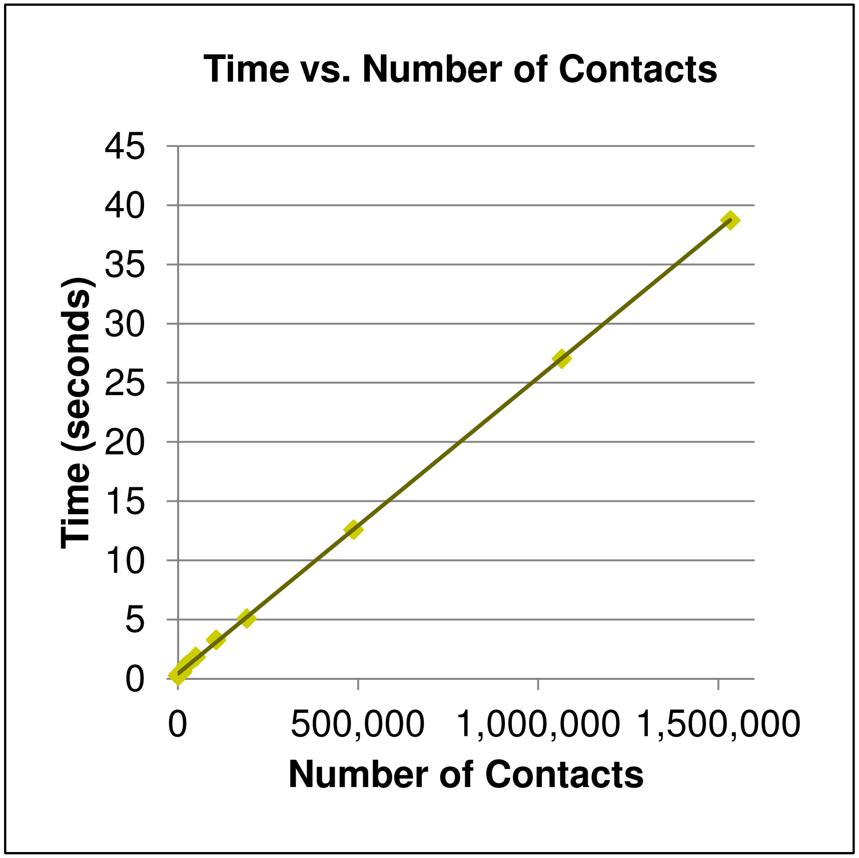
$$\lambda^2=\frac{1}{4}\mathbf{n}^T\mathbf{M}\mathbf{n}$$

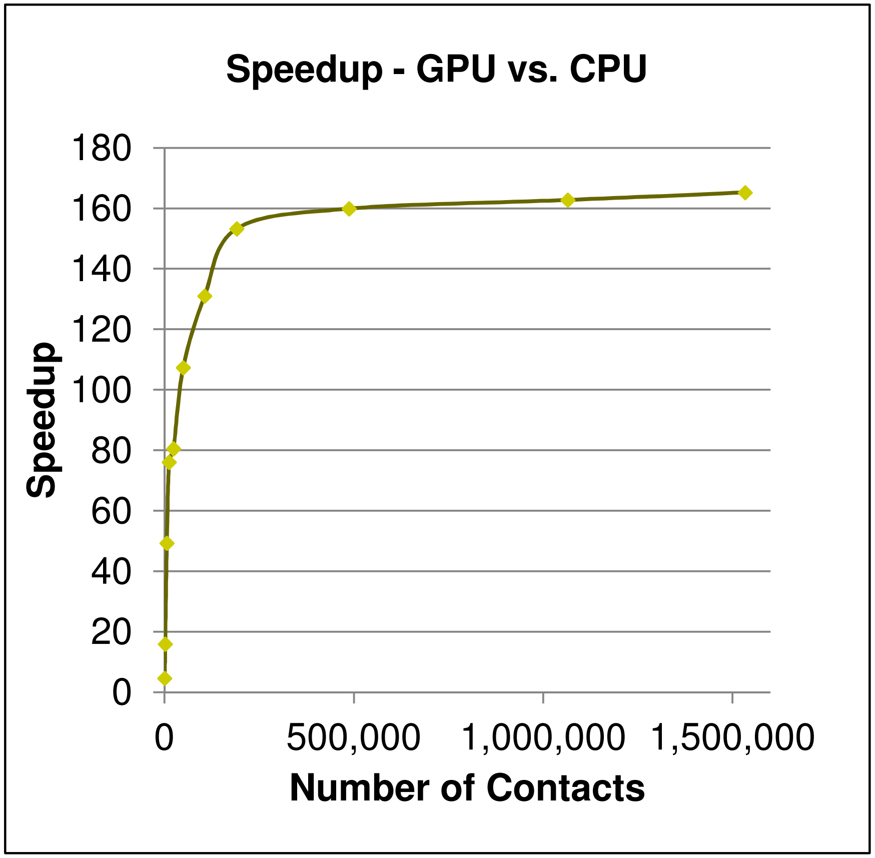


$$\mathbf{d}=\mathbf{P}_1-\mathbf{P}_2$$

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

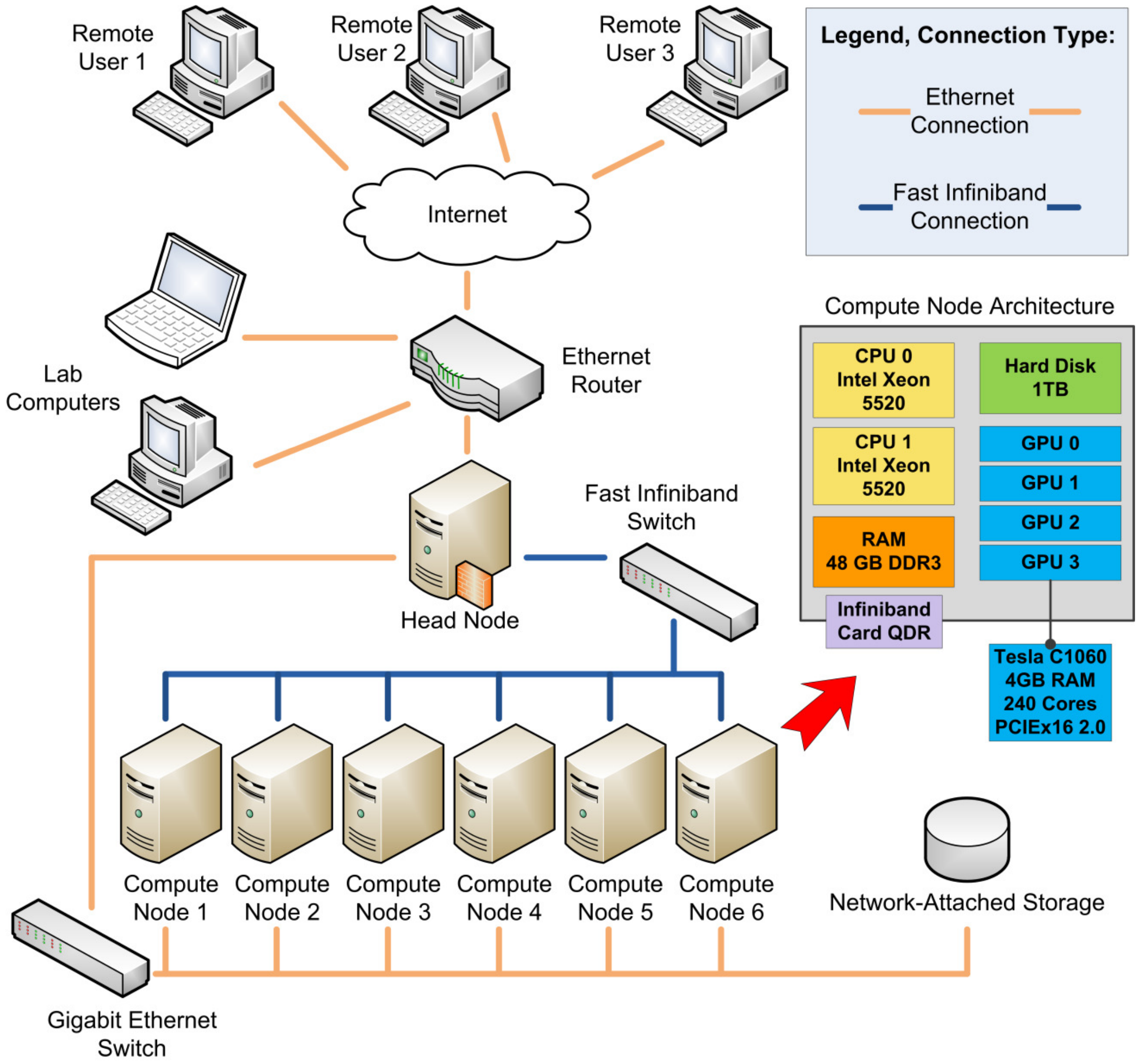
Ellipsoid-Ellipsoid CD: Results





- Multi-Physics targeted Computational Dynamics requires
  - Advanced modeling techniques
  - Strong algorithmic (applied math) support
  - Proximity computation
  - **Domain decomposition & Inter-domain data exchange**
  - Post-processing (visualization)

Heterogeneous Cluster



Remote User 1

Remote User 2

Remote User 3

Internet

Lab Computers

Ethernet Router

Head Node

Fast Infiniband Switch

Compute Node 1

Compute Node 2

Compute Node 3

Compute Node 4

Compute Node 5

Compute Node 6

Network-Attached Storage

Gigabit Ethernet Switch

Legend, Connection Type:

Ethernet Connection

Fast Infiniband Connection

Compute Node Architecture

CPU 0 Intel Xeon 5520

CPU 1 Intel Xeon 5520

RAM 48 GB DDR3

Infiniband Card QDR

Hard Disk 1TB

GPU 0

GPU 1

GPU 2

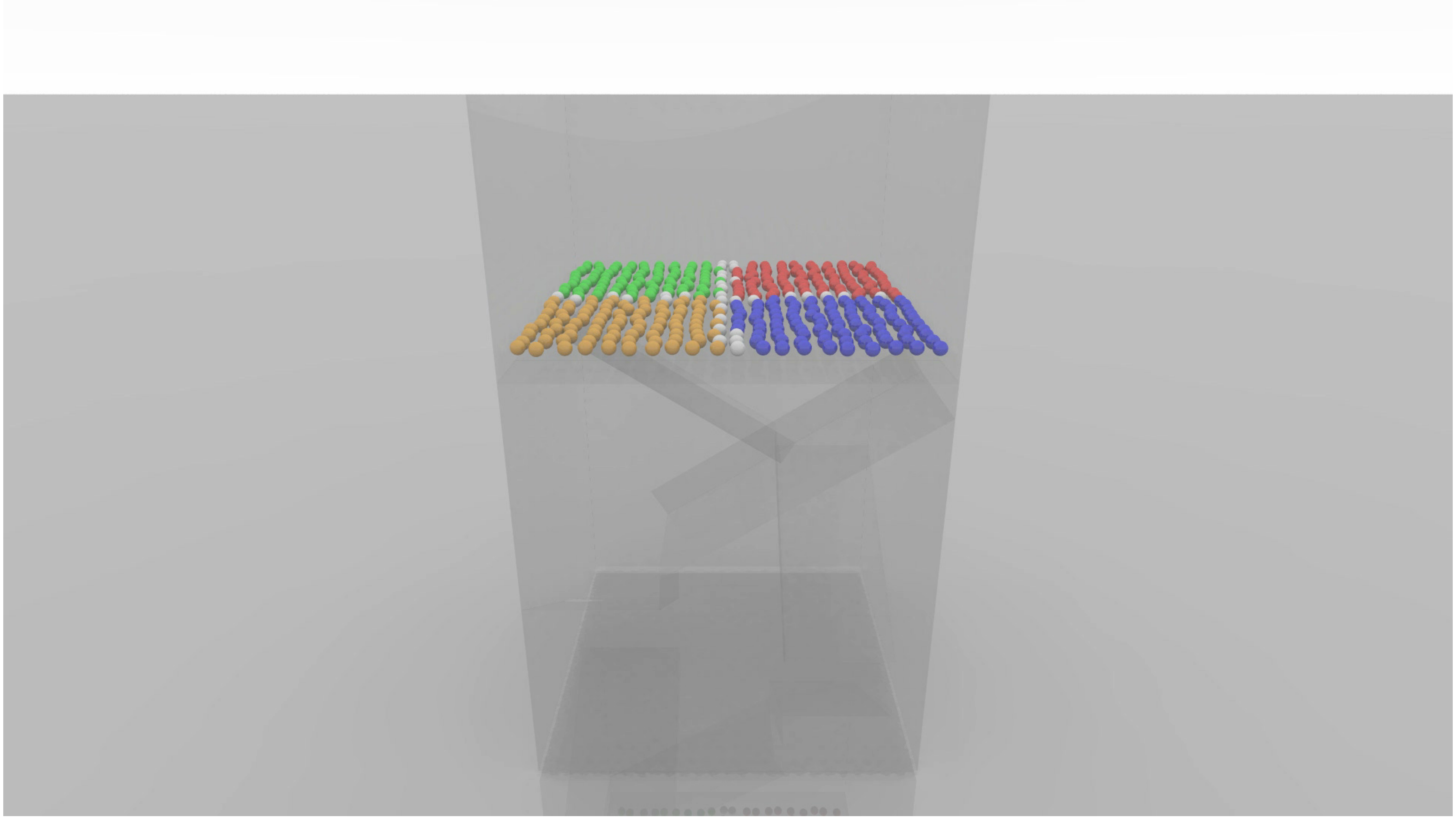
GPU 3

Tesla C1060 4GB RAM 240 Cores PCIEx16 2.0

Juggling World Record:

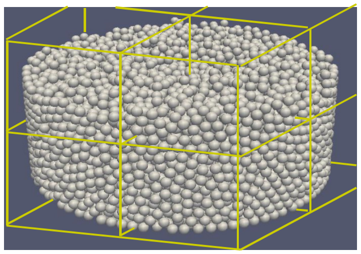
64 People Juggling (of all places) in Madison, Wisconsin

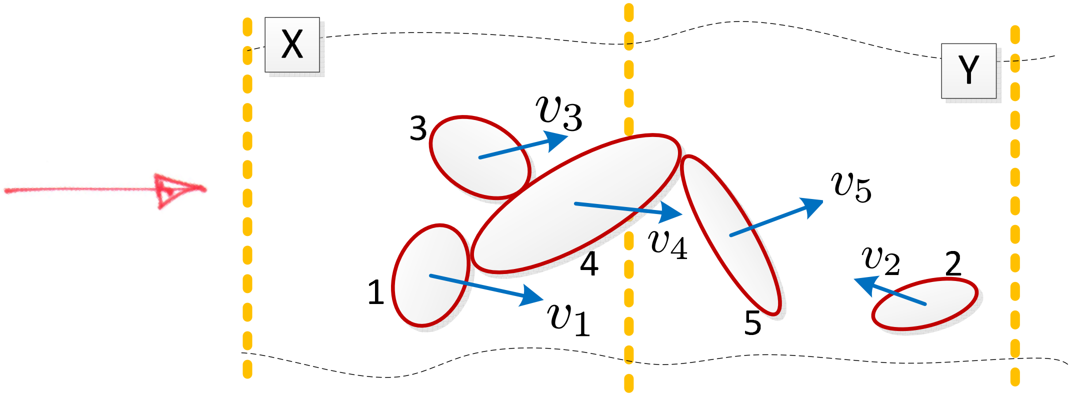
Computation Using Multiple CPUs



HCT: Domain decomposition & Inter-domain data exchange

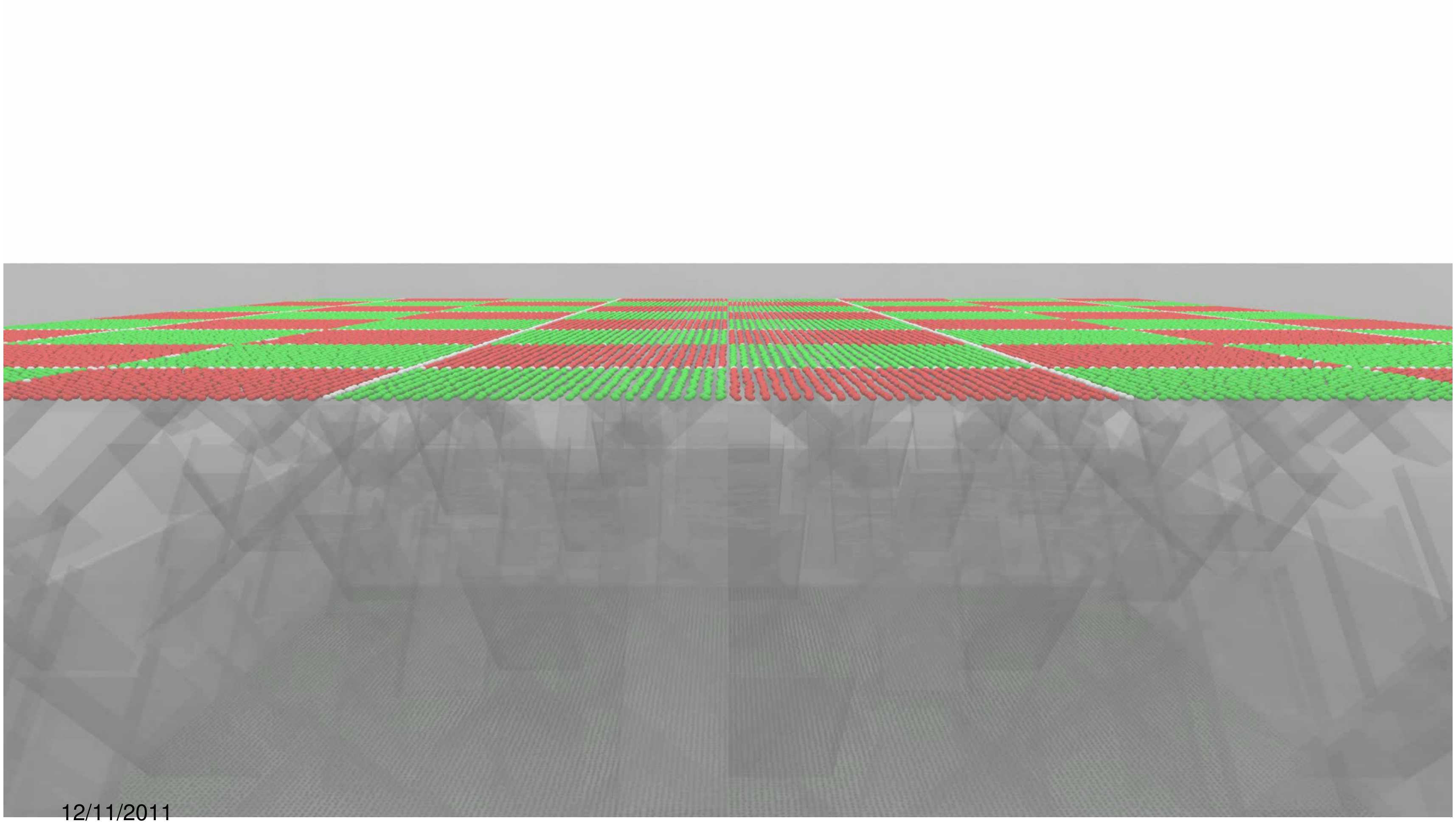
- Relates to the ability to divide the simulation into chunks and have multiple CPUs/GPUs exchange data during simulation as needed
- Elements leave one subdomain to move to a different one
- Key issues:
  - Dynamic load balancing
  - Establish a dynamic data exchange protocol (DDEP) between sub-domains



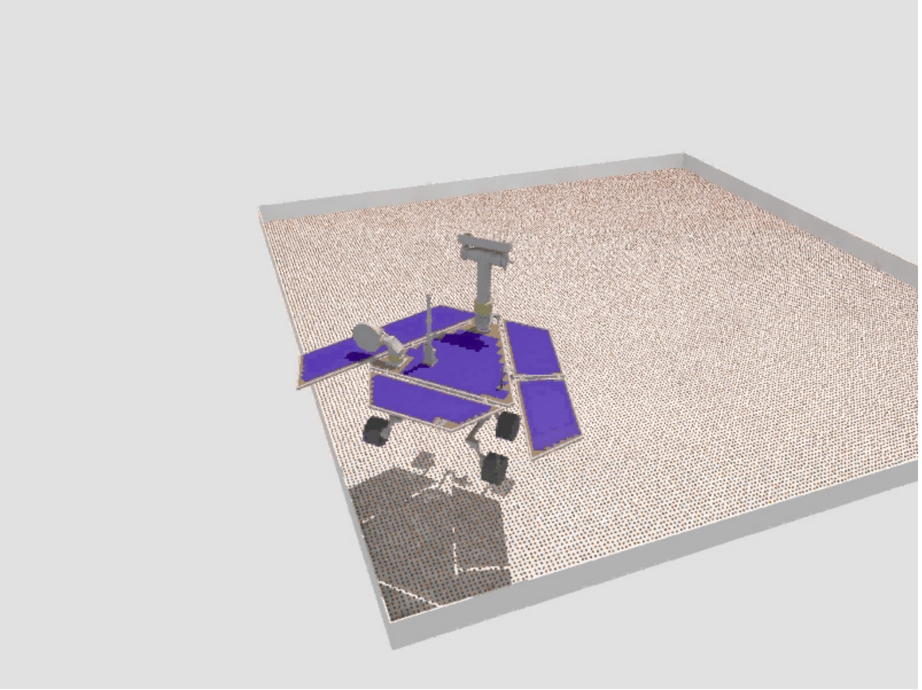


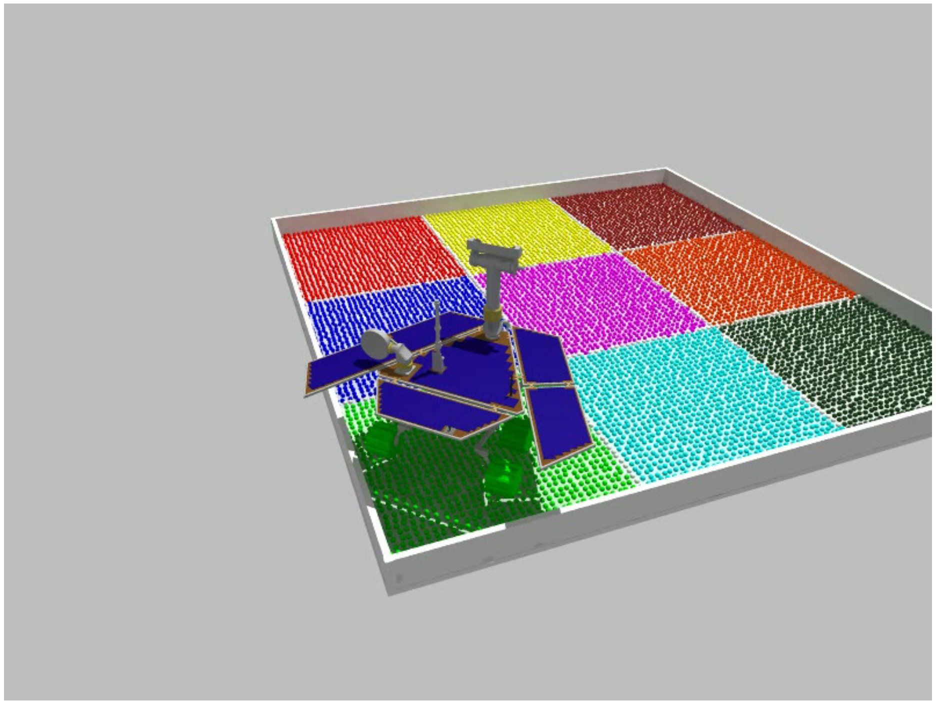
0.5 Million Bodies on 64 Cores

[Penalty Approach, MPI-based]



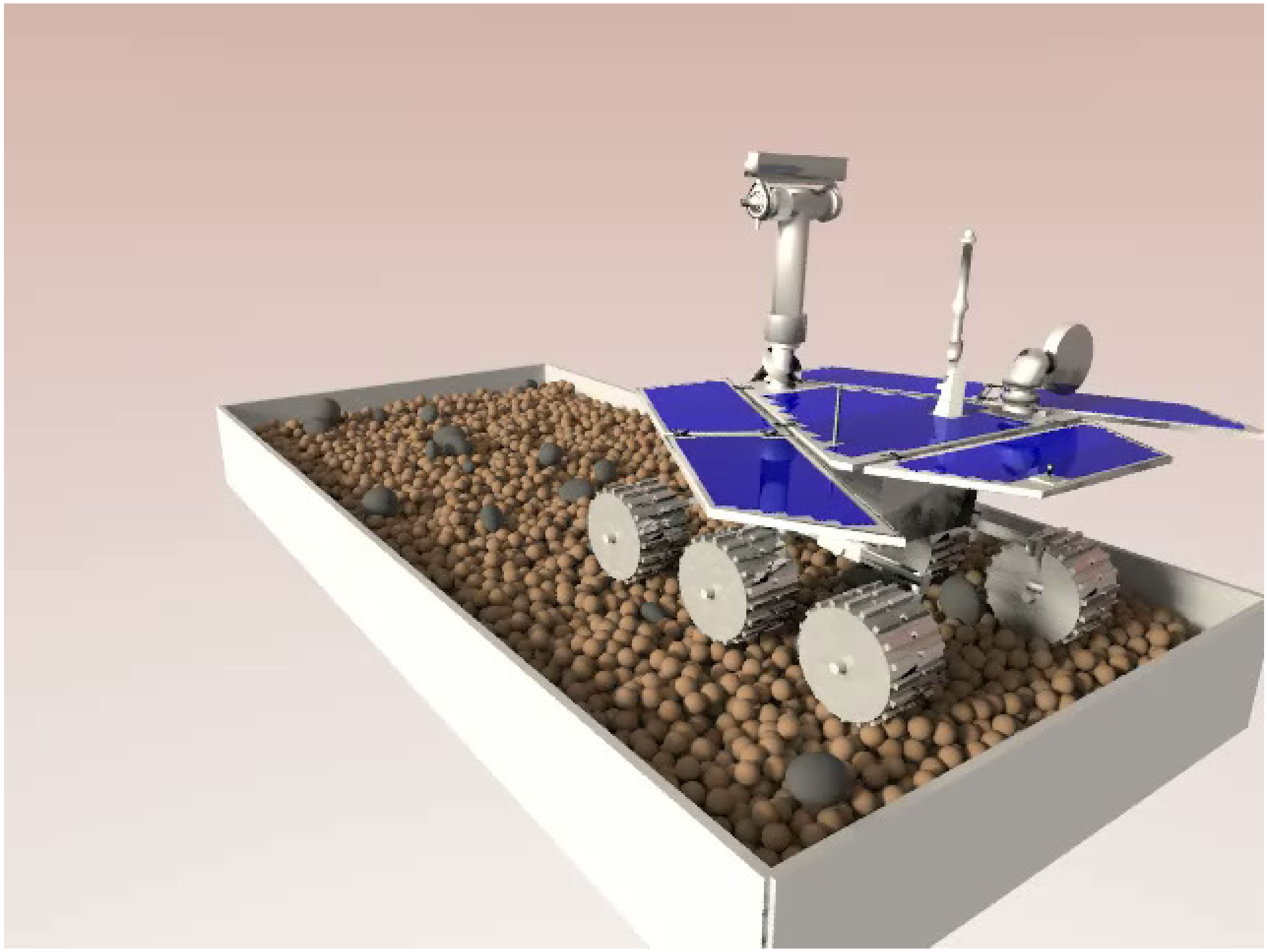
Computation Using Multiple CPUs





Three Years Ago…

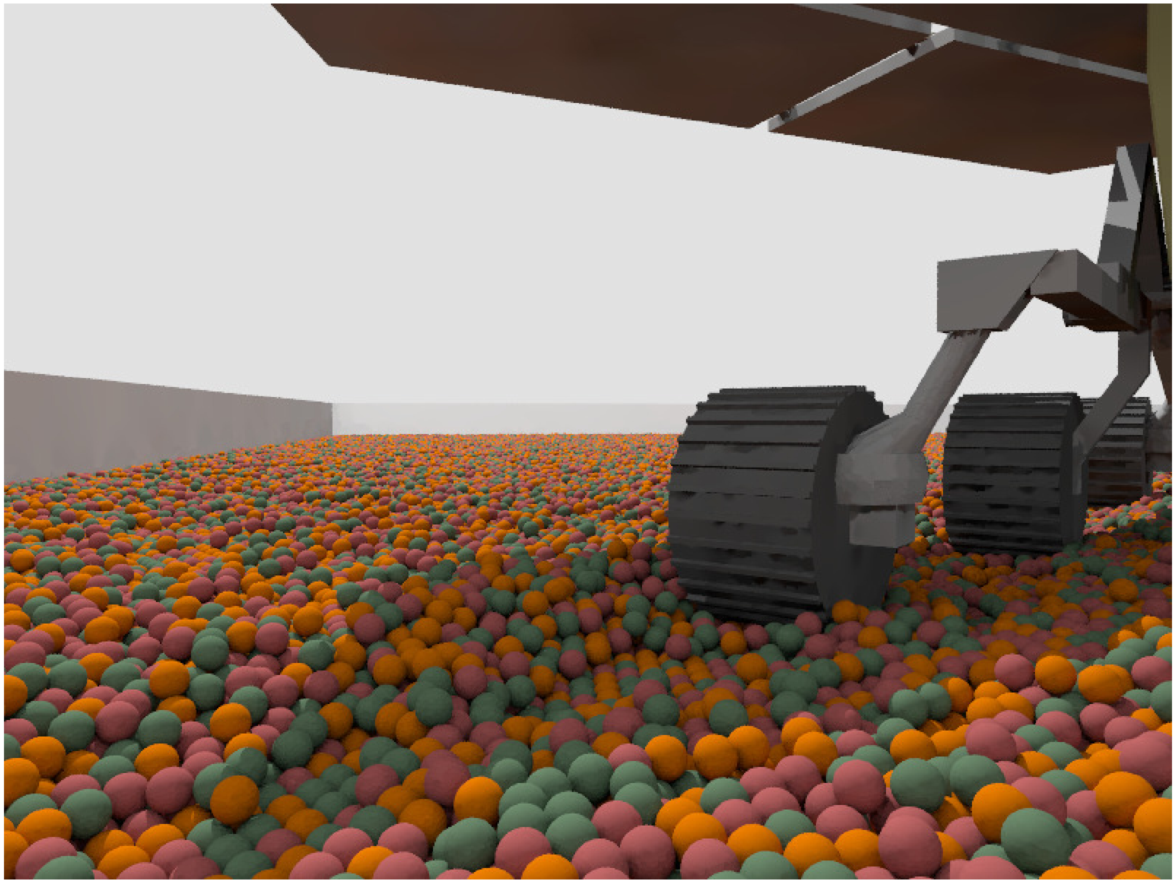
- Handling 50,000 bodies was challenging



Rover on Granular Terrain…

[0.522 million bodies]

- Can scale to thousands of cores
- Simulation uses 64 CPU cores
- Work in progress, we anticipate to get to 0.5 billion in 18 months



- Multi-Physics targeted Computational Dynamics requires
  - Advanced modeling techniques
  - Strong algorithmic (applied math) support
  - Proximity computation
  - Domain decomposition & Inter-domain data exchange
  - **Post-processing (visualization)**

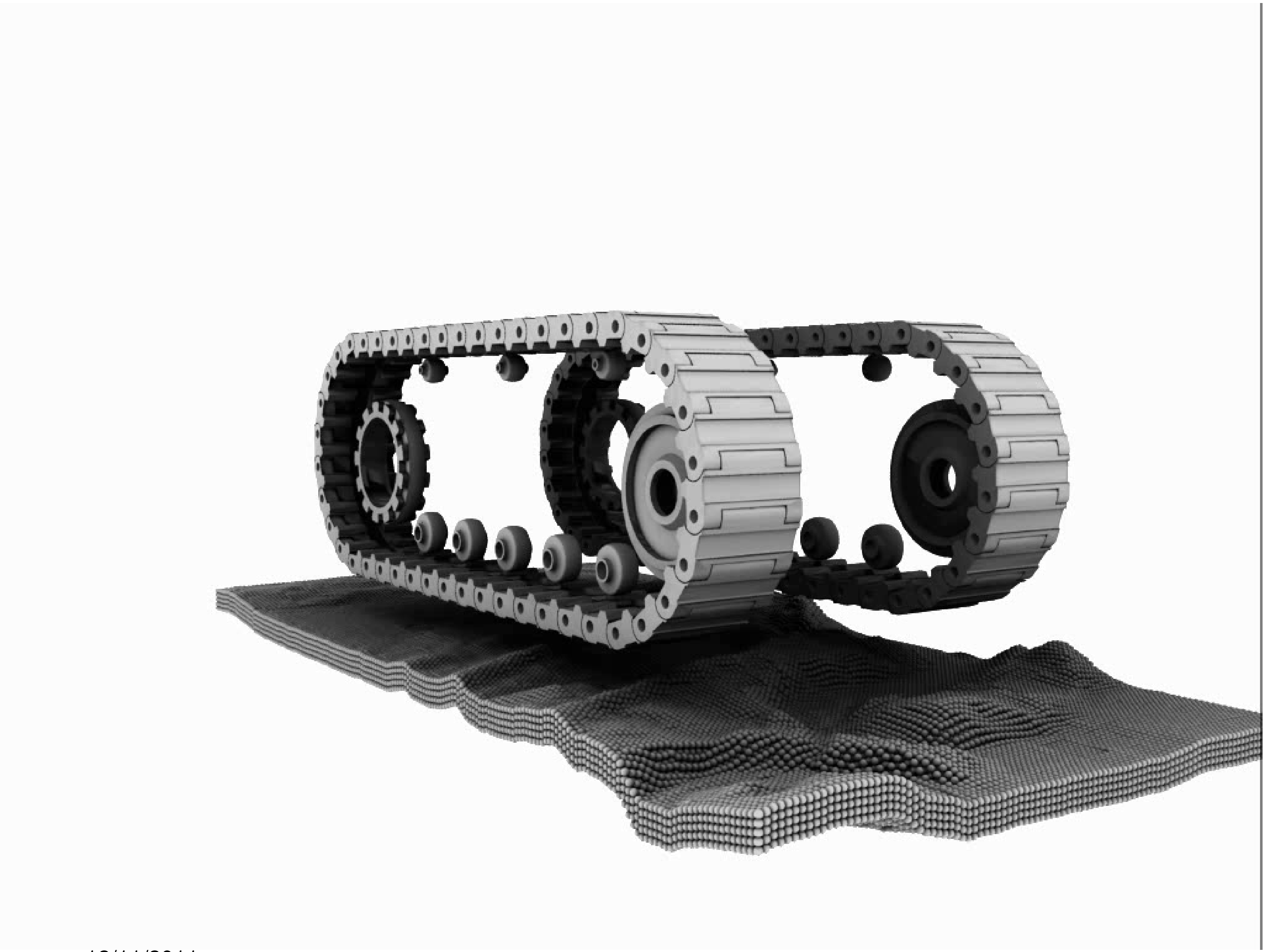
HCT:
Visualization and Post-Processing

- Rendering very complex scenes with more than one million components

- Rendering takes longer than simulating

- Pursuing a rendering pipeline that draws on multiple CPUs and GPUs

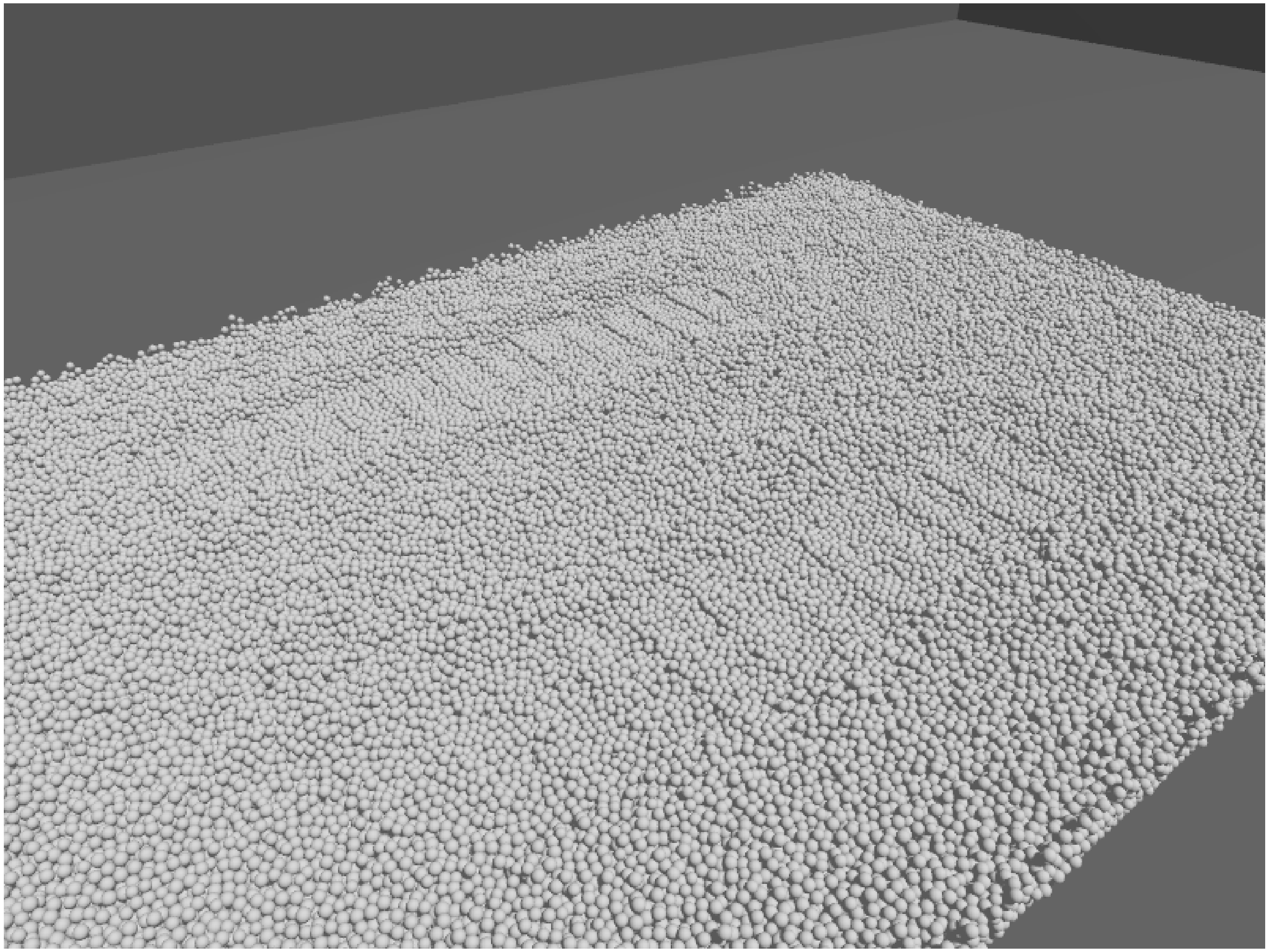
Track Simulation



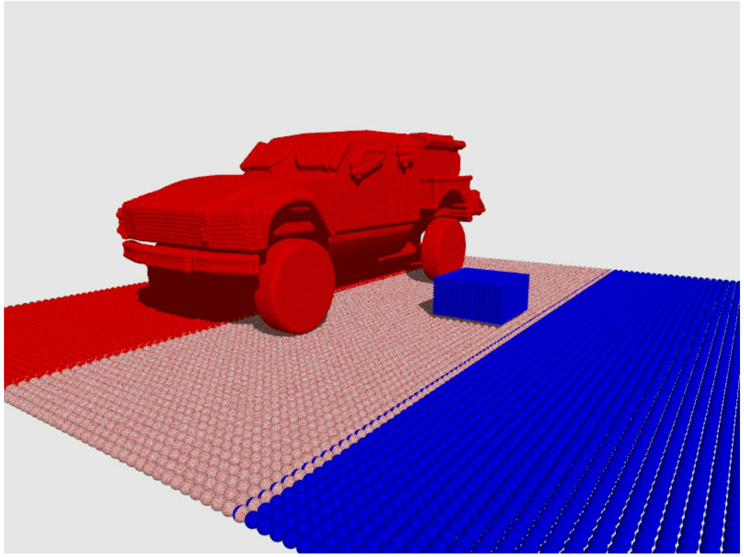
Parameters:

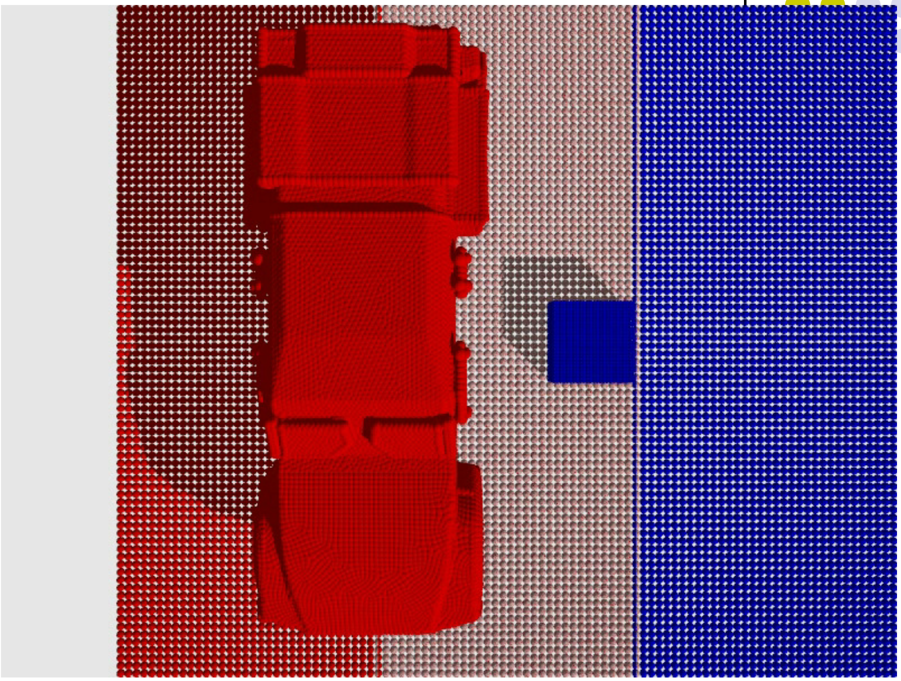
- Driving speed: 1.0 rad/sec
- Length: 12 seconds
- Time step: 0.005 sec
- Computation time: 18.5 hours
- Particle radius: .027273 m
- Terrain: 284,715 particles
- Inertia parameters of track are fake

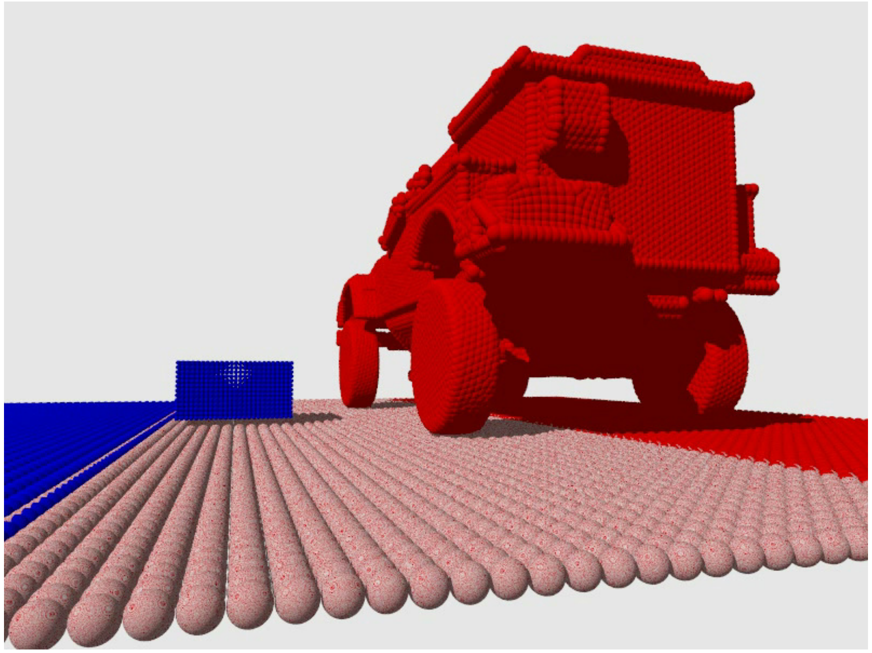
Dual Track ‘Footprint’



Simulation of MRAP Impacted by Debris



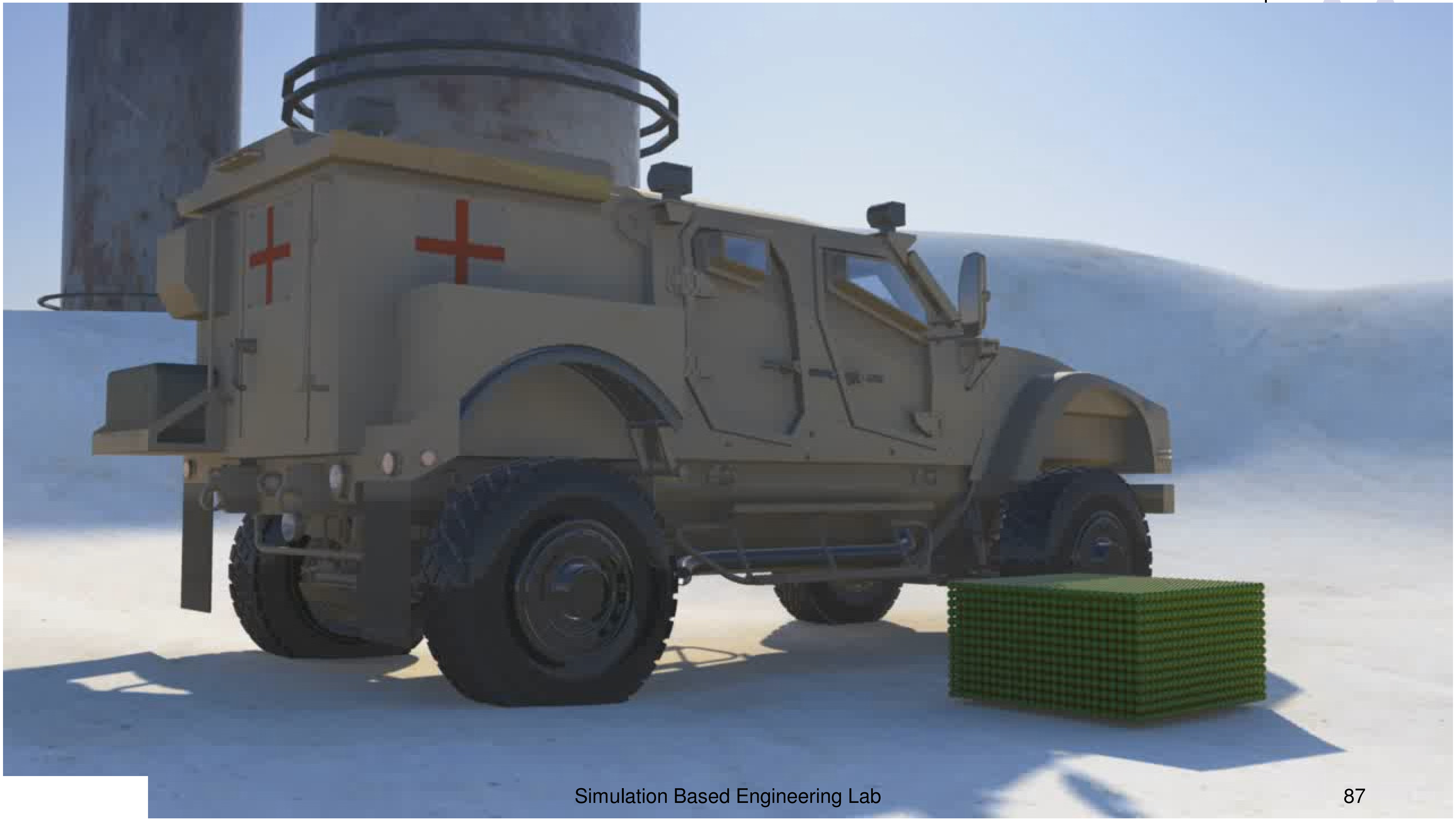




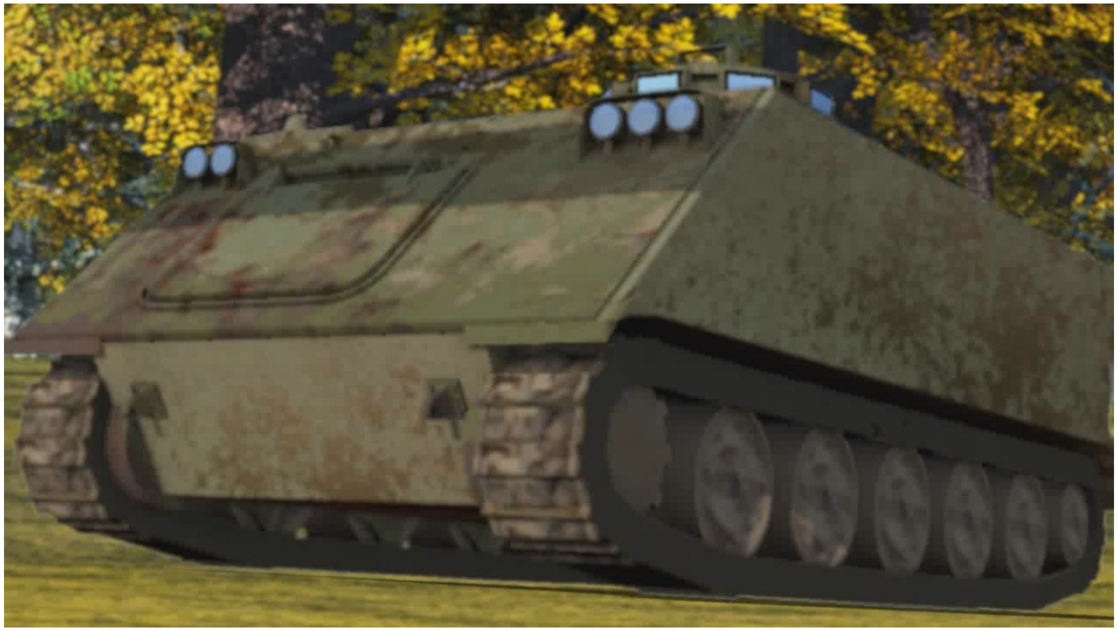
Animations show work in progress.
Run simultaneously on the CPU & GPU.

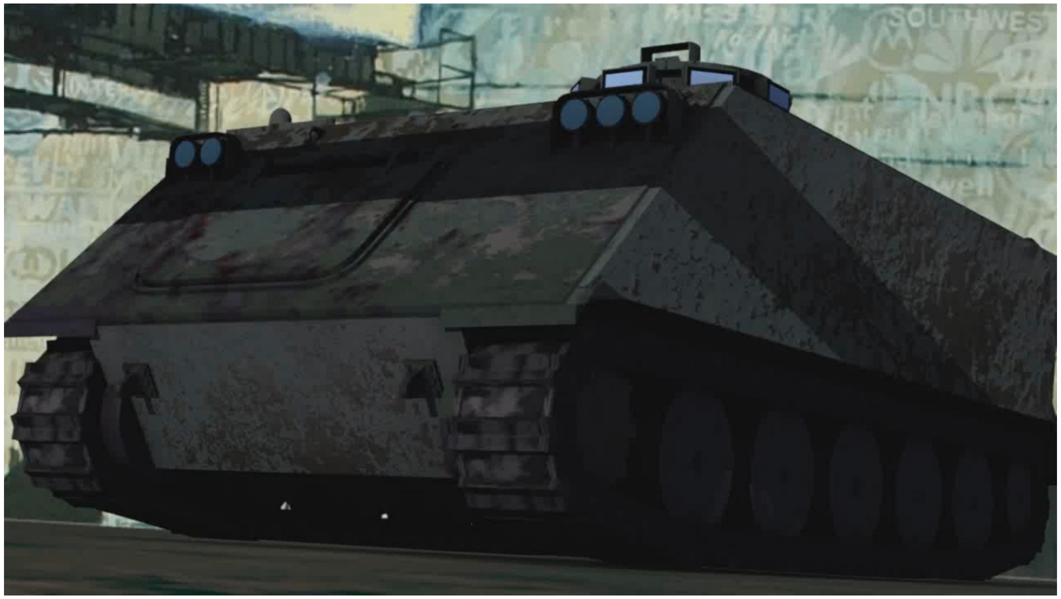
Simulation of MRAP Impacted by Debris

[work in progress]

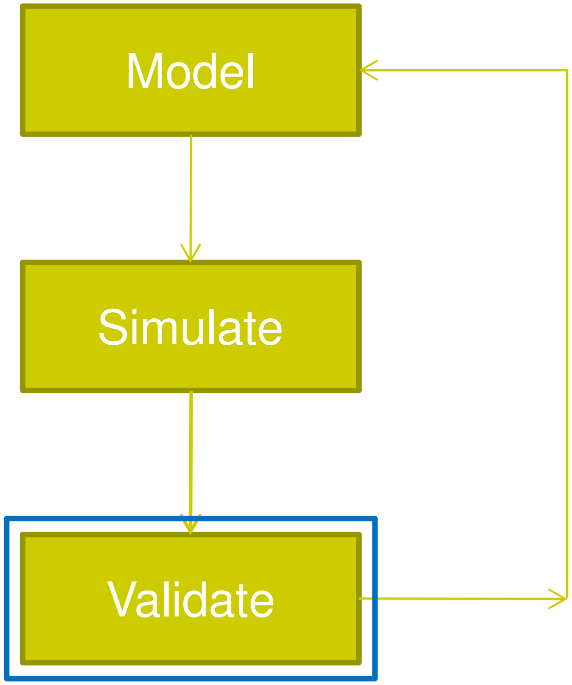


M113





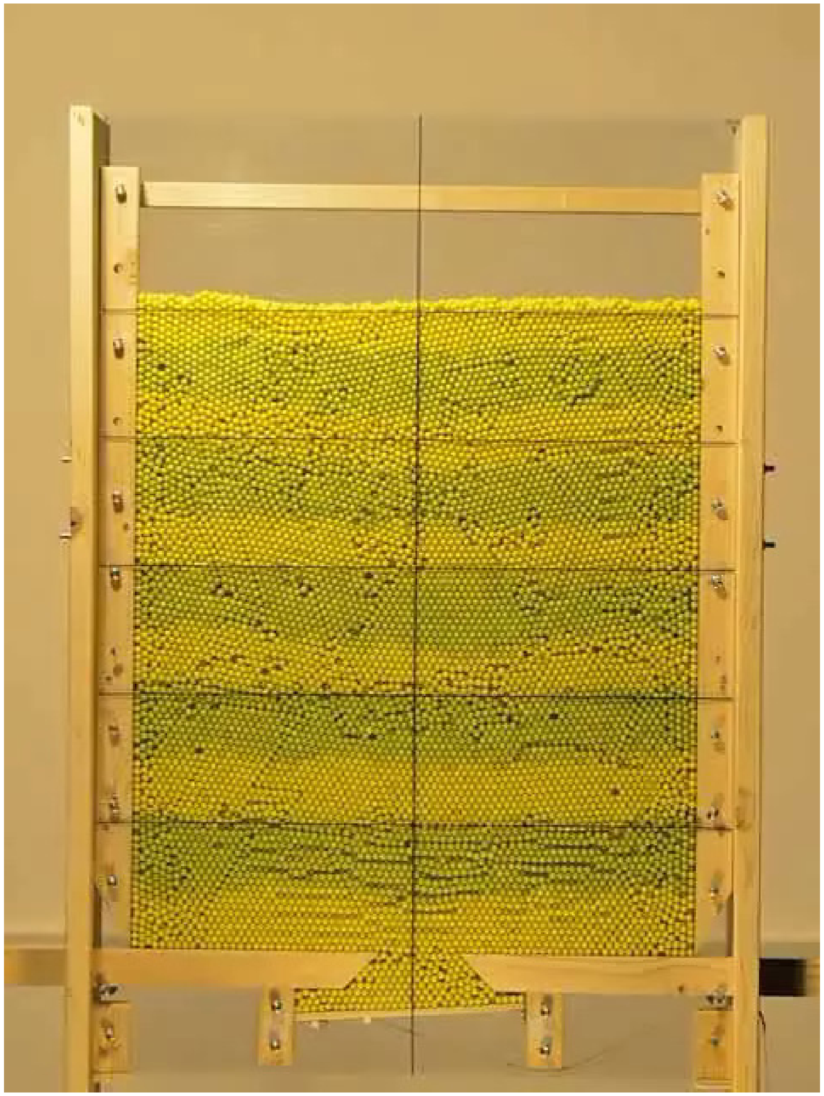
Validation.



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

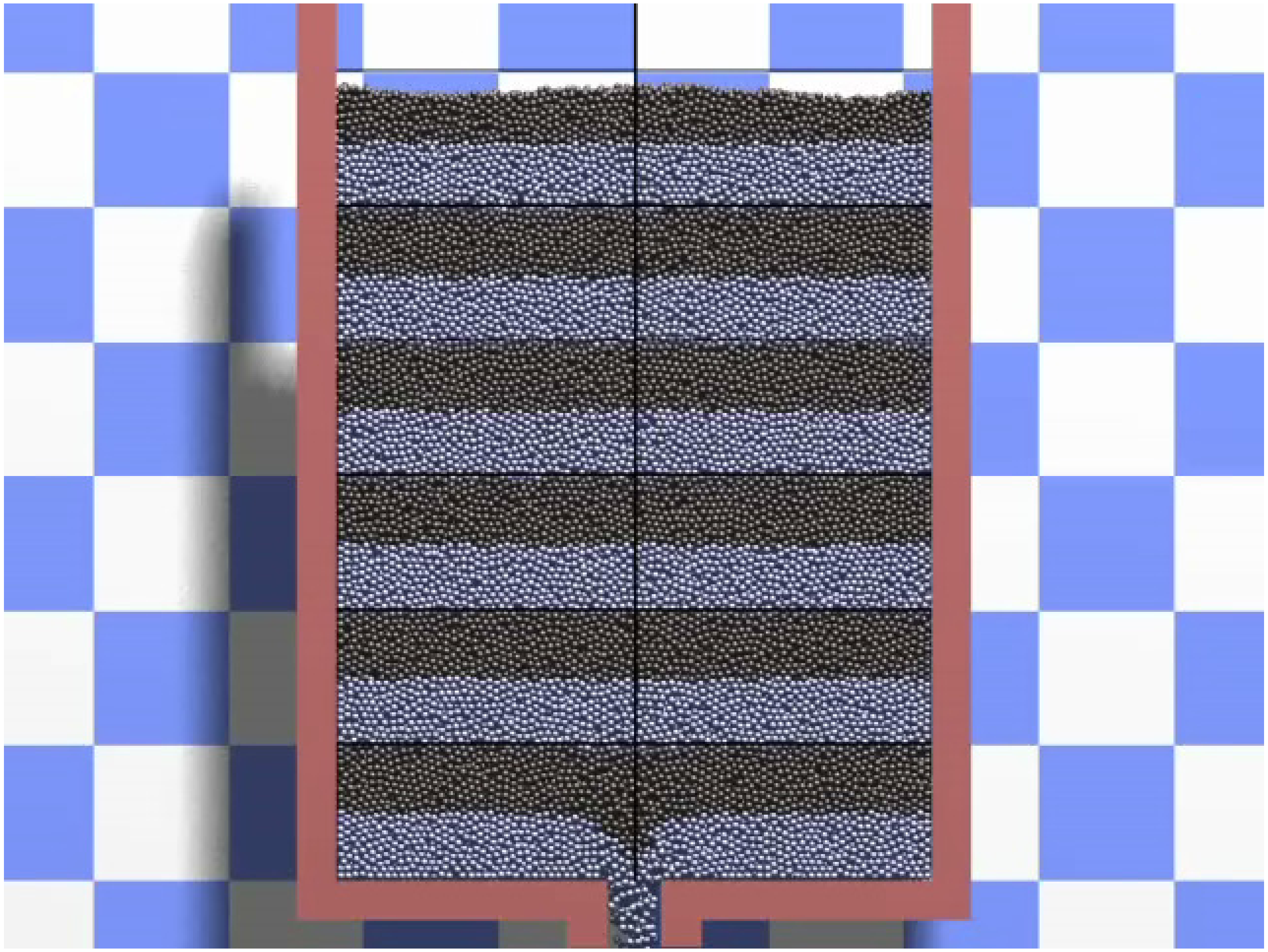
- Validation at “macroscale” – University of Parma, Italy

Flat Hopper Tests



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

Flat Hopper Tests

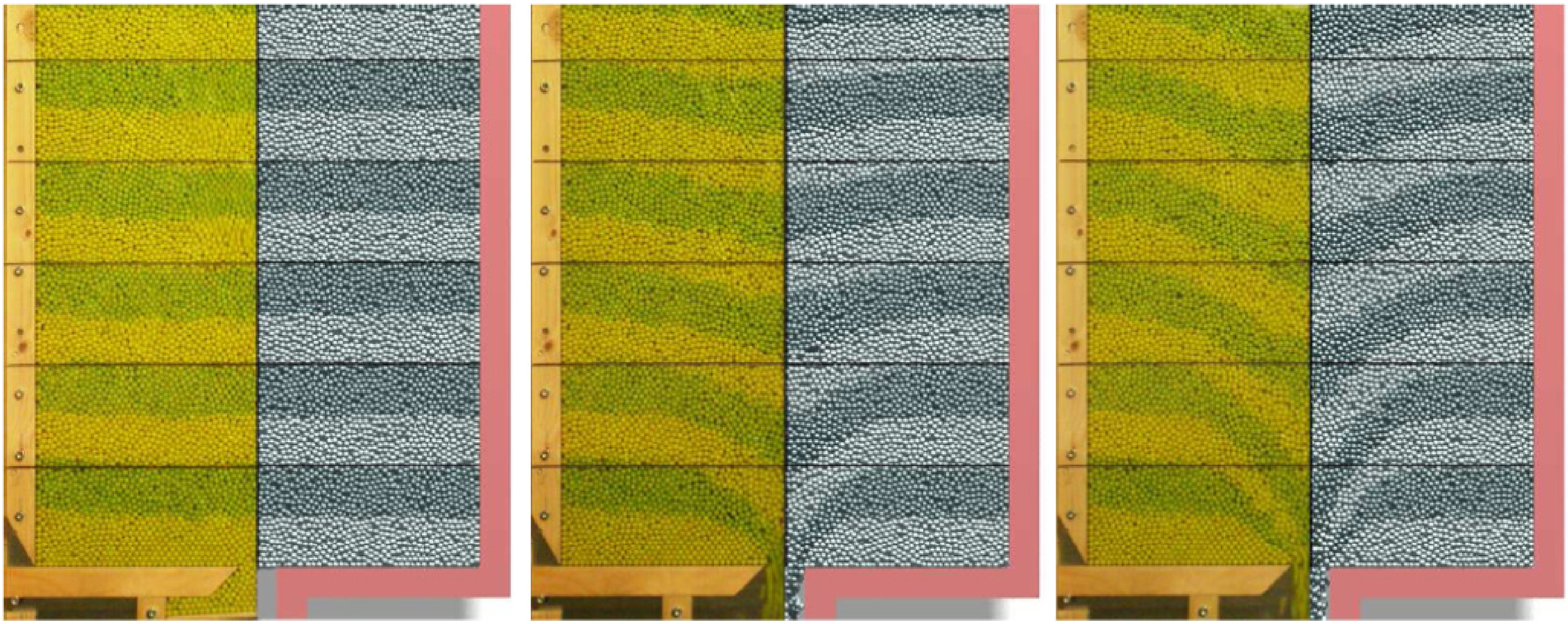


3D rendering from a simulation (4x slower than real-time)

Flat Hopper Tests

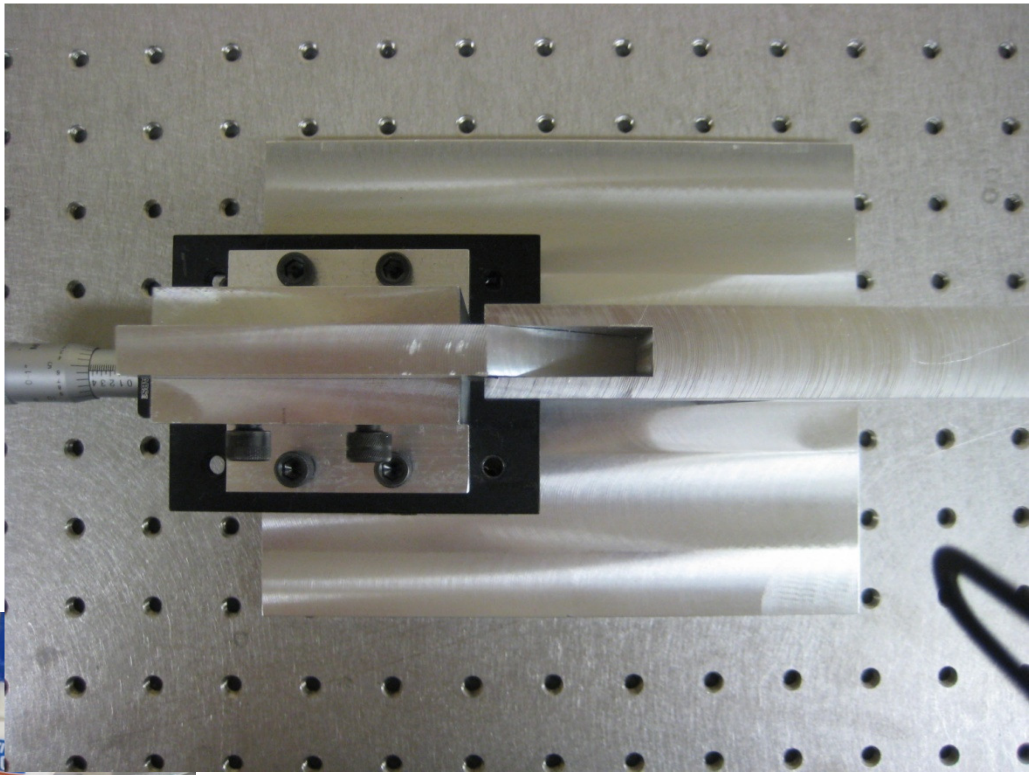
- Comparison experimental - simulated

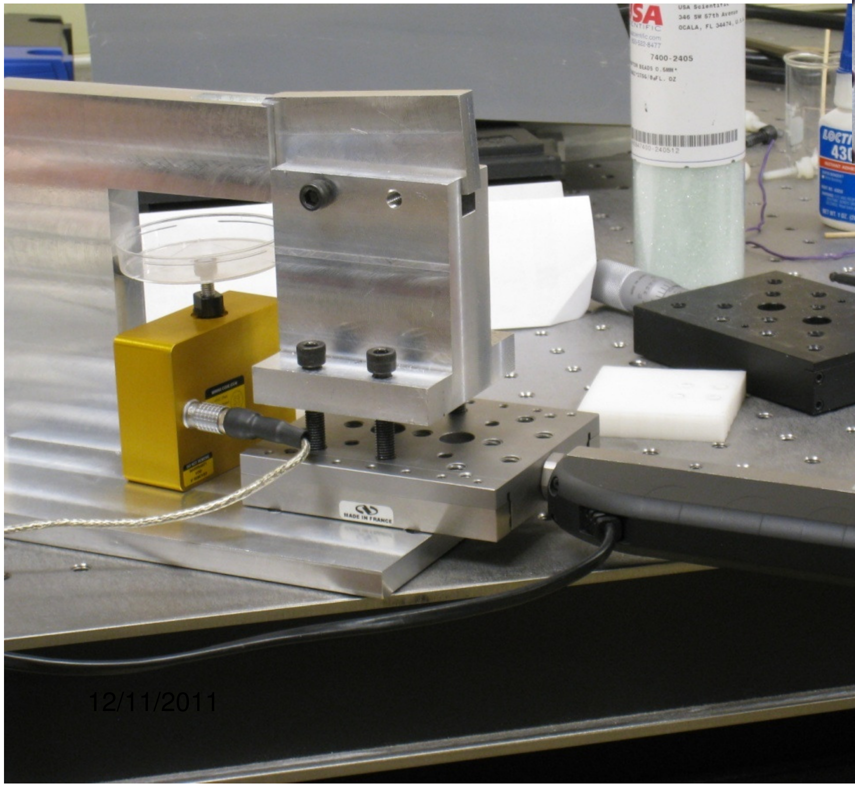
Experimental   Simulated

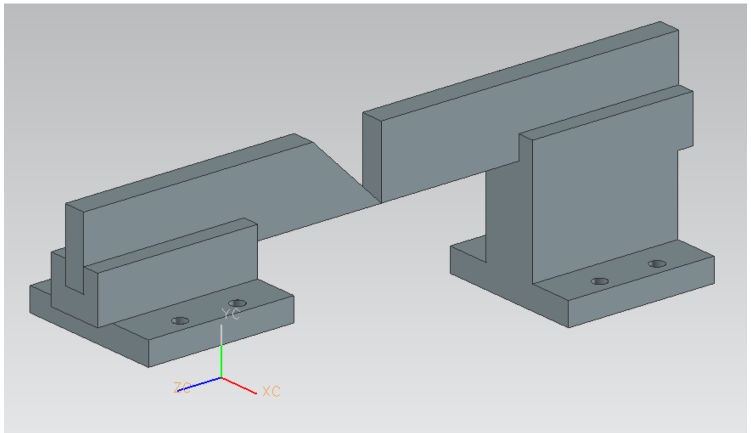


Validation at Microscale

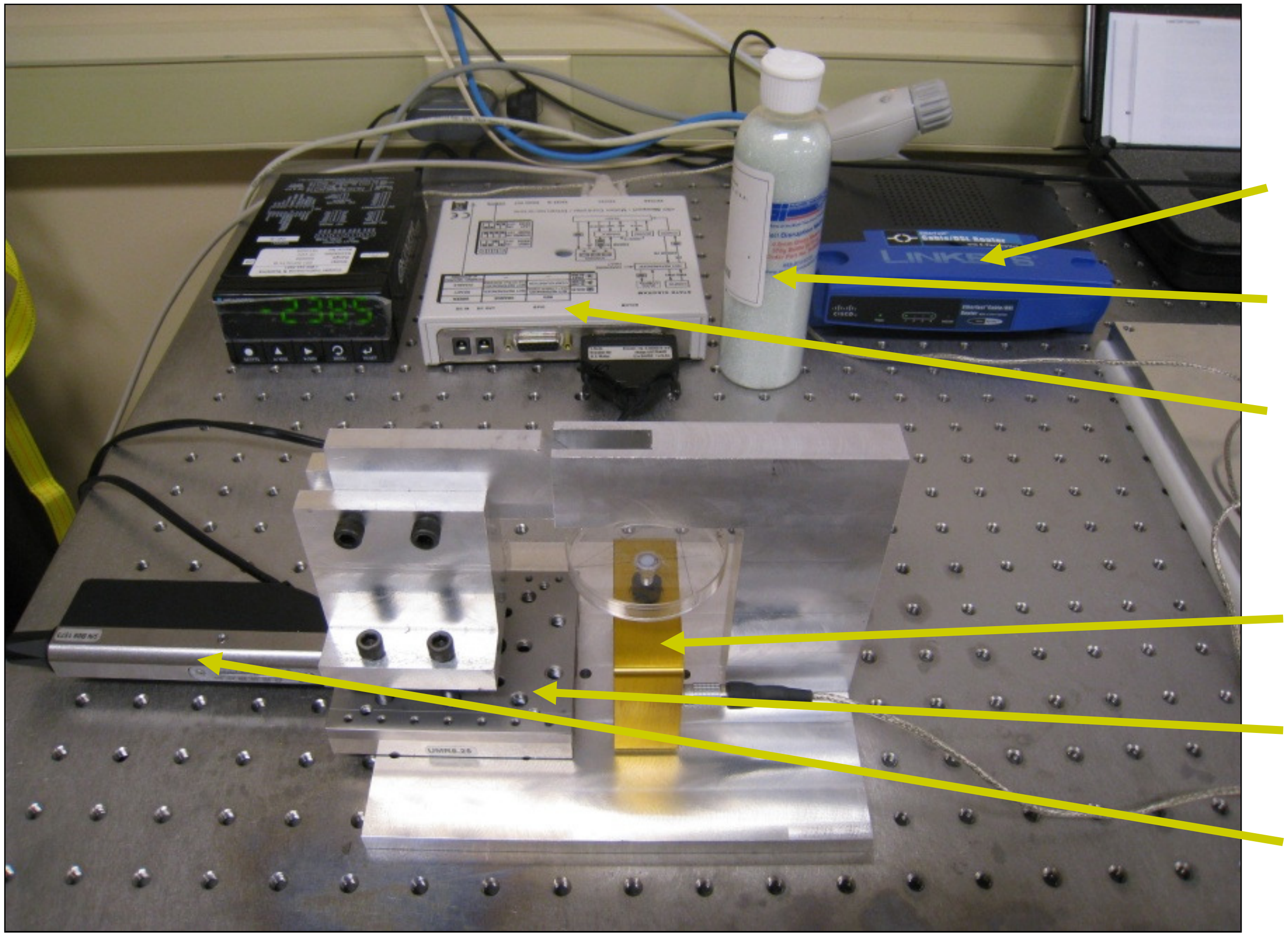
- Sand flow rate measurements
- Approx. 40K bodies
- Glass beads
- Diameter: 100-500 microns







Experimental Setup



CPU connection

Beads

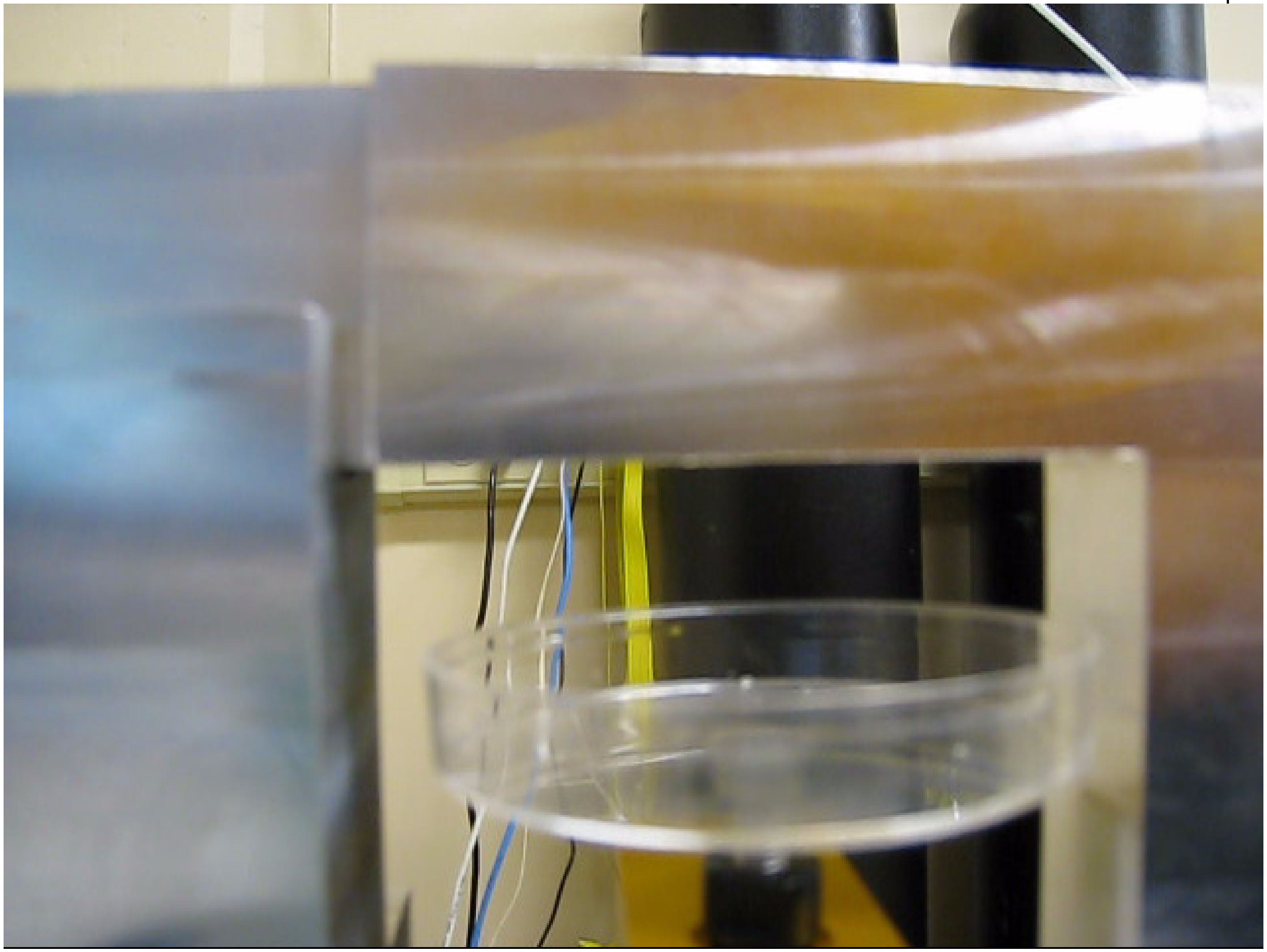
Nanopositioner controller

Load cell

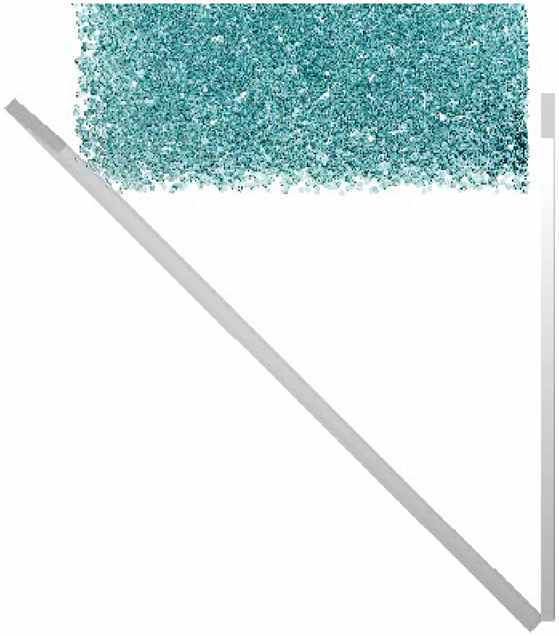
Translational stage

Nanopositioner

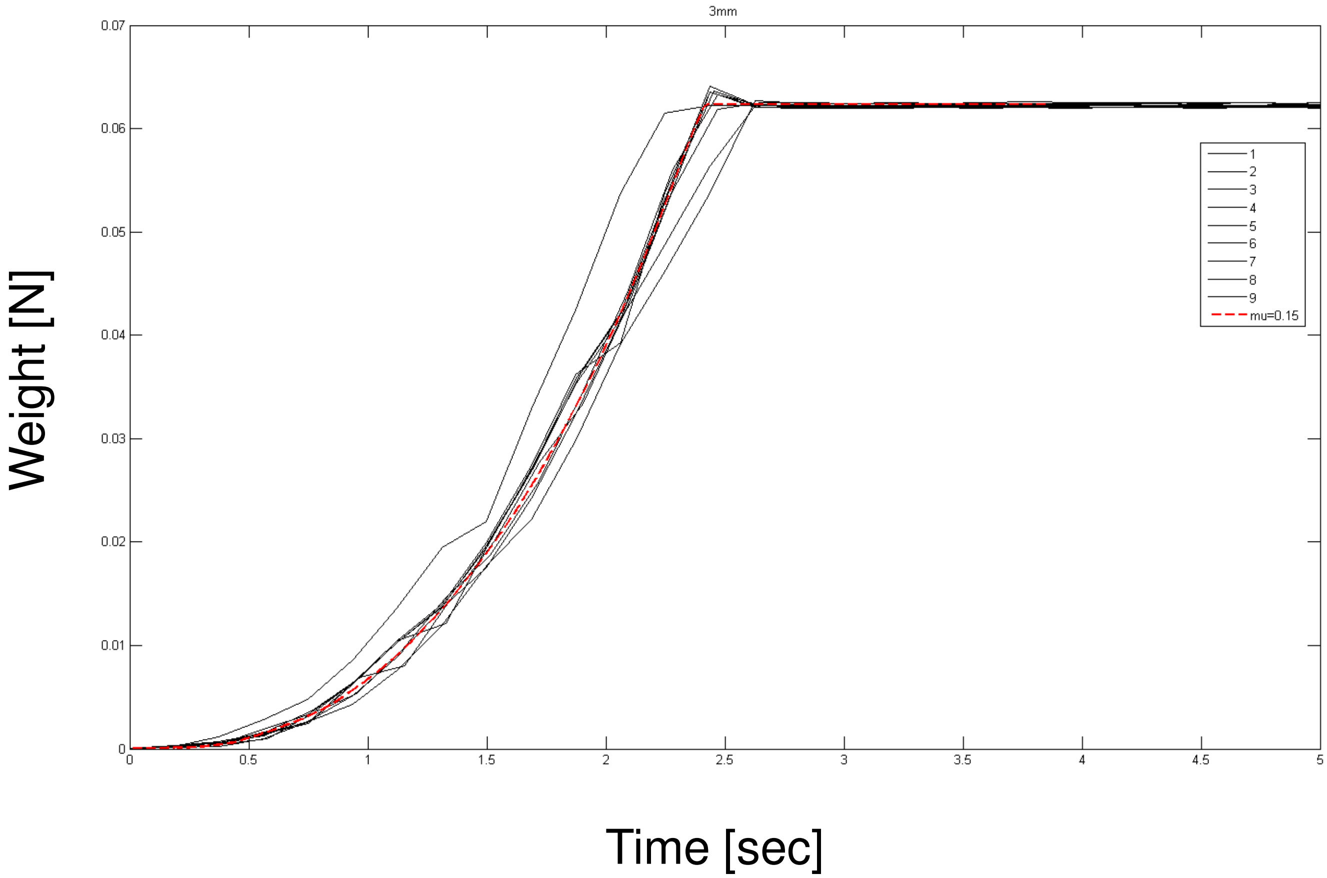
Flow Measurement, 500 micron Spheres



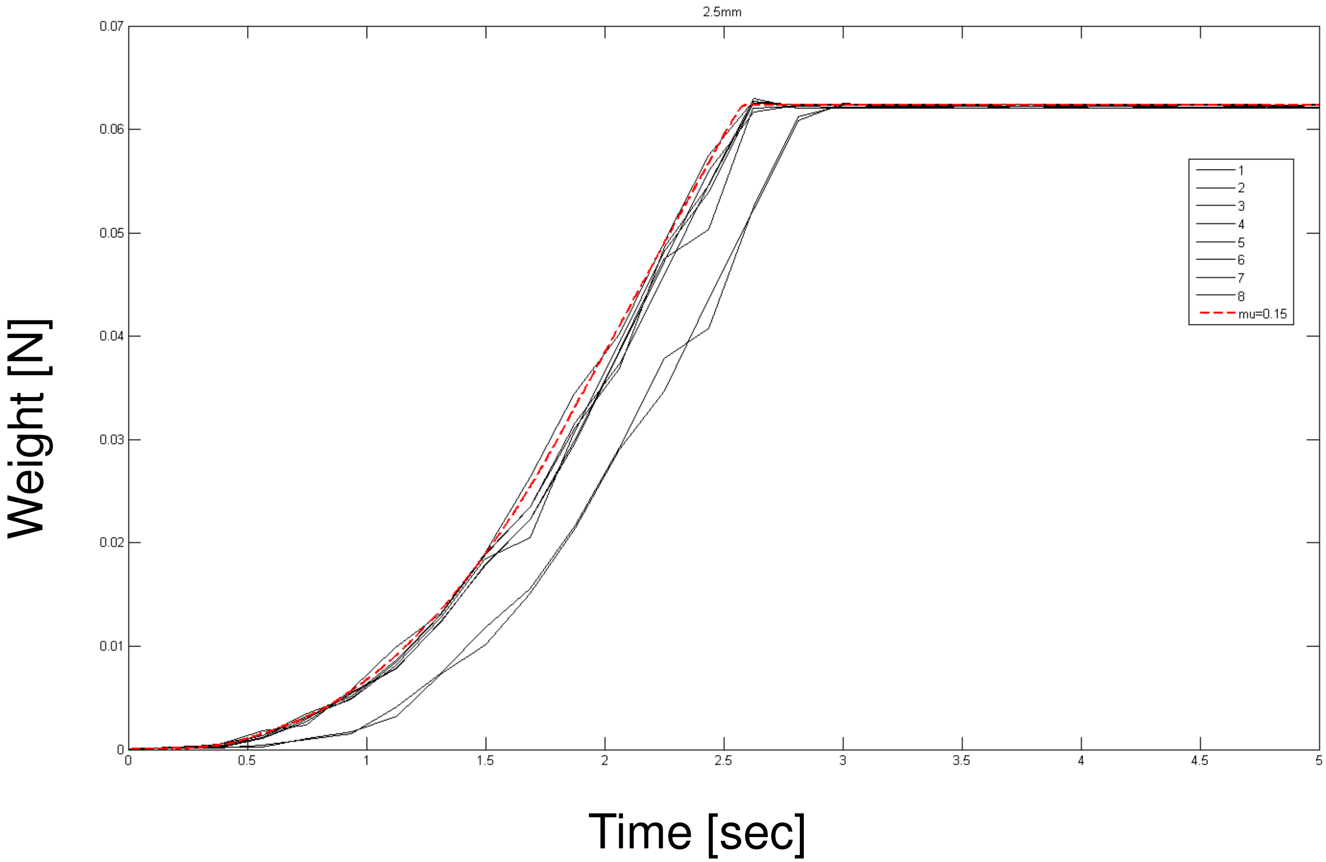
Flow Simulation, 500 micron Spheres



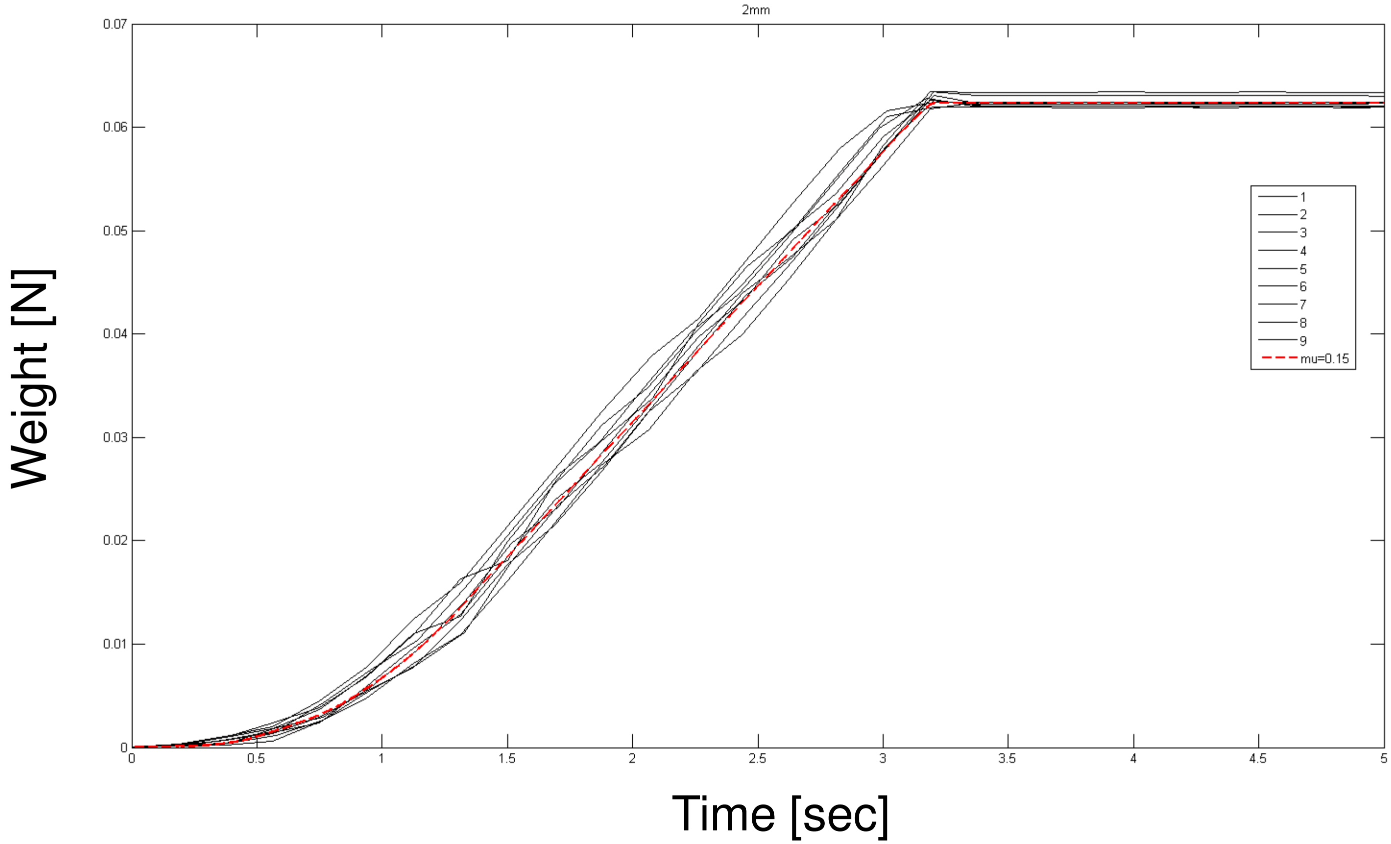
Flow Measurement Results, 3mm Gap Size



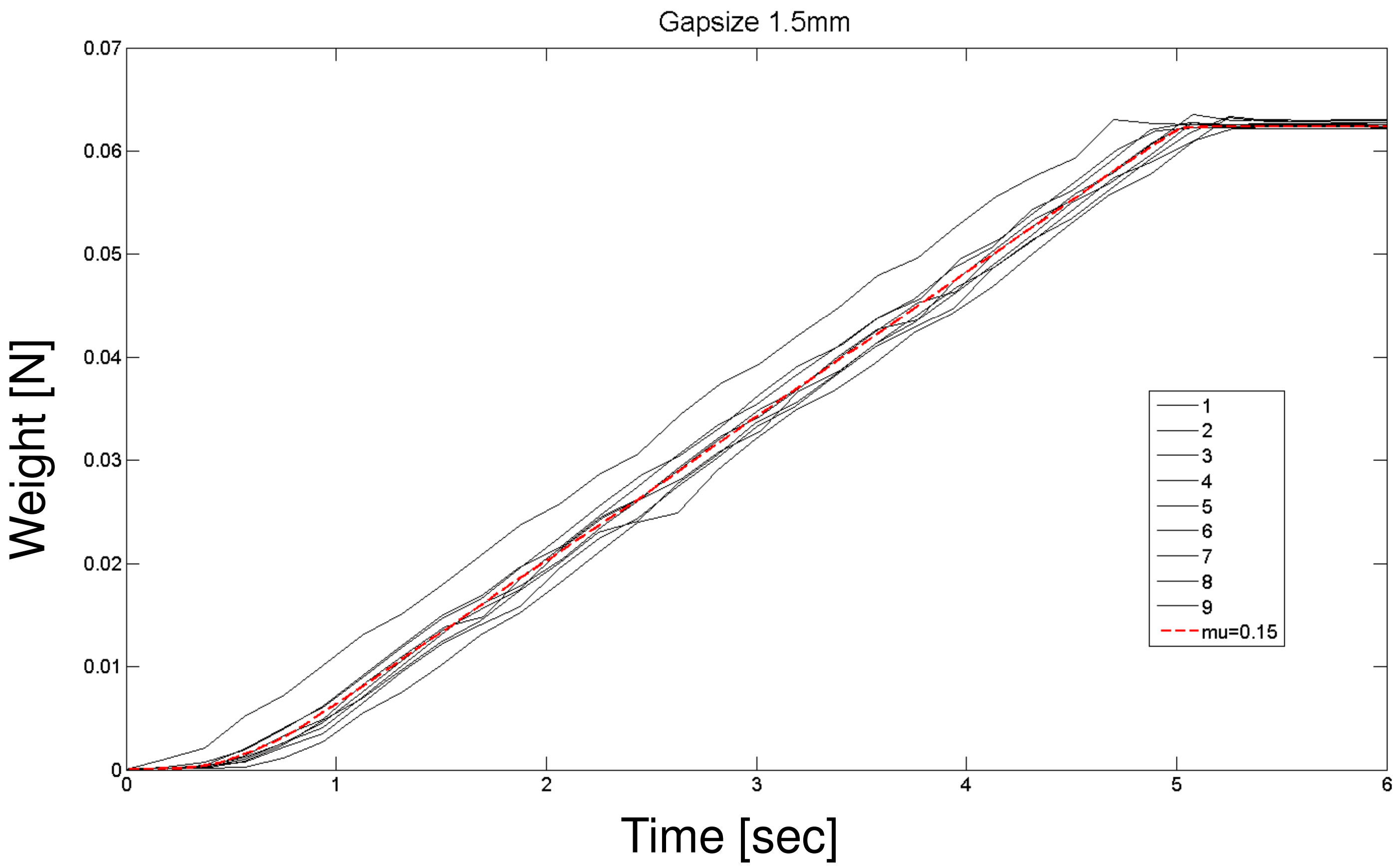
Flow Measurement Results, 2.5mm Gap Size



Flow Measurement Results, 2mm Gap Size



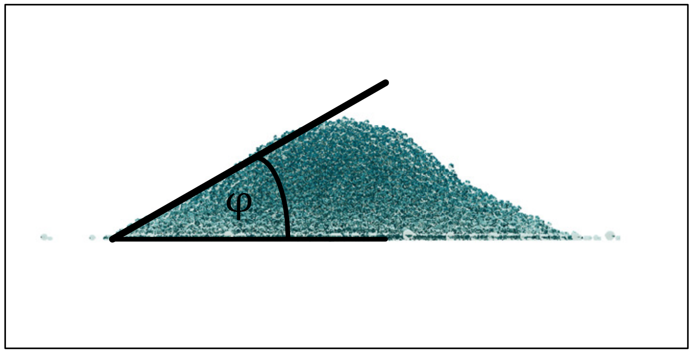
Flow Measurement Results, 1.5mm Gap Size



Validation Experiment: Repose Angle

- Experiment

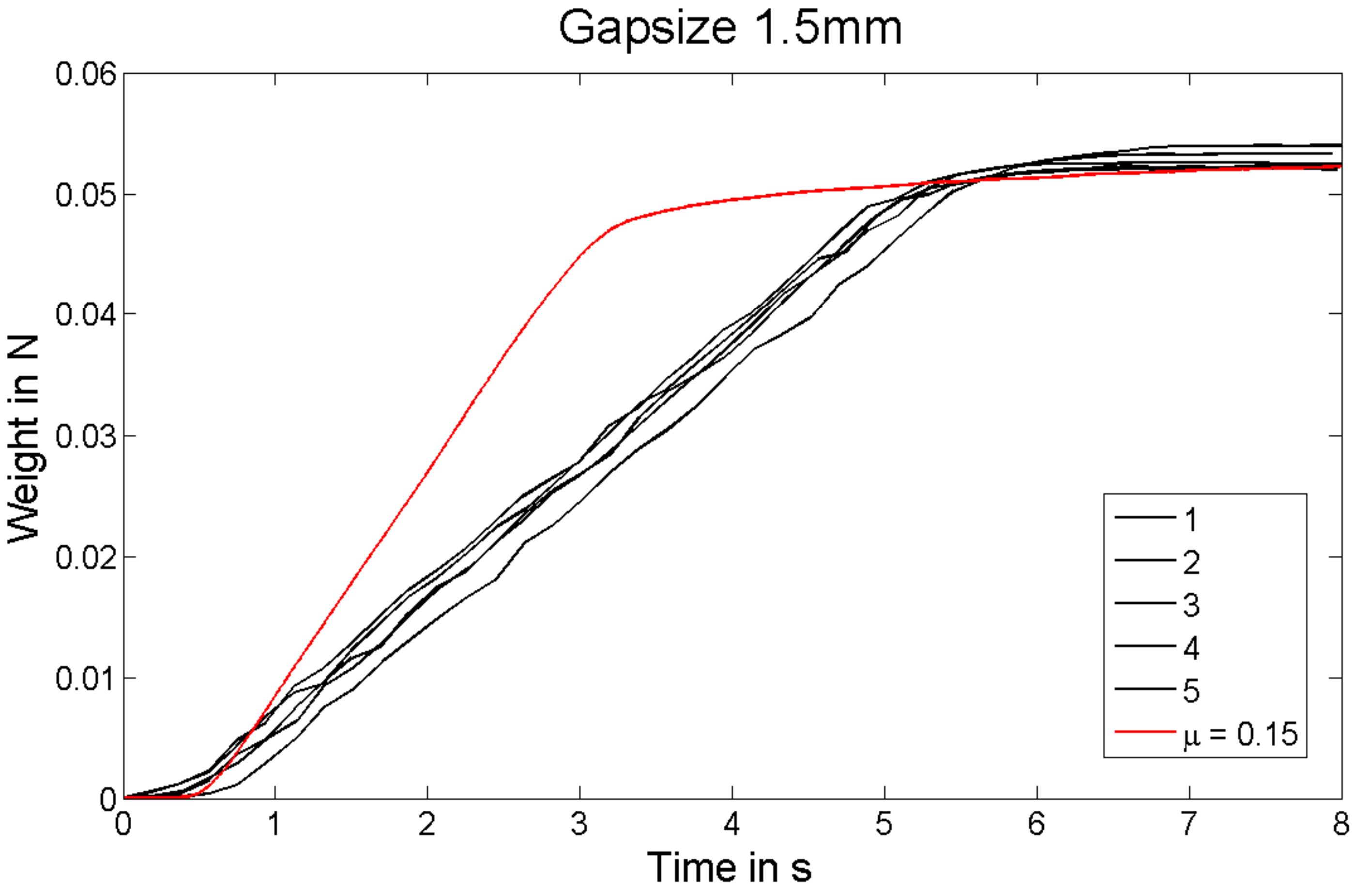
- Simulation



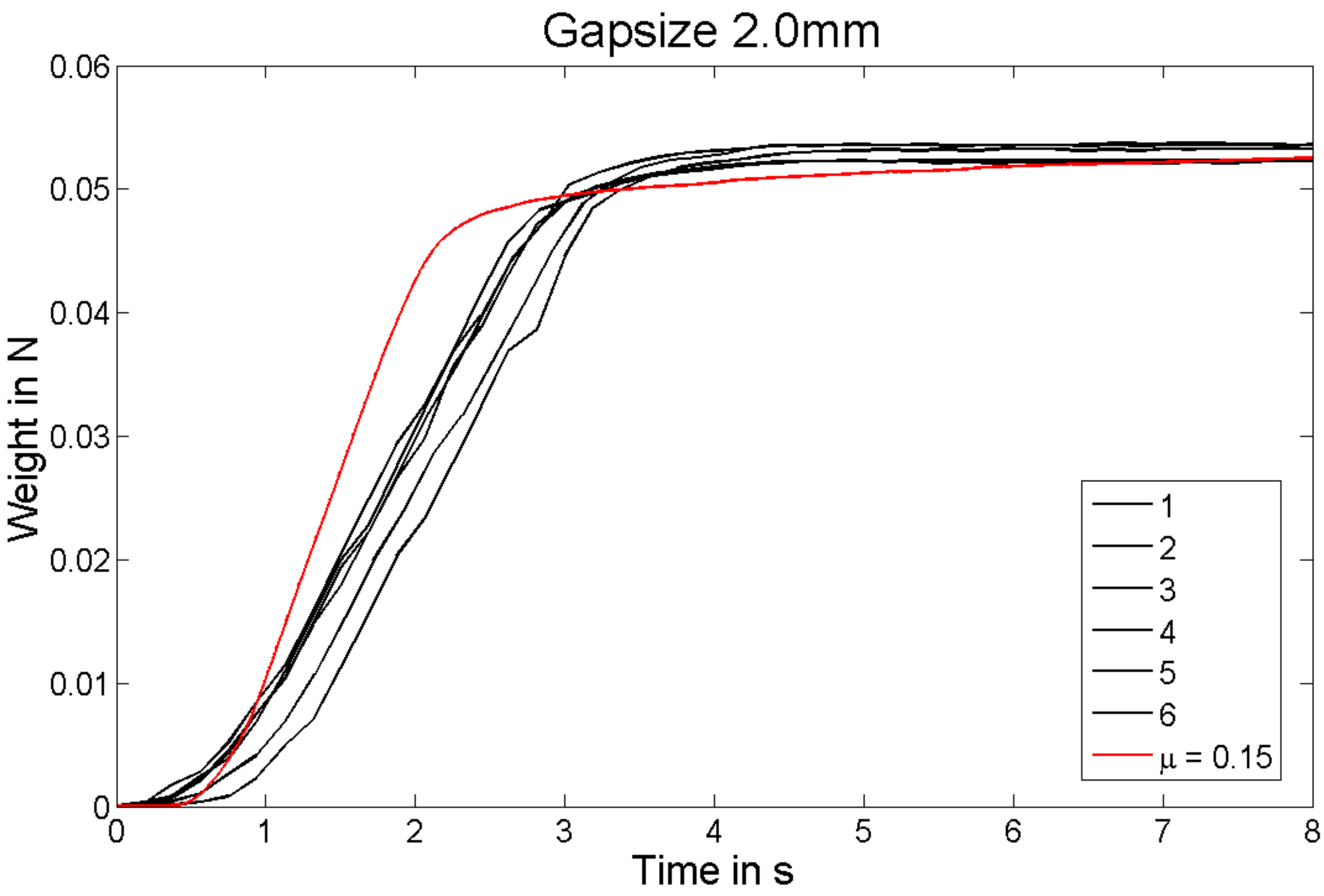
$\phi = 19.5^\circ \qquad \text{for} \quad \mu = 0.39$

Validation Experiment Flow and Stagnation

Validation, Flow and Stagnation



Validation,
Flow and Stagnation



Conclusions/Putting Things in Perspective

- Goal: Leverage hybrid CPU/GPU computing & new math to solve large engineering problems
  - Strategy: Develop an experimentally validated Heterogeneous Computing Template (HCT)

HCT: Five Major Components
[Looking Ahead]

- Novel modeling techniques
  - Rigid/Deformable bodies, fluid-solid interaction, electrostatics, cohesion
- Strong algorithmic (applied math) support
  - Sparse parallel direct preconditioner, Krylov type methods
- Proximity computation
  - Handling complex non-convex topologies + time continuous collision detection
- Domain decomposition & Inter-domain data exchange
  - Load balancing in distributed computing; focus on GPUDirect technology
- Post-processing (visualization)
  - Establish a feature-rich ready-to-use rendering pipeline that draws on High Throughput Computing

Thank You.

negrut@wisc.edu
http://sbel.wisc.edu