Simulation of Tracked Vehicles on Granular Terrain Leveraging GPU Computing

Toby Heyn

Acknowledgments

- Professor Dan Negut
- Professor Mario Trujillo
- Professor Chris Rutland
- Dr. David Lamb

- Professor Alessandro Tasora
- SBEL colleagues
  - Justin Madsen, Hammad Mazhar

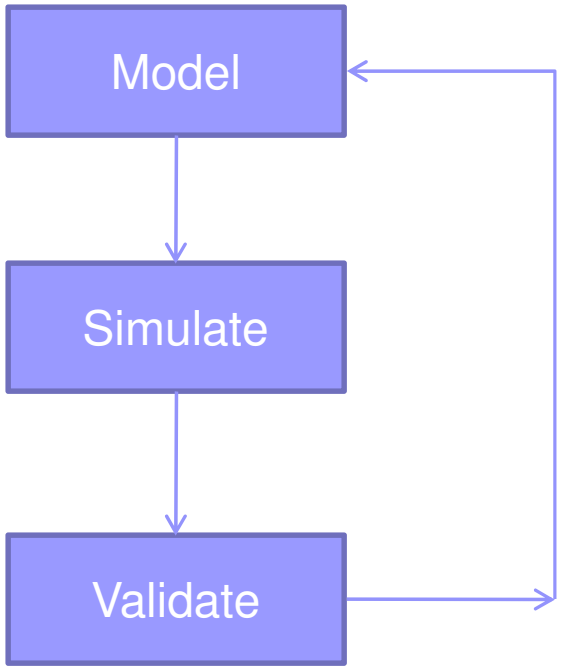
Outline

- Problem statement
- Parallel solution of dynamics problem
- Parallel collision detection
- Tracked vehicle model
- Granular terrain model
- Simulation results

Problem Statement

Simulation-Based Engineering

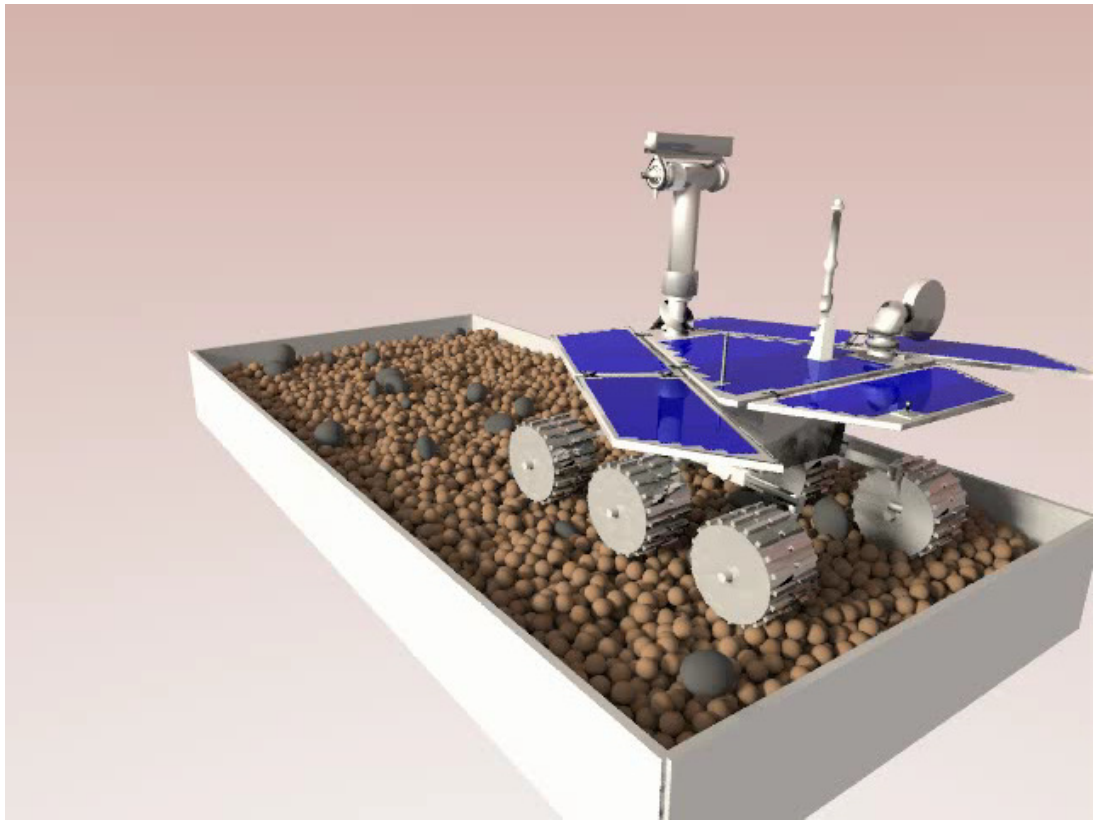
- Holy Grail in Computer Aided Engineering (CAE)
  - Do Virtual Prototyping before you build the first hardware prototype
  - Cut costs & time to market



- What’s pushing the need for high performance computing in multi-body dynamics simulation?

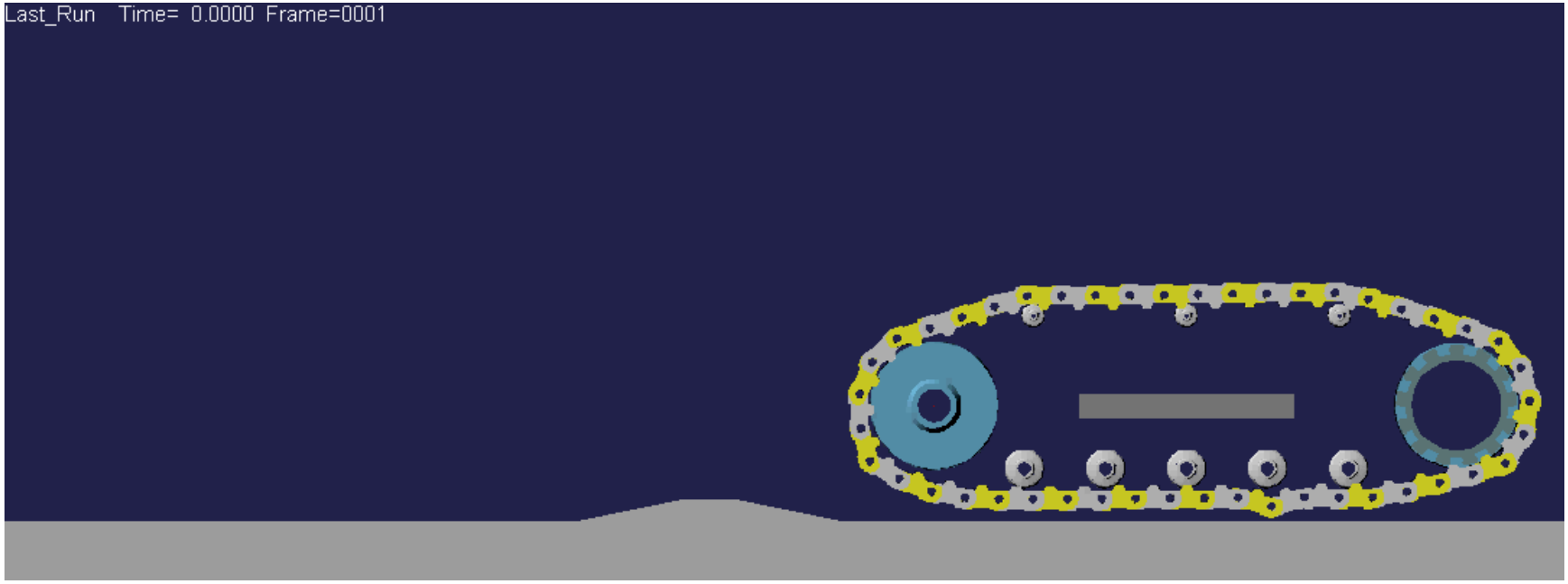
Rover on Granular Terrain

- Wheeled/tracked vehicle mobility on granular terrain

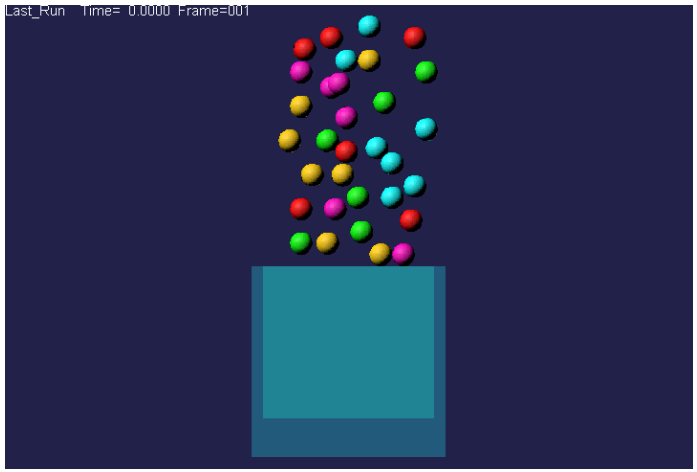


Simulation in Chrono::Engine

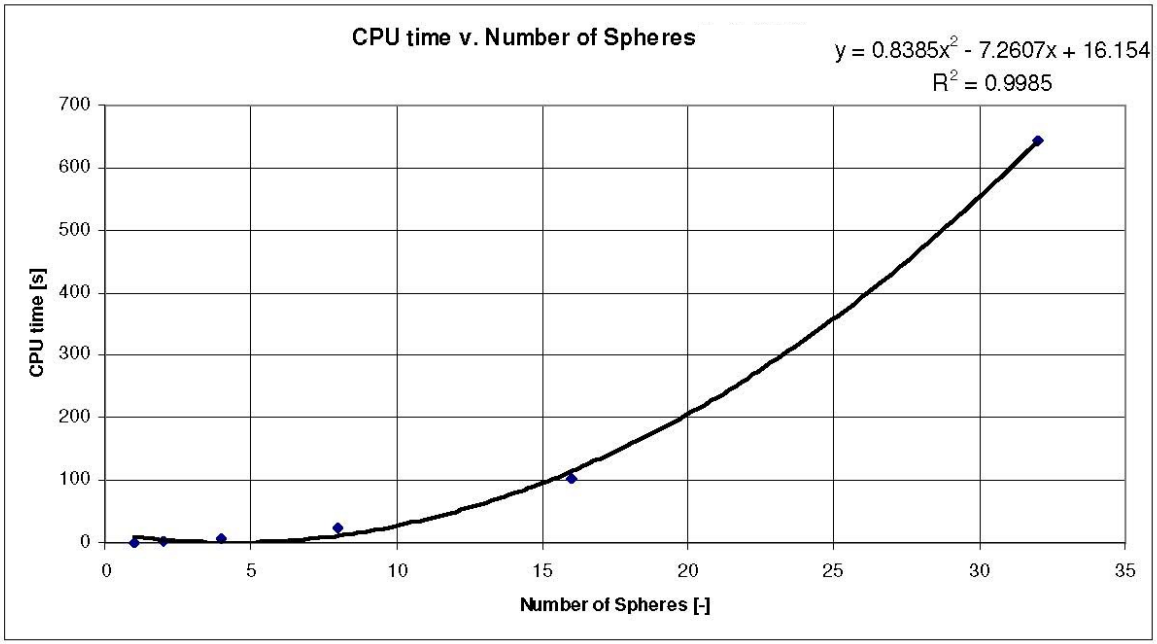
Track Simulation using Commercial Software



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 Faces of the Same Coin

- Long simulation times in off-the-shelf software traced back to the underlying formulation of the frictional contact problem
  - Draws on a “smoothing” (penalty) approach
    - Sophisticated but slow
    - General purpose tool

- Solution embraced draws on DVI (Differential Variational Inequalities)
  - A set of differential equations combined with inequality constraints
  - Less general than penalty approach

Dynamics Problem

Equations of Motion: DVI Approach

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"

Discretized EOMs

Time Step Index — Time Step — Velocity Transformation Matrix — Generalized Positions — Generalized Speeds — Reaction Force — Reaction Impulses

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

Force Balance Equations →

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

Holonomic Kinematic Constraints →

$$\underbrace{\frac{1}{h}\mathbf{g}(\mathbf{q}^{(l)}, t)} + \mathbf{g}_{\mathbf{q}}{}^{T}\mathbf{v}^{(l+1)} + \mathbf{g}_t = \mathbf{0}$$

Stabilization Terms

Contact Complementarity Conditions →

$$0 \le \overbrace{\frac{1}{h}\Phi^i(\mathbf{q}^{(l)}, t)} + \mathbf{D}_n^{T,i}\mathbf{v}^{(l+1)} \quad \perp \quad \gamma_n^i \ge 0 \qquad i = 1, 2, \ldots, \underbrace{N_c}_{\text{Total Number of Contacts}}$$

Coulomb Friction Model →

$$(\gamma_u^i, \gamma_w^i) = \underset{\mu^i \gamma_n^i \ge \sqrt{(\gamma_u^i)^2 + (\gamma_w^i)^2}}{\arg\min} \; (\gamma_u^i \mathbf{v}^T \mathbf{D}_u^i + \gamma_w^i \mathbf{v}^T \mathbf{D}_w^i)$$

(Stewart, 1998)

Outer Loop (Time-Stepping)

1. Set $t = 0$, step counter $l = 0$, provide initial values for $\mathbf{q}^{(l)}$ and $\mathbf{v}^{(l)}$.
2. Perform collision detection between bodies. For each contact i, compute $\mathbf{D}_{i,n}$, $\mathbf{D}_{i,u}$, $\mathbf{D}_{i,w}$.
3. For each body, compute forces $\mathbf{f}(t^{(l)}, \mathbf{q}^{(l)}, \mathbf{v}^{(l)})$.
4. Use **CCP Algorithm** to solve the cone complementarity problem and obtain unknown impulse γ and velocity $\mathbf{v}^{(l+1)}$.
5. Update positions using $\mathbf{q}^{(l+1)} = \mathbf{q}^{(l)} + h\mathbf{L}(\mathbf{q}^{(l)})\mathbf{v}^{(l+1)}$.
6. Increment $t := t + h$, $l := l + 1$, and repeat from step 2 until $t > t_{\text{end}}$

Preprocessing

"Inner Loop"

Post processing

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:

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

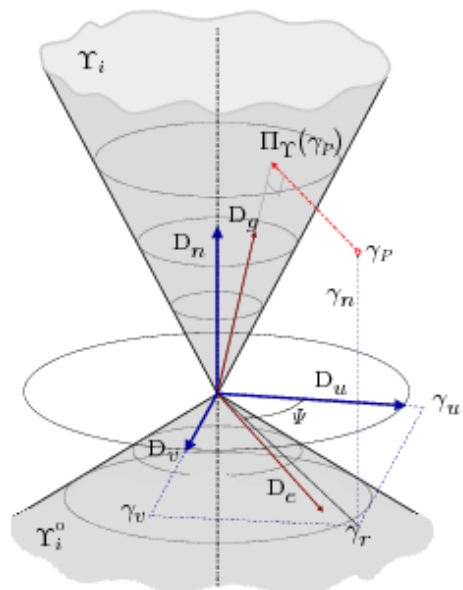
$$\Delta \gamma_i^{r+1} = \gamma_i^{r+1} - \gamma_i^r \ ;$$

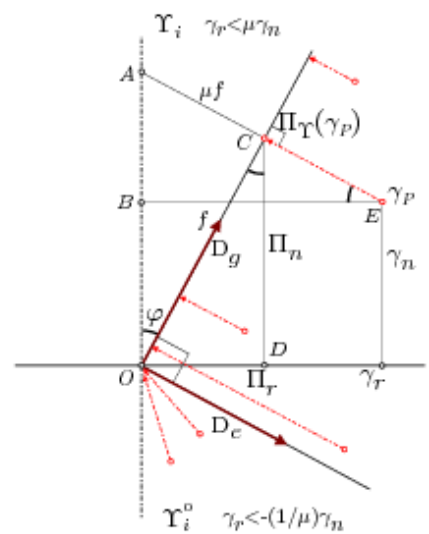
$$\Delta \mathbf{v}_i = \mathbf{M}^{-1} \mathbf{D}_i \Delta \gamma_i^{r+1}.$$

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 $r > r_{max}$





Parallelism, Opportunities

[at each integration time step]

1. Parallel Collision Detection
2. (Body parallel) Force kernel
3. (Contact parallel) Contact preprocessing kernel

Inner Loop

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

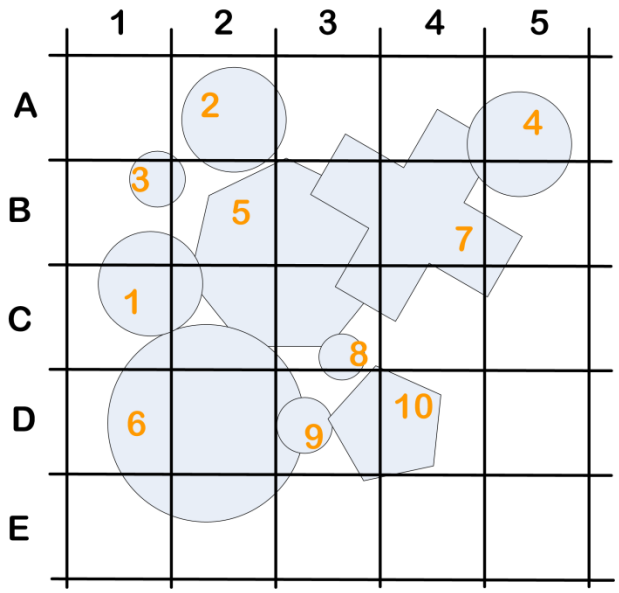
Collision Detection

Scalable 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
  - Similar in spirit to LeGrand algorithm (GPU Gems 3)
    - Yet capable of handling heterogeneous geometries (spheres, triangles, ellipsoids)

CD: Binning

- Example: 2D collision detection, bins are squares



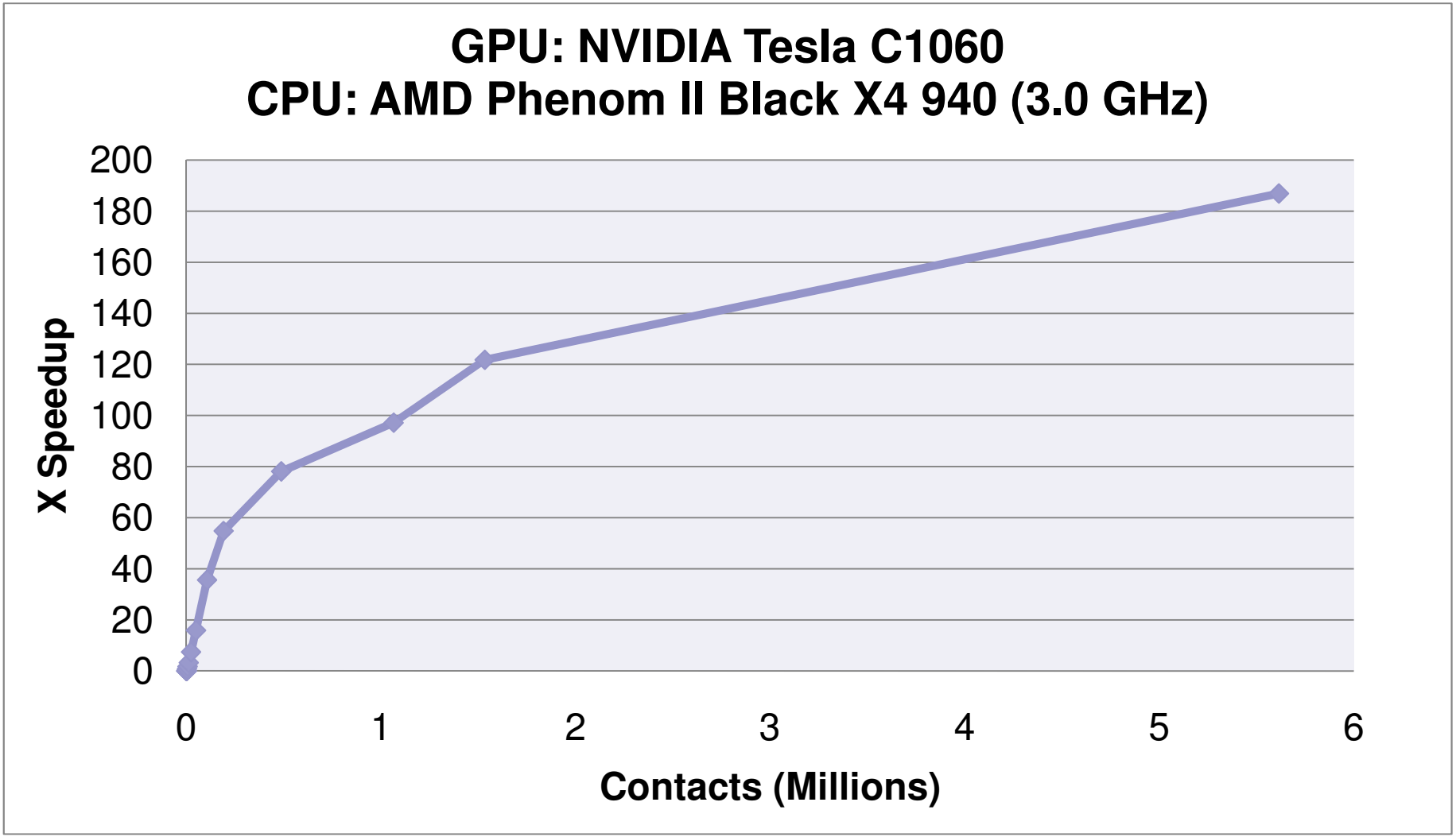
- Body 4 touches bins A4, A5, B4, B5
- Body 7 touches bins A3, A4, A5, B3, B4, B5, C3, C4, C5
- In proposed algorithm, bodies 4 and 7 will be checked for collision by three threads (associated with bin A4, A5, B4)

Parallel Binning: Summary of Stages

N – number of bodies
N_b – number of bins
M – total number of bins touched by the bodies present in the problem

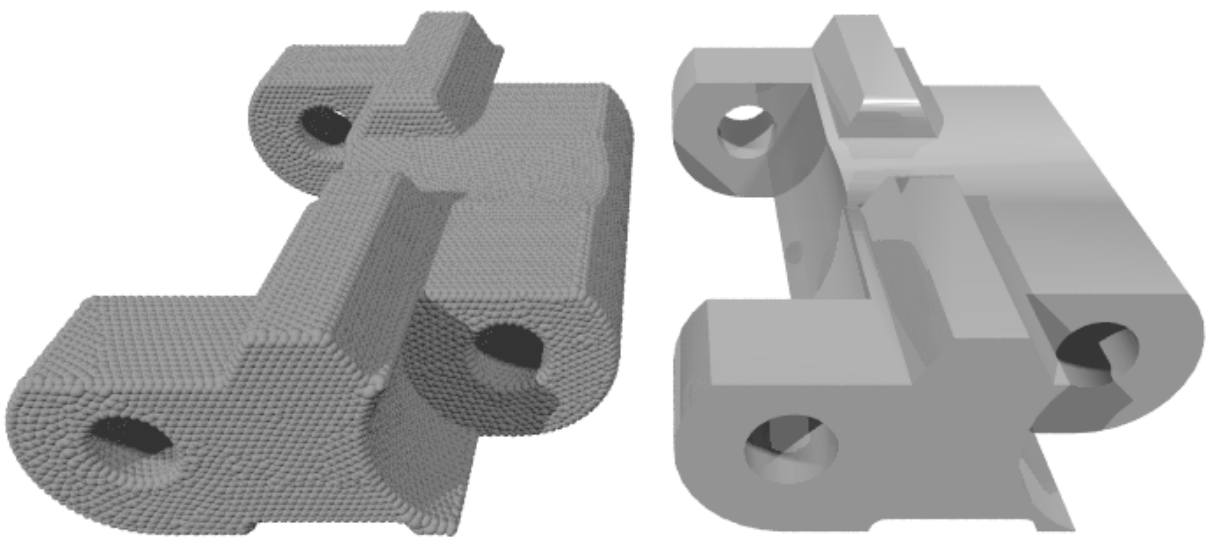
- Stage 1: Find number of bins touched by each body, populate **T** (body parallel)
- Stage 2: Parallel exclusive prefix scan of **T** (length of **T**: N)
- Stage 3: Determine body-to-bin association, populate **B** (body parallel)
- Stage 4: Parallel sort of **B** (length of **B**: M)
- Stage 5: Find bin starting index, populate **C** (bin parallel)
- Stage 6: Parallel sort of **C** for pruning (length of **C**: N_b)
- Stage 7: Determine # of collisions in each bin, store in **D** (bin parallel)
- Stage 8: Parallel prefix scan of **D** (length of **D**: N_b)
- Stage 9: Run collision detection and populate **E** with required collision info (bin parallel)

Speedup - GPU vs. CPU (Bullet library)



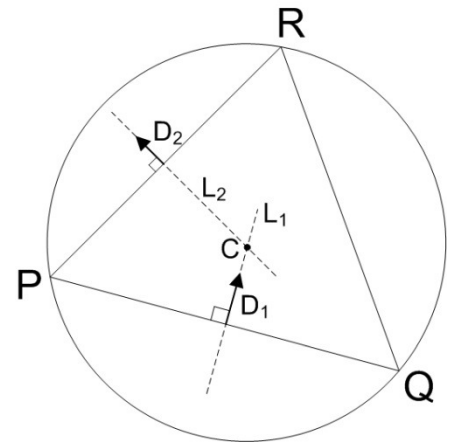
Spherical Decomposition

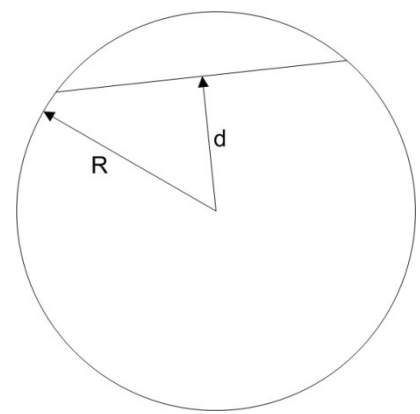
- Represent complex geometry as a union of spheres
- Allows use of fast parallel collision detection



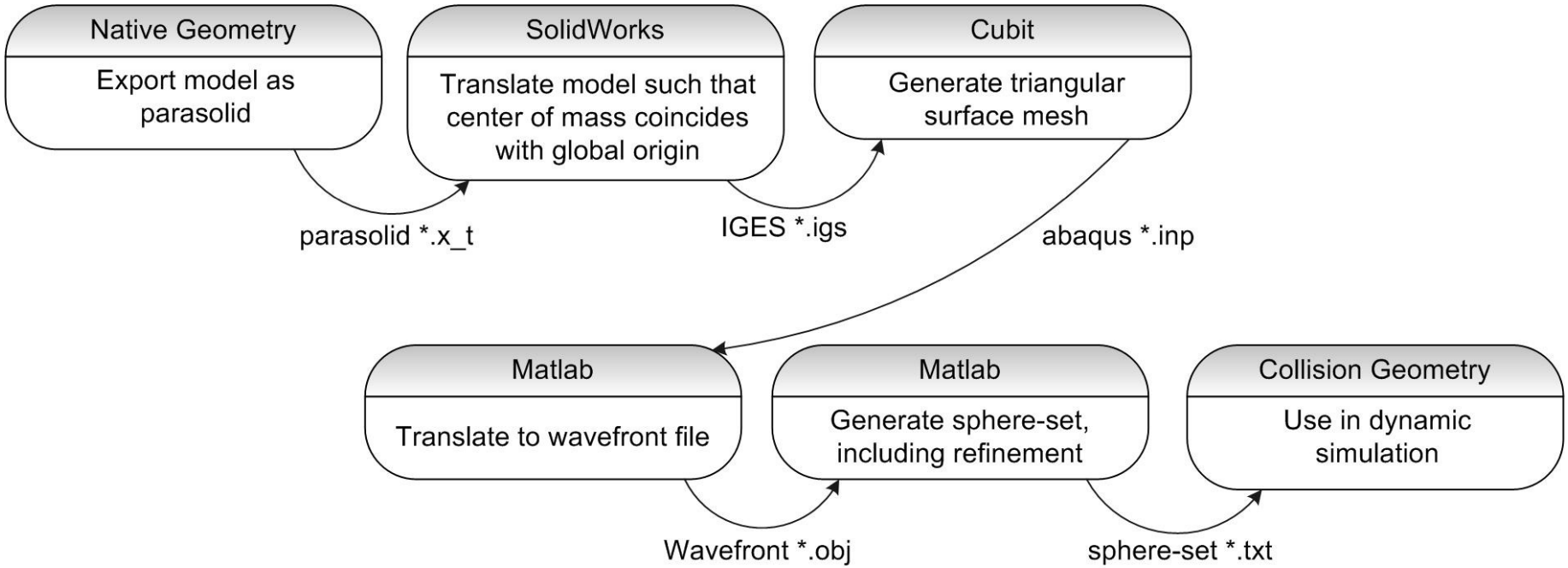
Spherical Decomposition

- Create triangular surface mesh
- Fit a single sphere through the vertices of each triangle
  - Surface normal defined by right hand rule
  - Compute center of circumcircle
  - Use iterative method to achieve target center ratio f=d/R

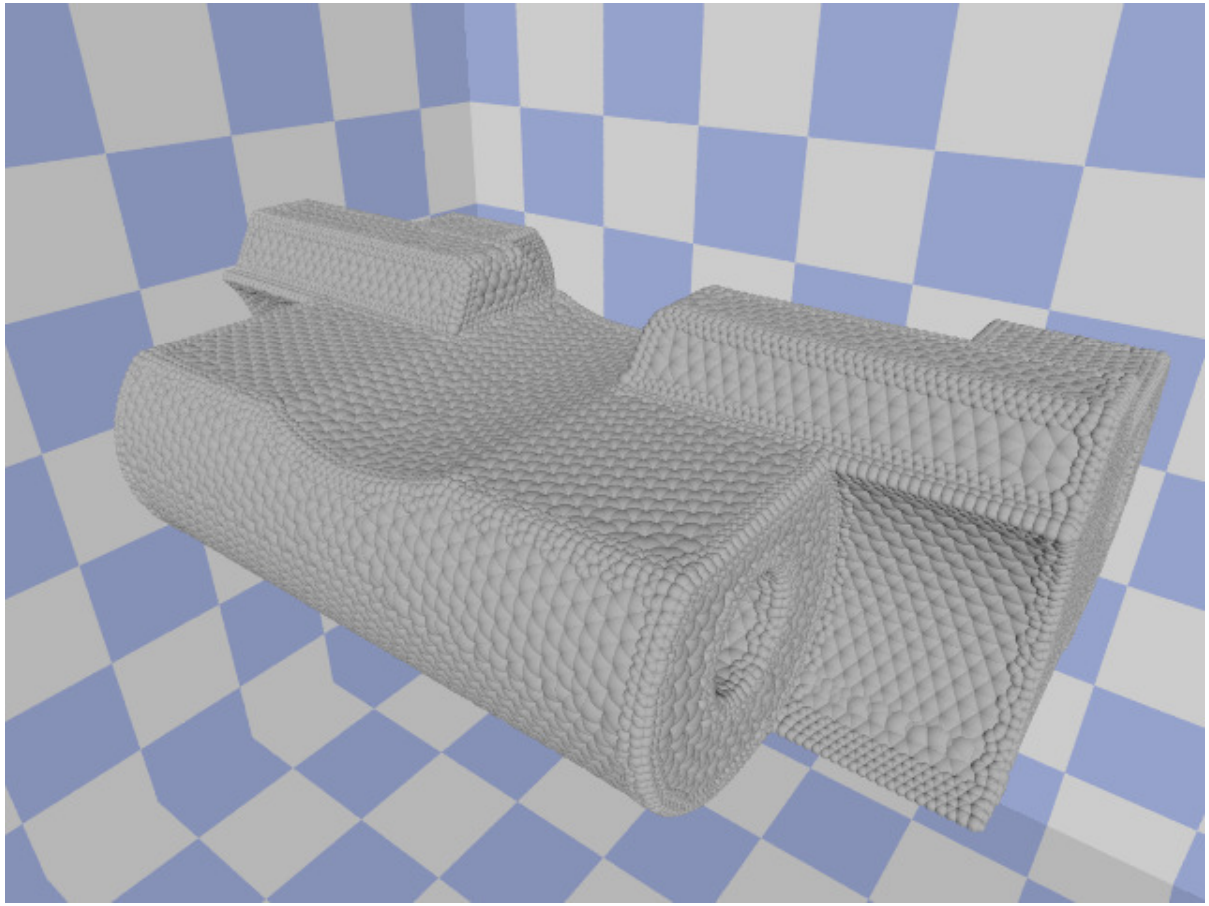




Spherical Decomposition

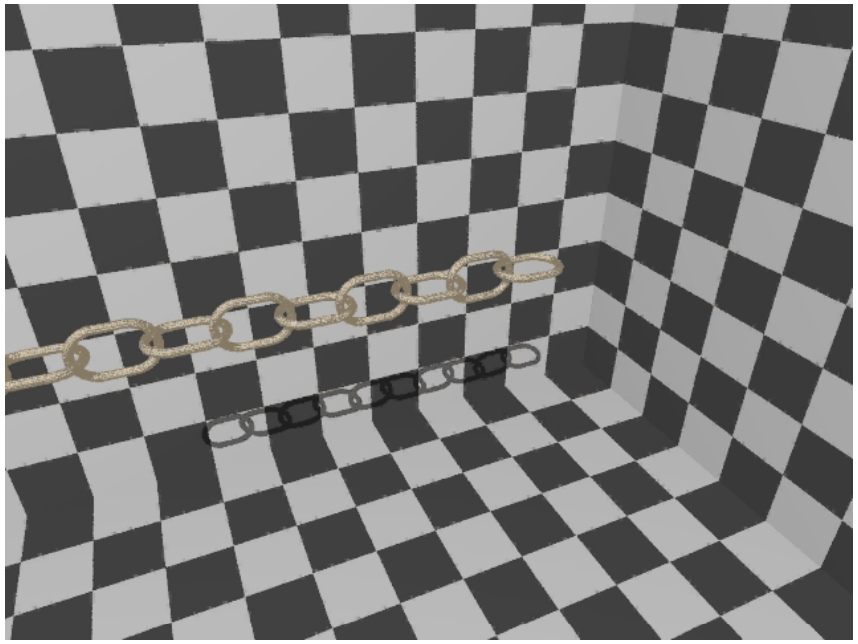


Sphere-set Refinement

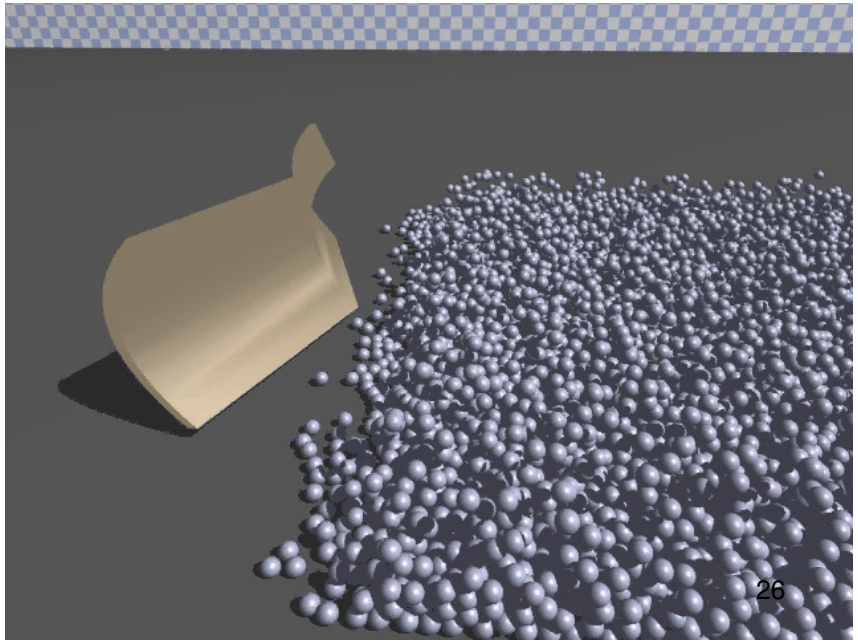


Examples…

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

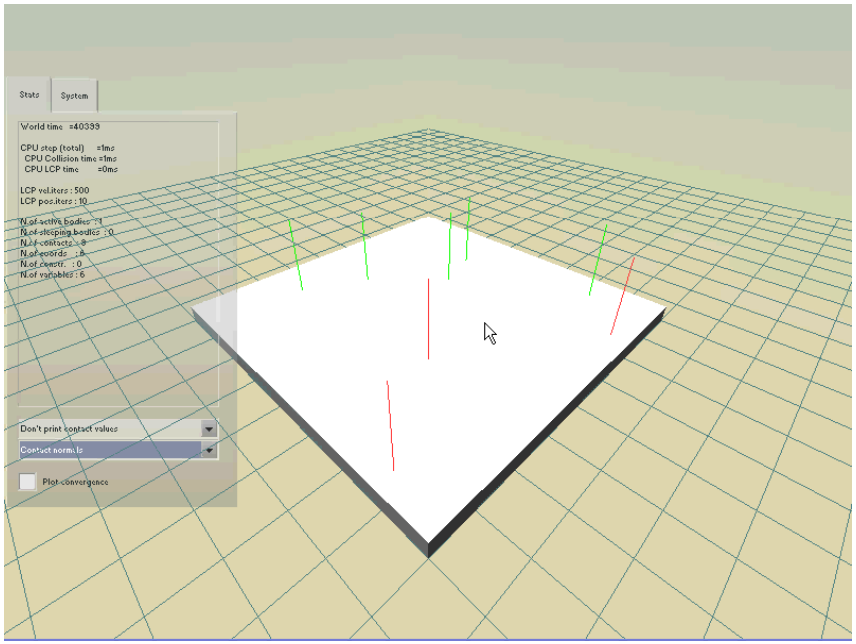


- Plow model
  - 31,791 spheres in plow blade model
  - 15,000 spheres representing terrain



Redundant Contacts

- When compound bodies collide, many contacts may be identified between the same pair of bodies
- Redundant contacts can lead to poor convergence for iterative CCP solver
- Use random subset of contacts acting between a given pair of bodies



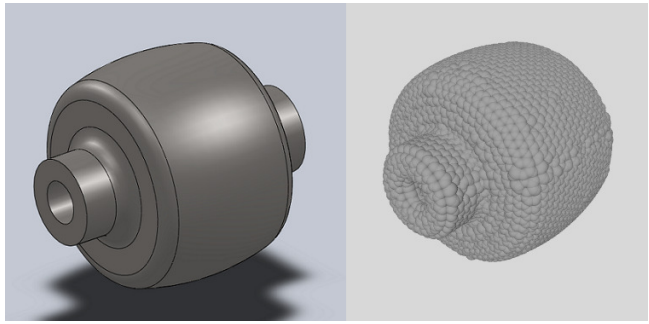
Collision geometry: 20x20 grid of spheres

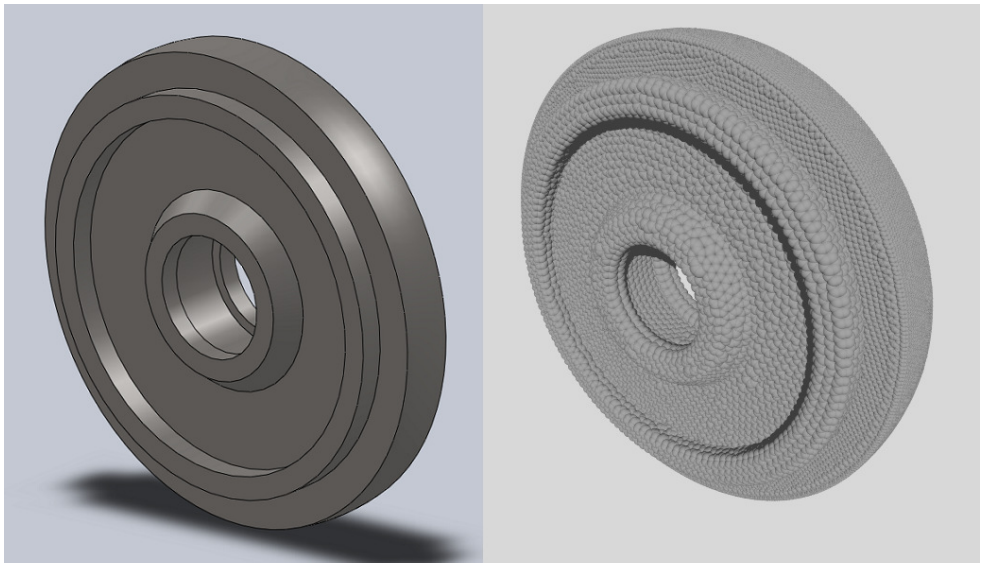
Subset: 8 contacts per pair

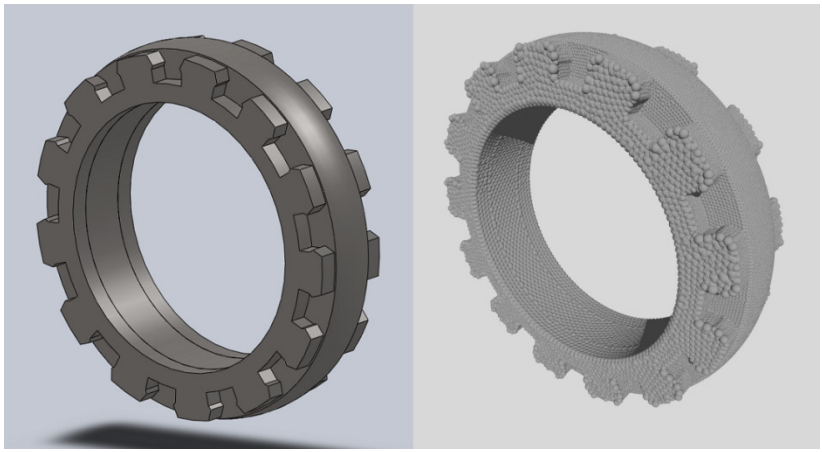
Tracked Vehicle Model

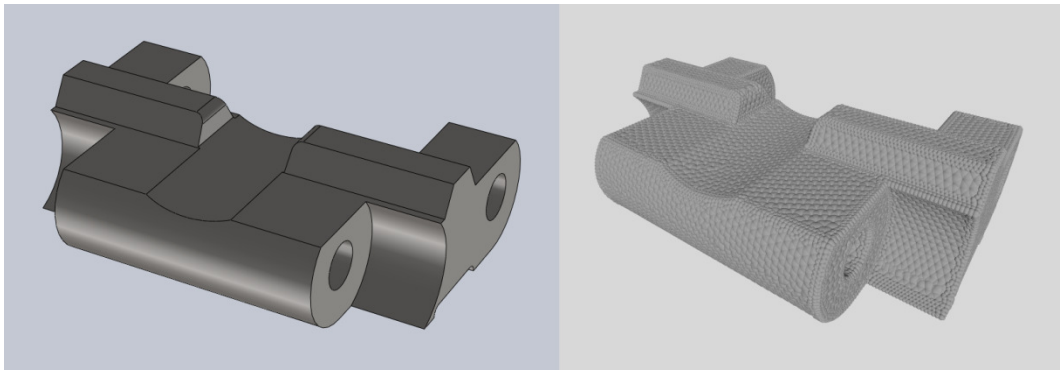
Track Components

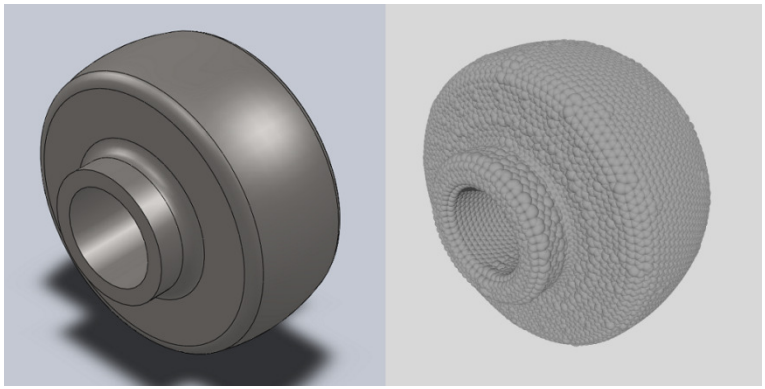
1,594,908 spheres per track



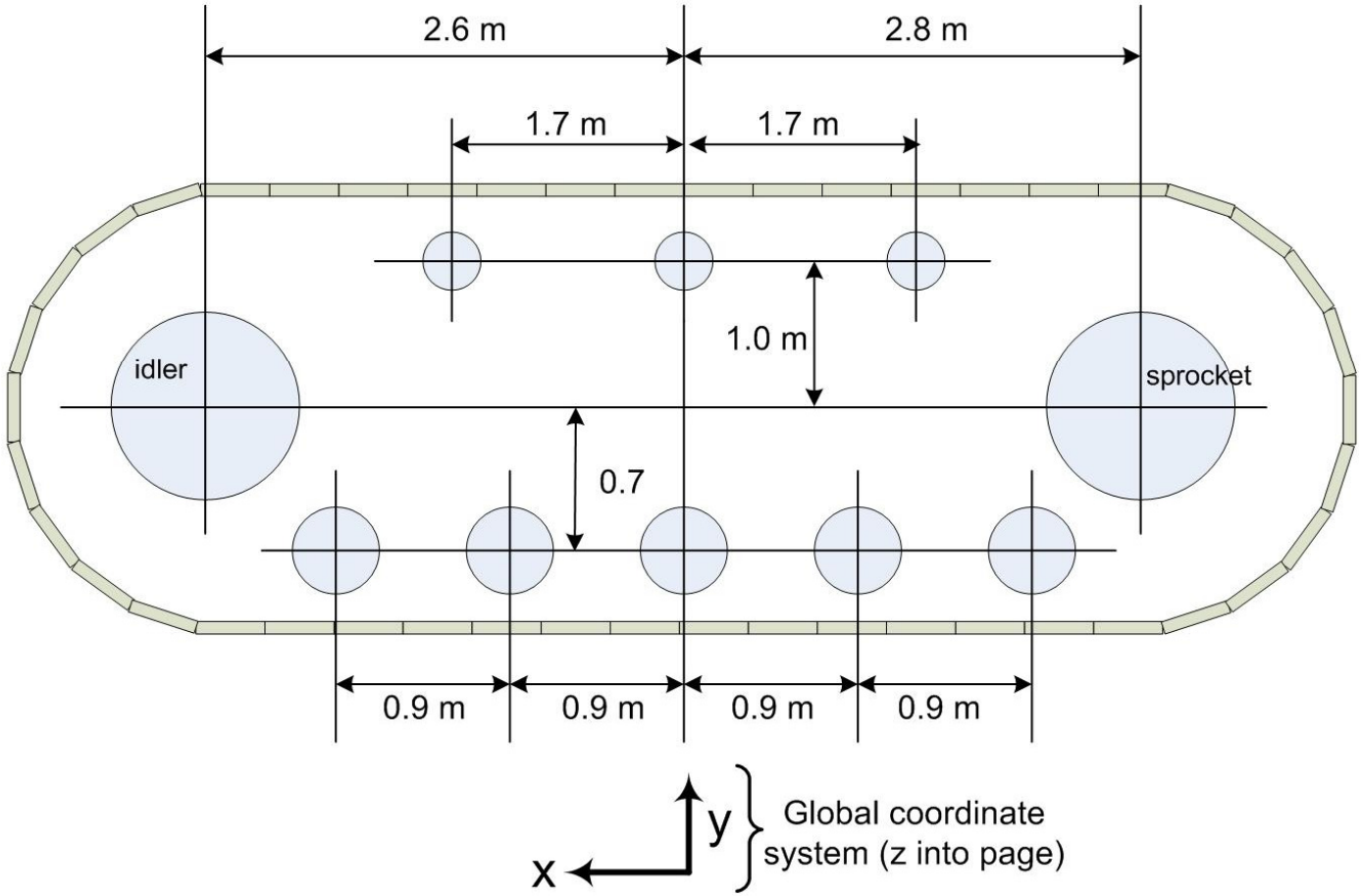




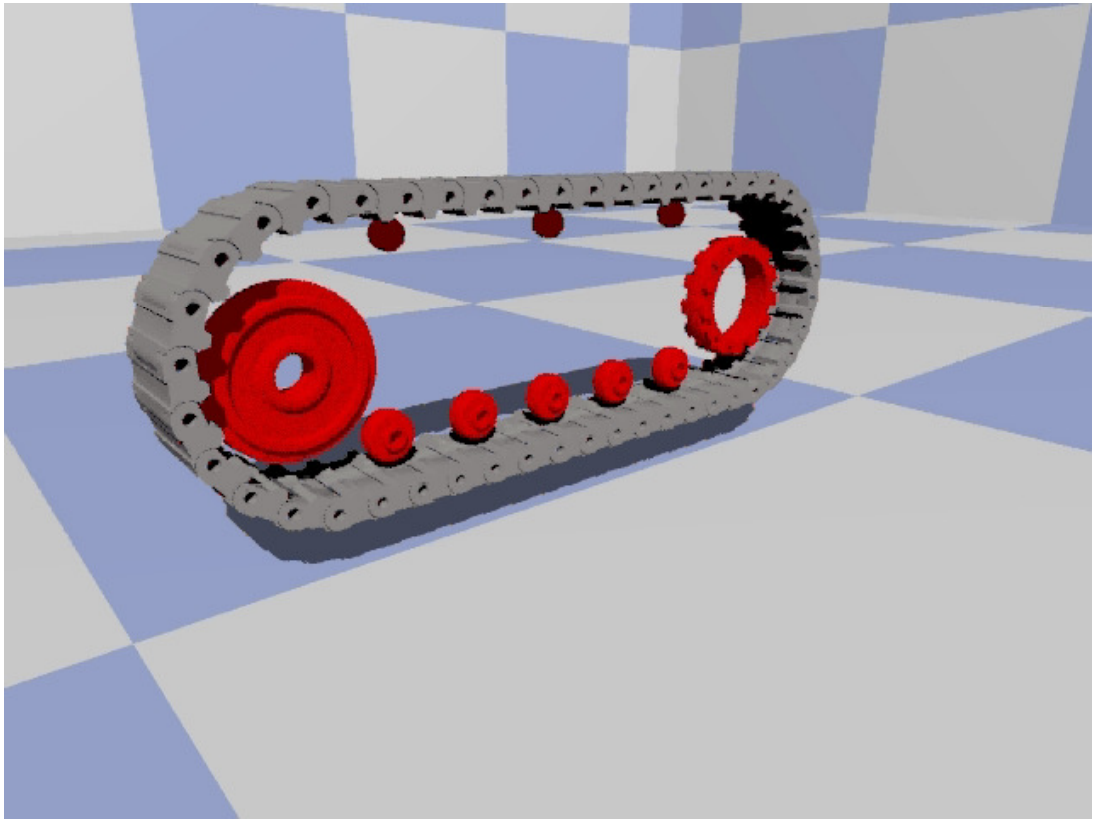




Track Model


2.6 m
2.8 m
1.7 m
1.7 m
1.0 m
idler
sprocket
0.7
0.9 m
0.9 m
0.9 m
0.9 m
y
x
Global coordinate system (z into page)

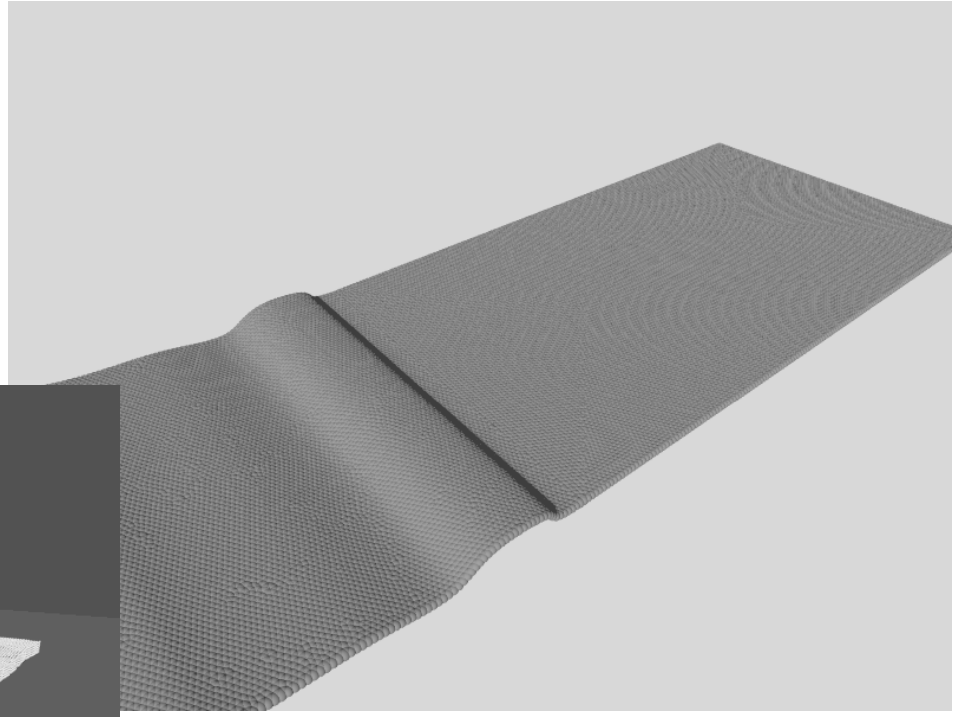
Track on Rigid Terrain

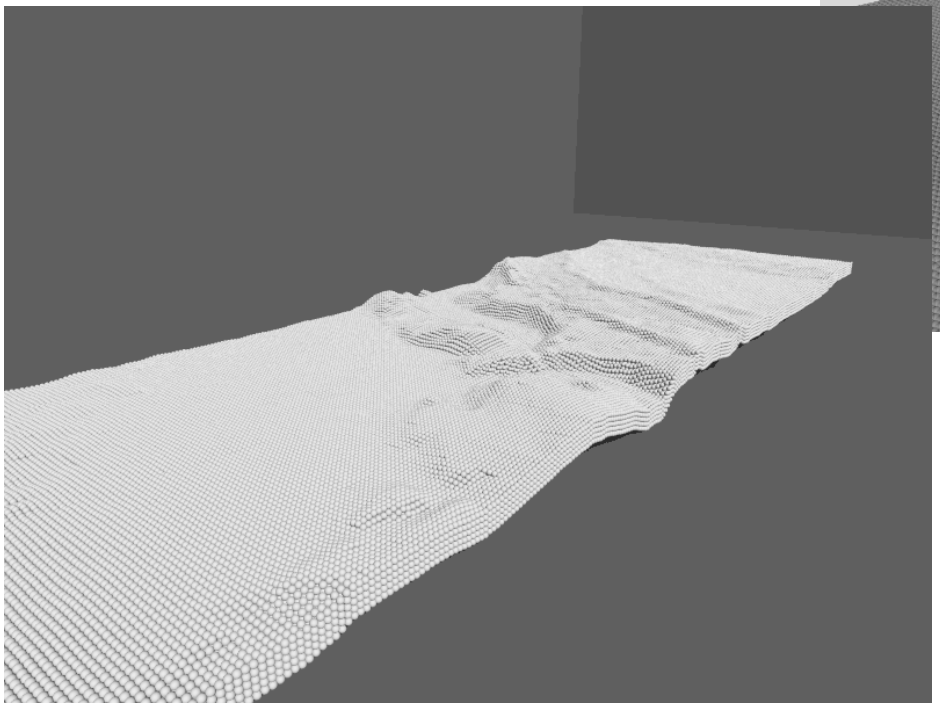


Granular Terrain Model

Terrain Representation

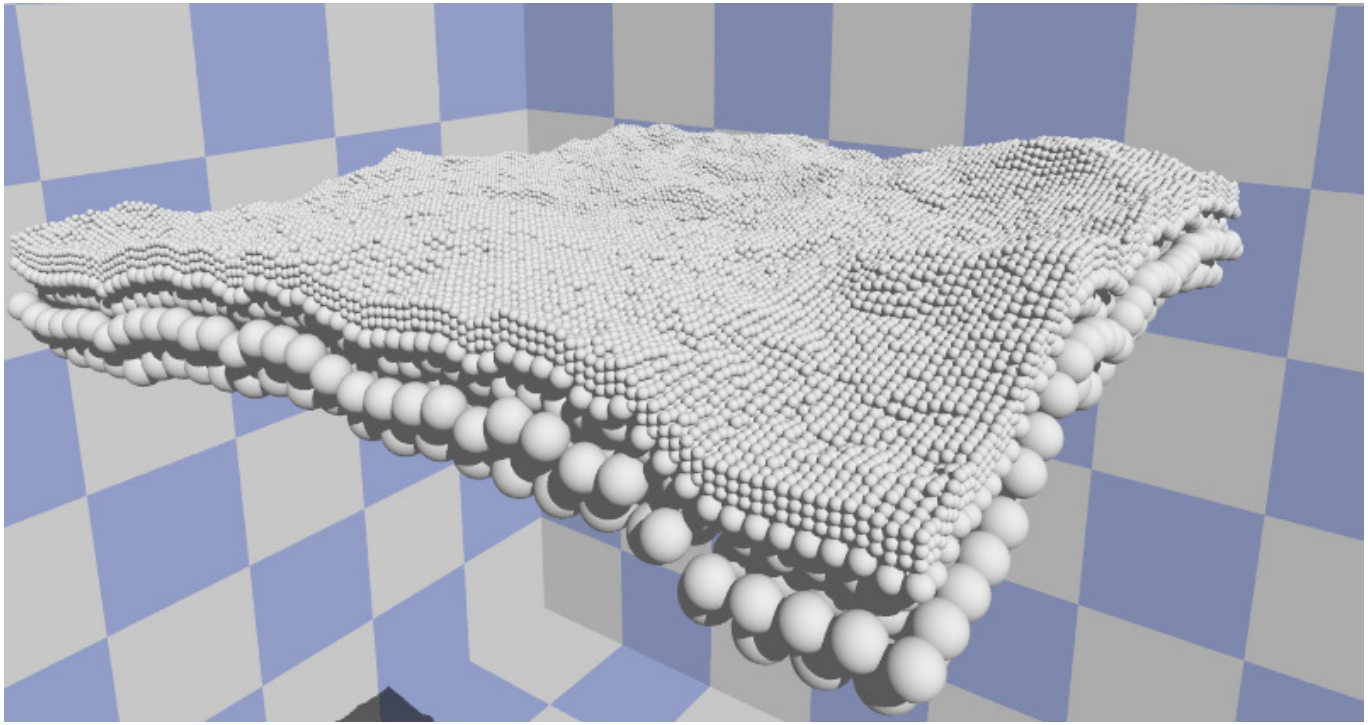
- Rigid plane
- Rigid sphere-set
- Discrete granular





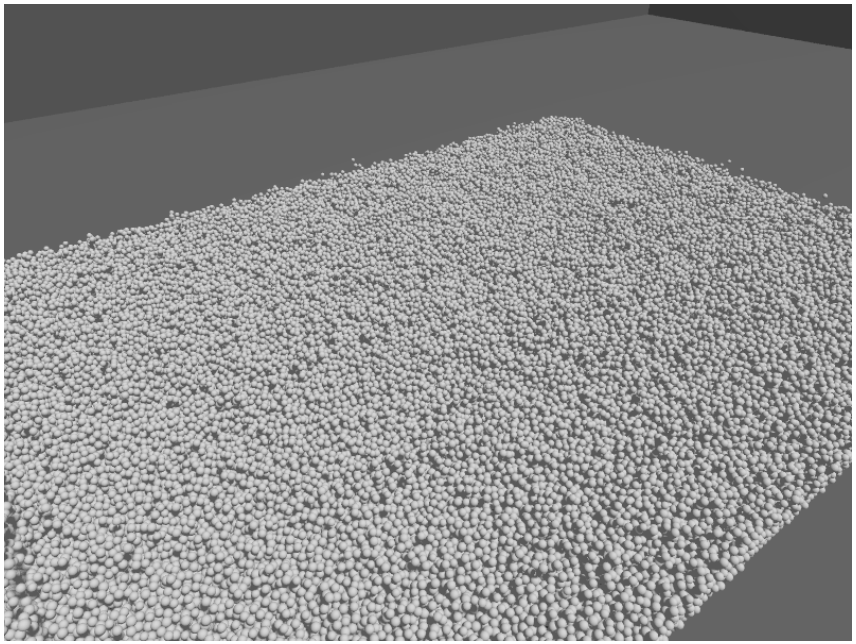
Granular Terrain Model

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

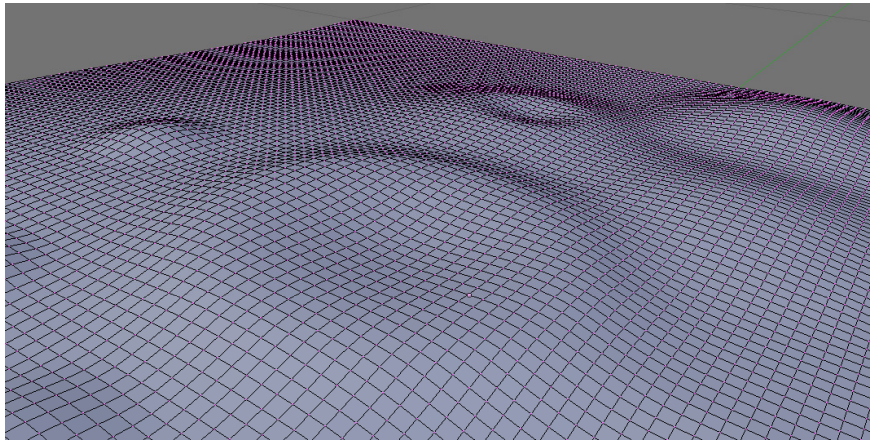


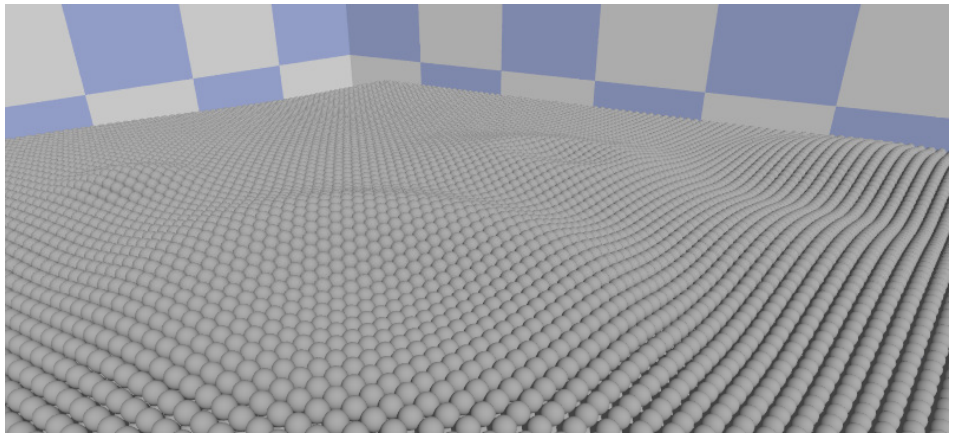
Random Filling

- Create particles at random locations in the domain
- Particle radius is r±0.1r, uniformly distributed
- Preprocessing simulation allows particles to settle



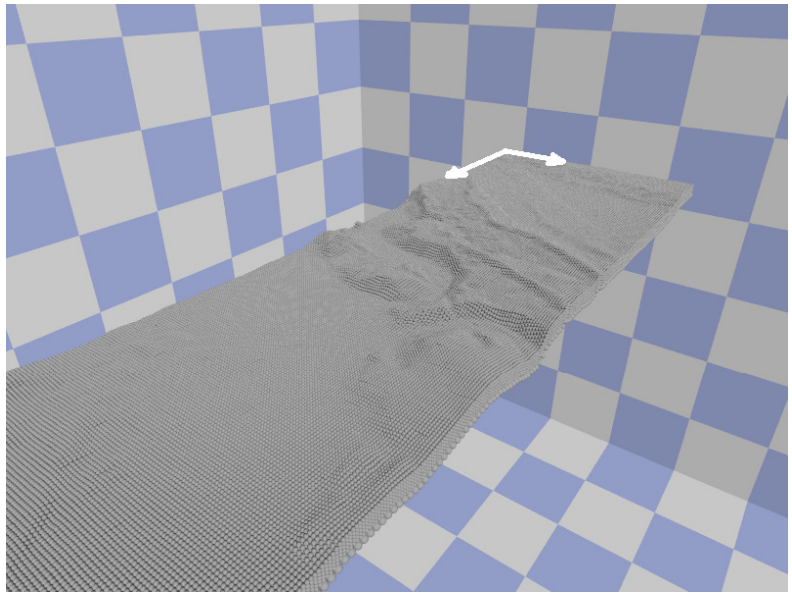
Mesh-Based Terrain





- Mesh file gives elevation of terrain on regular grid
- Choose desired particle size, sample grid to place each particle
- Offset mesh in global y direction to create different layers

Heightmap-Based Terrain

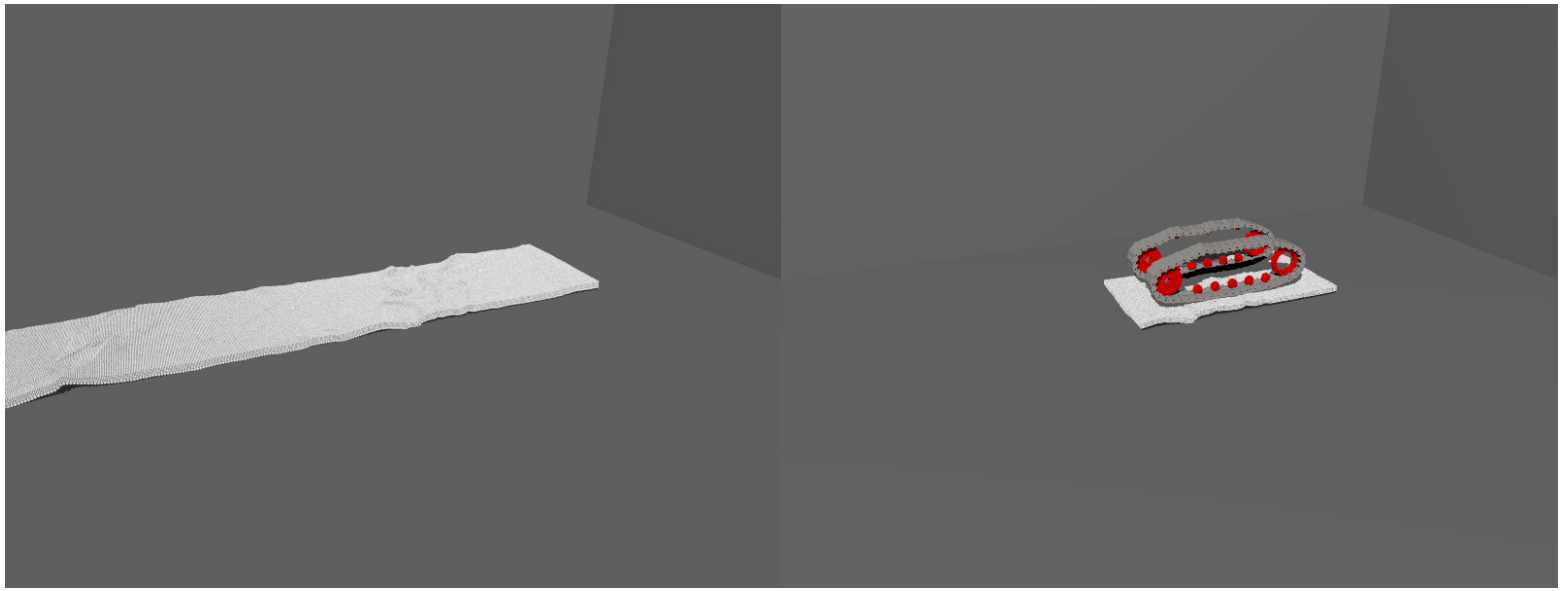


- Heightmap image defines elevation by grayscale
  - Black=0=lowest
  - White=1=highest
- Must know maximum feature size
- Sample image for grayscale value – convert to elevation
- Offset particles to create layers

Terrain Model

- Perform preprocessing step to generate initial conditions for terrain particles

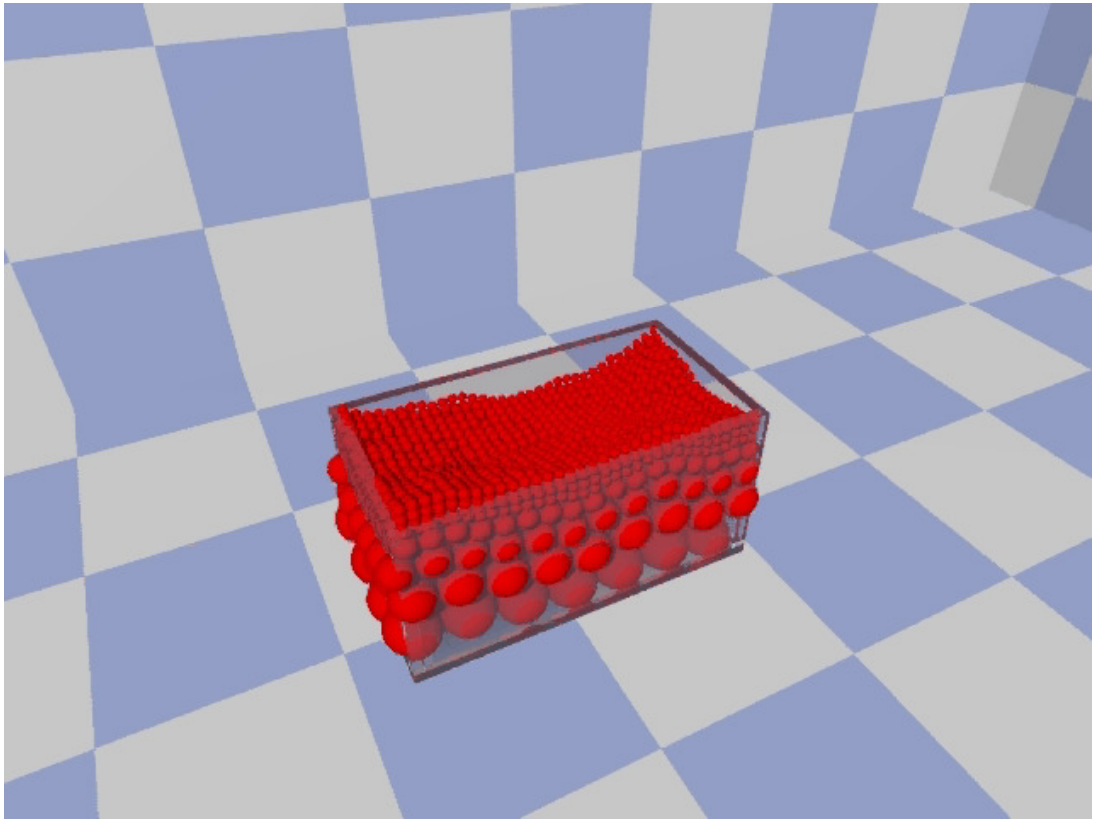
- Utilize ‘moving bounding box’ approach during simulation
  - No effort wasted simulating particles far from the vehicle
  - Allows larger terrain sets/smaller particles to be used within current memory limits



Moving Bounding Box

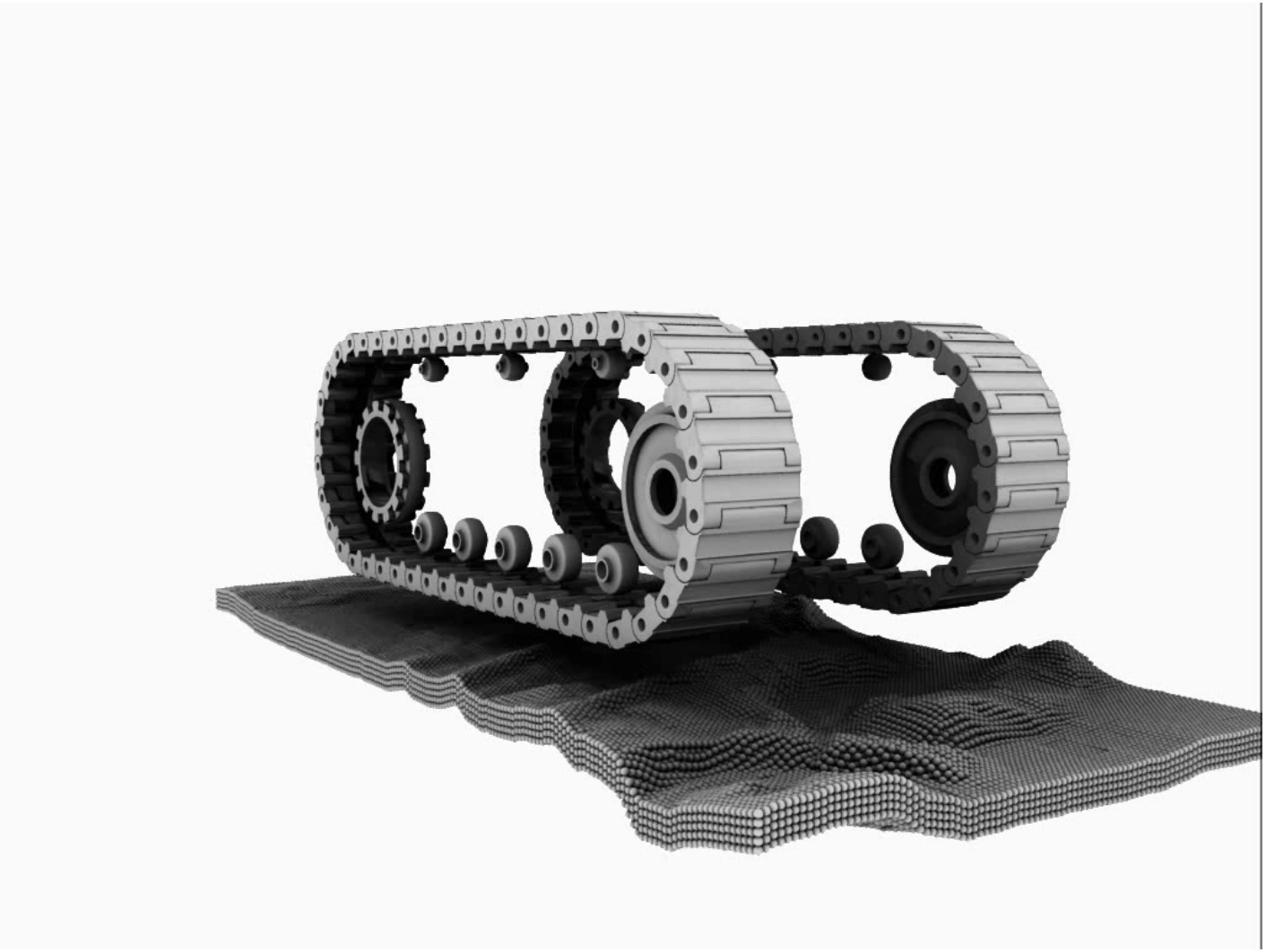
- terrainSphere
  - double x, y, z, r
  - bool active, bottomLayer
  - ChBody* tBody
- At each time step perform the following
  - Check position of each active body in box
    - If it is outside the box, remove it from the simulation
    - If it intersects the box, set it fixed
  - Check position of each inactive body in box
    - If it is inside the box, add it to the simulation
    - If it intersects the box, set if fixed
  - Update terrainSphere data for each body for persistence of terrain

Moving Bounding Box Demo



Simulation Results

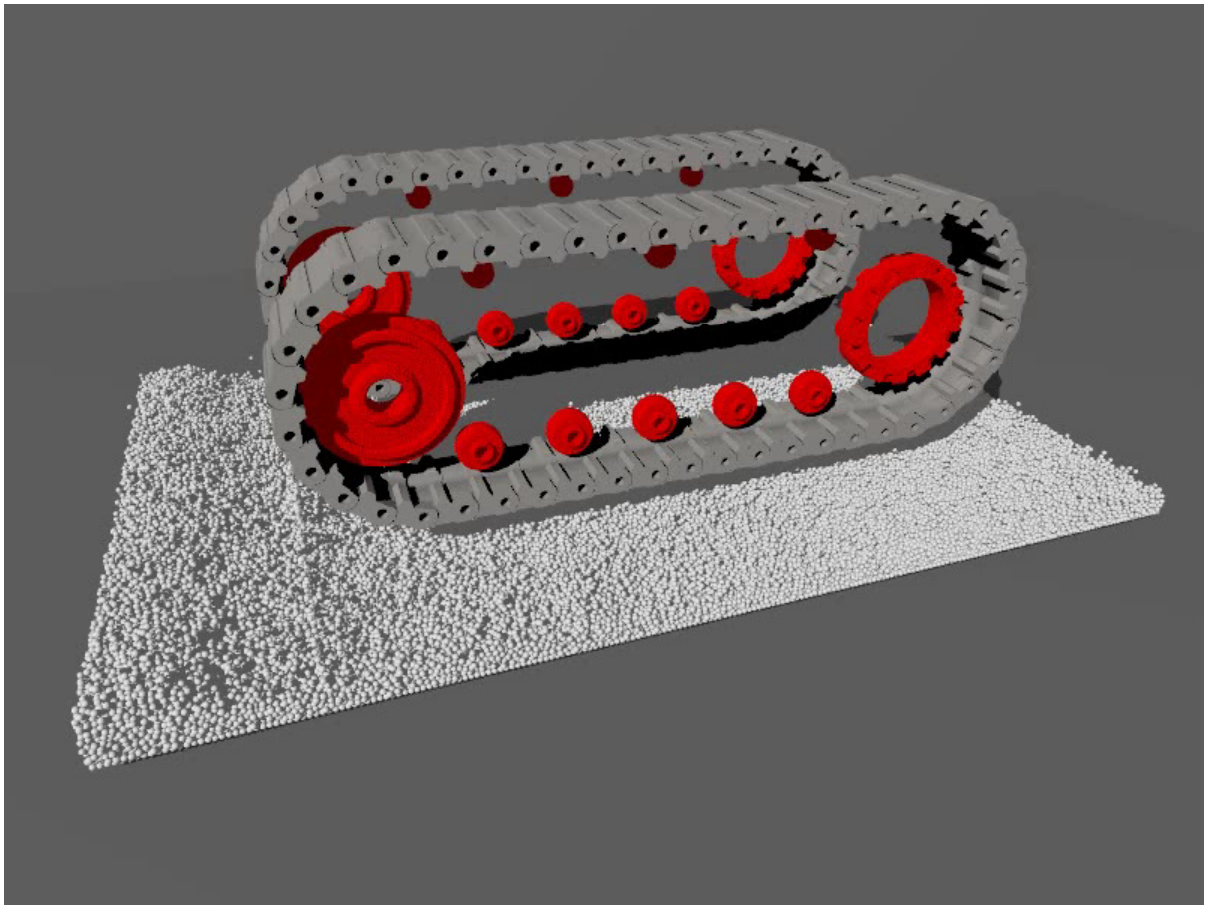
Track Simulation 1



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

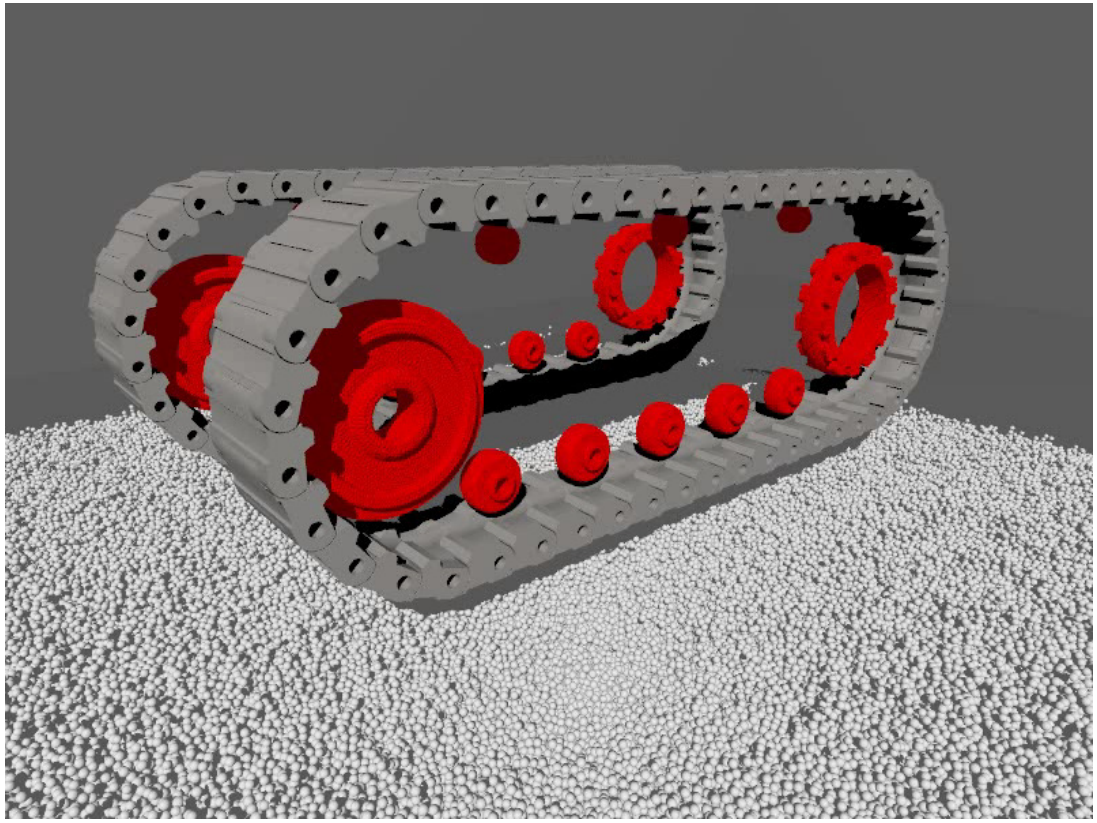
Track Simulation 2



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

Track Simulation 3

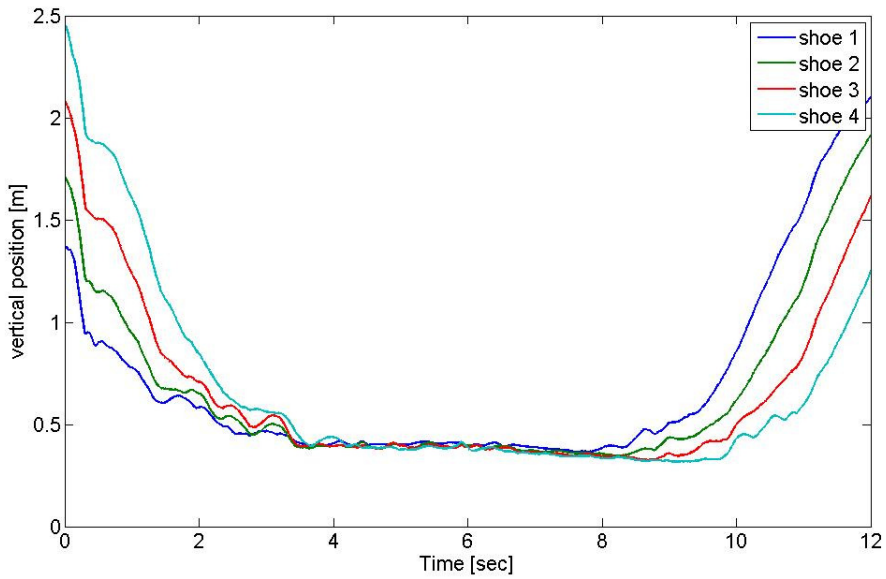


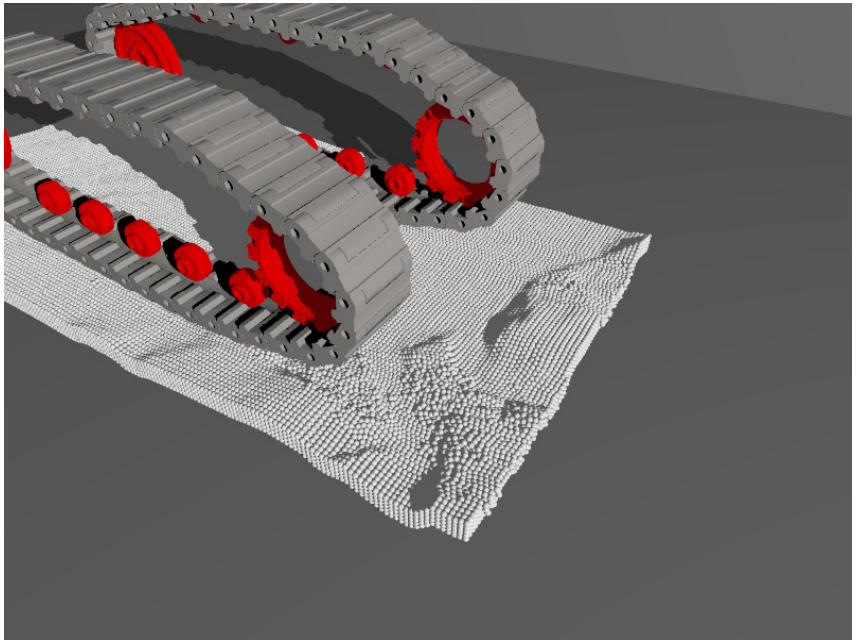
Parameters:

- Driving speed: ±1.0 rad/sec
- Length: 6 seconds
- Time step: 0.005 sec
- Computation time: 15.4 hours
- Particle radius: .025±.0025 m
- Terrain: 434,886 particles

Results: Positions

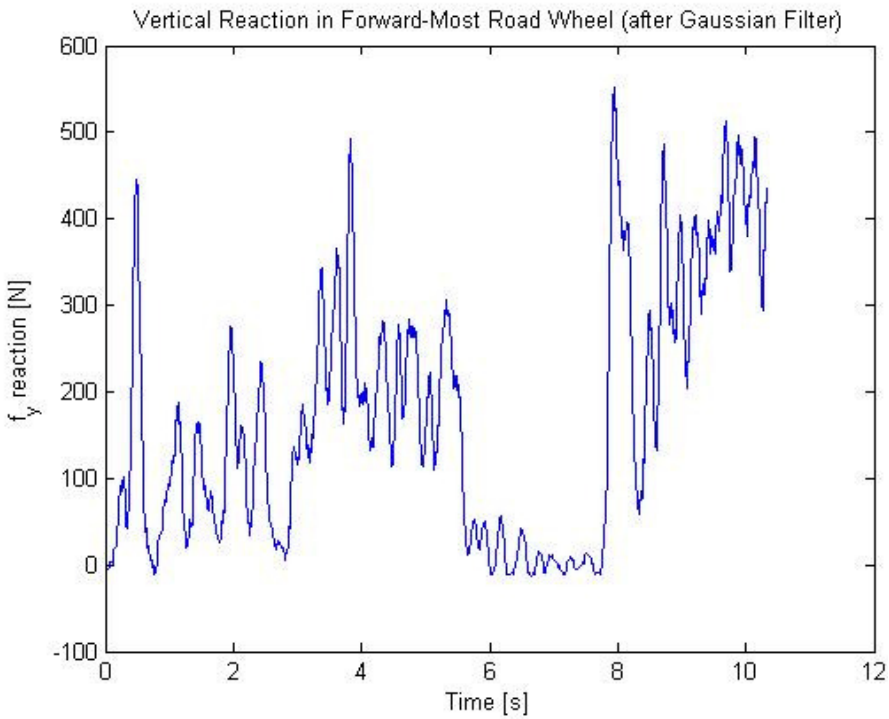
Track Simulation 1

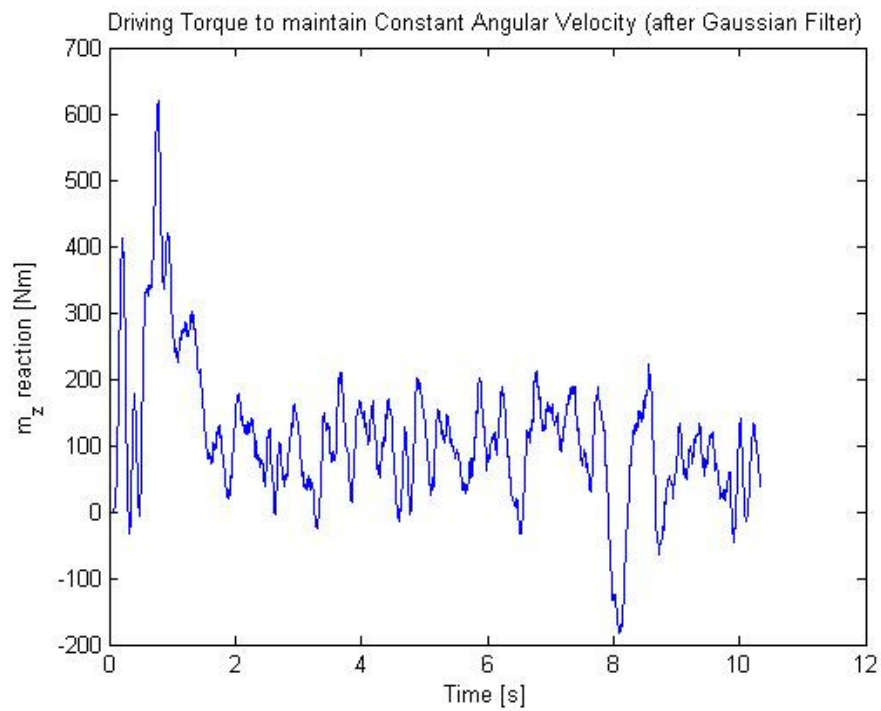




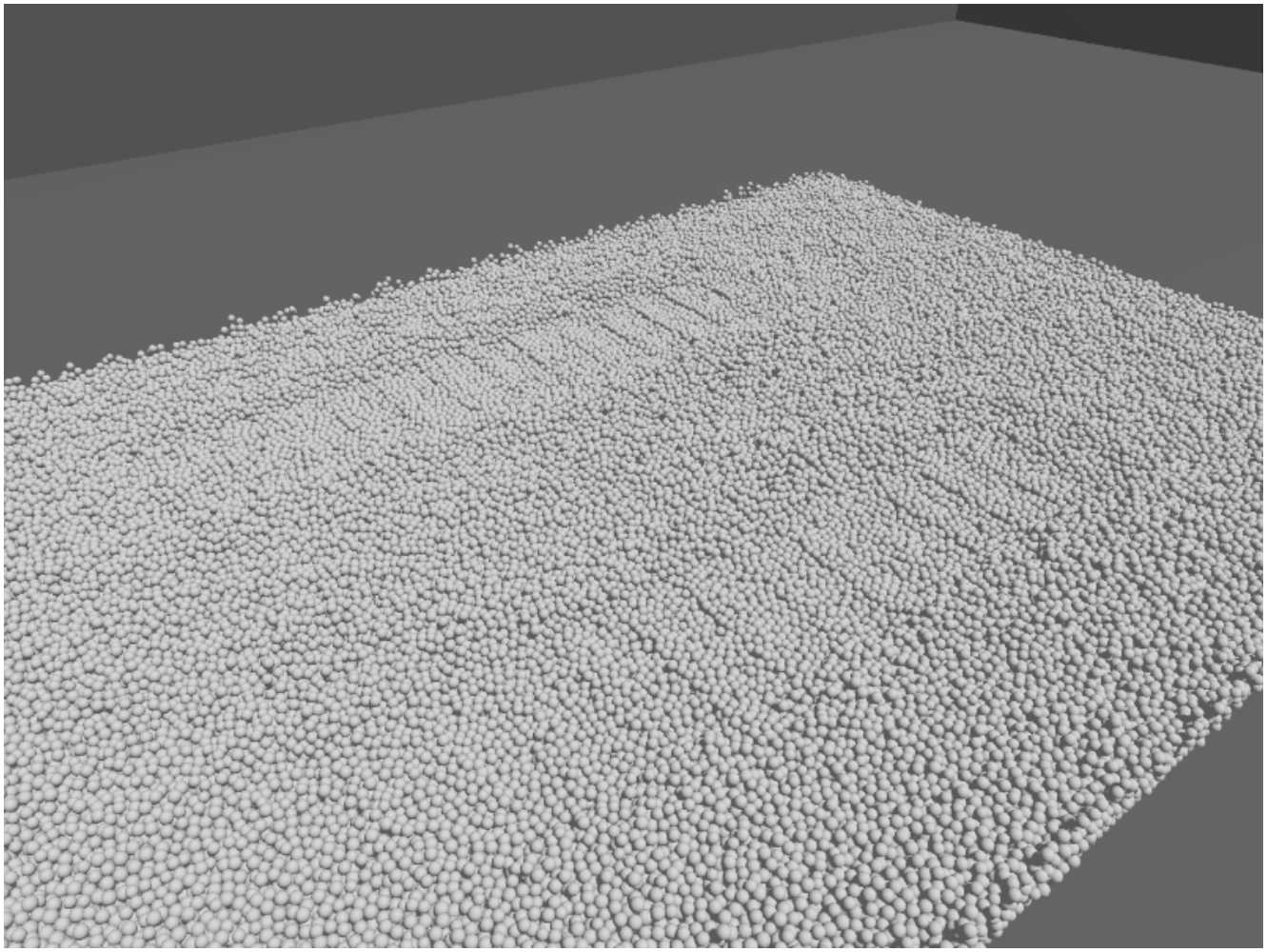
Results: Reaction forces

Track Simulation 2



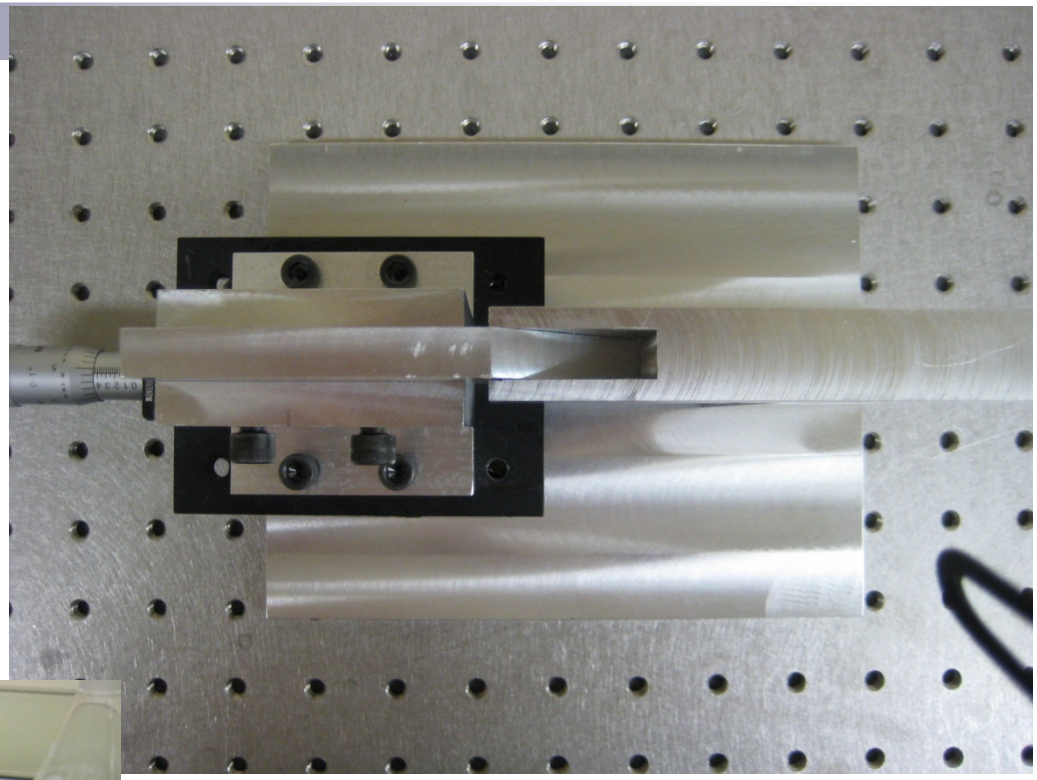


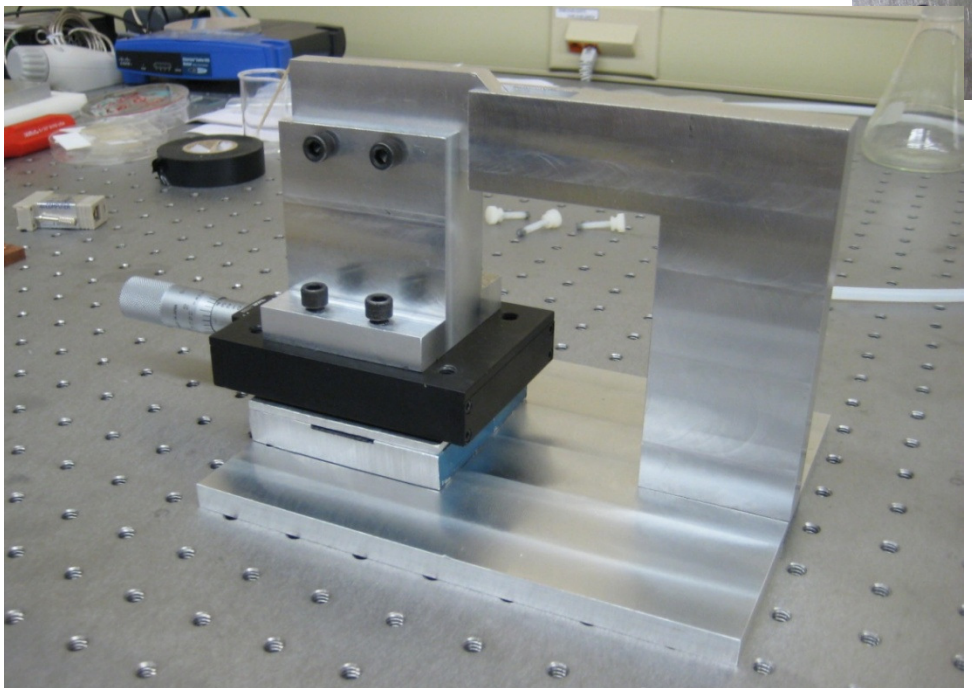
Results: Track ‘Footprint’

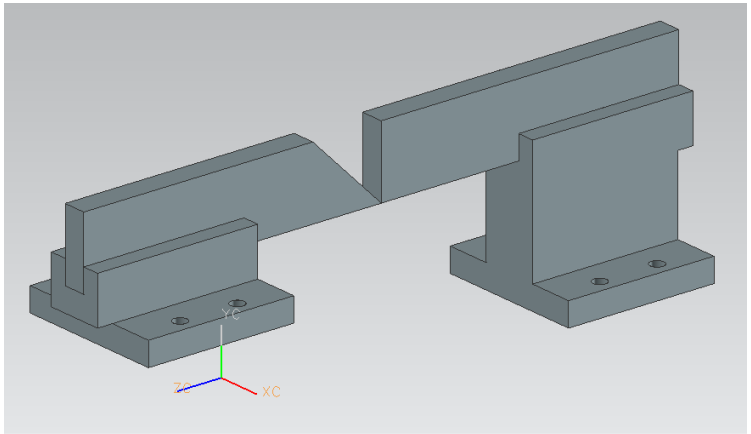


Validation at Microscale

- Sand flow rate measurements
- Approx. 1 million bodies
- Glass beads
- Diameter: 100-500 microns







Summary of Contributions

- Preliminary implementation of GPU collision detection
- Spherical decomposition method
- Reduction of redundant contacts to appropriate subset
- Three methods of generating granular terrain profiles from various input sources
- Moving bounding box approach for terrain simulation
- Extraction of reaction forces from GPU dynamics solver

Current limitations

- Large mass ratios lead to poor convergence
  - Mass of track shoe/mass of terrain particle ~100,000
- Handling of redundant constraints is not systematic
- Capability of CCP lags that of CD by three orders of magnitude
  - CCP: 1 million body simulation
  - CD: 1.4 billion contacts

Future Work

- Multi-GPU CCP solver
- Algebraic multi-grid for faster convergence
- Cluster of GPU machines
- Integration with smoothed-particle hydrodynamics (SPH) for multi-physics simulation
  - Interested in blast-worthiness of ground vehicles

Conclusions

- Achieved tracked vehicle simulation on granular terrain
  - Used GPU to speed up solution of CCP and CD problems
- Developed method for performing fast collision detection between complex geometries
- Developed method for representing complex granular terrain
  - Terrain preprocessing
  - Moving bounding box
- Demonstrated the capability to obtain reaction forces in track model

Thank you.