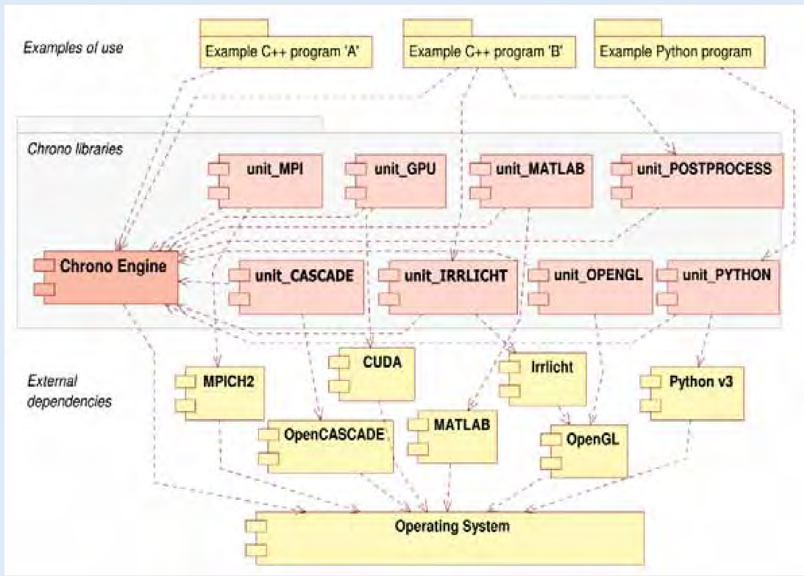
An open-source software for massive multibody simulation

Dario Mangoni[1], Dan Negrut[2], Alessandro Tasora[1], Hammad Mazhar[2]

[1]University of Parma, IT; [2]UW-Madison, WI

Introduction

Chrono::Engine is a powerful C++ middleware library especially designed to handle multibody dynamic problems. Developed to be platform-indipendent and thanks to its modular loose-coupled structure, it can be easily implemented in simulation softwares that aim to take advantage of the most modern state-of-the-art solutions in the field of non-smooth dynamics.



Chrono::Engine general-purpose modules; many other features available with task-specific add-ons.

Moreover many modules have been developed to enrich the features of the core library both in term of performance and connectivity with external third-party applications. In this way Chrono::Engine can be easily interfaced with Solidworks, Matlab, OpenCascade, but also can rely on efficient solver libraries such as Intel MKL Pardiso or Mumps.

Formulation

Chrono::Engine offers advanced solutions regarding frictional contacts. Instead of the most common Penalty Method, here contacts are modelled within the Differential Variational Inequality framework, that further leads to a Cone Complementarity Problem. The advantages are:

- larger timesteps that lead to lower computational expense
- effectively handles stick-slip conditions
- cohesion can be easily taken into account
- not dependent on contact model parameters (stiffness, damping, etc...) that could often lead to unreliable results

$$M^{(l)}(\boldsymbol{v}^{(l+1)} - \boldsymbol{v}^{l}) = \sum_{i \in \mathcal{G}_{\mathcal{A}}} \left(\gamma_n^i \boldsymbol{D}_n^i + \gamma_u^i \boldsymbol{D}_u^i + \gamma_v^i \boldsymbol{D}_v^i\right) + \sum_{i \in \mathcal{G}_{\mathcal{B}}} \left(\gamma_{\mathcal{B}}^i \nabla \Psi^i\right) + h \boldsymbol{f}_t(t^{(l)}, \boldsymbol{q}^{(l)}, \boldsymbol{v}^{(l)})$$

$$0 = \frac{1}{h}\Psi^i(\boldsymbol{q}^{(l)}) + \nabla \Psi^{i^T} \boldsymbol{v}^{(l+1)} + \frac{\partial \Psi^i}{\partial t}, \quad i \in \mathcal{G}_{\mathcal{B}}$$

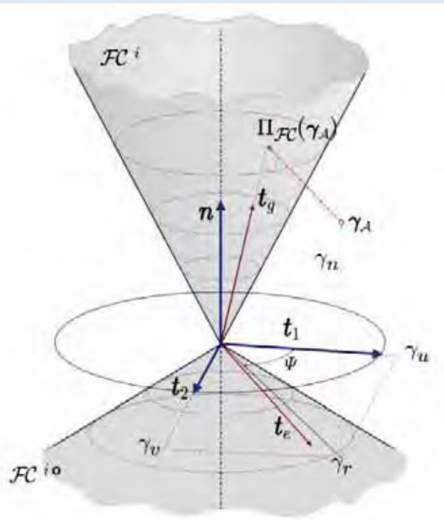
$$0 \le \frac{1}{h}\Phi^i(\boldsymbol{q}^{(l)}) + \nabla \Phi^{i^T} \boldsymbol{v}^{(l+1)} \perp \quad \gamma_n^i \ge 0, \; i \in \mathcal{G}_{\mathcal{A}}$$

$$(\gamma_u^i, \gamma_v^i) = \operatorname{argmin}_{\mu^i \gamma_n^i \ge \sqrt{(\gamma_u^i)^2 + (\gamma_v^i)^2}} \left[\boldsymbol{v}^T (\gamma_u \boldsymbol{D}_u^i + \gamma_v \boldsymbol{D}_v^i)\right] \quad i \in \mathcal{G}_{\mathcal{A}}$$

$$\boldsymbol{q}^{(l+1)} = \Lambda(\boldsymbol{q}^{(l)}, \boldsymbol{v}^{(l+1)}, h).$$

$$M^{(l)} v^{(l+1)} = \sum_{i \in \mathcal{G}_{\mathcal{A}}} \left(\gamma_n^i \boldsymbol{D}_n^i + \gamma_u^i \boldsymbol{D}_u^i + \gamma_v^i \boldsymbol{D}_v^i\right) + \sum_{i \in \mathcal{G}_{\mathcal{B}}} \left(\gamma_{\mathcal{B}}^i \nabla \Psi^i\right) + \tilde{\boldsymbol{k}}^{(l)},$$

$$- \boldsymbol{u}_{\mathcal{A}}^i \in \mathcal{FC}^{i\circ} \perp \boldsymbol{\gamma}_{\mathcal{A}}^i \in \mathcal{FC}^i, \quad i \in \mathcal{G}_{\mathcal{A}} \qquad - u_{\mathcal{B}}^i \in \mathcal{BC}^{i\circ} \perp \gamma_{\mathcal{B}}^i \in \mathcal{BC}^i, \quad i \in \mathcal{G}_{\mathcal{B}}.$$



The Coulomb Friction model and its formulation as a Cone Complementarity Problem (CCP)

Solvers

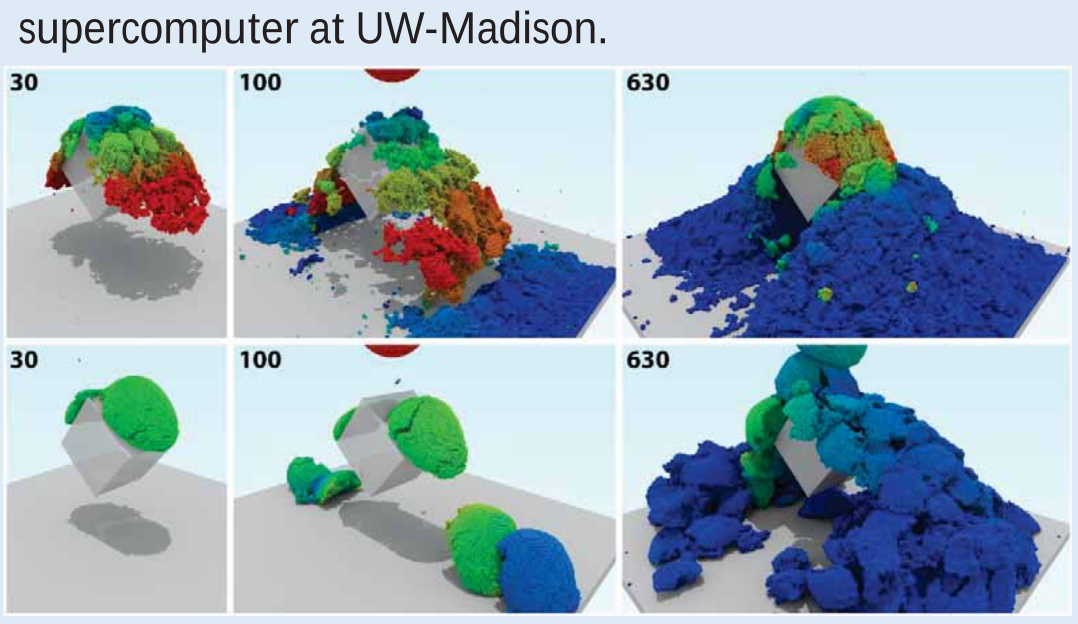
The CCP formulation advantages used in Chrono::Engine are exploited thanks to custom solvers designed and profiled to get the best performances in different scenario:

- Projected fixed-point solvers: SOR, SSOR, Jacobi, Gauss-Siedel
- Direct simplex/pivoting solver for Mixed-Linear CP (MLCP)
- Barzilai-Borwein gradient method
- Spectral Projected Gradient (P-SPG-FB)
- Nesterov's Accelerated Projected Gradient Descent Method (APGD)
- Krylov based methods: MinRes, CG
- Intel MKL Pardiso
- Mumps

Parallelism

Handling million-of-bodies problems couldn't be effectively achieved without taking into account the hardware on which the software can rely. Although the Chrono::Engine core library offers middle-scale parallelization capabilities thanks to OpenMP integration, great efforts are spent developing Chrono::GPU and Chrono::Parallel that unleash the computational power of small-scale modern GPUs as well as large-scale distributed computer using MPI.

Chrono library is also the choice for dynamic simulations on Euler supercomputer at UW-Madison.



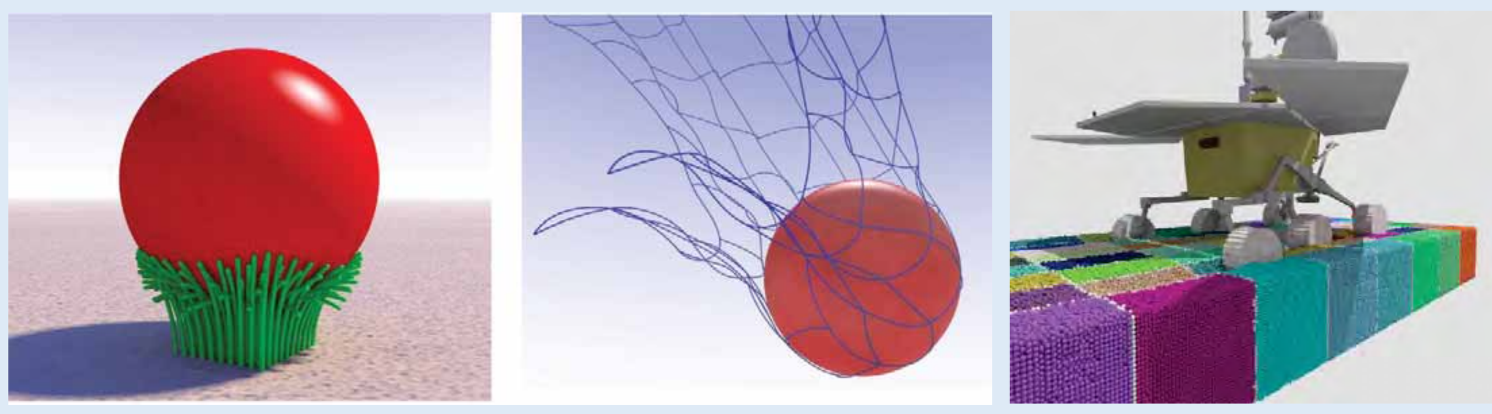
On the left: Snowballs with different cohesive force impact a rigid surface: an example of highly-parallelized simulation;
On the right: Euler supercomputer cluster. Chrono::Engine can also take advantage of large-scale parallelism thanks to MPI.

Collision Detection

Chrono::Engine collision detection relies on a high performance library especially built for parallel architecture, although Bullet library can be used for sequential simulations.

Sweep-and-prune with SAT, AABB and OOB algorithm are implemented from broad to narrow phases.

The collision can happen between a wide set of geometries including spheres, cubes, triangle meshes, but also convex shapes thanks to convex-decomposition algorithm.



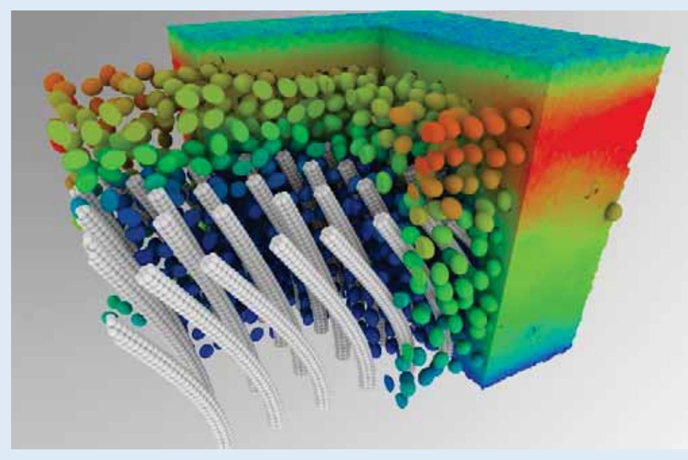
Collision detection algorithm at work with different objects among which ANCF elements and rigid bodies.

Real-World Applications

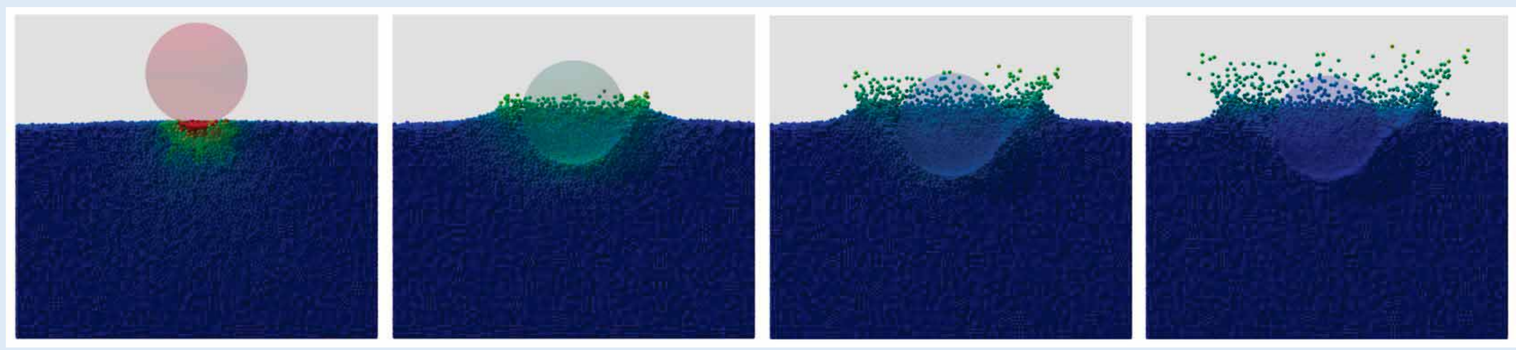
The wide set of Chrono::Engine features yield to a flexible library that can be effectively used in different applications involving:

- Smoothed-Particle Hydrodynamics (SPH)
- Finite Element Analysis (also ANCF cables and shells)
- tangled flexible structures with self contacts
- granular materials (e.g. tire-soil interaction)
- cohesive materials (food industry)
- particulate flows
- tracked vehicle mobility
- robotics (inverse kinematic included)

Humvee tire on granular soil.



Coupled fluid-flexible body analysis with ANCF elements.

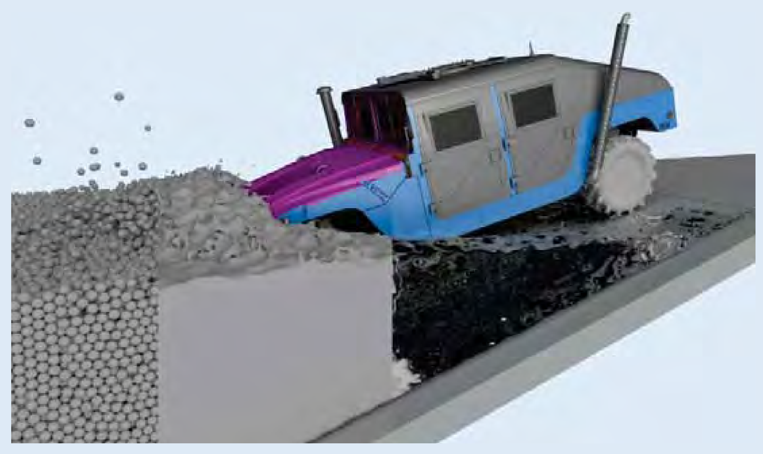


Ball drops in a bed of granular material; colours represent speed profile.

Conclusions

Chrono::Engine can tackle not only non-smooth dynamic problems with a huge number of bodies, but it can also handle contacts between them, also if different formulations are involved. With performances in mind, different effective solutions have been developed for leverage parallelism. This leads to a comprehensive open-source state-of-the-art solution that can be easily expanded and improved to meet the requirements of the end-user.

Flexible body interacting with rigid surface.



Vehicle through water basin; modelling fluids as particles can significantly decrease the simulation time while preserving a realistic effect.

References

Anitescu M, Tasora A. - *An iterative approach for cone complementarity problems for nonsmooth dynamics*

Tasora A., Anitescu M. - *A matrix-free cone complementarity approach for solving large-scale, nonsmooth, rigid body dynamics*

Mazhar H. et al. - *Chrono: A Parallel Multi-Physics Library for Rigid-Body, Flexible-Body, and Fluid Dynamics*

Heyn T., Anitescu M., Tasora A., Negrut D. - *Using Krylov Subspace and Spectral Methods for Solving Complementarity Problems in Many-Body Contact Dynamics Simulation*

Chrono::Engine website: chronoengine.info

SBEL labs website: sbel.wisc.edu