Chrono: An open source multi-physics dynamics engine

Alessandro Tasora[1], Radu Serban[2], Hammad Mazhar[2], Arman Pazouki[2], Daniel Melanz[2], Jonathan Fleischmann[2], Michael Taylor[2], Hiroyuki Sugiyama[3], and Dan Negrut[2]✉

[1] University of Parma, Italy
[2] University of Wisconsin–Madison, USA
[3] University of Iowa, USA

Abstract. We provide an overview of a multi-physics dynamics engine called Chrono. Its forte is the handling of complex and large dynamic systems containing millions of rigid bodies that interact through frictional contact. Chrono has been recently augmented to support the modeling of fluid-solid interaction (FSI) problems and linear and nonlinear finite element analysis (FEA). We discuss Chrono's software layout/design and outline some of the modeling and numerical solution techniques at the cornerstone of this dynamics engine. We briefly report on some validation studies that gauge the predictive attribute of the software solution. Chrono is released as open source under a permissive BSD3 license and available for download on GitHub.

Keywords: multi-physics modeling and simulation, rigid and flexible multi-body dynamics, friction and contact, fluid-solid interaction, granular dynamics, vehicle dynamics, parallel computing

1 Overview of Chrono and its Software Infrastructure

Chrono is an open source software infrastructure used to investigate the time evolution of systems governed by very large sets of differential-algebraic equations and/or ordinary differential equations and/or partial differential equations [**?**,**?**]. Chrono can currently be used to simulate (*i*) the dynamics of large systems of connected bodies governed by differential-algebraic equations; (*ii*) controls and other first-order dynamic systems governed by ordinary differential equations; (*iii*) fluid–solid interaction problems governed, in part, by the Navier-Stokes equations; and (*iv*) the dynamics of deformable bodies governed by partial differential equations. Chrono can handle multi-physics problems in which the solution calls for the mixing of (*i*) through (*iv*).

This dynamics simulation engine rests on five foundation components that provide the following basic functionality: equation formulation, equation solution, collision detection and proximity computation, support for parallel computing, and pre/post-processing, see Fig. 1. The first foundation component, called

✉ Corresponding Author: negrut@wisc.edu

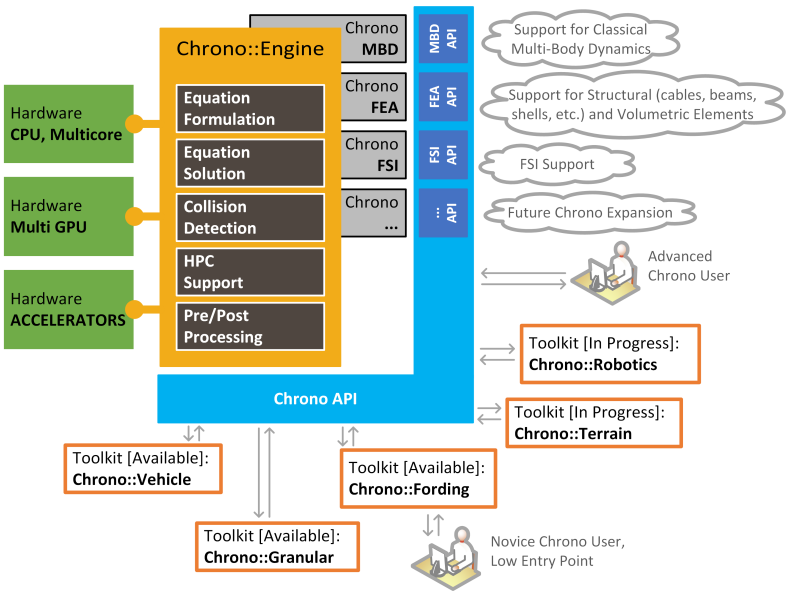


Fig. 1: An abstraction of the Chrono architecture. Chrono provides support for three application areas, namely multibody dynamics (MBD), FEA, and FSI, building on a five component foundation that handles the formulation of the equation of motion, numerical solution, collision detection, parallel and high-performance computing support, and pre/post processing tasks. An API (in blue) allows expert users to interact with Chrono. Toolkits such as Chrono::Vehicle, Chrono::Granular, etc. provide a low-entry point for users who need domain-specific Chrono support.

Equation Formulation, supports general-purpose modeling for large systems of rigid and flexible bodies and for basic FSI problems. The second component, called *Equation Solution*, provides the algorithmic support needed to numerically solve the resulting equations of motion. Proximity computation support, essential for collision detection and computation of short range interaction forces, is provided by the third foundation component. The fourth component enables the partitioning and farming out of very large dynamics problems for parallel execution on supercomputer architectures using the Message Passing Interface (MPI) paradigm [**?**]. The fifth component provides pre- and post-processing support.

Chrono is almost entirely written in C++. It is compiled into a library subsequently used by third party applications. A user can invoke functions implemented in Chrono via an Application Programming Interface (API) that comes in two options: C++ and Python. Chrono runs on Windows, Linux, and Mac OSX and is designed to leverage parallel computing. Depending on the solution mode in which it operates, it can rely on parallel computing on Graphics Processing Unit (GPU) cards using CUDA [**?**], multi-core computing using OpenMP [**?**], and multi-node parallel computing using MPI [**?**].

Chrono has been validated against experimental data, analytical results, and commercial software. Two correlation studies against MSC.ADAMS are summarized in [**?**, **?**]. The gradient-deficient beams and plates in Chrono::FEA have been validated against the commercial code ABAQUS [**?**]. The frictional-contact solution in Chrono has been validated against experimental data for angle of repose simulations [**?**], rate of flow [**?**, **?**], impact tests [**?**, **?**], and shear tests [**?**]. Finally, the fledgling Chrono::FSI module has been validated against analytical results and experimental data [**?**].

Chrono has been used for tracked and wheeled vehicle dynamics, in robotics (optimization and design of parallel kinematics machines), in the field of seismic engineering (simulation of earthquake effects on ancient buildings), in the field of waste processing (granular flows in separation machines), in additive manufacturing and 3D printing (see Figs. 2a and 2b), to simulate new types of escapements in mechanical clocks, and to characterize ice sheet dynamics for the oil industry. Approximately 150 movies that illustrate simulations run in Chrono are available at [**?**, **?**].

In terms of user support, the API documentation for the main Chrono modules is generated from their annotated C++ sources using Doxygen [**?**]. All Chrono software is configured and built using CMake [**?**] for a robust cross-platform build experience under Linux, Mac OSX, and Windows. Unit testing relies on the CTest suite of tools [**?**] to automate configuring, building, and executing the tests. Chrono operates under a continuous integration mode based on Buildbot [**?**]. Results and output from all builds, tests, and benchmarks are recorded to an external database accessible to all developers.

2 Rigid Body Dynamics Support in Chrono

The dynamics of articulated systems composed of rigid and flexible bodies is characterized by a system of index 3 Differential Algebraic Equations (DAEs) [?, ?, ?] shown using the terms in black font in Eqs. (1a)–(1c):

$$\dot{\mathbf{q}} = \mathbf{L}(\mathbf{q})\mathbf{v} \tag{1a}$$

$$\mathbf{M}(\mathbf{q})\dot{\mathbf{v}} = \mathbf{f}\left(t, \mathbf{q}, \mathbf{v}\right) - \mathbf{g}_{\mathbf{q}}^{\mathrm{T}}(\mathbf{q}, t)\hat{\lambda} \quad + \sum_{i \in \mathcal{A}(\mathbf{q}, \delta)} \underbrace{\left(\widehat{\gamma}_{i,n}\, \mathbf{D}_{i,n} + \widehat{\gamma}_{i,u}\, \mathbf{D}_{i,u} + \widehat{\gamma}_{i,w}\, \mathbf{D}_{i,w}\right)}_{i^{th}\,\text{frictional contact force}} \tag{1b}$$

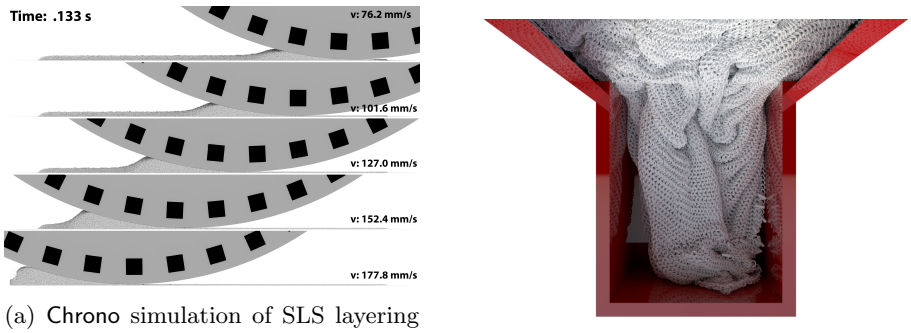
$$\mathbf{0} = \mathbf{g}(\mathbf{q}, t) \tag{1c}$$

$$i \in \mathcal{A}(\mathbf{q}(t)) : \begin{cases} 0 \leq \Phi_i(\mathbf{q}) \;\perp\; \widehat{\gamma}_{i,n} \geq 0 \\ (\widehat{\gamma}_{i,u}, \widehat{\gamma}_{i,w}) = \underset{\sqrt{\bar{\gamma}_{i,u}^2 + \bar{\gamma}_{i,w}^2} \leq \mu_i \widehat{\gamma}_{i,n}}{\operatorname{argmin}} \mathbf{v}^T \cdot \left(\bar{\gamma}_{i,u}\, \mathbf{D}_{i,u} + \bar{\gamma}_{i,w}\, \mathbf{D}_{i,w}\right). \end{cases} \tag{1d}$$

The differential equations in (1a) relate the time derivative of the generalized positions $\mathbf{q}$ and velocities $\mathbf{v}$ through a linear transformation defined by $\mathbf{L}(\mathbf{q})$. The force balance equation in (1b) ties the inertial forces to the applied and constraint forces, $\mathbf{f}\left(t, \mathbf{q}, \mathbf{v}\right)$ and $-\mathbf{g}_{\mathbf{q}}^{\mathrm{T}}(\mathbf{q}, t)\hat{\lambda}$, respectively. The latter are imposed by bilateral constraints that restrict the relative motion of the rigid or flexible bodies present in the system. These bilateral constraints, which lead to Eq. (1c), can be augmented by unilateral constraints associated with contact/impact phenomena. To that end, the concept of equations of motion is extended to employ differential inclusions [?]. The simplest example is that of a body that interacts with the ground through friction and contact, when the equations of motion become an inclusion $\mathbf{M}\ddot{\mathbf{q}} - \mathbf{f} \in \mathbf{F}(\mathbf{q}, t)$, where $\mathbf{M}$ is the inertia matrix, $\ddot{\mathbf{q}}$ is the body acceleration, $\mathbf{f}$ is the external force, and $\mathbf{F}(\mathbf{q}, t)$ is a set-valued function. The inclusion states that the frictional contact force lies somewhere inside the friction cone, with a value yet to be determined and controlled by the stick/slip state of the interaction between body and ground. In MBD the differential inclusion can be posed as a differential variational inequality problem [?], which brings along the red-font terms in Eq. (1) [?, ?, ?, ?, ?, ?, ?]. Specifically, the unilateral constraints define a set of contact complementarity conditions $0 \leq \Phi_i(\mathbf{q}) \perp \widehat{\gamma}_{i,n} \geq 0$, which make a simple point: for a potential contact i in the active set, $i \in \mathcal{A}(\mathbf{q}(t))$, either the gap Φ_i between two geometries is zero and consequently the normal contact force $\widehat{\gamma}_{i,n}$ is greater than zero, or vice versa. The last equation poses an optimization problem whose first order Karush-Kuhn-Tucker optimality conditions are equivalent to the Coulomb dry friction model [?]. The frictional contact force associated with contact i leads to a set of generalized forces, shown in red in Eq. (1b), which are obtained using the projectors $\mathbf{D}_{i,n}$, $\mathbf{D}_{i,u}$, and $\mathbf{D}_{i,w}$ [?].

The modeling methodology outlined above has been used in Chrono to analyze the dynamics of large multibody systems and granular material in a so called Discrete Element Method (DEM) framework. Since the methodology uses complementarity (C) conditions to enforce non-penetration of the discrete elements that come in mutual contact, this method is called DEM-C. This dif-

ferentiates it from DEM-P, a penalty (P) based methodology that is also implemented in Chrono and which accounts for the partial deformation of the bodies in mutual contact. Chrono, in its DEM-C embodiment that draws on Eq. (1), is shown at work in conjunction with two additive manufacturing processes in Fig. 2. Several frames of the Selective Laser Sintering (SLS) layering process simulations for various translational speeds are juxtaposed in 2(a). The model consists of 1 300 000 rigid spheres with an average diameter of 55 µm and a density of 930 kg/m^3 [**?**]. The radius of the powder particles was randomly distributed using a normal distribution. A roller with a diameter of 0.0762 m travels at various longitudinal speeds and rotates at a rate of 3.33 rad/s. An approximate equation tally is as follows: close to eight million from (1a), the same count for (1b), and almost no equations for (1c). In (1d), there are approximately 7.8 million contact complementarity conditions for the normal force and, because the problem accounts for normal, sliding, rolling and spinning of the bodies, there are six Lagrange multipliers for each contact. The optimization problem that provided the frictional contact forces for this simulation was posed in approximately 46.8 million variables and solved in Chrono using a methodology proposed in [**?**, **?**]. The handling of the rolling and spinning friction is described in [**?**]. The image in Fig. 2(b) shows a frame of a folding simulation that is a key step in predicting where each component of a dress is located inside the 3D printing volume. The position and orientation of each element is passed to the 3D printer which then prints the entire dress. The chain-mail dress is made up of 40 760 rigid rings and 55 clasps collapsing into a $10 \times 10 \times 10$ in^3 printing volume [**?**].



(a) Chrono simulation of SLS layering process.

(b) Chrono simulation of dress folding.

Fig. 2: Left – frames of the SLS layering process simulations for various translational speeds. Right – a frame of a folding simulation used in the 3D printing of a dress.

When using the DEM-P approach, or soft-body approach, Chrono regards the contacting bodies are "soft" in the sense that they are allowed to "overlap" or experience local deformation before a corrective contact force is applied at the point of contact. Once such an overlap δ_n is detected, by any one of a number of

contact algorithms, contact force vectors $\mathbf{F}_n$ and $\mathbf{F}_t$ normal and tangential to the contact plane at the point of contact are calculated using various constitutive laws [**?**, **?**, **?**] based on the local body deformation at the point of contact. In the contact-normal direction, $\mathbf{n}$, this local body deformation is defined as the penetration (overlap) of the two quasi-rigid bodies, $\mathbf{u}_n = \delta_n \mathbf{n}$. In the contact-tangential direction, the deformation is defined as a vector $\mathbf{u}_t$ that tracks the total tangential displacement of the initial contact points on the two quasi-rigid bodies, projected onto the current contact plane, as shown in Fig. 3.

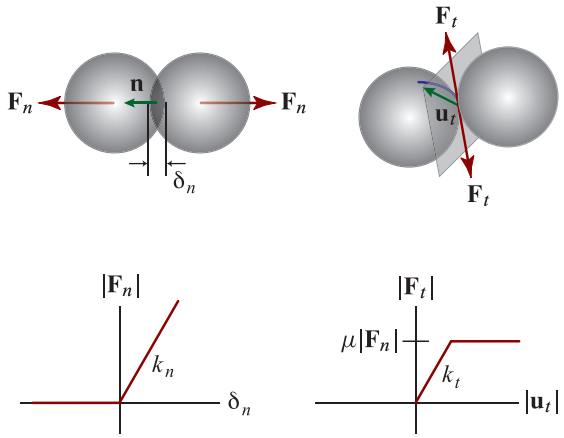


Fig. 3: DEM-P contact model, with normal overlap distance δ_n, contact-normal unit vector $\mathbf{n}$, and tangential displacement vector $\mathbf{u}_t$ in the plane of contact (top), and a Hookean-linear contact force-displacement law with constant Coulomb sliding friction (bottom).

An example of a DEM-P contact constitutive law, a slightly modified form of which is used in Chrono, is the following viscoelastic model based on either Hookean or Hertzian contact theory:

$$\begin{aligned} \mathbf{F}_n &= f(\bar{R}, \delta_n)\,(k_n \mathbf{u}_n - \gamma_n \bar{m} \mathbf{v}_n) \\ \mathbf{F}_t &= f(\bar{R}, \delta_n)\,(-k_t \mathbf{u}_t - \gamma_t \bar{m} \mathbf{v}_t)\ , \end{aligned} \tag{2}$$

where $\mathbf{u} = \mathbf{u}_n + \mathbf{u}_t$ is the overlap or local contact displacement of two interacting bodies. The quantities $\bar{m} = m_i m_j/(m_i + m_j)$ and $\bar{R} = R_i R_j/(R_i + R_j)$ represent the effective mass and effective radius of curvature, respectively, for contacting bodies with masses m_i and m_j and contact radii of curvature R_i and R_j. The vectors $\mathbf{v}_n$ and $\mathbf{v}_t$ are the normal and tangential components of the relative velocity at the contact point. For Hookean contact, $f(\bar{R}, \delta_n) = 1$ in Eq. (2); for Hertzian contact, one can let $f(\bar{R}, \delta_n) = \sqrt{\bar{R}\delta_n}$ [**?**, **?**, **?**]. The normal and tangential stiffness and damping coefficients k_n, k_t, γ_n, and γ_t are obtained, through various constitutive laws derived from contact mechanics, from physically measurable properties for the materials of the contacting bodies, such as Young's

modulus, Poisson's ratio, the coefficient of restitution, etc. Detailed descriptions of the DEM-P contact models implemented in Chrono, as well as alternative DEM-P contact models, are provided in [**?**].

The component of the contact displacement vector $\mathbf{u}$ in the contact-normal direction, $\mathbf{u}_n = \delta_n \mathbf{n}$, is obtained directly from the contact detection algorithm, which provides the magnitude of the "inter-penetration" δ_n between the bodies. The tangential contact displacement vector $\mathbf{u}_t$ is formulated as

$$\mathbf{u}_t = \int_{t_0}^{t} \mathbf{v}_t dt - \left(\mathbf{n} \cdot \int_{t_0}^{t} \mathbf{v}_t \, dt \right) \mathbf{n} \,, \tag{3}$$

where t is the current time and t_0 is the time at the initiation of contact [**?**]. For the true tangential contact displacement history model, the vector $\mathbf{u}_t$ must be stored and updated at each time step for each contact point on a given pair of contacting bodies from the time that contact is initiated until that contact is broken. On the importance of tangential contact displacement history in DEM-P contact models, see [**?**].

To enforce the Coulomb friction law, if $|\mathbf{F}_t| > \mu |\mathbf{F}_n|$ at any given time step, then before the contributions of the contact forces are added to the resultant force and torque on the body, the (stored) value of $|\mathbf{u}_t|$ is scaled so that $|\mathbf{F}_t| = \mu |\mathbf{F}_n|$, where μ is the Coulomb (static and sliding) friction coefficient. For example, if $f(x) = 1$ in Eq. (2), then

$$k_t |\mathbf{u}_t| > \mu |\mathbf{F}_n| \quad \Rightarrow \quad \mathbf{u}_t \leftarrow \mathbf{u}_t \frac{\mu |\mathbf{F}_n|}{k_t |\mathbf{u}_t|} \,. \tag{4}$$

Once the contact forces $\mathbf{F}_n$ and $\mathbf{F}_t$ are computed for each contact and their contributions are summed to obtain a resultant force and torque on each body in the system, the time evolution of each body in the system is obtained by integrating the Newton-Euler equations of motion, subject to the Courant-Friedrichs-Lewy (CFL) stability condition, which limits [**?**] the integration time step-size to $h < h_{\text{crit}} \sim \sqrt{m_{\min}/k_{\max}}$.

For multibody dynamics without frictional contact, or with frictional contact modeled using a penalty approach, Chrono implements index 3 DAE solutions [**?**,**?**]. For handling frictional contact within the differential variational inequality framework, Chrono implements a variant of the Nesterov algorithm [**?**]. Handling of cohesion is based on the an approach described in [**?**].

3 Flexible Body Dynamics Support in Chrono

Chrono implements flexible body dynamics support for problems in which bodies are expected to sustain large deformations that take place while the body might experience large translational and/or rotational accelerations. The current implementation does not support the so-called floating-frame-of-reference, modal, approach, in which the small deformation of the body is superimposed on top of a large reference motion, see for instance [**?**]. For large deformations,

Chrono currently resorts to the Absolute Nodal Coordinate Formulation (ANCF) for structural elements such as beams, plates, and shells and a corotational (CR) approach for both structural and volumetric elements.

3.1 Nonlinear Finite Element Analysis via the Absolute Nodal Coordinate Formulation

ANCF has proven to be successful in solving various challenging engineering problems of complex multibody systems including cables, belt drives, rotor blades, leaf springs, tires and many others [**?**,**?**,**?**,**?**]. In this finite element formulation, the large rotation and deformation of the element are parameterized by the global position vector gradients, and no rotational nodal coordinates such as Euler angles are utilized. This parameterization leads to a constant mass matrix for fully nonlinear dynamic problems while ensuring the exact modeling of rigid body reference motion [**?**]. Using the polar decomposition theorem, the displacement gradient tensor can be decomposed into the orthogonal rotation matrix and the stretch tensor that describes the most general six deformation modes. That is, use of the position vector gradient coordinates allows for describing the rotation and deformation within the element, thereby circumventing the complex nonlinear coupling of the rotation and deformation coordinates that appears in the inertia terms of flexible body models using rotational parameterization. The constant mass matrix of large deformable bodies not only leads to efficient solutions in nonlinear dynamics simulation, but also allows for the use of the non-incremental solution procedures utilized in general multibody dynamics computer algorithms. Using these important features and general motion description employed in ANCF, the structural beam and plate/shell elements can be implemented in the general multibody dynamics computer algorithms of Chrono without resorting to ad hoc co-simulation procedures. In what follows, the beam and plate/shell elements implemented in Chrono are summarized and important features of these elements are highlighted.

Thin Beam Element. The beam element implemented in Chrono is suited for modeling thin beam structures such as cables and belt drives, in which the transverse shearing effect of the beam cross section is considered negligible. The global position vector of the two-node ANCF Euler-Bernoulli beam element i on the centerline is defined by [**?**]

$$\mathbf{r}^i = \mathbf{S}^i(x^i)\mathbf{e}^i \ , \tag{5}$$

where $\mathbf{S}^i$ is the element shape function matrix obtained by the cubic Hermite polynomial [**?**], x^i is the element axial coordinate, and $\mathbf{e}^i$ is the nodal coordinate vector of element i. Node j of element i has the global position vector $\mathbf{r}^{ij}$ and global position vector gradient vector $\partial\mathbf{r}^{ij}/\partial x^i$ that is tangent to the beam centerline as follows [**?**]:

$$\mathbf{e}^{ij} = \begin{bmatrix} \left(\mathbf{r}^{ij}\right)^T & \left(\dfrac{\partial \mathbf{r}^{ij}}{\partial x^i}\right)^T \end{bmatrix}^T . \tag{6}$$

Using the Euler-Bernoulli beam theory (i.e., the beam cross section remains planar and normal to the bean centerline), the virtual work of the elastic forces can be obtained as

$$\delta W^i = EA \int_{x^i} \delta\varepsilon^i \varepsilon^i dx^i + EI \int_{x^i} \delta\kappa^i \kappa^i dx^i = \delta \mathbf{e}^{i^T} \mathbf{Q}_k^i \ , \tag{7}$$

where EA and EI are, respectively, the axial and flexural rigidity; ε^i and k^i are the axial strain and the curvature, respectively [**?**]. Notice here that this element does not account for pure torsion since rotation about the gradient vector is not considered. For this reason, this element is also called cable element [**?**, **?**]. Using the principle of virtual work in dynamics, the equations of motion of the ANCF beam element are written as

$$\mathbf{M}^i \ddot{\mathbf{e}}^i = \mathbf{Q}_k^i + \mathbf{Q}_e^i \ , \tag{8}$$

where $\mathbf{Q}_k^i$ is the vector of generalized element elastic forces, $\mathbf{Q}_e^i$ is the vector of generalized element external forces, and $\mathbf{M}^i$ is the constant element mass matrix defined by

$$\mathbf{M}^i = \int_{x^i} \rho^i A^i (\mathbf{S}^i)^T \mathbf{S}^i dx^i \ , \tag{9}$$

where ρ^i is the material density and A^i is the cross section area.

Thin Plate Element. There are two types of plate/shell elements in Chrono; the 4-node Kirchhoff thin plate element and the 4-node shear deformable shell element. In the thin plate element, the global position vector of a point in the middle plane of the plate element is defined by

$$\mathbf{r}^i = \mathbf{S}^i(x^i, y^i)\mathbf{e}^i \ , \tag{10}$$

where $\mathbf{S}^i(x^i, y^i)$ is the element shape function matrix obtained by the incomplete cubic polynomials [**?**, **?**]; x^i and y^i are the element coordinates in the middle plane. The nodal coordinates are defined as

$$\mathbf{e}^{ij} = \left[\left(\mathbf{r}^{ij}\right)^T \quad \left(\frac{\partial \mathbf{r}^{ij}}{\partial x^i}\right)^T \quad \left(\frac{\partial \mathbf{r}^{ij}}{\partial y^i}\right)^T \right]^T \ . \tag{11}$$

Using the Kirchhoff-Love plate theory (i.e., the plate section remains planar and normal to the middle surface), the virtual work of the elastic forces can be obtained as [**?**, **?**]

$$\delta W^i = \int_{V^i} \delta\boldsymbol{\varepsilon}^{i^T} \mathbf{D}^i \boldsymbol{\varepsilon}^i dV^i + \int_{A^i} \delta\boldsymbol{\kappa}^{i^T} \mathbf{D}_b^i \boldsymbol{\kappa}^i dA^i = \delta \mathbf{e}^{i^T} \mathbf{Q}_k^i \ , \tag{12}$$

where $\boldsymbol{\varepsilon}^i = [\varepsilon^i_{xx} \quad \varepsilon^i_{yy} \quad \gamma^i_{xy}]^T$; $\boldsymbol{\kappa}^i = [\kappa^i_{xx} \quad \kappa^i_{yy} \quad 2\kappa^i_{xy}]^T$; $\mathbf{D}^i$ and $\mathbf{D}^i_b$ are the elasticity matrices [**?**, **?**]. The in-plane strains and curvatures are defined as

$$\begin{aligned}
\varepsilon^i_{xx} =& \frac{1}{2}\left(\left(\frac{\partial \mathbf{r}^i}{\partial x^i}\right)^T\left(\frac{\partial \mathbf{r}^i}{\partial x^i}\right) - 1\right) \\
\varepsilon^i_{yy} =& \frac{1}{2}\left(\left(\frac{\partial \mathbf{r}^i}{\partial y^i}\right)^T\left(\frac{\partial \mathbf{r}^i}{\partial y^i}\right) - 1\right) \\
\gamma^i_{xy} =& \left(\frac{\partial \mathbf{r}^i}{\partial x^i}\right)^T\left(\frac{\partial \mathbf{r}^i}{\partial y^i}\right)
\end{aligned} \tag{13}$$

and

$$\kappa^i_{xx} = \mathbf{n}^{i^T}\frac{\partial^2 \mathbf{r}^i}{\partial {x^i}^2}, \quad \kappa^i_{yy} = \mathbf{n}^{i^T}\frac{\partial^2 \mathbf{r}^i}{\partial {y^i}^2}, \quad \kappa^i_{xy} = \mathbf{n}^{i^T}\frac{\partial^2 \mathbf{r}^i}{\partial x^i \partial y^i}, \tag{14}$$

where the vector $\mathbf{n}^i$ is a unit normal to the middle plane defined by $(\frac{\partial \mathbf{r}^i}{\partial x^i} \times \frac{\partial \mathbf{r}^i}{\partial y^i})/|\frac{\partial \mathbf{r}^i}{\partial x^i} \times \frac{\partial \mathbf{r}^i}{\partial y^i}|$.

Shear Deformable Shell Element. In the shear deformable shell element that can be applied to thick shell structures with various material models, the global position vector of an arbitrary point in element i is defined by

$$\mathbf{r}^i = \mathbf{S}^i(x^i, y^i, z^i)\mathbf{e}^i \,, \tag{15}$$

where $\mathbf{S}^i$ is the element shape function matrix obtained by the bi-linear polynomials [**?**, **?**]. The preceding equation can be expressed as a sum of the displacement on the middle surface $\mathbf{r}^i_m$ and the displacement on the cross section as $\mathbf{r}^i = \mathbf{r}^i_m(x^i, y^i) + z^i \partial \mathbf{r}^i / \partial z^i(x^i, y^i)$. z^i is the element coordinate along the shell thickness. The nodal coordinates are defined as [**?**, **?**].

$$\mathbf{e}^{ij} = \left[\left(\mathbf{r}^{ij}\right)^T \quad \left(\frac{\partial \mathbf{r}^{ij}}{\partial z^i}\right)^T\right]^T. \tag{16}$$

The virtual work of the elastic forces can then be obtained as [**?**]

$$\delta W^i = \int_{V^i_0} \delta {\boldsymbol{\varepsilon}^i}^T \frac{\partial W^i(\hat{\boldsymbol{\varepsilon}}^i)}{\partial \boldsymbol{\varepsilon}^i} dV^i_0 = \delta {\mathbf{e}^i}^T \mathbf{Q}^i_k \,, \tag{17}$$

where dV^i_0 is the infinitesimal volume at the initially curved reference configuration of element i, and W^i is an elastic energy density function. Due to the element lockings exhibited in this element, locking remedies need to be introduced to ensure the element convergence and accuracy. The lockings in the bi-linear shell element include the transverse shear locking; Poisson's thickness locking; curvature thickness locking; and in-plane shear locking. These lockings are systematically eliminated by applying the assumed natural strain method [**?**, **?**] and the enhanced assumed strain method [**?**, **?**]. By applying these locking remedies, the Green-Lagrange strain vector is defined as

$$\hat{\boldsymbol{\varepsilon}}^i = (\mathbf{T}^i)^{-T}\tilde{\boldsymbol{\varepsilon}}^i + \boldsymbol{\varepsilon}^{i,EAS} \,, \tag{18}$$

where $\tilde{\boldsymbol{\varepsilon}}^i$ is the covariant strain vector obtained from the covariant strain tensor given by:

$$\tilde{\mathbf{E}}^i = \frac{1}{2}\big((\bar{\mathbf{J}}^i)^T\bar{\mathbf{J}}^i - (\mathbf{J}^i)^T\mathbf{J}^i\big)\ . \tag{19}$$

In the preceding equation, $\bar{\mathbf{J}}^i = \partial\mathbf{r}^i/\partial\mathbf{x}^i$, $\mathbf{J}^i = \partial\mathbf{X}^i/\partial\mathbf{x}^i$ and $\mathbf{x}^i = [x^i \quad y^i \quad z^i]$, and $\mathbf{X}^i$ represents the global position vector of element i at an initially curved reference configuration. The transformation matrix $\mathbf{T}^i$ is as given in literature [**?**]. The assumed strain approach is introduced to the covariant transverse normal and transverse shear strains as follows:

$$\tilde{\boldsymbol{\varepsilon}}^i = [\tilde{\varepsilon}_{xx} \quad \tilde{\varepsilon}_{yy} \quad \tilde{\gamma}_{xy} \quad \tilde{\varepsilon}_{zz}^{ANS} \quad \tilde{\gamma}_{xz}^{ANS} \quad \tilde{\gamma}_{yz}^{ANS}]^T\ . \tag{20}$$

The enhanced assumed strain method is applied to the in-plain strains and transverse normal strain as follows:

$$\boldsymbol{\varepsilon}^{i,EAS} = [\varepsilon_{xx}^{EAS} \quad \varepsilon_{yy}^{EAS} \quad \gamma_{xy}^{EAS} \quad \varepsilon_{zz}^{EAS} \quad 0 \quad 0]^T\ . \tag{21}$$

It is important to note here that nonlinear constitutive models can be considered in a way same as solid elements. Using the principle of virtual work in dynamics, the equations of motion of the shear deformable shell element i can be expressed as

$$\mathbf{M}^i\ddot{\mathbf{e}}^i = \mathbf{Q}_k^i(\mathbf{e}^i, \boldsymbol{\alpha}^i) + \mathbf{Q}_e^i(\mathbf{e}^i, \dot{\mathbf{e}}^i, t)\ , \tag{22}$$

where vectors $\mathbf{Q}_k^i$ and $\mathbf{Q}_e^i$ are, respectively, vectors of the element elastic forces and external forces; and the matrix $\mathbf{M}^i$ is the constant element mass matrix defined by [**?**]

$$\mathbf{M}^i = \int_{V_0^i} \rho_0^i (\mathbf{S}^i)^T\mathbf{S}^i dV_0^i\ , \tag{23}$$

where ρ_0^{ik} is the material density at the reference configuration. The internal parameters $\boldsymbol{\alpha}^i$ in Eq. 22, which are introduced to define the enhanced assumed strain field, are determined by solving the following equations [**?**, **?**]

$$\int_{V_0^i} \left(\frac{\partial\boldsymbol{\varepsilon}^{i,EAS}}{\partial\boldsymbol{\alpha}^i}\right)^T \frac{\partial W^i(\hat{\boldsymbol{\varepsilon}}^i)}{\partial\boldsymbol{\varepsilon}^i} dV_0^i = \mathbf{0}\ . \tag{24}$$

The equations above can be solved at element level for the unknown internal parameters using the procedure presented in the literature [**?**].

3.2 Nonlinear Finite Element Analysis via the Corotational Approach

The CR approach, see for instance [**?**, **?**, **?**], can be regarded as an augmentation of the classic linear finite element analysis, of whom it inherits the fundamental functions for computing the stiffness matrices and the internal forces. This fosters the reuse of finite element algorithms and theories whose behavior in the linear field are already well known and tested. Yet, the paradigm reuse comes at a cost:

although displacements and rotations can be arbitrarily large, the CR framework requires that the strains must be small. In fact the CR concept is based on the idea that large deformations in beams, shells, etc., can be seen as local rigid body motions of finite elements, to whom small deformations can be superimposed - hence the possibility of using linear FEA formulations locally for the co-rotated elements.

Chrono uses the CR approach to model large deformations in meshes of 3D elements such as tetrahedrons and hexahedrons, as well as in beams discretized with classical elements such as Euler-Bernoulli beams. Fig. 4 shows the concept of the corotational formulation as implemented in Chrono. Although the figure shows a beam, it applies equally well to tetrahedrons and other elements. We introduce an auxiliary floating coordinate system F per each element, and require that the coordinate system follows the deformed element. For a proper choice of F position update, the overall gross motion from the undeformed state into the deformed state $\mathcal{C}_D$ can be seen as the superposition of a large rigid body motion from the reference configuration $\mathcal{C}_0$ to the so called *floating* or *shadow* configuration $\mathcal{C}_S$, plus a local small-strain deformation from $\mathcal{C}_S$ to $\mathcal{C}_D$.

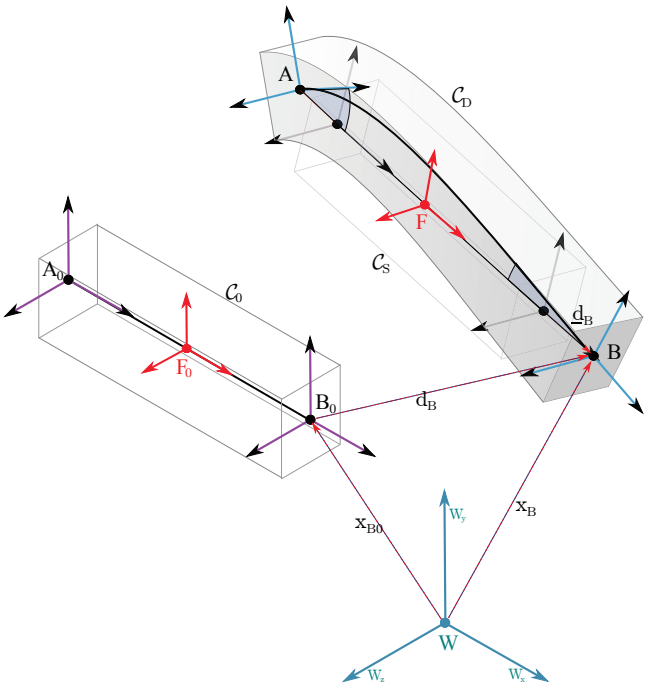


Fig. 4: The corotational finite element concept demonstrated in conjunction with a 3D beam element. For each finite element there is a floating frame F.

The idea of the corotational approach is that one can compute the global tangent stiffness $\boldsymbol{K}_e$ and a global force $\boldsymbol{f}_e$ for each element e, given its local $\underline{\boldsymbol{K}}$, its local $\underline{\boldsymbol{f}}$ and the rigid body motion of the frame F in $\mathcal{C}_0$ to F in $\mathcal{C}_S$. When the element moves, the position and rotation of F is updated. In the case of beams, to avoid dependence on connectivity we place the origin of F in the midpoint of the AB segment, as $\boldsymbol{x}_F = \frac{1}{2}(\boldsymbol{x}_B - \boldsymbol{x}_A)$, and align its $\boldsymbol{X}$ axis with $\boldsymbol{x}_B - \boldsymbol{x}_A$. The remaining $\boldsymbol{Y}$ and $\boldsymbol{Z}$ axes of F are obtained via a Gram-Schmidt orthogonalization, enforcing $\boldsymbol{Y}$ to bisect the $\boldsymbol{Y}$ axes of A and B when projected on the plane orthogonal to $\boldsymbol{X}$. In case of tetrahedrons, the F frame is placed in the barycenter of the tetrahedron, and the alignment of the three axes is obtained using a polar decomposition that minimizes the displacement of the nodes with respect to the rotated F as described in [**?**].

The matrix $\underline{\boldsymbol{K}}$ and the vector $\underline{\boldsymbol{f}}_{in}$ are evaluated using the classical theory for linear finite elements, since both are expressed in local coordinates. Then, both are transformed from the local to the global coordinates using the approach outlined in [**?**]. For the most part, this amounts to performing rotation transformations to the rows and columns corresponding to 3D nodes of $\underline{\boldsymbol{K}}$ and $\underline{\boldsymbol{f}}_{in}$, where the rotation matrix is the 3×3 matrix that contains the rotation of the floating coordinate system F. However, for completeness the mentioned approach also computes additional terms, especially the geometric stiffness matrix. Following [**?**] and [**?**] we also use projectors that filter the rigid body motion and that improve the consistency and the convergence of the method.

Fig. 5 pertains one of the benchmarks performed to validate the methodology. This is the so called Princeton beam experiment for which ample experimental results are available in the literature [**?**, **?**]. In this numerical experiment thin beams were each modeled with ten Euler-Bernoulli corotational beam elements. A force was applied to the tip, with increasing magnitude and inclination with respect to the vertical. Because of the diagonal nature of the force, beams are bent and slightly twisted. If a simple linear analysis were used, the twisting effect would not take place. Results, presented in Fig. 6, show that there is good agreement with experimental results and with other third party software; i.e., Dymore and MBDyn [**?**, **?**].

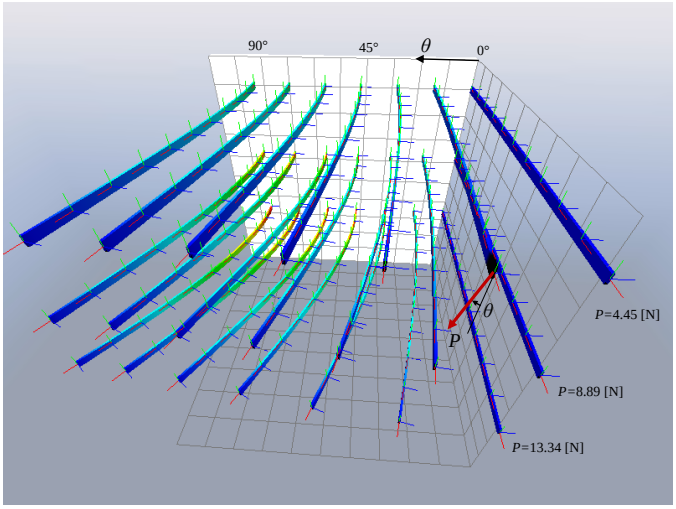


Fig. 5: Non-linear static analysis of the Princeton beam experiment, at different load magnitudes and θ angles.

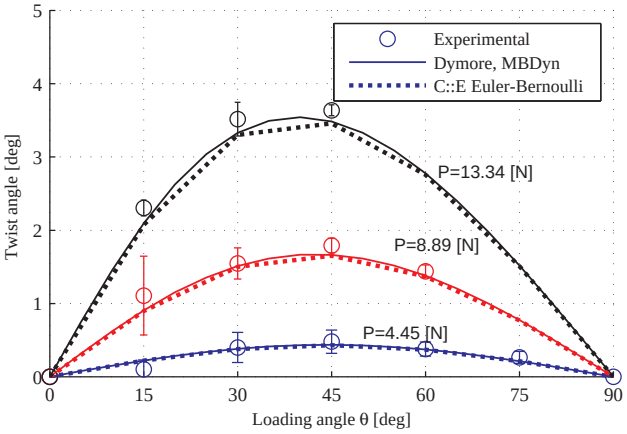


Fig. 6: Results from the non-linear analysis of the Princeton beam experiment.

4 Fluid–Solid Interaction Support in Chrono

In a Lagrangian framework, the continuity and momentum equations associated with the fluid dynamics assume the form of Ordinary Differential Equations (ODE) [**?**]

$$\frac{D\rho}{dt} = -\rho\nabla\cdot\mathbf{v} \tag{25a}$$

$$\frac{D\mathbf{v}}{dt} = -\frac{1}{\rho}\nabla p + \frac{\mu}{\rho}\nabla^2\mathbf{v} + \mathbf{f}\ , \tag{25b}$$

where μ is the fluid viscosity, ρ is fluid density, $\mathbf{v}$ and p are the flow velocity and pressure, respectively, and $\mathbf{f}$ is the body force. Assuming a Newtonian and incompressible flow, the first equation translates either in $\frac{D\rho}{dt} = 0$, or equivalently, imposing a divergence-free flow condition, in $\nabla\cdot\mathbf{v} = 0$.

The approach embraced in Chrono for the spatial discretization of the Navier-Stokes equations draws on the Smoothed Particle Hydrodynamics (SPH) methodology [**?**,**?**], a meshless method that dovetails well with the Lagrangian modeling perspective adopted for the dynamics of the solid phase. The term *smoothed* in SPH refers to the approximation of point properties via a smoothing kernel function W, defined over a support domain S. This approximation reproduces functions with up to second order accuracy, provided the kernel function: (i) approaches the Dirac delta function as the size of the support domain tends to zero, that is $\lim_{h\to 0} W(\mathbf{r},h) = \delta(\mathbf{r})$, where $\mathbf{r}$ is the spatial distance and h is a characteristic length that defines the kernel smoothness; (ii) is symmetric, i.e., $W(\mathbf{r},h) = W(-\mathbf{r},h)$; and (iii) is normal, i.e., $\int_S W(\mathbf{r},h)\mathrm{d}\mathbb{V} = 1$, where $\mathrm{d}\mathbb{V}$ denotes the differential volume. The term *particle* in SPH terminology indicates the discretization of the domain by a set of Lagrangian particles. To remove the ambiguity caused by the use of the term *rigid particles* in the context of FSI problems, the term *marker* is used herein to refer to the SPH discretization process. Each marker a has mass m_a associated with the representative volume $\mathrm{d}\mathbb{V}$ and carries all of the essential field properties. As a result, any field property at a certain location is shared and represented by the markers in the vicinity of that location [**?**]. Within this framework, Eqs. (25a) and (25b) are discretized at an arbitrary location $\mathbf{x}_a$ within the fluid domain as [**?**]:

$$\frac{d\rho_a}{dt} = \rho_a \sum_b \frac{m_b}{\rho_b}\left(\mathbf{v}_a - \mathbf{v}_b\right)\cdot\nabla_a W_{ab} \tag{26a}$$

$$\frac{d\mathbf{v}_a}{dt} = -\sum_b m_b\left[\left(\frac{p_a}{{\rho_a}^2} + \frac{p_b}{{\rho_b}^2}\right)\nabla_a W_{ab} - \frac{(\mu_a+\mu_b)\mathbf{x}_{ab}\cdot\nabla_a W_{ab}}{\bar{\rho}_{ab}^2(x_{ab}^2+\varepsilon h^2)}\mathbf{v}_{ab}\right] + \mathbf{f}_a\ , \tag{26b}$$

which are followed by

$$\frac{d\mathbf{x}_a}{dt} = \mathbf{v}_a\ , \tag{27}$$

to update the location of discretization markers. In the above equations, quantities with subscripts a and b are associated with markers a and b, respectively; $\mathbf{x}_{ab} = \mathbf{x}_a - \mathbf{x}_b$, $\mathbf{v}_{ab} = \mathbf{v}_a - \mathbf{v}_b$, $W_{ab} = W(\mathbf{x}_{ab}, h)$, $\bar{\rho}_{ab}$ is the average density of markers a and b, ∇_a is the gradient with respect to $\mathbf{x}_a$, i.e., $\partial/\partial\mathbf{x}_a$, and ε is a regularization coefficient to prevent infinite reaction between markers sharing the same location.

The current SPH implementation in Chrono relies on a weakly compressible model, namely Eq. (26a) followed by a equation of state to update pressure p [**?**]:

$$p = \frac{c_s^2 \rho_0}{\gamma} \left\{ \left(\frac{\rho}{\rho_0} \right)^{\gamma} - 1 \right\}. \tag{28}$$

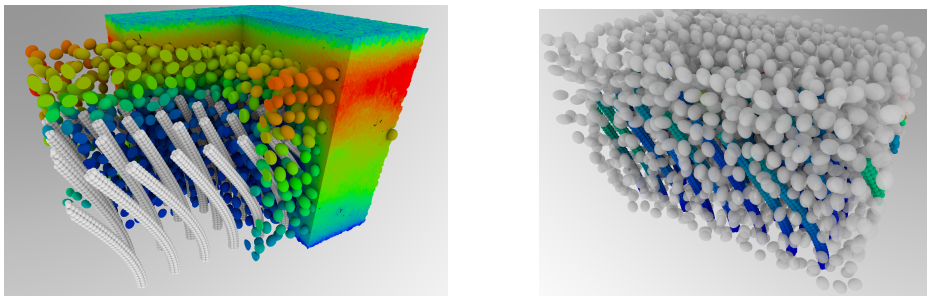
In this equation, ρ_0 is the reference density of the fluid, γ tunes the stiffness of the pressure-density relationship, and c_s is the speed of sound. The value c_s is adjusted depending on the maximum speed of the flow, $V_{\max}$, to keep the flow compressibility below some arbitrary value. Typically, $\gamma = 7$ and $c_s = 10V_{\max}$, which allows 1% flow compressibility [**?**].

Two additional refinements are adopted to improve the convergence properties of the weakly compressible SPH implementation. In the first adjustment, an extended SPH approach (XSPH) [**?**] is employed to modify the individual markers velocity based on collective; i.e., Eulerian, velocities. This regularization prevents excessive marker overlaps. The second modification is a re-initialization technique [**?**] which ensures consistency between marker densities updated through Eq. (26a) and those obtained directly from $\rho_a = \sum_b m_b W_{ab}$. The implementation details and algorithms can be found in [**?**, **?**].

Several methods are proposed to enforce a fixed or moving solid boundary [**?**, **?**, **?**, **?**, **?**]. In our previous work, we used an approach based on so-called Boundary Condition Enforcing (BCE) markers, which are distributed on the rigid [**?**] or flexible [**?**] bodies to capture the FSI interactions. At the fluid-solid interface, each BCE marker captures an interaction force due to its inclusion in the proximity of the nearby fluid markers through Eqs. (26). Two approaches were implemented to update the velocity and pressure of a BCE marker. In the first approach, the marker velocity is replaced by the local velocity of the moving boundary, while the pressure relies on a projection from the fluid domain [**?**]. In the second approach, the velocity of each BCE marker is calculated so that when combined with the fluid contribution, it results in the assigned wall velocity; additionally, BCE pressure is obtained from a force balance at the boundary. We showed that the second approach performs better in imposing the no-slip condition, particularly when the external body force is significant [**?**].

To achieve the computational efficiency required for complex multi-physics simulations, we adopted a parallel programming approach relying on GPU computing [**?**]. The resulting computational framework has been leveraged to investigate FSI problems in rigid body/particle suspension [**?**], three-way interaction of fluid, rigid, and flexible components [**?**], and microfluidic sorting of microtissues [**?**]. Fig. 7 shows a snapshot of a simulation that involves immersed rigid

and flexible components, where an ANCF method (see Section 3.1) is used to simulate the flexible beams.



(a) (b)

Fig. 7: GPU simulation of the flow of rigid bodies within an array of flexible beams. For a clear visualization, only parts of the domain are shown in each picture: (a) rigid bodies, flexible beams, and fluid flow; (b) rigid bodies and flexible beams only (fluid not rendered). The velocity field is color-coded: from zero (blue) to maximum (red), with $V_{max}^{fluid} = 0.045$ m/s, $V_{max}^{rigid} = 0.041$ m/s, and $V_{max}^{beam} = 0.005$ m/s.

5 Chrono Toolkits

A Chrono toolkit is a set of pre- and post-processing utilities that encapsulate domain expertise and provide a low entry point to Chrono by capturing expert knowledge in ready-to-use generic modeling templates in a focused research/application area. For instance, Chrono::Vehicle provides a collection of subsystem templates that can be quickly used to assemble a typical vehicle.

Chrono::Vehicle and Chrono::Granular are currently available and discussed herein. Chrono::Robotics and Chrono::Terramechanics are being developed.

5.1 Chrono::Vehicle

Available as an optional Chrono module, Chrono::Vehicle provides support for modeling, simulation, and visualization of ground vehicle systems. Modeling of vehicle systems is done in a modular fashion, with a vehicle defined as an assembly of instances of various subsystems (suspension, steering, driveline, etc.). Flexibility in modeling is provided by adopting a template-based design. In Chrono::Vehicle templates are parameterized models that define a particular implementation of a vehicle subsystem. As such, a template defines the basic modeling elements (bodies, joints, force elements), imposes the subsystem topology, prescribes the design parameters, and implements the common functionality for

a given type of subsystem (e.g., suspension) particularized to a specific template (e.g., double wishbone). The following vehicle subsystems and associated templates are available:

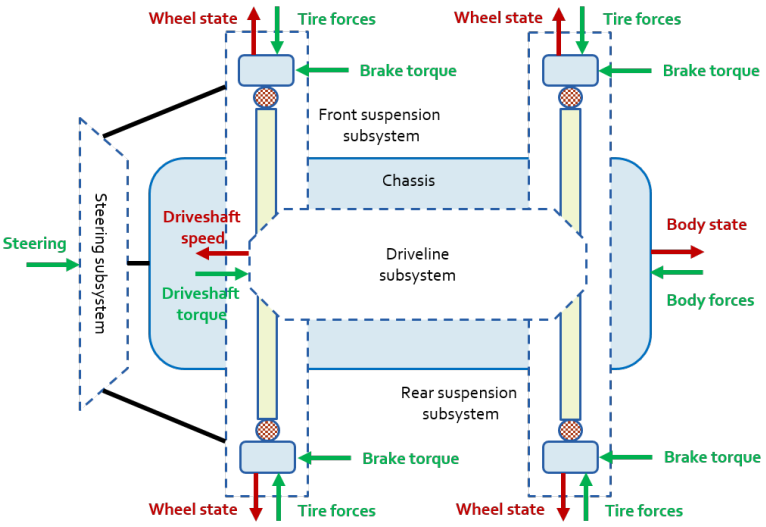
suspension: double wishbone, reduced double wishbone (with the A-arms modeled as distance constraints), multi-link, solid-axle, walking-beam;
steering: Pitman arm, rack-and-pinion;
anti-roll bar: two-body, rotational spring-damper-based anti-roll bar;
driveline: 2WD shaft-based, 4WD shaft-based; these templates are based on specialized Chrono modeling elements, named `ChShaft`, with a single rotational degree of freedom and various shaft coupling elements (gears, differentials, etc.);
wheel: in Chrono::Vehicle, a wheel only carries additional mass and inertia appended to the suspension's spindle body and, optionally, visualization information;
brake: simple brake (constant torque modulated by the driver braking input).

Fig. 8 illustrates the representation of a wheeled vehicle as an assembly of instances of various subsystem templates (here using double wishbone suspensions both in the front and rear and a Pitman-arm steering mechanism).

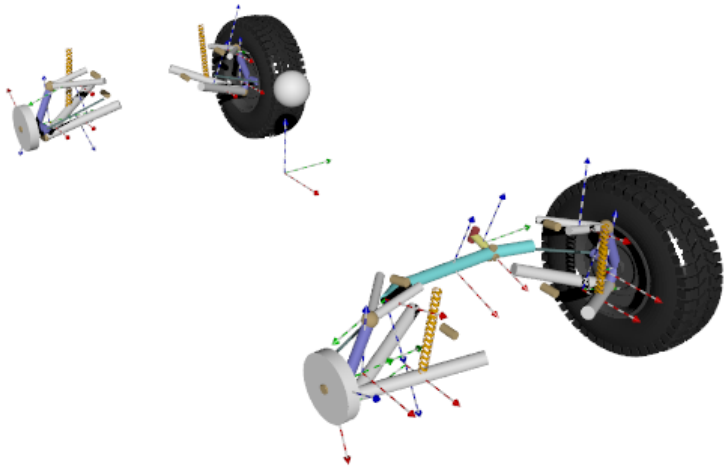
For additional flexibility and to facilitate inclusion in larger simulation frameworks, Chrono::Vehicle allows formally separating various systems (the vehicle itself, powertrain, tires, terrain, driver) and provides the inter-system communication API for a co-simulation framework based on force-displacement couplings. For consistency, these systems are themselves templatized:

vehicle: a collection of references used to instantiate templates for its constitutive subsystems;
powertrain: shaft-based template using an engine model based on speed-torque curves, torque converter based on capacity factor and torque ratio curves, and transmission parameterized by an arbitrary number of forward gear ratios and a single reverse gear ratio;
tire: rigid tire (based on the Chrono rigid contact model), Pacejka, and a LuGre friction tire model;
driver: interactive driver model (with user inputs from keyboard for real-time simulation), file-based driver model (interpolated driver inputs as functions of time).

Tire models supported in Chrono::Vehicle can be broken down into two categories: (*i*) a set of well developed traditional models, and (*ii*) a FEA-based tire model actively under development. In the first category, three tire models are currently supported: a rigid tire model, a tire model based on the LuGre friction model, and single contact point Pacejka tire model. The rigid tire model leverages the collision detection and frictional contact support in Chrono by using tires with user-specified geometry and contact material properties which are then treated like any other collision geometry within Chrono. The LuGre tire model described in [**?**] is based on a lumped brush-based LuGre friction model.



(a)



(b)

Fig. 8: A two-axle, independent suspension vehicle model: (a) diagram of the component subsystems in the vehicle assembly; (b) a possible realization in Chrono::Vehicle (in this particular case, using double wishbone suspensions, a Pitman-arm steering mechanism, and a 4WD driveline subsystem – the latter has no graphical representation in Chrono::Vehicle). The image on the left also illustrates the data exchange between the vehicle system and associated systems (powertrain, tires, etc.)

The tire is broken down into a user specified number of identical, connected two dimensional disks. For each disk, a normal force is calculated based on the disk–ground penetration distance and velocity; the lateral and longitudinal forces are calculated by integrating the differential equations for the friction force in each direction with no coupling between the lateral and longitudinal directions. The forces are then summed across all the disks and the resulting force is then transformed and applied to the wheel center. The third traditional tire model is a Pacejka-based formulation described in [**?**]. This model is a modification of the 2013 ADAMS/Tire PAC2002 non-linear transient tire model without belt dynamics, valid for tire responses up to approximately 15Hz. In the second category, a high-fidelity deformable tire model based on ANCF (see Section 3.1) is implemented in Chrono. The fiber reinforced rubber material of tires is modeled by the laminated composite shell element, and the distributed parameter LuGre tire friction model is utilized to allow for an accurate prediction of the shear contact stress distribution over the contact patch under various vehicle maneuvering scenarios [**?**].

For easy incorporation in and interoperability with third-party applications, Chrono::Vehicle is provided as a middleware library. System and subsystem templates are implemented through polymorphic classes and thus the library is extensible through C++ inheritance, allowing definition of new subsystems or new templates for existing subsystems. Systems and subsystems are defined with a three-layered C++ class hierarchy: (*i*) base abstract class for type (e.g., `ChSuspension`); (*ii*) derived, still abstract, class for specific template (e.g., `ChDoubleWishbone`); and (*iii*) concrete class that particularizes the template for a given vehicle (e.g., `HMMWV_DoubleWishboneFront`). We provide two different types for the concrete classes that specify a template. The first one, providing maximum flexibility, is through *user-defined* classes which implement all virtual functions imposed by the corresponding template base class. While not part of the Chrono::Vehicle library itself, several examples of this approach are provided with the package. A more convenient approach, allowing for fast experimentation and parametric studies, is offered through a set of concrete template classes (part of the Chrono::Vehicle library) that specify the subsystem for a particular vehicle through specification data files in the JSON format [**?**].

Visualization of vehicle systems is provided both for run-time, interactive simulation (through the Chrono built-in Irrlicht visualization support) and for high-quality post-processing rendering (using for example the POV-Ray ray-tracing package, see Fig. 9).

Currently only wheeled vehicles are supported, but work is underway for extending Chrono::Vehicle to include templates appropriate for modeling tracked vehicles.

5.2 Chrono::Granular

To facilitate the accurate modeling of particulate or granular materials used in Chrono simulations, a module called Chrono::Granular is available. This module provides pre-processing and post-processing tools specific to granular materials,

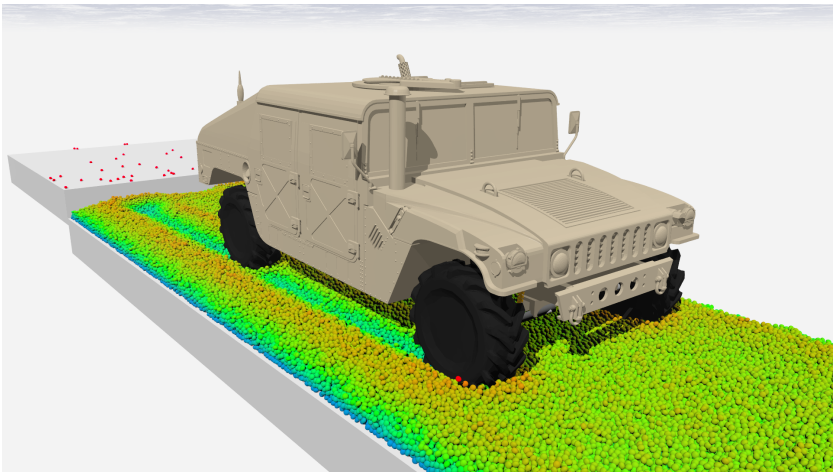


Fig. 9: A Chrono::Vehicle simulation of a HMMWV vehicle operating on granular terrain composed of over 150 000 bodies. Each granular particle is an ellipsoid that can be circumscribed in a sphere with a radius of 2 cm. The solver uses the complementarity form of contact with a time step of 0.001 seconds and 40 APGD iterations per step. The simulation took approximately 3.2 computation hours per second on a 3 GHz Intel Xeon E5-2690v2 processor using 20 cores.

as well as C++ templates for a variety of material tests, including the standard geomechanics tests (e.g., direct shear, triaxial) to determine elastic and plastic properties of the bulk granular material, as well as standard industrial processes (e.g., hopper flow, conveyor transport). Chrono::Granular can be used on its own or in conjunction with other modules, such as Chrono::Vehicle, e.g., in scenarios where vehicle-ground interactions are critical and where the ground is a granular material, such as sand, soil, or gravel.

The pre-processing tools provided by Chrono::Granular include the specification of granular materials with either user-definable or preset particle size and shape distributions, e.g., for ASTM standard graded sand; and various bulk geometries, e.g., cubical for the direct shear box test or cylindrical for the standard triaxial test. Once specified, granular material specimens can be subjected to the standard tests provided by the Chrono::Granular templates, to determine bulk elastic and plastic properties, such as Young's modulus, Poisson's ratio, and the friction and dilation angles of the bulk granular material. These bulk material properties for the simulated granular material are extracted from the simulations by the templates, and they can be compared to the bulk physical properties of the granular material that the user wishes to model for validation before subsequent Chrono simulations are run.

In addition to the determination of macro-scale or bulk granular material properties, Chrono::Granular also provides post-processing tools for the determination and visualization of micro-scale or local granular material properties,

including particle trajectories, inter-particle force chains, and the local stress distribution [?], defined as $\sigma_{ij} = \left(\sum_c f_i^c l_j^c\right) / V_\sigma$, where σ_{ij} is the average local stress tensor over a representative volume including at least two particles, f_i^c is the contact force vector, and l_j^c is the branch vector connecting the centers of contacting pairs, as shown in Fig. 10. The sum is over the contacts between the particles within the representative volume, and V_σ is the volume of the region containing those particles.

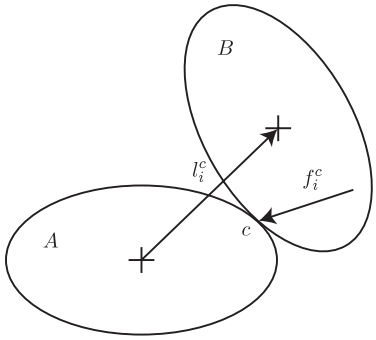


Fig. 10: An inter-particle contact c between two particles within a granular material, with the branch vector l_i^c and the contact force vector f_i^c shown.

6 Chrono Validation Studies

Validation against ADAMS. To validate basic modeling elements, a series of systems were created within Chrono. Multiple models were generated to test different aspects of each modeling element examined: revolute, spherical, universal, prismatic, and cylindrical joints; distance and revolute-spherical joints; translational and rotational spring-dampers; and translational actuators. Since the majority of the basic test systems did not have closed form analytical solutions, equivalent models were constructed and solved within MSC.ADAMS. The simulation results between Chrono and ADAMS were compared for each model to ensure close agreement between the two programs. For all of the test models examined, the results between the simulation programs were in close alignment as described in [?]. As an example, consider one of the unit tests for the universal joint. In this test, a pendulum body, initially at rest and aligned along the $x = y$ line, is connected to the ground through a universal joint; the initial directions of the universal joint's cross are $[1/2, -1/2, \sqrt{2}/2]$ and $[-1/2, 1/2, \sqrt{2}/2]$. The comparison plots in Fig. 11 show that the maximum difference in angular acceleration between Chrono and ADAMS is on the order of 0.5% of the peak angular acceleration seen in the system. Although not shown, the results between Chrono and ADAMS would converge further if the solver tolerances were tightened beyond the settings used in this study.

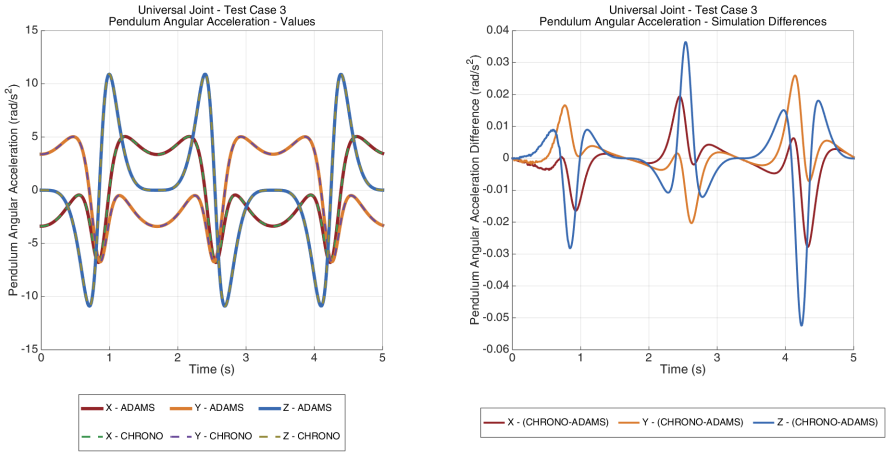


Fig. 11: Comparison of the pendulum angular acceleration in the global frame between CHRONO and ADAMS for the universal joint test system 3

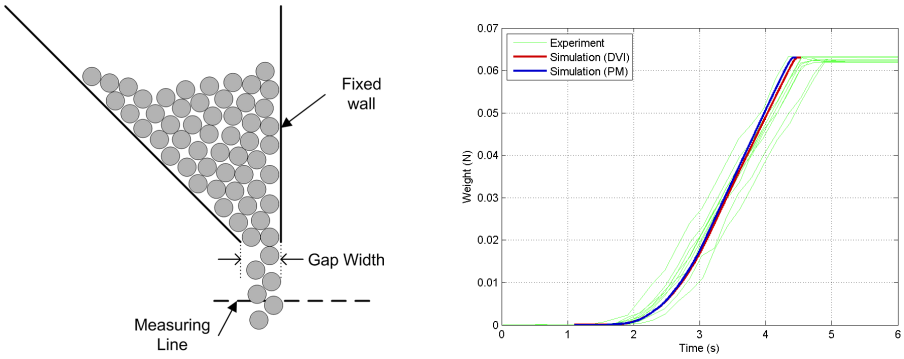
Validation against experimental data. The DEM-P and DEM-C contact approaches were validated against several fundamental terramechanics experiments. The first validation test, shown in Fig. 12a, used an aluminum rig designed and fabricated to measure the gravity-induced mass flow rate through a gap of a specified amount of granular material. The flow of the 500 micron diameter glass spheres was simulated for each gap size used in the lab experiments. Fig. 12b shows the simulation results plotted next to experimental measurements (weight as a function of time). Good statistical correlation was observed between simulation and experimental results for both DEM-P and DEM-C.

A second validation test, shown in Fig. 13, was run for an impact problem in a series of simulations reported in [**?**]. A relation of the form

$$d = \frac{0.14}{\mu}\left(\frac{\rho_b}{\rho_g}\right)^{1/2} D_b^{2/3} H^{1/3} \tag{29}$$

has been empirically established [**?**], where d is the depth of penetration, h is the height from which a ball of density ρ_b and diameter D_b is dropped, $H = d + h$ is the total drop distance, and ρ_g is the density of the granular material. Finally, μ was the friction coefficient in the granular material obtained from an angle of repose experiment as $\mu = \tan(\theta_r)$, where θ_r is the repose angle. In this test, the emergent behavior, i.e. the empirical equation above, is the result of the coordinated motion of 0.5 million bodies. Again, Chrono provided good statistical correlation between simulation and experimental results for both DEM-P and DEM-C, shown in Fig. 14.

Lastly, to verify that the Chrono DEM-P contact model with true tangential displacement history does indeed accurately model the micro-scale physics and emergent macro-scale properties of a simple granular material, we have used



(a) (b)

Fig. 12: Schematic of mass flow rate validation experiment in the open configuration (left). Weight of collected granular material vs. time for 2.0 mm gap size (right).

the templates of Chrono::Granular to simulate physical tests typical of the field of geomechanics; i.e., direct shear tests on a mono-disperse material, shown in Fig. 15. This third validation against experimental data, shown in Fig. 16, shows shear versus displacement curves obtained from both experimental [**?**] (top) and Chrono-simulated (bottom) direct shear tests, performed under constant normal stresses of 3.1, 6.4, 12.5, and 24.2 kPa, on $5,000$ uniform glass beads. The inside dimensions of the shear box were 12 cm in length by 12 cm in width, and the height of the granular material specimen was approximately 6 cm. In both the experimental and simulated direct shear tests, the glass spheres had a uniform diameter of 6 mm, and the random packing of $5,000$ spheres was initially obtained by a "rainfall" method, after which the spheres were compacted by the confining normal stress without adjusting the inter-particle friction coefficient. The DEM-P simulations were performed in Chrono using a Hertzian normal contact force model and true tangential contact displacement history with Coulomb friction. The material properties of the spheres in the simulations were taken to be those corresponding to glass [**?**], for which the density is $2,550$ kg/m^3, the inter-particle friction coefficient is $\mu = 0.18$, Poisson's ratio is $\nu = 0.22$, and the elastic modulus is $E = 4 \times 10^{10}$ Pa, except that the elastic modulus was reduced by several orders of magnitude, to $E = 4 \times 10^7$ Pa, to ensure a stable simulation with a reasonable time integration step-size of $h = 10^{-5}$ s. The shear speed was 1 mm/s.

From Fig. 16, one can calculate the friction angle ϕ for the bulk granular material, which is the inverse tangent of the ratio of shear stress to normal stress at the initiation of yield (the peak friction angle ϕ_p) and post yield (the residual friction angle ϕ_r) during the direct shear test [**?**]. Specifically, for constant normal stresses of 3.1, 6.4, 12.5, and 24.2 kPa, the peak friction angles ϕ_p are

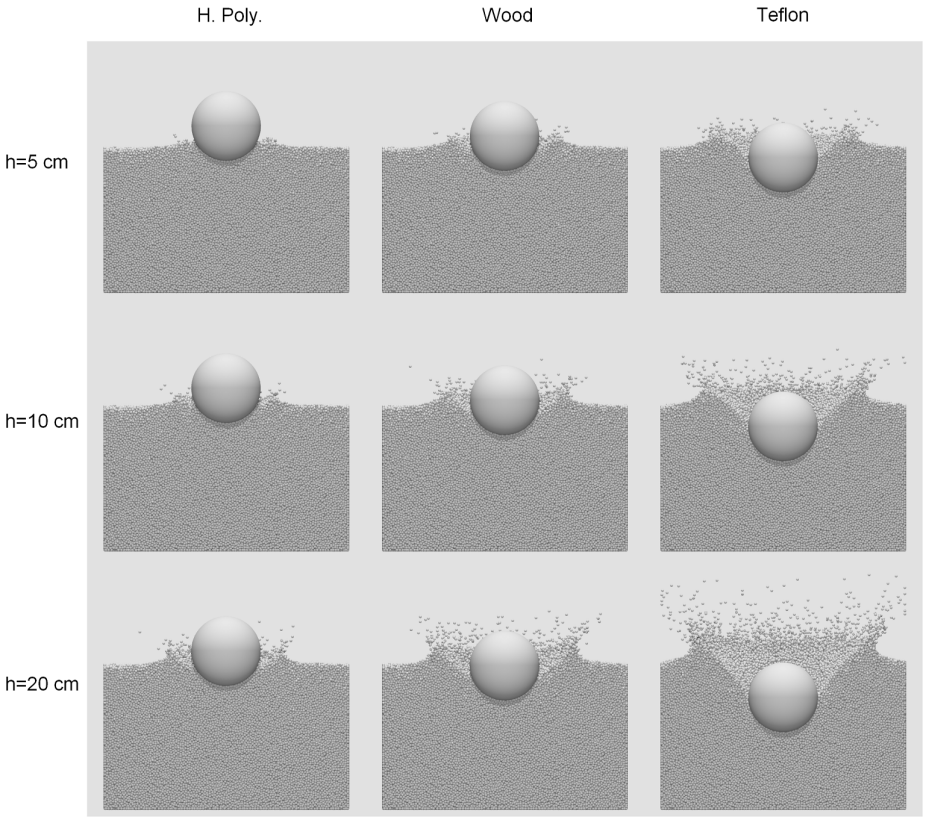
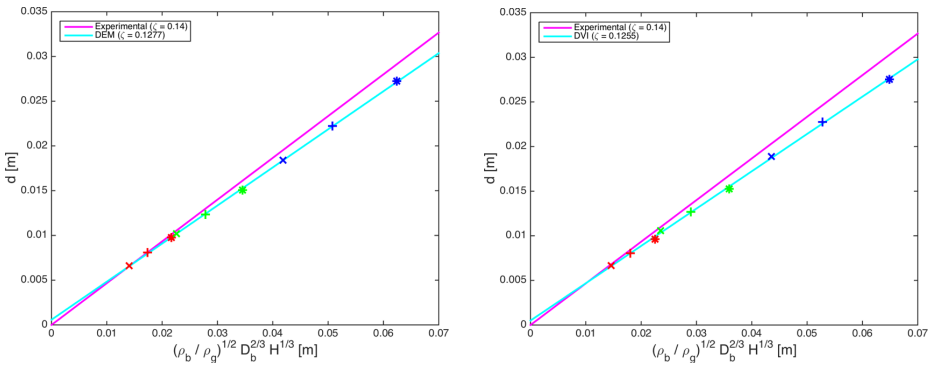


Fig. 13: Snapshot of the instant of deepest penetration from each impact simulation.



(a) (b)

Fig. 14: Penetration depth vs. scaled total drop distance for the DEM-P (left) and DEM-C (right) contact methods.

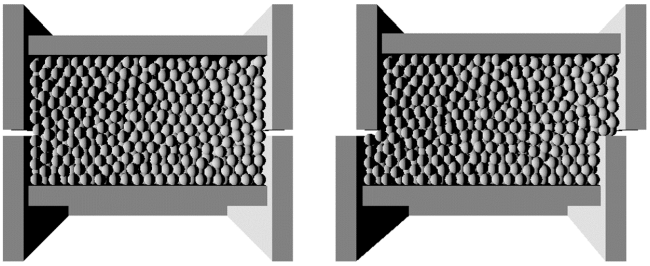


Fig. 15: Direct shear test simulations performed on 5,000 randomly packed uniform glass beads using Chrono, in initial (left) and final (right) positions.

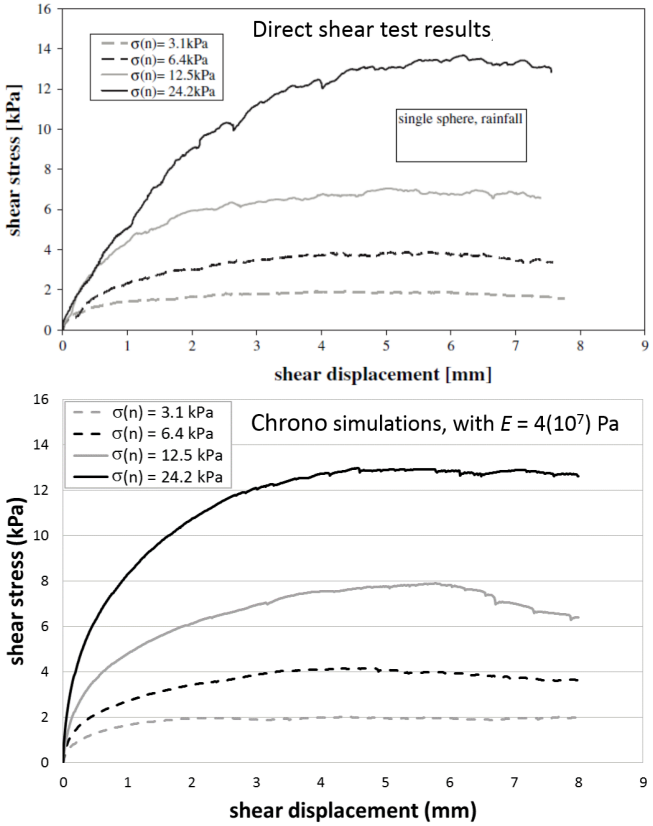


Fig. 16: Direct shear test results for 5,000 randomly packed uniform glass beads, obtained by experiment [**?**] (top) and DEM-P simulations using Chrono (bottom), under constant normal stresses of 3.1, 6.4, 12.5, and 24.2 kPa.

approximately 33, 32, 29, and 28 deg., respectively, for both the experimental and Chrono-simulated direct shear tests.

7 Conclusions and Future Work

Chrono is an ongoing open source software development project aimed at establishing a dynamics engine that is experimentally validated and which draws on parallel computing and a broad spectrum of modeling techniques to solve large problems of practical relevance. Chrono is used in a spectrum of applications in additive manufacturing, rheology, ground-vehicle interaction, soft-matter physics, and geomechanics. Its most salient attribute is the ability to solve multidisciplinary problems that require the solution of coupled differential algebraic equations, ordinary differential equations, partial differential equations and variational inequalities. Chrono has been validated for basic granular dynamics problems, fluid-solid interaction problems, and efforts are underway to validate it for ground-vehicle mobility problems. Chrono is distributed with several toolkits, such as Chrono::Vehicle and Chrono::Granular, which provide a low entry point for individuals who are interested in using the tool without understanding its implementation. For a focused application area, the toolkits attempt to encapsulate best-practice solutions through ready-to-use templates that can reduce modeling time and improve numerical solution robustness. Chrono is available on GitHub [**?**] and can be used under a very permissive license. More than 150 animations of Chrono simulations are available online at [**?**, **?**]. Looking ahead, effort is underway to improve Chrono in terms of (*i*) modeling prowess by building up the support for fluid dynamics and nonlinear finite element analysis; (*ii*) time to solution by leveraging parallel computing; and (*iii*) solution accuracy and robustness by embedding more refined numerical methods for solving differential equations and variational problems.

Acknowledgments

This work has been possible owing to US Army Research Office Rapid Innovation Funding grant W56HZV-14-C-0254, US National Science Foundation grant GOALI-CMMI 1362583, and US Army Research Office grant W911NF-12-1-0395. Milad Rakhsha is gratefully acknowledged for his help in the preparation of this manuscript.

References