Project Chrono

Overview, structure, capabilities
Project Chrono

- Growing ecosystem of software tools
- Multi-physics simulation engine

- **Open source**, released under permissive BSD-3 license

- Provides support for simulation of
  - Many-body dynamics
  - Nonlinear Finite Element Analysis
  - Fluid-Solid Interaction Problems
What is Chrono

- **Middleware**: can be embedded in third-party applications
- **Modular**: based on optional linking of specialized modules
- **Expandable**: via C++ inheritance
- **Efficient**: fast and robust data structures and algorithms
- **Cross-platform**: builds on Windows, Linux, OS X (MSVC, GCC, ICC, clang)
Modular and hierarchical structure

Chrono::Engine

Chrono MBD
- Equation Formulation
- Equation Solution
- Collision Detection
- HPC Support
- Pre/Post Processing

Chrono FEA
- Equation Solution

Chrono FSI
- FSI API

Support for Classical Multi-Body Dynamics

Support for Structural and Volumetric Elements

Support for Fluid-Solid Interaction

Future Chrono Expansion

Advanced Chrono Use

Low-Entry Point Chrono Use

Chrono API

Toolkit Chrono::GeoMech

Toolkit Chrono::Robotics

Toolkit Chrono::Vehicle

Toolkit Chrono::Granular

Toolkit Chrono::Fording

Hardware
- CPU, Multicore
- Multiple GPU
- Multiple Nodes

Hardware
- CPU, Multicore
- Multiple GPU
- Multiple Nodes
## Chrono modules

<table>
<thead>
<tr>
<th>Module</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chrono::Engine</td>
<td>Core Functionality</td>
</tr>
<tr>
<td>CASCADE</td>
<td>Interoperability with CAD tools</td>
</tr>
<tr>
<td>COSIMULATION</td>
<td>Support for co-simulation</td>
</tr>
<tr>
<td>FEA</td>
<td>Finite Element Analysis</td>
</tr>
<tr>
<td>FSI</td>
<td>Fluid-Solid Interaction</td>
</tr>
<tr>
<td>IRRLICHT</td>
<td>Runtime Visualization with Irrlicht</td>
</tr>
<tr>
<td>MATLAB</td>
<td>Interoperability with MATLAB</td>
</tr>
<tr>
<td>MKL</td>
<td>Interface to Intel MKL</td>
</tr>
<tr>
<td>OGRE</td>
<td>WIP: Runtime Visualization with Ogre</td>
</tr>
<tr>
<td>OPENGL</td>
<td>Runtime Visualization with OpenGL</td>
</tr>
<tr>
<td>PARALLEL</td>
<td>Parallel (multi-core) simulation module</td>
</tr>
<tr>
<td>POSTPROCESS</td>
<td>Tools for post-processing (POV-Ray output, Gnuplot)</td>
</tr>
<tr>
<td>PYTHON</td>
<td>Python Interoperability</td>
</tr>
<tr>
<td>VEHICLE</td>
<td>Template-based Ground Vehicle Modeling and Simulation</td>
</tr>
</tbody>
</table>
Modeling features

- Rigid **bodies, markers, forces, torques**
- **Springs** and **dampers**, with user-defined non-linear features
- Wide set of **joints**, e.g. spherical, revolute, prismatic, universal, etc.
- Impose **trajectories** to parts and markers
- **Constraint motion** on splines, surfaces, etc.
- Constraints can have **limits** (e.g. elbow joint)
Modeling features

- Custom constraint for motors, reducers etc.
- Custom constraint for linear motors.
- 1-DOF elements for powertrains, drivelines, etc.
- Brakes and clutches, with stick-slip effect
Modeling features

- Fast collision detection algorithms
- Collision families and groups
- Coulomb friction model, with stick-slip
- Rolling and spinning friction
- Restitution coefficients for rebouncing
- Collision detection between compound shapes
- Bodies activation/deactivation and sleeping
- Conveyor belts
Chrono::FEA module

• Co-rotational formulation
  • Bar element, Euler beam, Hexa8, Hexa20, Tetra4, Tetra10

• ANCF
  • Cable element, Shell element (isotropic, orthotropic, composite)

• Other
  • EAS brick element (isotropic and hyperelastic Mooney-Rivlin)

• Support for concentrated and distributed loads
  • Linear, surface, volumetric
  • Built-in classes for pressure, gravitational forces

• Support for constraints
  • Between two nodes, node and point on body, gradient and body direction

• Support for contact (penalty-based formulation)
  • Mesh-mesh and mesh-rigid
  • Surfaces represented as node clouds or triangular mesh
Chrono::Vehicle module

• Chrono vertical app (module) modeling, simulation, and visualization of wheeled ground vehicles and (soon) tracked vehicles

• Middleware: can be embedded in third parties software

• Modular: vehicle are modeled from instances of subsystems (suspension, steering, driveline, etc.)

• Flexible: use parameterized templates

• Expandable, via C++ inheritance
  • New subsystems
  • New templates for existing subsystems
  • New vehicle types

• Dependencies: Chrono::Engine and (optionally) the Chrono::Irrlicht and Chrono::FEA modules
Chrono::Parallel module

- Chrono vertical app (module)
  - library for OpenMP-based parallel simulation of Chrono models

- Middleware: can be embedded in third parties software

- Chrono-Parallel relies on Chrono for all its modeling capabilities

- Supports a subset of Chrono modeling elements:
  - Rigid bodies with frictional contact (DEM-C or DEM-P)
  - Kinematic joints (revolute, spherical, translational, etc.)
  - Force elements (spring-dampers, actuators, etc.)
  - 1-D shafts and associated elements and constraints (shaft-body connection, gears, motors, etc.)

- No support for FEA

- Implements only the *Implicit Euler Linearized* time-stepper

- Chrono-Parallel uses different data structures and algorithms
Chrono::FSI module

• Current State
  • Fluid interaction with multibody dynamics.
    • MBD includes contact and constraint
    • Supports flexible beam
  • Heterogeneous computing
    • GPU-based parallelism for fluid and OMP/AVX/SSE parallelism for MBD
  • Constraint-based fluid simulation
• Under development
  • Implicit incompressible CFD approach for fluid dynamics
  • Support for fluid interaction with flexible plate and shell
  • Distributed memory parallelism using Charm++
Code availability and documentation
Chrono source code

• Project Chrono GitHub repository
  https://github.com/projectchrono/chrono

• Clone/fork develop branch

• Planned major release: January 2017
Chrono dependencies and requirements

• C++ 11
  • Visual Studio 2013 or newer
  • GCC version 4.9 or newer

• Various Chrono modules have additional external dependencies
  • Chrono::Parallel: OpenMP, Blaze (v 2.4), Boost, Thrust
  • Chrono::OpenGL: GLEW, GLFW, GLM
  • Chrono::Python: SWIG, Python 3

• Build system based on CMake
Chrono documentation and support

• Project website: [http://projectchrono.org/](http://projectchrono.org/)

  - Installation guides
  - API documentation
  - Manuals, white papers, tutorials, etc.

• Support:
  - User mailing list (Google group): [http://projectchrono.org/forum/](http://projectchrono.org/forum/)
  - Bug tracking, issue tracking, and feature requests through GitHub