Manual for SPHERHARM user package

Release 0.1

Mohammad Imaran, Kevin J Hanley

Jan 19, 2023

CONTENTS:

0.1 Getting Started

This document describes the implementation of the DEM using Spherical Harmonic within the Large-scale Atomic/Molecular Massively Parallel Simulator (LAMMPS).

LAMMPS is a particle simulation code, developed and maintained at Sandia National Laboratories, USA. While is primarily aimed at Molecular Dynamics simulations of atomistic systems, it provides a general, fully parallelized framework for particle simulations governed by Newton's equations of motion.

0.2 SPHERHARM Theory

Spherical harmonics can be used to represent a closed star-like i.e., any line segment drawn from an origin, O, inside the particle crosses the contour of the particle's only once three-dimensional surface . This is achieved using both the spherical harmonic functions $Y_n^m(\theta, \phi)$ and the spherical harmonic coefficients a_{nm}:

$$r(\theta, \phi) = \sum_{n=0}^{\infty} \sum_{m=-n}^{n} a_{nm} Y_n^m(\theta, \phi) \approx r(N, \theta, \phi) = \sum_{n=0}^{N} \sum_{m=-n}^{n} a_{nm} Y_n^m(\theta, \phi)$$

where $r(\theta, \phi)$ is any smooth function defined on the unit sphere with $0 \leq \theta \leq \pi$ and $0 \leq \phi \leq 2\pi$.

In this case, $r(\theta, \phi)$ describes the particle radius which is measured from its centre of mass. In practice, the above is approximated by truncating the initial sum from infinity to θ. Spherical harmonic function, Y_n^m of a degree n and order m is given as

$$Y_n^m(\theta, \phi) = \sqrt{\frac{(2n+1)(n-m)!}{4\pi(n+m)!}} P_n^m(\cos(\theta)) e^{im\phi}$$

where $P_n^m(x)$ is the associated Legendre function, which can be calculated efficiently through the recurrence relations described by Press et al.@~cite{Press2007}.

Garboczi details how the coefficients a_{nm} can be extracted from a real particle using X-ray tomography and selecting radius measurement points that correspond with point of Gaussian quadrature such that can be solved, where the asterisk in the above denotes the complex conjugate .

$$a_{nm} = \int_0^{2\pi} \int_0^{\pi} d\phi d\theta \sin(\theta) r(\theta, \phi) Y_n^m(\theta, \phi)^*$$
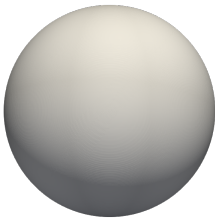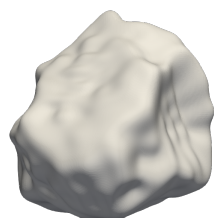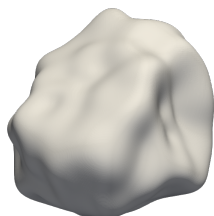
0.3 Implementation of SPHERHARM in LAMMPS

0.3.1 Atom style

Atom style defines the set of properties of an atom used during a simulation in communication and input-output operations. Unlike traditional atom styles in LAMMPS, spherical harmonic particles are constructed in the simulation by supplying the shape coefficients. Any instance of this atom style accesses the appropriate coefficients. Spherical harmonic representation of the particles requires the multiple per-shape variables: shape coefficients a_{nm}, principle inertia $I_{1,2,3}$, initial quaternion q_i, and degree of spherical harmonic expansion n. Modified set commands further link these per-shape variables to per-atom or per-group properties. In order to add support for these quantities, a new data structure was created, which can be accessed using the following command:

Syntax

```
atom_style spherharm N numerical_quad1 shape_coeff.dat
```

- ID, group-ID are documented in compute command
- erotate/sphere = style name of this compute command

Note: It should be noted that per-shape properties are not the same as the per-type properties. A spherical harmonic atom type can contain only one shape, although a shape can belong to multiple types of spherical harmonic particles.

0.3.2 Pair style

In LAMMPS, two-body and multi-body interactions are implemented as the inter-particle potentials. In the current implementation of the *SPHERHARM*, inter-particle interaction is calculated as a function of the overlapped volume between the particles [feng paper]. The particles are represented by their bounding spheres when building neighbour lists and checking for potential contacts. Calculation of overlap volume, in addition to other properties required by the contact theory, is handled through numerical integration over the spherical caps formed by the overlap of bounding spheres as detailed in section [ref section contact detection]. This is managed through the addition of standalone routines for spherical harmonic and quadrature functions, defined similarly to those in MathExtra. The commands for pair style are as follows:

Syntax

```
pair_style spherharm
pair_coeff I J Kn exponent numerical_quad2
```

0.3.3 Fix style

The dynamics of a model are implemented by the use of different fix styles in LAMMPS. It contains a list of commands to perform specified operations during the dynamic time step. Two variants of the fix are implemented in the *SPHERHARM* package. Due to additional properties of the spherical harmonic particles, i.e. quaternion, and shtype, a new method based on the Velocity Verlet algorithm for time-stepping has been added. It can be accessed by the following command.

fix NVE/sh

In the simulation of granular particles, walls are often used to describe the boundary condition or flow to a certain region in space. This boundary condition can be stationary as well as moving. LAMMPS offers various fix wall/* commands to implement such rigid boundaries. However, these walls can not be used with a particle defined using the spherharm atom style. The spherharm atom style necessitates using the spherharm pair style to compute the interaction forces. Thus the newly implemented walls calculate the repulsive force for a particle-wall interaction by following the same methodology as for particle-particle interactions. The tangential friction force is calculated by *calc_vel Coulomb_force_torque* method, based on the velocity-dependent Coulomb friction described in texttt{pair granular}. A suitable wall surface can be provided by

Syntax

```
fix ID group_ID wall/spherharm Kn exponent Kt wallstyle args keyword values
```

This wall style takes **translate** keyword and velocities :math: v_x, v_y, v_z as values to implement moving walls.

0.3.4 Compute styles

Compute styles in LAMMPS are used to calculate the properties of the system at different instances as they are invoked. They operate on specified groups or chunks of atoms, and they produce output which is stored internally for use by other commands. In *SPHERHARM* package, two compute styles are implemented to calculate the system's rotational kinetic energy o and potential energy. The following compute style is used to calculate the rotational kinetic energy o of a group of spherical harmonic particles.

```
compute ID group-ID erotate/spherharm
```

The rotational energy is computed as $E_{rot} = 0.5 * I.\omega^2$, where :math: I is the moment of inertia and ω is the particle's angular velocity. Similarly, another compute is implemented to calculate the potential energy of a particle group by calculating the incremental work done by the potential. This compute style can be invoked by

```
compute ID group-ID efunction/spherharm
```

The work done by the potential over a time step is a combination of that done from the "back half" $t- > t + dt/2$ and the "front half" $t + dt/2- > t + dt$ of the time step. At the end of the time step, i.e $t + dt$, it is not possible to calculate the "back half" of the work done, so this must be carried forward from the previous time step. These compute styles calculate global scalars and can be used by other commands that use a global scalar value from a compute as input.

0.4 Validation tests

0.4.1 Energy conservation test

The elastic collision between two irregularly shaped particles is used as the first benchmark test to verify and demonstrate the energy conservation of the spherical-harmonic-based contact detection algorithm and the volume-based energy-conserving contact model. Two identical and randomly oriented particles are generated with the spherical harmonic degree of expansion of $N=10$. Initially, these particles with equal and opposite velocities are separated by a smaller distance.

```
# Test

variable    name string spherical_harmonics_testing

atom_style  spherharm 1     120 A-anm-00001.dat
units               si
newton off
# newton on


##############################################
# Geometry-related parameters
##############################################

variable    boxx equal 100
variable    boxy equal 100
variable    boxz equal 100



############
processors * * 1
region              boxreg block 0 ${boxx} 0 ${boxy} 0 ${boxz}
```

```
create_box   2 boxreg
change_box   all boundary f f f

create_atoms        1 single 35.5 50.0 50.0
create_atoms        2 single 52.5 50.0 50.0

#mass *       1.0

set               atom 1 vx 1.0 vy 0.0 vz 0.0
set               atom 2 vx -1.0 vy 0.0 vz 0.0

set        atom 1 sh/shape 1
set        atom 2 sh/shape 1
set        group all sh/quat/random 4

fix        time_fix all nve/sh

neigh_modify       delay 0 every 1 check yes
comm_modify vel yes

thermo 1

compute TransKE all ke
compute RotKE all erotate/spherharm
compute PEcalc all efunction/spherharm
variable ETotal equal "c_PEcalc + c_TransKE + c_RotKE"
variable ERot equal "C_RotKE"
thermo_style custom step c_PEcalc c_TransKE c_RotKE v_ETotal

pair_style   spherharm
pair_coeff   * * 5.0e-6 2.0   30

timestep    1.0e-3
variable    step equal step
#fix    1 all ave/time 100 1 100 c_PEcalc c_TransKE c_RotKE v_ETotal  file energy_spher.
↪dat
#fix               2 all print 1 "${step} ${ETotal}" file energy_spher.dat screen no
run         1000
```

The normal stiffness coefficient k_n is fixed to allow a large overlap. Fig. 1 displays the total energy of the system over time, where the contact work done is the sum of the incremental contact work done:
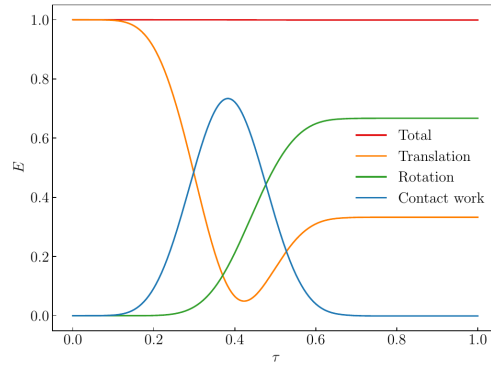
$$dW = -dt(F \cdot v + M \cdot \omega)$$

Absfsifdoa

Fig. 1: Plot of energy (total, translational, rotational, and contact work) over time for an impact and excessive overlap of two particles

0.5 Indices and tables

- genindex
- modindex
- search