The Journal of Open Source Software

PeriDEM – High-fidelity modeling of granular media consisting of deformable complex-shaped particles

Prashant Kumar Jha [1]

1 Department of Mechanical Engineering, South Dakota School of Mines and Technology, Rapid City, SD 57701, USA

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Open Journals

Reviewers:

- @openjournals

Submitted: 01 January 1970
Published: unpublished

Summary

Granular materials are crucial in various sectors, including geotechnical, manufacturing, and mining. Predictive modeling of these materials under extensive loading becomes challenging due to the deformation and breakage of particles and the complex contact mechanisms between complex-shaped particles undergoing considerable deformation. Focusing on the scenarios when particle deformation and breakage are crucial, the PeriDEM model introduced in (Prashant K. Jha et al., 2021) is implemented in the PeriDEM library. The underlying idea is that individual particles are modeled as deformable solids using peridynamics theory, and the contact between two deforming particles is applied locally at the contact region, allowing the modeling of complex-shaped particles. Integrating peridynamics within the discrete element method (DEM) provides a flexible, hybrid framework that handles the contact mechanics at the particle boundary while accounting for the internal material response, including deformation and fracture. This opens up new avenues for exploring the interactions in granular systems, including developing constitutive laws for phenomenological continuum models, understanding effective behavior when subjected to extensive loading, and the impact of particle shape on particle dynamics.

Statement of Need

As stated earlier, granular materials are prevalent in numerous industrial sectors, and the predictive modeling and simulation of these materials will play a fundamental role in designing processes. Current modeling techniques, such as DEM, are used widely, and numerous open-source libraries exist. However, the models based on DEM struggle to model particle deformation and breakage and accurately capture the behavior of granular materials under extreme conditions, especially when dealing with complex geometries and deformable particles. PeriDEM overcomes the challenges and implements a high-fidelity framework combining DEM and peridynamics to allow for accurate simulations of granular systems under extreme loading conditions. PeriDEM library makes the implementation of the high-fidelity approach transparent. The library depends on limited external libraries and is easier to build on Ubuntu and Mac systems, allowing quick testing and extension to user-specific needs.

Background

The PeriDEM model was introduced in (Prashant K. Jha et al., 2021), demonstrating its ability to model inter-particle contact and intra-particle fracture for complex-shaped particles. It is briefly described next.

The Journal of Open Source Software

Brief Introduction to PeriDEM Model

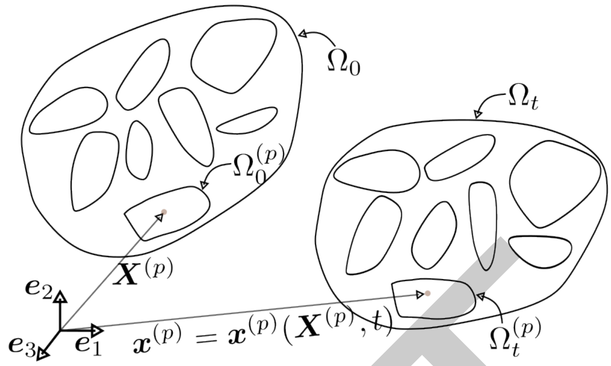

Figure 1: Motion of particle system.

Consider a fixed frame of reference and $\{e_i\}_{i=1}^d$ are orthonormal bases. Consider a collection of N_P particles $\Omega_0^{(p)}$, $1 \leq p \leq N_P$, where $\Omega_0^{(p)} \subset \mathbb{R}^d$ with $d = 2, 3$ represents the initial configuration of particle p. Suppose $\Omega_0 \supset \cup_{p=1}^{N_P} \Omega_0^{(p)}$ is the domain containing all particles; see Figure 1. The particles in Ω_0 are dynamically evolving due to external boundary conditions and internal interactions; let $\Omega_t^{(p)}$ denote the configuration of particle p at time $t \in (0, t_F]$, and $\Omega_t \supset \cup_{p=1}^{N_P} \Omega_t^{(p)}$ domain containing all particles at that time. The motion $x^{(p)} = x^{(p)}(X^{(p)}, t)$ takes point $X^{(p)} \in \Omega_0^{(p)}$ to $x^{(p)} \in \Omega_t^{(p)}$, and collectively, the motion is given by $x = x(X, t) \in \Omega_t$ for $X \in \Omega_0$. We assume the media is dry and not influenced by factors other than mechanical loading (e.g., moisture and temperature are not considered). The configuration of particles in Ω_t at time t depends on various factors, such as material and geometrical properties, contact mechanism, and external loading. Essentially, there are two types of interactions present in the media: - *Intra-particle interaction* that models the deformation and internal forces in the particle and - *Inter-particle interaction* that accounts for the contact between particles and the boundary of the domain in which the particles are contained. In DEM, the first interaction is ignored, assuming particle deformation is insignificant compared to the inter-particle interaction. On the other hand, PeriDEM accounts for both interactions.

The balance of linear momentum for particle p, $1 \leq p \leq N_P$, takes the form:

$$\rho^{(p)} \ddot{u}^{(p)}(X, t) = f_{int}^{(p)}(X, t) + f_{ext}^{(p)}(X, t), \qquad \forall (X, t) \in \Omega_0^{(p)} \times (0, t_F)\,, \tag{1}$$

where $\rho^{(p)}$, $f_{int}^{(p)}$, and $f_{ext}^{(p)}$ are density, and internal and external force densities. The above equation is complemented with initial conditions, $u^{(p)}(X, 0) = u_0^{(p)}(X), \dot{u}^{(p)}(X, 0) = \dot{u}_0^{(p)}(X), X \in \Omega_0^{(p)}$.

Internal force - State-based peridynamics

Since all expressions in this paragraph are for a fixed particle p, we drop the superscript p, noting that material properties and other quantities can depend on the particle p. Following (Silling et al., 2007) and simplified expression of state-based peridynamics force in (Prashant K. Jha et al., 2021), the internal force takes the form, for $X \in \Omega_0^{(p)}$,

$$f_{int}^{(p)}(X, t) = \int_{B_\epsilon(X) \cap \Omega_0^{(p)}} (T_X(Y) - T_Y(X))\, \mathrm{d}Y, \tag{2}$$

where $T_X(Y) - T_Y(X)$ is the force on X due to nonlocal interaction with Y. Let $R = |Y - X|$ be the reference bond length, $r = |x(Y) - x(X)|$ current bond length, $s(Y, X) = (r - R)/R$

bond strain, then $T_X(Y)$ is given by (Prashant K. Jha et al., 2021; Silling et al., 2007)

$$T_X(Y) = h(s)J(R/\epsilon)\left[R\theta_X\left(\frac{3\kappa}{m_X} - \frac{15G}{3m_X}\right) + (r-R)\frac{15G}{m_X}\right]\frac{x(Y)-x(X)}{|x(Y)-x(X)|}, \tag{3}$$

where

$$\begin{aligned} m_X &= \int_{B_\epsilon(X)\cap\Omega_0^{(p)}} R^2 J(R/\epsilon)\,\mathrm{d}Y, \\ \theta_X &= h(s)\frac{3}{m_X}\int_{B_\epsilon(X)\cap\Omega_0^{(p)}} (r-R)\,R\,J(R/\epsilon)\,\mathrm{d}Y, \\ h(s) &= \begin{cases} 1, & \text{if } s < s_0 := \sqrt{\frac{\mathcal{G}_c}{(3G+(3/4)^4[\kappa-5G/3])\epsilon}}, \\ 0, & \text{otherwise}. \end{cases} \end{aligned} \tag{4}$$

In the above, $J : [0, \infty) \to \mathbb{R}$ is the influence function, $\kappa, G, \mathcal{G}_c$ are bulk and shear moduli and critical energy release rate, respectively. These parameters, including nonlocal length scale ϵ, could depend on the particle p.

DEM-inspired contact forces

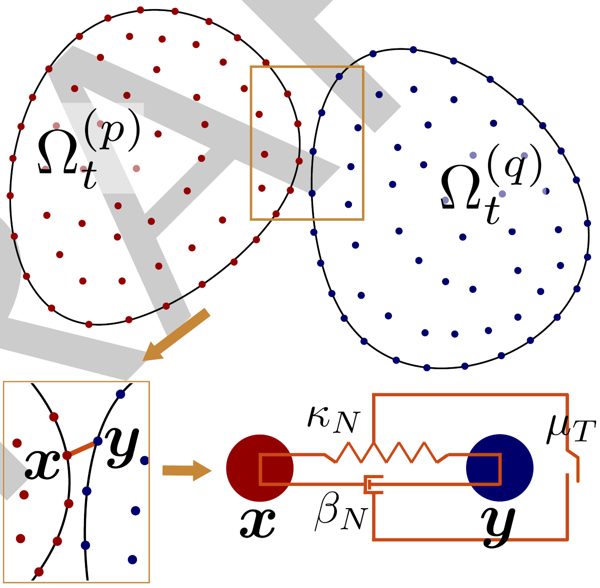

Figure 2: High-resolution contact approach in PeriDEM model for granular materials**?** between arbitrarily-shaped particles. The spring-dashpot-slider system shows the normal contact (spring), normal damping (dashpot), and tangential friction (slider) forces between points x and y.

The external force density $f_{ext}^{(p)}$ is generally expressed as

$$f_{ext}^{(p)} = \rho^{(p)} b + f^{\Omega_0,(p)} + \sum_{q \neq p} f^{(q),(p)}, \tag{5}$$

where b is body force per unit mass, $f^{\Omega_0,(p)}$ and $f^{(q),(p)}$ are contact forces due to interaction between particle p and container Ω_0 and neighboring particles q, respectively. In (Prashant K. Jha et al., 2021), the contact between two particles is applied locally where the contact takes place; this is exemplified in Figure 2 where contact between points y and x of two distinct particles p and q is activated when they get sufficiently close. The contact forces are shown using a spring-dashpot-slider system. To fix the contact forces, consider a point $X \in \Omega_0^{(p)}$ and let $R_c^{(q),(p)}$ be the critical contact radius (points in particles p and q interact if the distance is below this critical distance). Further, define the relative distance between two points $Y \in \Omega_0^{(q)}$ and $X \in \Omega^{(p)}$ and normal and tangential directions as follows:

$$
\begin{aligned}
\Delta^{(q),(p)}(Y,X) &= |x^{(q)}(Y) - x^{(p)}(X)| - R_c^{(q),(p)}\,, \\
e_N^{(q),(p)}(Y,X) &= \frac{x^{(q)}(Y) - x^{(p)}(X)}{|x^{(q)}(Y) - x^{(p)}(X)|}\,, \\
e_T^{(q),(p)}(Y,X) &= \left[I - e_N^{(q),(p)}(Y,X) \otimes e_N^{(q),(p)}(Y,X)\right] \frac{\dot{x}^{(q)}(Y) - \dot{x}^{(p)}(X)}{|\dot{x}^{(q)}(Y) - \dot{x}^{(p)}(X)|}\,.
\end{aligned}
\tag{6}
$$

Then the force on particle p at X due to contact with particle q can be written as (Prashant K. Jha et al., 2021):

$$
f^{(q),(p)}(X,t) = \int_{Y \in \Omega_0^{(q)} \cap B_{R_c^{(q),(p)}}(X)} \left(f_N^{(q),(p)}(Y,X) + f_T^{(q),(p)}(Y,X)\right) \, \mathrm{d}Y, \tag{7}
$$

with normal and tangential forces following (Desai et al., 2019; Prashant K. Jha et al., 2021) given by, if $\Delta^{(q),(p)}(Y,X) < 0$,

$$
f_N^{(q),(p)}(Y,X) = \left[\kappa_N^{(q),(p)} \Delta^{(q),(p)}(Y,X) - \beta_N^{(q),(p)} \dot{\Delta}^{(q),(p)}(Y,X)\right]\,, \tag{8}
$$

else $f_N^{(q),(p)}(Y,X) = 0$, and

$$
f_T^{(q),(p)}(Y,X) = -\mu_T^{(q),(p)}\, |f_N^{(q),(p)}(Y,X)|\, e_T^{(q),(p)}\,. \tag{9}
$$

Here, $\kappa_N^{(q),(p)}, \beta_N^{(q),(p)}, \mu_T^{(q),(p)}$ are coefficients for normal contact, normal damping, and tangential friction forces, and generally depend on the material properties of two particles p and q.

Implementation

PeriDEM is implemented in GitHub. It is based on C++ and uses only a handful of external libraries, which are included in the library in the `external` folder, allowing the code to be built and tested in Ubuntu and Mac systems relatively quickly. Specifically, we use taskflow (Huang et al., 2021) for asynchronous multithreaded computation, nanoflann (Blanco & Rai, 2014) for tree search to calculate neighbors for contact forces, and VTK for output. MPI and metis (Karypis & Kumar, 1997) have recently been integrated to implement distributed parallelism in the near future. This work is based on the previous research on analysis and numerical methods for peridynamics; see (Prashant K. Jha & Lipton, 2018a, 2018b, 2019; Prashant K. Jha & Lipton, 2020; Lipton et al., 2019).

Features

- Hybrid modeling using peridynamics and DEM for intra-particle and inter-particle interactions.
- It can simulate the deformation and breakage of a single particle with complex boundary conditions using peridynamics.
- Support for arbitrary shaped particles, allowing for realistic simulation scenarios.
- MPI will be used for distributed computing in the near future.
- Future work includes developing an adaptive modeling approach to enhance efficiency without compromising accuracy.

The Journal of Open Source Software

Brief implementation details

The primary implementation of the model is carried out in the model directory dem and the PeriDEM model is implemented in class DEMModel. The function `DEMModel::run()` performs the simulation. We next look at some key methods in `DEMModel` in more detail:

DEMModel::run()

This function does three tasks:

```
void model::DEMModel::run(inp::Input *deck) {
    // initialize data
    init();

    // check for restart
    if (d_modelDeck_p->d_isRestartActive)
      restart(deck);

    // integrate in time
    integrate();
}
```

In `DEMModel::init()`, the simulation is prepared by reading the input files (such as `.yaml`, `.msh`, and `particle_locations.csv`).

DEMModel::integrate()

Key steps in `DEMModel::integrate()` are

```
void model::DEMModel::run(inp::Input *deck) {
    // apply initial condition
    if (d_n == 0)
      applyInitialCondition();

    // apply loading
    computeExternalDisplacementBC();
    computeForces();

    // time step
    for (size_t i = d_n; i < d_modelDeck_p->d_Nt; i++) {
      // advance simulation to next step
      integrateStep();

      // perform output if needed
      output();
    }
}
```

In `DEMModel::integrateStep()`, we either utilize the central-difference scheme, implemented in `DEMModel::integrateCD()`, or the velocity-verlet scheme, implemented in `DEMModel::integrateVerlet()`. As an example, we look at `DEMModel::integrateCD()` method below:

```
void model::DEMModel::integrateVerlet() {
    // update current position, displacement, and velocity of nodes
    {
      tf::Executor executor(util::parallel::getNThreads());
      tf::Taskflow taskflow;
```

```
      taskflow.for_each_index(
        (std::size_t) 0, d_fPdCompNodes.size(), (std::size_t) 1,
          [this, dt, dim](std::size_t II) {
            auto i = this->d_fPdCompNodes[II];

            const auto rho = this->getDensity(i);
            const auto &fix = this->d_fix[i];

            for (int dof = 0; dof < dim; dof++) {
              if (util::methods::isFree(fix, dof)) {
                this->d_v[i][dof] += 0.5 * (dt / rho) * this->d_f[i][dof];
                this->d_u[i][dof] += dt * this->d_v[i][dof];
                this->d_x[i][dof] += dt * this->d_v[i][dof];
              }
            }
          } // loop over nodes
      ); // for_each

      executor.run(taskflow).get();
    }

    // advance time
    d_n++;
    d_time += dt;

    // update displacement bc
    computeExternalDisplacementBC();

    // compute force
    computeForces();

    // update velocity of nodes (similar to the above)
}
```


DEMModel::computeForces()

The key method in time integration is `DEMModel::computeForces()` In this function, we compute internal and external forces at each node of a particle and also account for the external boundary conditions. This function looks like

```
void model::DEMModel::computeForces() {
    // update the point cloud (make sure that d_x is updated along with displacement)
    auto pt_cloud_update_time = d_nsearch_p->updatePointCloud(d_x, true);
    pt_cloud_update_time += d_nsearch_p->setInputCloud();

    // reset forces to zero ...

    // compute peridynamic forces
    computePeridynamicForces();

    // compute contact forces between particles
    computeContactForces();

    // Compute external forces
    computeExternalForces();
```

}

Further reading

The above gives the basic idea of implementation. For a closer look, interested readers can look at demModel.cpp.

Examples

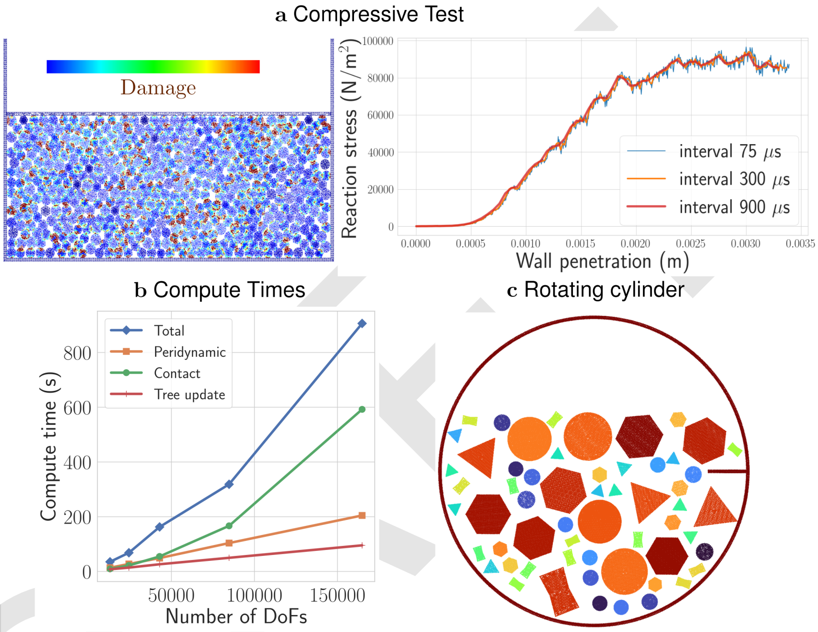

Figure 3: (a) Nonlinear response under compression, (b) exponential growth of compute time due to nonlocality of internal and contact forces, and (c) rotating cylinder with nonspherical particles.

Examples are described in examples/README.md of the library. One key result is the compression of 500+ circular and hexagon particles in a rectangular container by moving the top wall. The stress on the moving wall as a function of wall penetration becomes increasingly nonlinear, and media shows signs of yielding as the damage becomes extensive; see Figure 3a. Preliminary compute time analysis with an increasing number of particles shows an exponential increase in compute time of contact and peridynamics forces, which is unsurprising as both computations are nonlocal. This also shows the bottleneck with the PeriDEM approach, motivating us to consider MPI-parallelism and multi-fidelity framework. Demonstration examples also include attrition of various non-circular particles in a rotating cylinder Figure 3c.

References

Blanco, J. L., & Rai, P. K. (2014). *Nanoflann: A C++ header-only fork of FLANN, a library for nearest neighbor (NN) with KD-trees*. https://github.com/jlblancoc/nanoflann.

Desai, P. S., Mehta, A., Dougherty, P. S., & Higgs III, C. F. (2019). A rheometry based calibration of a first-order DEM model to generate virtual avatars of metal additive manufacturing (AM) powders. *Powder Technology*, *342*, 441–456. https://doi.org/https://doi.org/10.1016/j.powtec.2018.09.047

Huang, T.-W., Lin, D.-L., Lin, C.-X., & Lin, Y. (2021). Taskflow: A lightweight parallel and heterogeneous task graph computing system. *IEEE Transactions on Parallel and Distributed Systems*, *33*(6), 1303–1320. https://doi.org/https://doi.org/10.1109/TPDS.2021.3104255

Jha, Prashant K., Desai, P. S., Bhattacharya, D., & Lipton, R. (2021). Peridynamics-based discrete element method (PeriDEM) model of granular systems involving breakage of arbitrarily shaped particles. *Journal of the Mechanics and Physics of Solids*, *151*, 104376. https://doi.org/https://doi.org/10.1016/j.jmps.2021.104376

Jha, Prashant K., & Lipton, R. (2018a). Numerical analysis of nonlocal fracture models in holder space. *SIAM Journal on Numerical Analysis*, *56*(2), 906–941. https://doi.org/https://doi.org/10.1137/17M1112236

Jha, Prashant K., & Lipton, R. (2018b). Numerical convergence of nonlinear nonlocal continuum models to local elastodynamics. *International Journal for Numerical Methods in Engineering*, *114*(13), 1389–1410. https://doi.org/https://doi.org/10.1002/nme.5791

Jha, Prashant K., & Lipton, R. (2019). Numerical convergence of finite difference approximations for state based peridynamic fracture models. *Computer Methods in Applied Mechanics and Engineering*, *351*, 184–225. https://doi.org/https://doi.org/10.1016/j.cma.2019.03.024

Jha, Prashant K., & Lipton, R. P. (2020). Kinetic relations and local energy balance for LEFM from a nonlocal peridynamic model. *International Journal of Fracture*. https://doi.org/10.1007/s10704-020-00480-0

Karypis, G., & Kumar, V. (1997). *METIS: A software package for partitioning unstructured graphs, partitioning meshes, and computing fill-reducing orderings of sparse matrices*. https://hdl.handle.net/11299/215346

Lipton, R. P., Lehoucq, R. B., & Jha, P. K. (2019). Complex fracture nucleation and evolution with nonlocal elastodynamics. *Journal of Peridynamics and Nonlocal Modeling*, *1*(2), 122–130. https://doi.org/https://doi.org/10.1007/s42102-019-00010-0

Silling, S. A., Epton, M., Weckner, O., Xu, J., & Askari, E. (2007). Peridynamic states and constitutive modeling. *Journal of Elasticity*, *88*, 151–184. https://doi.org/https://doi.org/10.1007/s10659-007-9125-1