Yade’s (undeniable) transformation into a model project of optimization, multi-physics couplings, and user support

Robert A. Caulk¹, Janek Kozicki², Deepak Kunhappan¹, Raphaël Maurin³, Eduard P. Montella¹, Thomas Sweijen⁴, Chao Yuan⁵, Bruno Chareyre¹

¹ Univ. Grenoble Alpes
CNRS, Grenoble INP, 3SR
1270 rue de la Piscine, 38610 Gières, France
² Gdańsk Univ. of Technology
Department of Theoretical Physics and Quantum Information
³ Univ. de Toulouse
INPT, UPS, IMFT, Institut de Mécanique des Fluides de Toulouse
Alle Camille Soula, 31400 Toulouse, France
⁴ CRUX engineering
Pedro de Medinalean 3c, Amsterdam, The Netherlands
⁵ Univ. of Calgary
Calgary, AB, Canada T2N 1N4

Keywords Yade, developments, extensions, DEM, CFD, THM, MPI

Abstract This paper provides an update to the 2015 paper titled “Yade-DEM: (not so) recent advances and steps toward multiphase couplings”. Since 2015, Yade has continued its maturation from a robust open source DEM code into a high performance multi-physics code with over 65 total contributors, countless extensions, and a well supported user base. Many of the recent advancements and extensions are highlighted within this paper to raise awareness of new Yade functionalities and the corresponding publications.

1 INTRODUCTION

Yade open source DEM software is experiencing sustained growth of its active user base supported by a development group of over 21 academic and industrial contributors in the recent 12 months¹. The impressive progress of the Yade project is evidenced by an increased citation rate, an increased rate of source code contributions, and finally, an increased rate of user questions. Starting with academic prominence, Yade documentation alone received an increased yearly citation count from 70 in 2014 to 123 in 2018 (Table 1). These citations stem from an influx of almost 2,000 unique commits comprising 25,000 lines of new code (Table 1) between 2015 and 2019. Within these 25,000 lines of code

¹https://www.openhub.net/p/yade
Recent advancements in Yade open source DEM software

hides a multitude of new packages, modified packages, bug fixes, and optimizations which all bolster the active Yade user base as observed in the increased number of monthly questions fielded in the Launchpad Q&A website (Fig. 1). This combination of active contributors and diligent users has resulted in a vibrant open source project that boasts new modules published in academic journals, high scrutiny bug control, and user support for packages long after the original developers depart from the project. The newest of these modules include fundamental code management process changes, an MPI parallelization rehaul, two-phase flow packages, CFD couplings, thermal physics couplings, and more. It is clear that Yade is undergoing a transformation from a robust DEM code to a widely extended multi-physics project worth almost $1.4 million (Tab. 1).

Table 1: Facts and figures

<table>
<thead>
<tr>
<th></th>
<th>2015</th>
<th>2019</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lines of code</td>
<td>74,384</td>
<td>101,894</td>
</tr>
<tr>
<td>Languages</td>
<td>C++11, Python 2</td>
<td>C++11, Python 2&amp;3</td>
</tr>
<tr>
<td>License</td>
<td>GPL2.0</td>
<td>GPL2.0</td>
</tr>
<tr>
<td>Commits</td>
<td>4,752</td>
<td>6,353</td>
</tr>
<tr>
<td>Total Contributors</td>
<td>52</td>
<td>65</td>
</tr>
<tr>
<td>Cost estimate</td>
<td>18 person years, $1,006,365</td>
<td>26 person years, $1,398,978</td>
</tr>
<tr>
<td>Branch host</td>
<td>GitHub</td>
<td>GitLab</td>
</tr>
</tbody>
</table>

2 CFD-DEM SIMULATIONS WITH OPENFOAM

Yade-DEM was recently connected to the OpenFOAM coupling engine and two CFD solvers in OpenFOAM for realizing Euler-Lagrange multiphase flow simulations [9] [10]. Compared to the previous version of CFD-DEM coupling by Feng Chen [7], the present implementation is flexible and permits the usage of multiple cores both in Yade (based on OpenMP) and OpenFOAM (MPI). The two OpenFOAM solvers, icoFoamYade and pimpleFoamYade, provide access to simple point force coupling for dilute suspensions (and for low $Re$) and volume averaged coupling (semi-dilute to dense suspensions), which

---

3https://scholar.google.fr/citations?user=hZB8GGcAAAAJ&hl=en
3https://en.wikipedia.org/wiki/COCOMO

---

Figure 1: Threads started on Yade’s launchpad Question & Answer website.
Recent advancements in Yade open source DEM software accounts for the particle volume fraction with Gaussian interpolation of field and particle variables, respectively. Both the solvers utilize fast range based mesh search (with parallel support) which enables an accurate calculation of the volume fraction field in the fluid, and circumvents the need for solving a diffusion equation for smoothing the volume fraction field. The current implementation can handle up to $10^5$ particles. The current framework enables the future inclusion of fully resolved flow description around the particle by Immersed Boundary Method and full parallel coupling between Yade-MPI and OpenFOAM (currently under development).

3 TWO-PHASE FLOWENGINE

Recent two-phase flow developments extend the original Yade FlowEngine [5] to simulate wet granular materials. Two-Phase FlowEngine employs a “Two-phase pore-scale finite volume” method (2PFV-DEM) [20, 19] which simulates pendular bridges and larger liquid clusters (Fig 3a) where air-water distribution is governed by capillarity. Fluid distribution in 2PFV-DEM is controlled by local rules that are described by the Incircle method [18] or the Mayer-Stowe-Princen (MS-P) method [20] providing accurate results for primary drainage curves (Fig 3b). Users can look forward to applying 2PFV-DEM to many engineering applications, such as the wetting and drying processes, powder technology or enhanced oil recovery. Besides the previous approximation methods used to predict drainage and imbibition curves, lattice Boltzmann method was also integrated into Yade to perform direct simulations enhancing an exhaustive description of the volume-pressure relationship for the throats ([17] and [6] in this volume) and accurate results on the liquid morphology (Fig 4).

4 THERMALENGINE

The recent addition of a ThermalEngine in Yade opens up a wide range of Thermal-Hydraulic-Mechanical (THM) modelling possibilities such as non-isothermal flow through a conductive sphere packing and geomechanical thermoporoelasticity. Conductive heat transfer is simulated between particles using a pipe network, while advective heat transfer conserves energy in pore spaces by reusing fluxes from Yade’s existing FlowEngine. [3] describes the framework in detail and shows a complex THM experimental validation. Further, Yade’s python wrapping enables users to easily modify, debug, visualize and
Recent advancements in Yade open source DEM software

Figure 3: a) Phase distribution of a primary drainage simulation using the 2PFV-DEM for a packing of 40,000 spheres. b) 2PFV-DEM primary drainage curve compared to experimental data [8].

Figure 4: Decomposition of the granular assembly (a) into small subsets (b). Each subset (c) is made up of 3 spheres in which the non-wetting phases (air) invades the pore throat by means of a lattice Boltzmann method simulation.
extend the existing THM framework.

5 HYDROFORCEENGINE

A new module, called HydroForceEngine, was designed with the objective of simulating turbulent bedload transport configurations. It is a one-dimensional (1D) volume-averaged fluid resolution scheme that allows the user to (i) apply buoyancy force, as well as drag and lift forces evaluated from an input 1D vertical fluid velocity profile, (ii) solve the 1D vertical fluid momentum balance taking into account the presence of the particles, (iii) solve the 1D vertical fluid momentum balance alone. Yade users can now easily perform both one-way and two-way coupled simulations for unidirectional flows of any kind, as evidenced by example scripts in the code. For more details on the theoretical aspect, the numerical implementation and validations, see [12, 11, 13, 14].

6 ACOUSTIC EMISSIONS MODULE

Yade now has the capacity to simulate acoustic emissions in cohesive discrete element assemblies. Following a strain energy approach, Yade tracks the release of strain energy in bonds surrounding recently broken bonds to estimate Acoustic Emission magnitudes. Further, Yade is capable of clustering bonds that break in close proximity to one another which enables the realistic quantification of acoustic emission magnitude during material failure. Yade users can easily post process and visualize acoustic emission magnitudes, acoustic emission clusters, and more using the existing data export tools in Yade. [1] demonstrates the new functionality by running three point bending tests on rock samples containing various levels of heterogeneity to ultimately show the spatial distributions and magnitudes of acoustic emissions during pre and post failure.

7 MPI PARALLELIZATION

New DEM users often compare DEM codes based on parallelization schemes instead of package availability and support accessibility. Commonly visited threads online con-
sider Yade’s shared memory parallelization (OpenMP) to be a weakness compared to the massively parallel MPI schemes employed by other codes. What is sometimes overlooked is that the shared memory approach on a single node is probably more efficient than distributed memory. However, it sets an upper bound to the number of threads (typically 24 on current best CPUs). On the other hand, taking advantage of hundreds of nodes when they are available is only possible with a distributed memory approach, which was not available with Yade until recently. Luckily, this is changing. In the Fall of 2018, Bruno Chareyre began the process of adding MPI support to Yade. After a successful hackathon[4] and several additional months of effort aided by François Kneib, Janek Kozicki and Deepak Kunhappan, MPI was successfully added and it should be a stable feature soon.

The strategy is as follows: the full domain is decomposed so that each available node integrates a subset of the particles, where boundary particle positions and velocities are passed by MPI messages to neighboring subdomains. The bodies can be exchanged between subdomains as they move to maintain computational balance and compactness of the subdomains.

Implementing a distributed memory approach is usually considered a very intrusive technique in terms of source code (which is a problem with a large multi-author code base like Yade), yet interestingly our implementation did not require substantial changes in the C++ source code. Instead, the implementation was started with Python (module mpi4py), such that one master Python process would control multiple running instances of Yade and communicate data and instructions to them interactively. Each individual instance was essentially MPI-blind internally. Only a few critical functions have been translated from Python to C++ in a later stage. That was the case, namely, for functions involving multiple loops on bodies and on interactions to form a large message. In such case, because of the loops and pickling overhead, C++ clearly outperforms Python. Nevertheless, the core of Yade code is, and will remain, independent of those utility functions. On the other hand each instance can exploit shared memory parallelism on a single node via OpenMP, which enables an hybrid framework.

8 GPU ACCELERATION OF FLOWENGINE

Yade’s FlowEngine was accelerated using a variety of methods including matrix factor reuse, parallel task management, and GPU computing. Combined, these techniques yield an increase of performance by an astonishing 170x as highlighted by [2]. Yade users interested in testing the first two methods can simply run FlowEngine and manipulate

Figure 7: Example of acoustic emission locations along tensile fracture, colored by clusters and labeled with estimated magnitude.
the “remeshInterval” parameter as they desire. Other Yade users interested in testing the GPU acceleration should follow the online tutorial to get started with installation.

9 GITLAB MIGRATION

The recent migration of Yade development to GitLab.com in January 2019 has improved the speed and accuracy of its open sourced development process. The most notable feature of the new development process is the GitLab continuous integration pipeline combined with branch merge requests. As Yade developers make branch merge requests, the integration pipeline compiles the source code, runs the code tests, and builds the website. Other developers can review the changes, inspect the build, and discuss the merge before approving. After approval, the compilation and tests are performed on 5 different Linux distributions: Ubuntu 16.04, 18.04, Debian Buster, Debian Stretch and OpenSUSE15. Beyond the integration pipeline, bug-tracking tools allow developers to seamlessly connect bug discussion with merge requests and repository commits. All developer activity is quickly accessed via, activity, which lets developers and users quickly view recent development efforts. As a very positive effect of GitLab migration, Yade already experienced a surge of activity as shown at these convenient links: commit activity, charts, and analytics.

![Figure 8: Effect of acceleration techniques on FlowEngine performance for packings comprised of 10k and 350k spheres.](image-url)
REFERENCES


