Critical Comparison of Motion Integration Strategies and Discretization Choices in the Material Point Method

Sacha Duverger, Jérôme Duriez, Pierre Philippe, Stéphane Bonelli

Critical comparison of motion integration strategies and discretization choices in the Material Point Method

Sacha Duverger[1], Jérôme Duriez[1*], Pierre Philippe[1] and Stéphane Bonelli[1]

[1]INRAE, Aix Marseille Univ, RECOVER, Aix-en-Provence, France.

*Corresponding author(s). E-mail(s): jerome.duriez@inrae.fr;
Contributing authors: sacha.duverger@inrae.fr;
pierre.philippe@inrae.fr; stephane.bonelli@inrae.fr;

Abstract

To simulate large, history-dependent material displacements, the Material Point Method (MPM) solves for the kinematics of Lagrangian material points being embedded with mechanical variables while moving freely within a fixed mesh. The MPM procedure makes use of the latter mesh as a computational grid, where the momentum balance equation with the acceleration field are first projected onto nodes, before material points can be moved. During that process, a number of different choices have been adopted in the literature for what concerns the computational definition of time increments of velocity and position, from the knowledge of nodal acceleration. An overview of these different motion integration strategies is herein proposed, with a particular emphasis on their impact onto the MPM conservative properties. Original results illustrate the discussion, considering either simple configurations of solid translation and rotation or a more complex collapse of a frictional mass. These analyses furthermore reveal hidden properties of some motion integration strategies regarding conservation, namely a direct influence of the time step value during a time integration being inspired by the Particle In Cell (PIC) ancestor of

the MPM. The spatial, resp. temporal (in comparison with vorticity), discretizations are also shown to affect the angular momentum conservation of the FLIP method, resp. an affine extension of PIC (APIC).

Keywords: Material Point Method; PIC; FLIP; APIC; XPIC

1 Introduction

Anticipating the deformations of large-scale constructions, e.g. water dams possibly rising up to hundreds of meters, under various mechanical loads is crucial for safety concerns. These studies are typically carried out with numerical methods based on the Finite Element Method (FEM) and corresponding pioneering works [1, 2] for continuous materials. The FEM unfortunately suffers from a number of limitations related with its underlying Lagrangian mesh for solid mechanics. In particular, many scenarios of interest involve large deformations, e.g. possible structure failure, where extreme mesh distortions would prevent the FEM procedure to pursue. A solution to overcome this issue can be to include a conditional remeshing step in the FEM framework [3, 4, 5] but this can turn to be computationally expensive. If present, e.g. for elasto-plastic solids, a field of hardening variables has also to be recast into the new mesh which makes things even more intricate. These issues have been reviewed e.g. in [6], with a particular focus onto the Particle Finite Element Method (PFEM) [7] as an appropriate remeshing FEM technique in presence of hardening variables.

In contrast, the historical Particle In Cell method (PIC) [8] has avoided distortion issues by fixing the mesh, turning it into an Eulerian frame within which the integration points of the FEM are free to move. As such, the PIC method came as an hybrid Eulerian-Lagrangian method with a robust nature in the case of large displacements. Then, in [9], the FLuid Implicit Particle

method (FLIP) has revisited PIC with the idea of limiting the numerical diffusion and energy dissipation the latter suffers from, adapting in particular the way integration points velocities are defined from the acceleration at grid points. These methods eventually led to the slightly more recent Material Point Method (MPM) [10], possibly in a GIMP [11] or CPDI [12] variants, which aims to extend the hybrid Eulerian-Lagrangian formulation to history-dependent materials and has become a popular tool to simulate large displacements in solid mechanics, see e.g. [13, 14, 15] for general MPM reviews.

Following PIC and FLIP, the MPM essentially solves continuum mechanics equations and computes accelerations on fixed mesh grid points in order to eventually update positions of Lagrangian material points. Subtleties then arise during the MPM procedure with a frequent transport, i.e. mapping, of mechanical quantities between these two uncoupled spatial discretizations, which deteriorates the proper definitions of classical kinematic fields of accelerations, velocities and positions. As it will be further exposed in the manuscript, it is for instance to note that material point velocities may not always be the time integral of their accelerations, nor the time derivatives of their positions. These subtleties lead the MPM to face similar challenges as PIC and FLIP for what concerns the conservation of energy and momentum. As a matter of fact, these fundamental objectives are not always fulfilled and prompted a number of previous studies.

As for momentum conservation, the MPM is formulated such that linear momentum is automatically conserved but strong issues may concern angular momentum and rotational motion, as discussed, e.g., in [16, 17, 18] and further recalled in the remainder of the manuscript. It was actually shown in [16, 17] that the angular momentum is lost both with PIC and FLIP during the two-ways transport between material points and grid points with classical lumped

expressions of nodal masses and an Affine-augmented Particle In Cell method (APIC) was accordingly proposed in [17] aiming to solve the issue. Before being further improved in [18], APIC was shown in [17] to conserve the energy better than PIC if not perfectly, while being more stable than FLIP. Since the remaining energy dissipation sources in APIC [17] basically come from its share of PIC transport equations, an affine-augmented variation of the FLIP velocity mapping, AFLIP, was tested in [19] and logically found to conserve the energy even better than both APIC and FLIP.

Similar to APIC and AFLIP which eventually aim at better capturing affine velocity fields when mapping one field from another, a spatial gradient-based Taylor expansion has been proposed in [20] and later denoted as Taylor-PIC (TPIC) in [21].

Another recent study [22] highlights that the motion integration procedure actually corresponds to choosing a velocity field among an infinity of solutions to the velocity transportation equations. With this point of view, the PIC strategy appears as the solution which minimizes the velocity by filtering out all the noise, while the FLIP strategy minimizes the acceleration and does not filter any noise. Using this insight, a new XPIC(m) strategy was proposed in the same study [22], for adopting the velocity field that minimizes the difference with respect to its previous values filtered at the order m. The XPIC(m) strategy can thus be used to adapt the noise filtration for a specific problem, giving a great adjustable compromise between PIC and FLIP strategies. Based on the XPIC(m) strategy, a FMPM (Full mass matrix MPM) formulation was proposed in [23], considering an approximation of the full mass matrix (inverse) instead of the traditional lumped one. Although this approach complexifies the imposition of boundary conditions, results were shown to be less dissipative and more accurate than both FLIP and XPIC(m).

In connection with the evergrowing popularity of the MPM, the present manuscript then aims to clarify the underlying reasons for so many choices regarding motion integration in a MPM implementation and review most of the currently available alternatives. The discussion is illustrated from original numerical results evidencing their respective performances in terms of energy and momentum conservation.

After recalling the general MPM background in Section 2, Section 3 proposes a grouped presentation of the various possible motion integration strategies. Section 4 then investigates the conservative nature of several of those in the simple cases of an elastic solid moving in a purely translational motion or a rigid solid subject to a rotational motion, where analytical ground-truth results are available for reference. The analysis is finally extended to the more complex and rather classical case of the granular column collapse in Section 5, together with a discussion of the (time and spatial) discretization parameters.

Although the calculations performed in this study are likely to involve large transformations (large deformations and large rotations), our model is based on an infinitesimal transformation formulation. Taking large transformations into account, similar to [12, 23, 24, 25], would lead to more accurate results, but would not change the conclusions of the present work which is focused on the comparison of motion integration strategies and discretization choices in the MPM formulation. All simulations are executed with a modified version of the open-source code CB-Geo MPM [26], used together with its PyCBG interface [27].

2 Common MPM principles

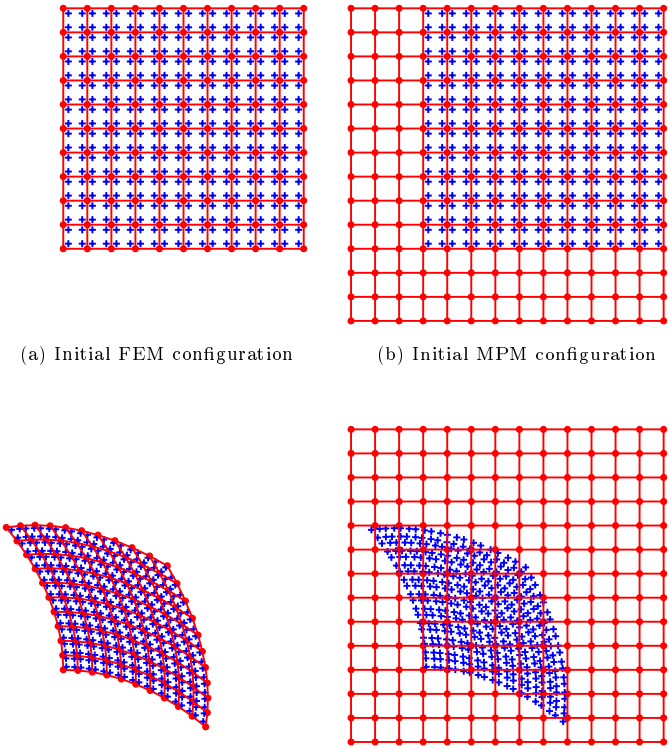
2.1 Governing equation and double spatial discretization

Considering a deformable solid being continuously present in the domain Ω, mass density, velocity and stress fields are respectively denoted ρ, $\vec{V}$ and $\boldsymbol{\sigma}$, omitting time and space variables for simplicity. External forces may apply onto Γ, the surface of Ω, as tractions $\overrightarrow{\tau} = \boldsymbol{\sigma} \cdot \overrightarrow{n}$, with $\overrightarrow{n}$ the outward normal to Γ. If present, a constant, uniform, gravitational field is denoted as $\overrightarrow{g}$. With $\overrightarrow{w}$ a test function (a kinematically admissible velocity field, typically) the weak form of the virtual work principle classically reads:

$$\int_\Omega \rho \overrightarrow{w} . \frac{\partial \overrightarrow{V}}{\partial t} d\Omega = \int_\Omega \rho \overrightarrow{w} . \overrightarrow{g} d\Omega + \int_\Gamma \overrightarrow{w} . \overrightarrow{\tau} d\Gamma - \int_\Omega \nabla \overrightarrow{w} : \boldsymbol{\sigma} d\Omega \tag{1}$$

As proposed in [10], the MPM solves Eq. (1) for a Lagrangian velocity field, which is defined on a first layer of spatial discretization in terms of a finite set $\{p\}$ of N_{mp} material points. Those material points will also carry each a constant mass quantity m^p and history-dependent, e.g. elasto-plastic hardening, variables and accordingly express material behaviour.

Being geometrically uncoupled to the set of material points, the MPM additionally relies on an Eulerian mesh with a number N_{gp} of grid points i located at constant positions $\vec{x}_i$ and connecting into mesh cells i.e. elements, see Figure 1b. While being fixed in space as illustrated in Figure 1d in contrast to Figure 1c for the classical FEM in solid mechanics, that Eulerian grid plays a FEM-like computational role in which the weak form (1) is eventually transposed at every grid point. Grid points actually combine with associated shape functions $N_i(\overrightarrow{x})$ to serve as a nodal basis for expressing quantities, e.g.



(a) Initial FEM configuration (b) Initial MPM configuration

(c) Final FEM configuration (d) Final MPM configuration

Fig. 1: Mesh (in red) and integration or material points (in blue) evolutions during similar FEM and MPM simulations

for the test function:

$$\vec{w}(\vec{x}) = \sum_i \vec{w}_i N_i(\vec{x}) \tag{2}$$

while obeying partition of unity:

$$\sum_i N_i(\vec{x}) = 1 \tag{3}$$

Eq. (3) can indeed be seen as a necessary condition for (i.e., a consequence of) Eq. (2).

From this double spatial discretization, the classical MPM procedure leads to the following expression for nodal acceleration [10]:

$$\overrightarrow{a}^i = \frac{\overrightarrow{f}^i}{m^i} \tag{4}$$

$$\overrightarrow{f}^i = \overrightarrow{f}^i_{int} + \overrightarrow{f}^i_{ext} \tag{5}$$

$$m^i = \sum_p N_i(\vec{x}^p) m^p \tag{6}$$

where the first two terms on the right-hand side of Eq. (1), describing body forces and external surface loads, have been turned into a nodal force quantity, $\overrightarrow{f}^i_{ext}$:

$$\overrightarrow{f}^i_{ext} = \sum_p N_i(\overrightarrow{x}^p) m^p \overrightarrow{g} + \int_\Gamma N_i \overrightarrow{\tau} d\Gamma \tag{7}$$

while the last term of Eq. (1) being representative of internal forces corresponds to the following nodal force quantity $\overrightarrow{f}_{int}$ obtained from the material points stresses:

$$\overrightarrow{f}^i_{int} = -\sum_p \frac{m^p}{\rho} \boldsymbol{\sigma}^p \overrightarrow{\nabla N}_i(\overrightarrow{x}^p) = -\sum_p v^p \boldsymbol{\sigma}^p \overrightarrow{\nabla N}_i(\overrightarrow{x}^p) \tag{8}$$

It is to note that Eqs. (4)-(6) correspond to a lumped mass matrix approach with known consequences for the conservative properties of the method [9, 10, 16, 28]. While mass is naturally conserved, in the sense that $\sum_i m^i = \sum_p m^p$, the cases of momenta and energy are more intricate and will be reviewed below. It is also to note that, unlike the initial MPM formulation [10], mass density is not considered constant here but instead computed from the constant m^p and a variable material point volume v^p. The latter is initialized from $v^{j(p)}$ the volume of the mesh element, j, including p and N^j_{mp} the total number of

material points in that cell:

$$v^p(t=0) = \frac{v^{j(p)}}{N_{mp}^{j(p)}} \tag{9}$$

before being updated at each iteration depending on a volumetric strain measure at the center of the cell j.

2.2 Material points or material domains in the MPM discretization

The above discrete equations, Eqs (4)-(8), have been obtained in [10] from the consideration of a point-wise mass distribution within $\{p\}$, with material point-centered Dirac distributions for the mass density and the stress fields. As an alternative to this pure "material point" point of view, finite "material domains" have been proposed in a GIMP variant [11] to carry a piece-wise constant stress field, through the concept of particle characteristic function with finite support, unlike the Dirac distribution of classical MPM. Shape functions are then directly replaced in Eq. (7) (for its first right-hand side term) and Eq. (8) with weighting functions that convolute shape functions with particle characteristic functions and show more suitable properties, e.g. a non-linear nature even with linear shape functions (see below Section 2.3).

As imposed by partition of unity considerations, such material domains should however stay contiguous and non-overlapping for results to stay consistent, e.g., [29]. Particle domains then need to be updated according to material point kinematics, way beyond pure translation, and the CPDI [12] extension to GIMP for instance enables one to update those as deformable parallelograms while considering piece-wise constant deformation gradients and stresses across domains.

One can note that the discrete equations, Eq. (8) in particular, could also be seen as a quadrature expression for a numeric integration of the weak form terms, avoiding any assumption on the mechanical (e.g., stress) fields and considering material points just as integration points or numeric tracers. From this point of view, quadrature precision issues arise in connection with material point displacements, as discussed in e.g. [13, 30] but those could possibly be solved following [31], which is still out of the present scope.

Here, the remainder of the sequel adopts the "material point" point of view of classical MPM to avoid the complexity of defining particle domains that maintain a rigorous tessellation of Ω whatever the kinematics.

2.3 Chosen shape functions

In this "material point" framework, piecewise linear shape functions such as used in [32, 33, 34] are avoided due to their C^0 nature that may lead to ringing, i.e. cell crossing, instabilities [35] through the so-called internal force $\overrightarrow{f}^i_{int}$ in Eq. (8) with the gradient of the shape functions, which would induce discontinuous changes in $\overrightarrow{f}^i_{int}$ when a material point crosses the boundary between two cells [13, 30, 36]. Boundary modified cubic splines [15] with (twice) continuous derivatives are adopted instead.

3 Various MPM motion integration strategies

3.1 Intricacy of mappings between material points and grid points as motion integration strategies

While the nodal acceleration Eq. (4) is intended to serve as a time update of nodal velocities, a salient keypoint of the MPM then appears in the yet unspecified choices for initial nodal velocities $\overrightarrow{V}^i$ and for the exact role of updated values, being denoted $\widetilde{\overrightarrow{V}^i}$ likewise to [8], in the material point kinematics, in

necessary connection with material point velocities $\overrightarrow{V}^p$ for both cases. The corresponding operations are generically denoted in this paper as the "motion integration strategy". Before being reviewed in details below, these could be introduced, broadly speaking, as a particular choice of back-and-forth mappings between grid points and material points. From the grid points to the particles, a shape-function-inspired mapping for a given quantity s (which can be any tensor of order $n \geq 0$) would be:

$$s^p = \sum_i N_i(\overrightarrow{x}^p) s^i \tag{10}$$

which can be recast in matrix form:

$$\overrightarrow{s}^{\mathbb{P}} = \boldsymbol{G}^{\bullet P} \overrightarrow{s}^{\mathbb{G}} \tag{11}$$

$$(G^{\bullet P})_{pi} = N_i(\overrightarrow{x}^p) \tag{12}$$

Here $\overrightarrow{s}^{\mathbb{P}}$ denotes the whole set of material point values and $\overrightarrow{s}^{\mathbb{G}}$ its nodal counterpart. The arrow in Eq. (11) corresponds to the increase in tensor order from n to $n+1$ in order to cover the whole sets of material points or grid points, with each line of $\overrightarrow{s}^{\mathbb{G}}$ (among N_{gp}), resp. $\overrightarrow{s}^{\mathbb{P}}$ (among N_{mp}), corresponding to the quantity for a specific grid point, resp. material point. In the case of s being a vector quantity, $\overrightarrow{s}^{\mathbb{G}}$ and $\overrightarrow{s}^{\mathbb{P}}$ are both second order tensors in Eq. (11) and will be denoted $\boldsymbol{s}^{\mathbb{G}}$ and $\boldsymbol{s}^{\mathbb{P}}$ in the rest of this paper.

From the particles to the grid, one could adopt, likewise to the lumped mass expression in Eq. (6), the following expression which obeys $\sum_i s^i = \sum_p s^p$:

$$s^i = \sum_p N_i(\overrightarrow{x}^p) s^p \tag{13}$$

or in matrix form:

$$\overrightarrow{s}^{\mathbb{G}} = \boldsymbol{P}^{\blacktriangleright G} \overrightarrow{s}^{\mathbb{P}} \tag{14}$$

$$\boldsymbol{P}^{\blacktriangleright G} = (\boldsymbol{G}^{\blacktriangleright P})^T \tag{15}$$

It has to be observed that in the general case $N_{mp} \neq N_{gp}$, making it impossible for $\boldsymbol{G}^{\blacktriangleright P}$ to be square. There is furthermore even no reason for the latter to be at least semi-orthogonal, i.e. $\boldsymbol{G}^{\blacktriangleright P}\boldsymbol{P}^{\blacktriangleright G}\overrightarrow{s}^{\mathbb{P}} \neq \overrightarrow{s}^{\mathbb{P}}$, contributing to the subtleties of the mappings discussed below, through the introduction of so-called null space errors [37].

3.2 From known particle velocities to nodal velocities

As for the nodal velocities $\overrightarrow{V}^i$ at the beginning of a time step, those have been usually expressed since the FLIP method from material point velocities by explicitly conserving the linear momentum $\vec{u}$ through the application of Eq. (14) from $\vec{u}^p = m^p \overrightarrow{V}^p$ to $\vec{u}^i$ and then dividing by the nodal masses to give:

$$\overrightarrow{V}^i = \frac{\sum_p N_i(\overrightarrow{x}^p) m^p \overrightarrow{V}^p}{m^i} \tag{16}$$

The result of Eq. (16) can also be seen as a mass-weighted transport procedure that directly applies to velocities $\overrightarrow{V}$ and which can be written in matrix form as:

$$\boldsymbol{V}^{\mathbb{G}} = \boldsymbol{P}_m^{\blacktriangleright G} \boldsymbol{V}^{\mathbb{P}} \tag{17}$$

$$(P_m^{\blacktriangleright G})_{ip} = \frac{m^p N_i(\overrightarrow{x}^p)}{\sum_{p'} m^{p'} N_i(\overrightarrow{x}^{p'})} \tag{18}$$

where $\boldsymbol{V}^{\mathbb{G}}$ and $\boldsymbol{V}^{\mathbb{P}}$ are respectively $(N_{gp} \times N_{dim})$ and $(N_{mp} \times N_{dim})$ matrices with N_{dim} the number of dimensions, and $\boldsymbol{P}_m^{\star G}$ a mass-weighted mapping matrix that directly applies to velocity quantities. Similar to the previous case of Eqs. (11)-(12), each line of $\boldsymbol{V}^{\mathbb{G}}$, resp. $\boldsymbol{V}^{\mathbb{P}}$, corresponds to the velocity of a specific grid point $\overrightarrow{V}^i$, resp. material point $\overrightarrow{V}^p$.

Even though linear momentum is conserved by construction through Eq. (17), it is to note that the present lumped mass approach makes the nodal kinetic energy, computed from nodal masses in Eq. (6) and nodal velocities in Eq. (16), to be less than the kinetic energy defined on material points [9, 10, 16, 38, 39]. In line with the absence of nodal history, this feature nevertheless does not necessarily hinder overall conservation of energy in itself, see [16] and a simple translation illustration under free fall in the forthcoming Section 4.1 and Figure 3.

The Eqs. (16)-(18) are however unable to correctly infer nodal velocities from the ones of material points in a number of situations. Considering for instance a case where a grid node i would be under the influence (through $N_i(\overrightarrow{x}^p) \neq 0$) of just one material point p, one would have $\overrightarrow{V}^i = \overrightarrow{V}^p$ no matter the position offset between i and p, which is in particular incorrect if velocities are to correspond to an affine field, as discussed in [20]. As such, it has been proposed in [20], and later coined as a Taylor-PIC (TPIC) strategy in [21], to use a gradient-based extrapolation, i.e. a first order Taylor expression, for a better description of affine velocity fields from material point values:

$$\boldsymbol{V}^{\mathbb{G}} = \boldsymbol{P}_m^{\star G} \boldsymbol{V}^{\mathbb{P}} + \boldsymbol{W}^{\mathbb{G}} \tag{19}$$

$$(W^{\mathbb{G}})_{ik} = \sum_p \sum_j (P_m^{\star G})_{ip} (\nabla \overrightarrow{V}^p)_{jk} (b_i^p)_j \tag{20}$$

$$\overrightarrow{b}_i^p = \overrightarrow{x}^i - \overrightarrow{x}^p \tag{21}$$

where Eq. (19) introduces into Eq. (17) an additional term that combines the branch vector between i and p, $\overrightarrow{b}_i^p$, with the velocity gradient $\boldsymbol{\nabla}\overrightarrow{V}^p$. The latter is stored for every material point after being computed in a previous iteration as:

$$\boldsymbol{\nabla}\overrightarrow{V}^p = \sum_i \overrightarrow{\nabla N}_i(\overrightarrow{x}^p) \otimes \overrightarrow{V}^i \tag{22}$$

A rotational ensemble motion on a set of material points is an example of an affine velocity field, with an antisymmetric $\boldsymbol{\nabla}\overrightarrow{V}^p$, and the limitations of the initial Eq. (17) for projecting velocity fields naturally appear in this case, with a loss of angular momentum that is actually due to the lumped mass approach of Eqs. (4)-(6) and (17) [16, 17, 18]. While keeping the convenient lumped mass transport Eq. (6), it has been proposed in [17] as an APIC formulation to explicitly consider angular momentum in the transport Eq. (17) through additional terms at the material point level, namely:

$$\boldsymbol{V}^{\mathbb{G}} = \boldsymbol{P}_m^{\blacktriangleright G}\boldsymbol{V}^{\mathbb{P}} + \boldsymbol{U}^{\mathbb{G}} \tag{23}$$

$$(U^{\mathbb{G}})_{ik} = \sum_p \sum_j (P_m^{\blacktriangleright G})_{ip}(C^p)_{jk}(b_i^p)_j \tag{24}$$

$$\boldsymbol{C}^p = \boldsymbol{B}^p(\boldsymbol{D}^p)^{-1} \tag{25}$$

$$\boldsymbol{B}^p = \sum_i N_i(\overrightarrow{x}^p)\widetilde{\overrightarrow{V}}^i \otimes \overrightarrow{b}_i^p \tag{26}$$

$$\boldsymbol{D}^p = \sum_i N_i(\overrightarrow{x}^p)\,\overrightarrow{b}_i^p \otimes \overrightarrow{b}_i^p \tag{27}$$

Here, the affine augmentation of Eqs. (23)-(27) (with a more complex version proposed in [18]) embeds every material point with an angular velocity-like quantity $\boldsymbol{C}^p$ which is computed from an angular momentum-like quantity $m^p\boldsymbol{B}^p$ and an inertia-like quantity $m^p\boldsymbol{D}^p$ (or its inverse). That angular

velocity-like quantity is then combined in Eq. (24) with $\overrightarrow{b}_i^p$, giving an additional rotational term, $\boldsymbol{U}^{\mathbb{P}}$, in the velocity field to be mapped into the grid in Eq. (23). In such an APIC formulation, both matrices $\boldsymbol{B}^p, \boldsymbol{D}^p$ are stored for every material point in order to form $\boldsymbol{C^p}$ and Eq. (26) actually corresponds to an end-of-step update (from updated nodal velocities $\widetilde{\overrightarrow{V}}^i$ precised in Section 3.3) for $\boldsymbol{B^p}$, providing its value that will be used in a subsequent step.

A proper initialization of $\boldsymbol{\nabla}\overrightarrow{V}^p$, for the Taylor formulation, or $\boldsymbol{B}^p$, for the APIC formulation, is however critical in case of assigned initial velocities to material points. For this purpose, the initial value of $\boldsymbol{\nabla}\overrightarrow{V}^p$ is determined by repeating Eqs. (19) to (22) until $\boldsymbol{\nabla}\overrightarrow{V}^p$ converges. Because it can usually be observed that the APIC $\boldsymbol{C}^p$ is numerically similar to velocity gradient, as suggested by the formal similarities between Eq. (20) and (24), the former is incidentally initialized to the same value of the latter.

3.3 From nodal acceleration to particle kinematics

Whether $\overrightarrow{V}^i$ is affine-augmented or not, an updated nodal velocity field $\widetilde{\overrightarrow{V}}^i$ is subsequently obtained after solving Eq. (1) in its discrete form Eq. (4) for nodal acceleration and applying a simple, explicit, finite difference scheme in time:

$$\widetilde{\overrightarrow{V}}^i = \overrightarrow{V}^i + \overrightarrow{a}^i \cdot \Delta t \tag{28}$$

Eq. (28) naturally leaves aside possible grid points serving as Dirichlet boundary conditions with constant velocity components ($a_k^i = 0$ and V_k^i fixed to a chosen $V_k^i|_{\text{lim}}$ for one or several directions k). Time step Δt is computed as a fraction of the characteristic time obtained with the material and mesh parameters:

$$\Delta t = a_\tau l_{cell} \sqrt{\frac{\rho}{E}} \tag{29}$$

with l_{cell} the size of a mesh element, ρ the mass density of the material, E its Young modulus for the subsequent examples considering, at least in part, Hooke's law, and a_τ a proportion coefficient. The latter is determined empirically in each specific case, in order to ensure a stable simulation.

In line with the absence of nodal history, the updated velocity field $\widetilde{\overrightarrow{V}}^i$ is only temporary, before being overwritten at the beginning of the next iteration with the one transported from the material points with one or another mapping among Eqs. (17), (19) or (23). It rules however material point kinematics during current time step, where different MPM strategies can again be found for this purpose. Consistently with the above use of a mapping matrix $\boldsymbol{P}_m^{\star G}$, this procedure can be considered as equivalent to solving the following equation for $\boldsymbol{V}^{\mathbb{P},new}$:

$$\begin{aligned}\boldsymbol{P}_m^{\star G}\boldsymbol{V}^{\mathbb{P},new} &= \widetilde{\boldsymbol{V}}^{\mathbb{G}} \\ &= \begin{cases} \boldsymbol{P}_m^{\star G}\boldsymbol{V}^{\mathbb{P}} + \boldsymbol{W}^{\mathbb{G}} + \boldsymbol{a}^{\mathbb{G}}\Delta t & \text{for TPIC} \\ \boldsymbol{P}_m^{\star G}\boldsymbol{V}^{\mathbb{P}} + \boldsymbol{U}^{\mathbb{G}} + \boldsymbol{a}^{\mathbb{G}}\Delta t & \text{for APIC} \\ \boldsymbol{P}_m^{\star G}\boldsymbol{V}^{\mathbb{P}} + \boldsymbol{a}^{\mathbb{G}}\Delta t & \text{for other strategies} \end{cases} \end{aligned} \tag{30}$$

In the general case with $N_{mp} \neq N_{gp}$ and a non-square $\boldsymbol{P}_m^{\star G}$, the set of solutions to Eq. (30) can be infinite, justifying the existence of several strategies. The most direct solution to Eq. (30) was inspired by the PIC method and essentially maps the velocity from grid points back to the material points with Eq. (11), leading to:

$$\boldsymbol{V}_{PIC}^{\mathbb{P},new} = \boldsymbol{G}^{\star P}\widetilde{\boldsymbol{V}}^{\mathbb{G}} \tag{31}$$

$$\boldsymbol{x}_{PIC}^{\mathbb{P},new} = \boldsymbol{x}^{\mathbb{P}} + \boldsymbol{V}_{PIC}^{\mathbb{P},new} \times \Delta t \tag{32}$$

Although this strategy is straightforward to implement, it is known to improperly filter the velocity which leads to an unrealistic damping [22, 40]. One can note for instance that Eq. (31) erases the individual memory for material point velocity one could expect since $\boldsymbol{V}^{\mathbb{P},new}$ is not directly integrated from $\boldsymbol{V}^{\mathbb{P}}$. Moreover, it has been demonstrated in [22] that $\boldsymbol{V}^{\mathbb{P}}_{PIC}$ is the solution to Eq. (30) that minimizes $\boldsymbol{V}^{\mathbb{P}}$.

Aiming to achieve a better energy conservation than PIC, the FLIP method [9] has been proposed with alternative time velocity increments from the same nodal acceleration. In a FLIP approach, the latter is actually transported into $\mathbb{P}$ towards an incremental computation of the material points' velocity, reinstating a direct, individual, link between $\overrightarrow{V}^{p,new}$ and $\overrightarrow{V}^{p}$:

$$\boldsymbol{V}^{\mathbb{P},new}_{FLIP} = \boldsymbol{V}^{\mathbb{P}} + \boldsymbol{G}^{\blacktriangleright P}\boldsymbol{a}^{\mathbb{G}} \times \Delta t \tag{33}$$

Thinking in terms of solution to Eq. (30), it was established in [22] that $\boldsymbol{V}^{\mathbb{P}}_{FLIP}$ is the solution which minimizes the fluctuations in $\boldsymbol{V}^{\mathbb{P}}$, i.e. acceleration. However, FLIP still uses the same Eq. (32) as PIC to displace the material points, which can be recast as:

$$\begin{aligned} \boldsymbol{x}^{\mathbb{P},new} &= \boldsymbol{x}^{\mathbb{P}} + \boldsymbol{V}^{\mathbb{P},new}_{PIC} \times \Delta t \\ &= \boldsymbol{x}^{\mathbb{P}} + \boldsymbol{G}^{\blacktriangleright P}\left(\boldsymbol{V}^{\mathbb{G}} + \boldsymbol{a}^{\mathbb{G}}\Delta t\right)\Delta t \end{aligned} \tag{34}$$

Since $\boldsymbol{G}^{\blacktriangleright P}\boldsymbol{V}^{\mathbb{G}} \neq \boldsymbol{V}^{\mathbb{P}}$, one could consider from comparing Eq. (34) and (33) the FLIP scheme implies an unnatural description of the material points motion in the sense that the time derivative of $\overrightarrow{x}^{p}_{FLIP}$ is different from the FLIP material point velocity (even though this was intended in [9]):

$$\overrightarrow{V}^{p}_{FLIP} \neq \frac{d\overrightarrow{x}^{p}_{FLIP}}{dt} \tag{35}$$

As such, $\overrightarrow{V}^p_{PIC}$ in Eq. (32) has been replaced in [41] with $\overrightarrow{V}^p_{FLIP}$, forming another NFLIP strategy (Naturally modified FLIP), as coined by [19]:

$$\boldsymbol{x}^{\mathbb{P},new}_{NFLIP} = \boldsymbol{x}^{\mathbb{P}} + \boldsymbol{V}^{\mathbb{P},new}_{FLIP} \times \Delta t \tag{36}$$

As previously mentioned, FLIP and NFLIP conserve better energy than PIC but they are more prone to instabilities. A common practice is thus to blend FLIP or NFLIP with PIC in order to make the simulation more stable [13, 42, 43, 44], based on a blending parameter P_{PIC}. These blended strategies are denoted FLIPX and NFLIPX, where $X = 1 - P_{PIC}$ indirectly highlights the proportion of PIC. Namely, a FLIPX velocity is given by:

$$\begin{aligned} \boldsymbol{V}^{\mathbb{P},new}_{FLIPX} &= X\boldsymbol{V}^{\mathbb{P},new}_{FLIP} + (1-X)\boldsymbol{V}^{\mathbb{P},new}_{PIC} \\ &= (1-P_{PIC})\boldsymbol{V}^{\mathbb{P},new}_{FLIP} + P_{PIC}\boldsymbol{V}^{\mathbb{P},new}_{PIC} \end{aligned} \tag{37}$$

Such a FLIPX velocity would also be used by the NFLIPX strategy to compute new positions for the material points, similarly to NFLIP Eq. (36):

$$\boldsymbol{x}^{\mathbb{P},new}_{NFLIPX} = \boldsymbol{x}^{\mathbb{P}} + \boldsymbol{V}^{\mathbb{P},new}_{FLIPX} \times \Delta t \tag{38}$$

On the other hand, it has been proposed in [22, 23, 45] , that the blended velocity update of Eq. (37) should be accompanied with deeper changes in position updates that would eventually conform neither Eq. (34) nor Eq. (38) but include a second order term.

Such a second order position update is actually found in the XPIC(m) strategy [22], which first aims for a definition of the updated velocity field $\boldsymbol{V}^{\mathbb{P},new}$ that shows minimal variations from its previous value, in a smoothed

version $\boldsymbol{V_{sm}}^{\mathbb{P}}$:

$$\boldsymbol{V_{sm}}^{\mathbb{P}} = (\boldsymbol{I} - (\boldsymbol{I} - \boldsymbol{G}^{\star P}\boldsymbol{P}_m^{\star G})^{m-1})\boldsymbol{V}^{\mathbb{P}} \tag{39}$$

with $\boldsymbol{I}$ the $(N_{mp} \times N_{mp})$ identity matrix and m a chosen smoothing parameter. The resulting solution to Eq. (30) is then [22]:

$$\boldsymbol{V}_{XPIC(m)}^{\mathbb{P},new} = \boldsymbol{V}^{\mathbb{P}} - (\boldsymbol{I} - \boldsymbol{G}^{\star P}\boldsymbol{P}_m^{\star G})^m \boldsymbol{V}^{\mathbb{P}} + \boldsymbol{G}^{\star P}\boldsymbol{a}^{\mathbb{G}} \times \Delta t \tag{40}$$

Material points are then moved using a second order time integration scheme:

$$\boldsymbol{x}_{XPIC(m)}^{\mathbb{P},new} = \boldsymbol{x}^{\mathbb{P}} + \boldsymbol{G}^{\star P}\widetilde{\boldsymbol{V}}^{\mathbb{G}}\Delta t - \frac{1}{2}\left(\boldsymbol{G}^{\star P}\boldsymbol{a}^{\mathbb{G}}\Delta t^2 + (\boldsymbol{I} - \boldsymbol{G}^{\star P}\boldsymbol{P}_m^{\star G})^m \boldsymbol{V}^{\mathbb{P}}\Delta t\right) \tag{41}$$

Note that with $m = 1$, a XPIC(1) strategy is equivalent to PIC, see Eq. (40) vs Eq. (31) for velocity, and that the corresponding position update of Eq. (41) has latter been further modified in [23] and be proposed as an optimal position update for a PIC framework.

Most of the motion integration strategies mentioned above have been declined into their affine-augmented version (APIC, AFLIP and ANFLIP, AFLIPX and ANFLIPX) just by using Eq. (23) to express the grid points velocity $\boldsymbol{V}^{\mathbb{G}}$ depending on the material points velocity $\boldsymbol{V}^{\mathbb{P}}$. Similarly, a TFLIP counterpart to TPIC is defined as using Eq. (19) for initializing grid velocities from material point ones, before using all other FLIP equations.

Since the FLIP logic is to incrementally update the material point velocity rather than over-writing it through a grid interpolation, one could expect similar time update equations for the matrices $\boldsymbol{B}^p, \boldsymbol{D}^p, \boldsymbol{C}^p$ for AFLIP or $\boldsymbol{\nabla}\overrightarrow{V}^p$ for TFLIP. Since it is instead chosen to use the same equations than APIC or

Motion integration operations / Motion integration strategy	$\boldsymbol{V}^{\mathbb{G}}$ measure from current $\boldsymbol{V}^{\mathbb{P}}$	$\boldsymbol{V}^{\mathbb{P}}$ update from $\boldsymbol{a}^{\mathbb{G}}$, i.e. updated $\boldsymbol{V}^{\mathbb{G}}$	$\boldsymbol{x}^{\mathbb{P}}$ update
PIC [8]	$\boldsymbol{P}_m^{G} \boldsymbol{V}^{\mathbb{P}}$ Eq. (17)	$\boldsymbol{V}^{\mathbb{P}}_{PIC}$ Eq. (31)	$\boldsymbol{x}^{\mathbb{P}}_{PIC}$ Eq. (32)
APIC [17]	$\boldsymbol{P}_m^{G} (\boldsymbol{V}^{\mathbb{P}} + \boldsymbol{U}^{\mathbb{P}})$ Eq. (23)		
TPIC [20, 21]	$\boldsymbol{P}_m^{G} (\boldsymbol{V}^{\mathbb{P}} + \boldsymbol{W}^{\mathbb{P}})$ Eq. (19)		
FLIP [9, 10]	$\boldsymbol{P}_m^{G} \boldsymbol{V}^{\mathbb{P}}$ Eq. (17)	$\boldsymbol{V}^{\mathbb{P}}_{FLIP}$ Eq. (33)	
AFLIP [17, 19]	$\boldsymbol{P}_m^{G} (\boldsymbol{V}^{\mathbb{P}} + \boldsymbol{U}^{\mathbb{P}})$ Eq. (23)		
TFLIP [20]	$\boldsymbol{P}_m^{G} (\boldsymbol{V}^{\mathbb{P}} + \boldsymbol{W}^{\mathbb{P}})$ Eq. (19)		
NFLIP [41, 19]	$\boldsymbol{P}_m^{G} \boldsymbol{V}^{\mathbb{P}}$ Eq. (17)		$\boldsymbol{x}^{\mathbb{P}}_{NFLIP}$ Eq. (36)
ANFLIP	$\boldsymbol{P}_m^{G} (\boldsymbol{V}^{\mathbb{P}} + \boldsymbol{U}^{\mathbb{P}})$ Eq. (23)		
FLIPX	$\boldsymbol{P}_m^{G} \boldsymbol{V}^{\mathbb{P}}$ Eq. (17)	$X\,\boldsymbol{V}^{\mathbb{P}}_{FLIP} + (1-X)\,\boldsymbol{V}^{\mathbb{P}}_{PIC}$ Eq. (37)	$\boldsymbol{x}^{\mathbb{P}}_{PIC}$ Eq. (32)
AFLIPX	$\boldsymbol{P}_m^{G} (\boldsymbol{V}^{\mathbb{P}} + \boldsymbol{U}^{\mathbb{P}})$ Eq. (23)		
NFLIPX	$\boldsymbol{P}_m^{G} \boldsymbol{V}^{\mathbb{P}}$ Eq. (17)		$\boldsymbol{x}^{\mathbb{P}}_{NFLIPX}$ Eq. (38)
ANFLIPX	$\boldsymbol{P}_m^{G} (\boldsymbol{V}^{\mathbb{P}} + \boldsymbol{U}^{\mathbb{P}})$ Eq. (23)		
XPIC(m) [22]	$\boldsymbol{P}_m^{G} \boldsymbol{V}^{\mathbb{P}}$ Eq. (17)	$\boldsymbol{V}^{\mathbb{P}}_{XPIC(m)}$ Eq. (40)	$\boldsymbol{x}^{\mathbb{P}}_{XPIC(m)}$ Eq. (41)

Table 1: Designations of the different MPM motion integration strategies with their chosen underlying equations

TPIC to express those matrices, with a direct transport from the grid, a better terminology could adopt AFLIP/PIC and TFLIP/PIC notations, which is nevertheless not done here for the sake of simplicity.

Table 1 summarizes all these motion integration strategies.

3.4 Stress update scheme

Following [10, 46] and for the sake of simplicity, a small deformations (notwithstanding possible large displacements) strain tensor $\boldsymbol{\epsilon}$ is defined from its rate

$\dot{\boldsymbol{\epsilon}}$:

$$\dot{\boldsymbol{\epsilon}} = \frac{1}{2}\left(\nabla\overrightarrow{V} + (\nabla\overrightarrow{V})^T\right) \tag{42}$$

and the corresponding increment $\boldsymbol{d\epsilon}$ is related with $\boldsymbol{d\sigma}$ through a material-specific constitutive relation for non-viscous solids, taking into account history-dependent variables when necessary. If necessary, large strain-compliant more general formulations based on deformation gradient and, possibly, objective stress rate (e.g., of Jaumann type) can be found for instance in [12, 23, 24, 25]. The MPM algorithm naturally applies the incremental stress update at each material point, after computing a finite $\Delta\boldsymbol{\epsilon}^p$ from nodal velocities $\overrightarrow{V}^i$:

$$\Delta\boldsymbol{\epsilon}^p = \dot{\boldsymbol{\epsilon}}^p \Delta t \tag{43}$$

$$\dot{\boldsymbol{\epsilon}}^p = \frac{1}{2}\sum_i \left(\boldsymbol{A}^{ip} + \boldsymbol{A}^{ip^T}\right) \tag{44}$$

$$\boldsymbol{A}^{ip} = \overrightarrow{\nabla N_i}(\overrightarrow{x}^p) \otimes \overrightarrow{V}^i \tag{45}$$

with $\boldsymbol{A}^{ip}$ defined for each pair of grid point i and material point p. Due to the intricate relations between nodal velocity or even material point velocity and time increments of material positions, it is again to note that such a strain tensor may not be directly interpreted from the relative displacements among material points.

The location of that stress update in the sequence of the MPM algorithm is a matter of choice, with no prior justification for an execution before or after the internal force computation of Eq. (8). The two immediate choices of updating stresses before, resp. after, solving the equation of motion have been denoted USF ("Update Stress First"), resp. USL ("Update Stress Last") and analysed in [38] for their consequences on energy conservation. It was shown

therein the USF scheme is more likely to conserve energy in average through a mutual cancellation of two sources of errors that affect both material points kinetic and strain energies at each MPM iteration. It has also been observed herein during preliminary simulations that USF is better suited to use with linear shape functions, unlike USL which exacerbates cell crossing instability issues in this case.

The stress update scheme was further studied in [47], where a variation of the USL scheme initially proposed in [48] was coined MUSL ("Modified Update Stress Last") and found to be very similar to the USF scheme in its results. In its definition, the MUSL scheme executes the stress update likewise to USL, after relating internal forces to nodal acceleration in Eq. (8), but uses for this purpose a strain increment computed from updated nodal velocities, with $\widetilde{\overrightarrow{V}}^i$ replacing $\overrightarrow{V}^i$ in Eq. (45). In line with these observations, [47] proposed the USAVG ("Update Stress Averaged") scheme which conserves almost perfectly the energy, at the cost of computing the material behaviour twice in the same MPM iteration.

In the present manuscript, a USF formulation is adopted unless otherwise specified, including an updated stress value in Eq. (8). While a USL choice may be more usual in the literature, the USF scheme is herein chosen since it was shown in [38] that it can be completely conservative energy-wise, while the USL scheme is systematically strictly dissipative.

4 Energy conservation on heuristic stiff examples for different motion integration strategies

This section determines the influence of the different strategies detailed in the previous section on the MPM capacity to conserve energy, for two simple

cases with basic configurations that enable a ground-truth comparison. Proposed results are necessarily affected by the explicit first order time integration scheme (except for the XPIC case) chosen in the above equations for velocity or positions update, e.g., Eq. (28). Other integration schemes, e.g., proposed in [18], especially if symplectic, would improve energy conservation.

4.1 Energy conservation in translation

The first case extends a similar analysis in [16, 18] (that is also directly considered in Appendix B.1) by simulating a bouncing cube conforming a translational motion under gravity, with a gravitational acceleration $g = 9.81$ m/s^2, and illustrated in Figure 2 for its initial configuration. The mesh contains $5 \times 5 \times 13$ cubic cells along the different axes, each with a side of $l_{cell} = 20$ cm. Each cell located between $x_3 = 1$ m and $x_3 = 2$ m contains initially $2^3 = 8$ material points located at the Gauss-Legendre integration points, for a total of $8 \times 125 = 1000$ material points. The normal velocity is imposed to zero on all grid points on the left, right, front, back and bottom boundaries.

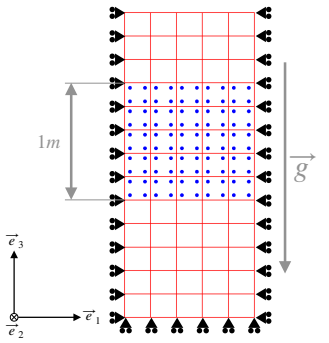


Fig. 2: Simulation setup of the bouncing cube example (plane view)

Hyperelastic Hooke's law is assigned to the solid material (e.g., to describe the deformations of the cube upon impact on the floor), with Young's modulus being set to $E = 12.84\ MPa$ and Poisson's ratio to $\nu = 0.16$. The density of the material is set to $\rho = 1748\ kg \times m^{-3}$, meaning that each material point will have a mass of $m_p \approx 1.75\ kg$. Using Eq. (29) along with $a_\tau \approx 5 \times 10^{-2}$ or $a_\tau \approx 5 \times 10^{-3}$, the time step is chosen as $\Delta t = 1.17 \times 10^{-4}\ s$ or $\Delta t = 1.17 \times 10^{-5}\ s$.

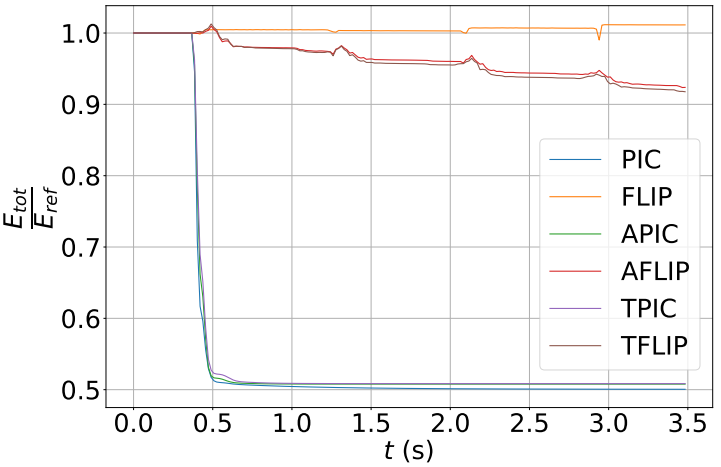
In line with the conservative nature of the problem, energy should theoretically conserve and just converts during time between elastic strain energy E_{el}, kinetic energy E_k and gravitational energy E_p (with $x_3 = 0$ serving as reference for $E_p = 0$). Measuring those quantities over all material points and using Einstein's convention for summing over repeated indices, we namely have:

$$E_{el} = \sum_{p \in \{p\}} \frac{v^p}{2} \left(\sigma_{ij} \epsilon_{ij}\right) \tag{46}$$

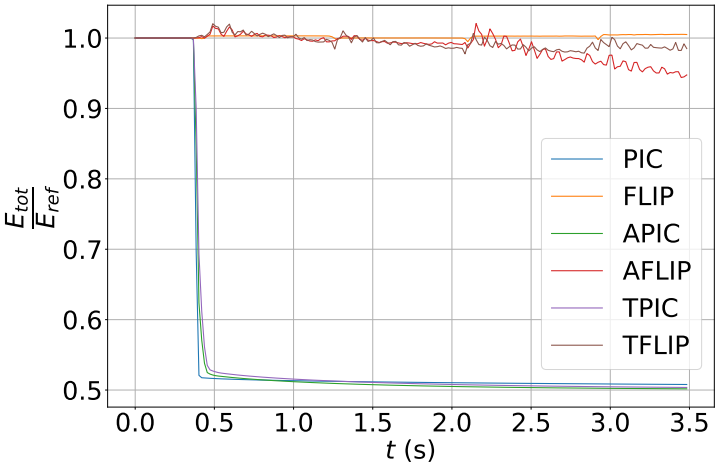
$$E_k = \sum_{p \in \{p\}} \frac{1}{2} m^p \| \overrightarrow{V}^p \|^2 \tag{47}$$

$$E_g = \sum_{p \in \{p\}} m^p g x_3^p \tag{48}$$

The following Figures 3 and 4 illustrate the obtained energy balance in MPM for different motion integration strategies, namely PIC, FLIP, APIC, AFLIP, TPIC, and TFLIP; when using the USF stress update scheme (USL results being also presented for the same simulations in Appendix A). The NFLIP formulation was also considered during preliminary simulations with a coarser mesh and linear shape functions and was observed to yield unrealistic results (e.g., segregation of material points within the cube cells for various time steps ranging from $a_\tau \approx 5 \times 10^{-6}$ to 5×10^{-2}), as already reported in [23].



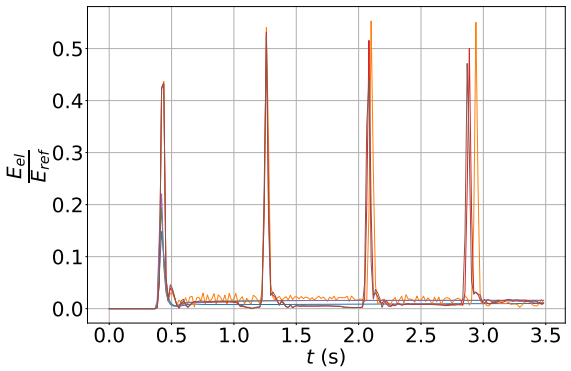
(a) $a_\tau = 5 \times 10^{-2}$



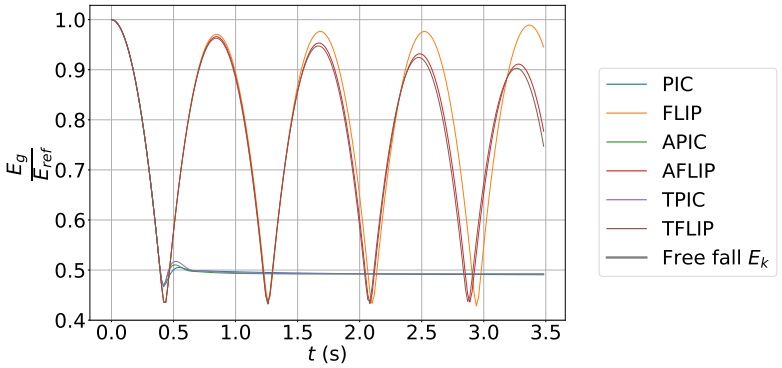
(b) $a_\tau = 5 \times 10^{-3}$

Fig. 3: Total energy for the bouncing cube example simulated with different motion integration strategies for different time steps ($E_{ref} \approx 25.7\ kJ$)

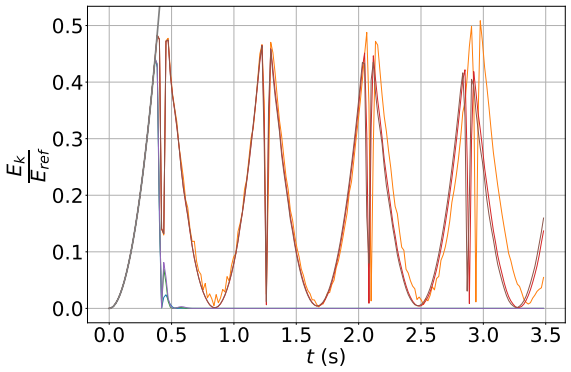
Figure 3 first shows the total energy $E_{tot} = E_{el} + E_k + E_g$ relative to its initial value of $E_{ref} \approx 25.7\ kJ$, for the two different time steps. At first, during the initial free fall phase with neither strains nor stress, all results are strictly equivalent and theoretically correct, whatever the motion integration strategy. After the first contact with the floor, all PIC-based cases (PIC, APIC and TPIC) dissipate so much energy that the cube does not bounce, no



(a) Elastic energy E_{el}



(b) Gravitational energy E_g



(c) Kinetic energy E_k (the grey curve materializes the theoretical free fall case)

Fig. 4: Energy balance for the bouncing cube example simulated with different MPM motion integration strategies ($E_{ref} \approx 25.7\ kJ$, $a_\tau = 5 \times 10^{-2}$)

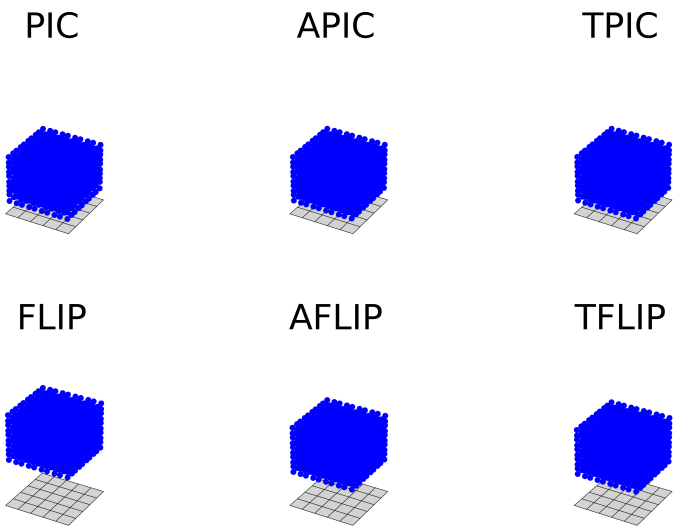


Fig. 5: Material points positions at $t \approx 3.5\ s$ ($E_{ref} \approx 25.7\ kJ$, see a corresponding video as a supplementary material)

matter the time step used. On the other hand, FLIP-based strategies appear to be much more conservative, allowing for several bounces. In more details, FLIP conserves energy almost perfectly, slightly increasing it, while AFLIP and TFLIP let E_{tot} decrease by approximately 3% between each bounce (these strategies are shown to be equivalent to FLIP when using the USL scheme in Appendix A), although these defects are limited when using a lower time step.

Figure 4 then discriminates between the various energy terms in E_{tot} when using $a_\tau = 5 \times 10^{-2}$. PIC-induced artificial damping therein appears both in terms of elastic energy E_{el} (Figure 4 (a)) with no energy oscillations after impacting the floor, and in terms of gravitational energy E_g (Figure 4 (b)) with the height of the cube reaching its final value after the first contact with the floor. With FLIP-based strategies, the cube continues to deform even during its ascending phase, especially for FLIP, and the material points almost bounce

back to their initial heights. In fact, after the last bounce, the FLIP set of material points bounces higher than its previous peak position, in correspondence to E_{tot} observations. As for E_k (Figure 4 (c)), all motion integration strategies reproduce the theoretical velocity obtained during the free fall part of the simulation. Then, all FLIP-based strategies give a similar variation rate of E_k, even though it is shifted in time because of the different bouncing altitudes the cube reaches. One can lastly note that, due to the MPM discretization and the extended support of present cubic B-spline shape functions (two cells wide for the boundary case), impacts and bounces occur as soon as the lowest material points reach $x_3 = 0.4\ m$.

On this very simple case involving translation only, FLIP thus appears a possible best candidate for conserving energy (both with present USF choice or with USL, see Appendix A).

4.2 Conservation of rotational kinetic energy and angular momentum

4.2.1 Simulation description

A second simplified case study includes a combination of rotational and linear motion, without gravity, which extends a somewhat similar previous analysis in [18] (also directly considered in Appendix B.2). Basically, a cube with a side $l_{cube} = 3\ m$ is thrown in space, i.e. is given an initial velocity with both a linear and rotational motion, with

$$\begin{aligned} \overrightarrow{V}_c &= \frac{V_c}{\sqrt{3}}(\overrightarrow{e}_1 + \overrightarrow{e}_2 + \overrightarrow{e}_3) \\ V_c &= 1.71 \text{ cm/s} \end{aligned} \tag{49}$$

the linear velocity of the cube, computed so that the cube stays within the mesh during the whole simulation, and

$$\begin{aligned} \overrightarrow{\omega} &= \omega\sqrt{\frac{2}{3}}(0.5\overrightarrow{e}_1 + 0.5\overrightarrow{e}_2 + \overrightarrow{e}_3) \\ \omega &= 0.108 \text{ rad/s} \end{aligned} \tag{50}$$

its angular velocity, computed so that the cube performs several revolutions during the simulation.

The whole space domain is covered by a mesh of cubic elements with a side length of $l_{cell} = l_{cube}/3 = 1\ m$, the solid cube thus spans over $l_{cube}^3 = 3^3 = 27$ mesh cells. Initially, the cube is located in a corner of the domain while including 2^3 material points per cell which are regularly spaced in the cells, and have initial velocities assigned in accordance with the desired linear and angular velocities mentioned in the above. The time step Δt is computed with Eq. (29) from the simulation parameters and a variable a_τ (equal to 5×10^{-2} unless otherwise specified, providing a time step of $5.83 \times 10^{-4}\ s$).

While the same elastic material properties are used here as in previous Section 4.1, inertial (centrifuge) effects are small enough to consider the cube as rigid. As a matter of fact, inertial effects can be quantified from the following dimensionless number C_a which is considerably small:

$$C_a = \frac{\rho\omega^2 R^2}{E} \approx 1.07 \cdot 10^{-5} \tag{51}$$

with $R = \frac{\sqrt{3}}{2} l_{cube}$ the radius of the circumscribed sphere. While being analog to the Cauchy number in fluid mechanics, C_a is built from $\rho\omega^2 R^2$ that rules stress quantities in a rotating elastic solid [49] and material stiffness E.

Strains being theoretically negligible, use can be made of rigid bodies equations, e.g., Eq. (52) below, with the consideration of angular momentum $\boldsymbol{I}\vec{\omega}$ (where $\boldsymbol{I}$ is the inertia matrix), in order to theoretically predict the movement. Since angular momentum is here constant in a body-attached local frame due to the absence of external loads, while $\boldsymbol{I}$ for a cube is a spherical tensor with a constant expression in any frame, angular velocity can also directly be considered as constant with its expression in global frame such as given in Eq. (50). The position of any material point p can thus be predicted over time instants being separated by a given Δt and corresponding to the MPM time discretization as follows:

$$\vec{V}^p = \vec{V}_c + \vec{\omega} \times (\vec{x}^p - \vec{x}^c) \tag{52}$$

$$\vec{x}^{p,new} = \vec{x}^p + \vec{V}^p \Delta t \tag{53}$$

with $\times$ denoting the cross product when being applied like in Eq. (52) to two vectors, $\vec{x}^p$ and $\vec{x}^{p,new}$ corresponding to the positions at two successive integration times, and

$$\vec{x}^c = \frac{\sum_p \vec{x}^p}{N_{mp}} \tag{54}$$

the position of the center of mass of the cube.

These will be compared with actual MPM results, together with the consideration of the traveled distance $d_c = \|\vec{x}^c(t) - \vec{x}^c(t_0)\|$, with a corresponding relative error as:

$$D_{err} = \left| \frac{d_c}{V_c t} - 1 \right| \tag{55}$$

The total angular momentum L^{tot} is also computed on the material points as per the following Eq. (56):

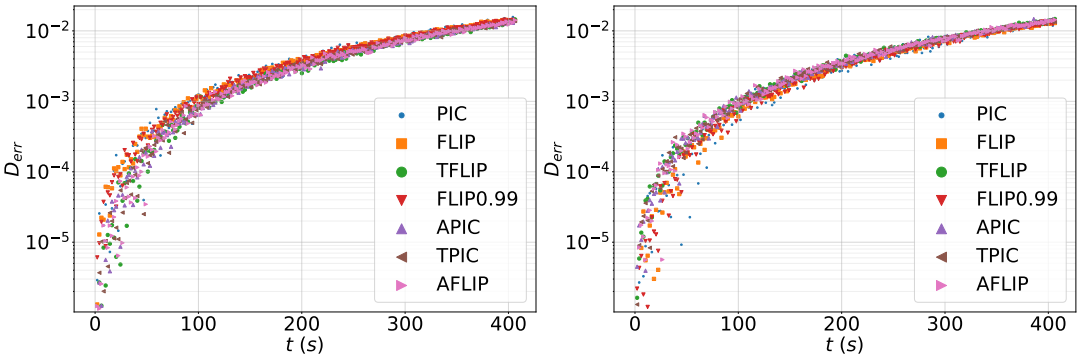
$$L^{tot} = \|\sum_p m^p(\overrightarrow{x}^p - \overrightarrow{x}^c) \times \overrightarrow{V}^p\| \tag{56}$$

in order to check to which extent MPM results do conform conservation of angular momentum. The Eq. (56), also used in, e.g., [15] is chosen to be consistent with the present "material point" point of view on the method discussed in previous Section 2.2, by considering that material points carry only linear velocity, and also consistent with the kinetic energy Eq. (47). On the other hand, with a "material domain" point of view, it could be considered that material points are centers of mass of some finite domain with a space-variable velocity field that could define a material point angular velocity and additional terms would enter Eq. (56). Those additional terms would account from angular momentum contributions corresponding to this material domain-scale velocity field, which is a choice done in, e.g., [17, 18], in connection with the consideration of an affine velocity field for the APIC transfers, and should in principle be accompanied with corresponding modifications to kinetic energy expression, necessitating additional terms in Eq. (47).

Without claiming for exhaustivity, the PIC, FLIP, FLIP0.99, APIC, AFLIP, TPIC and TFLIP motion integration strategies are herein tested.

4.2.2 Results

For what concerns first the (unconstrained) linear motion, and similarly to the free fall part of the previous case, all motion integration strategies provide accurate predictions as shown in Figure 6 where the traveled distance exhibits a negligible error with respect to its expected value.



(a) $a_\tau = 5 \times 10^{-2}$ (b) $a_\tau = 2.5 \times 10^{-2}$

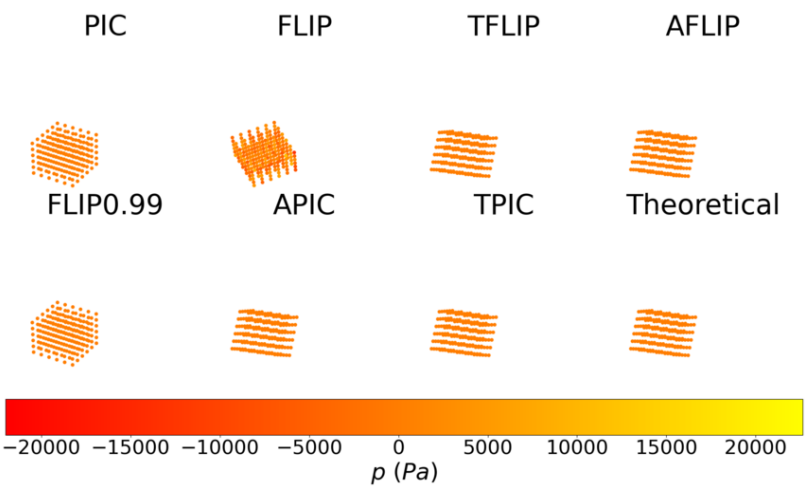
Fig. 6: Relative error D_{err} on the distance traveled by the center of the rotating cube

However, inspecting the individual movements of material points (Figure 7) or the time evolution of angular momentum (Figure 8) recalls how rotational motion gets lost for all strategies others than the affine-augmented or Taylor-based ones.

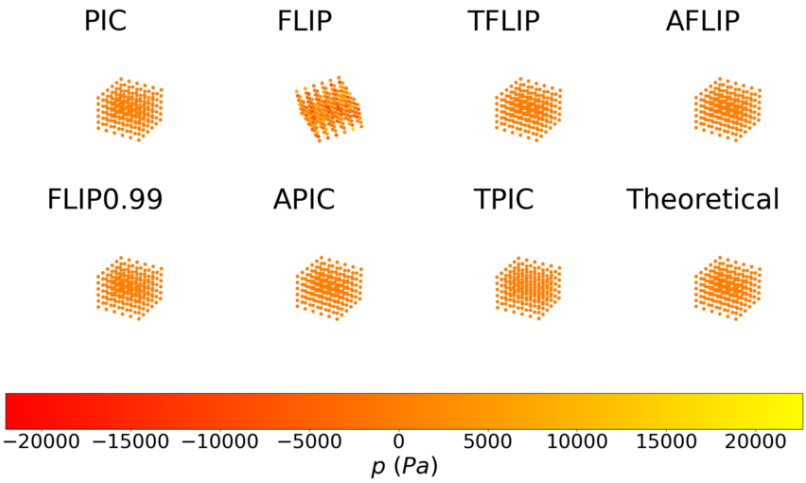
In particular, PIC-based non-affine-augmented strategies (PIC and FLIP0.99) cancel L^{tot} at the very beginning of the simulation (virtually immediately for PIC), in line with PIC Eq. (31) which induces a drastic averaging of velocity field that is unable to conserve the theoretical spatially-variable velocity field inherent to rotational motion.

The FLIP strategy allows the cube to somewhat preserve a slight rotation but it is clearly not the one imposed initially (Figure 7) and the angular momentum eventually reaches a small value after evolving quite erratically (Figure 8(a)). While those observations also apply even with a smaller time step ($a_\tau = 2.5 \times 10^{-2}$ vs 5×10^{-2}), the Figure 26 in Appendix B.2 shows how FLIP conserves much better angular momentum if the mesh is fine enough.

On the other hand, affine augmented strategies show themselves to be much more conservative regarding this rotational motion, though with influences of



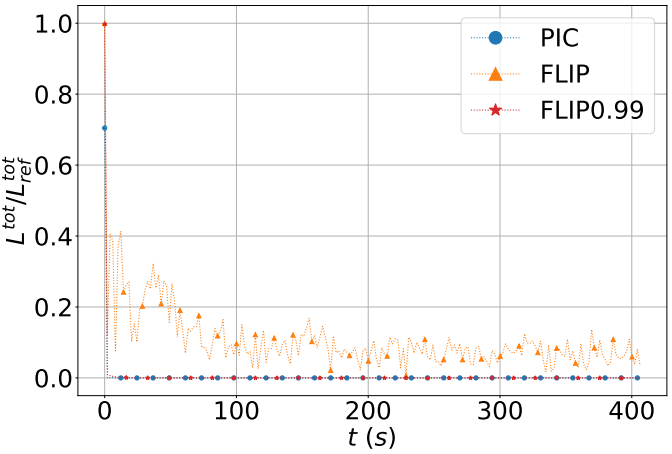
(a) At $t \approx 31\ s$



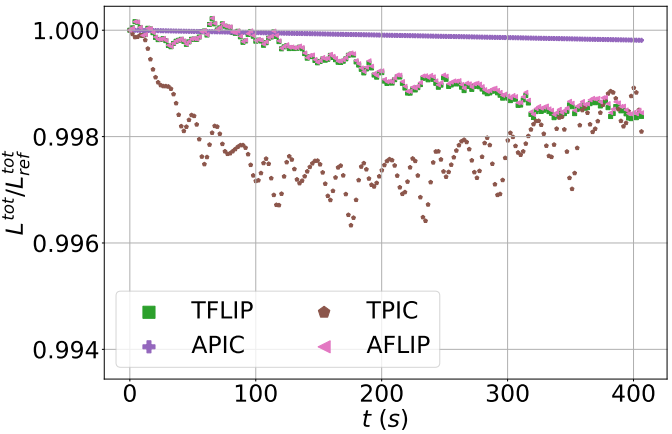
(b) At $t \approx 292\ s$

Fig. 7: Material point positions in the case of the rotating cube (see corresponding video as a supplementary material)

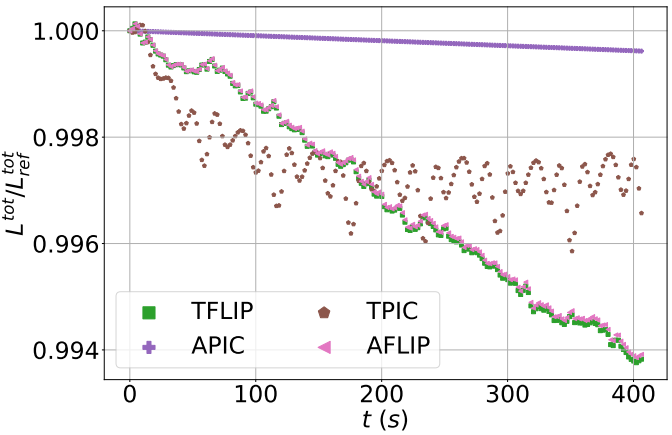
both the time step (Figure 8(b) and (c)) and the magnitude of the angular velocity $\vec{\omega}$ (Figure 9). More precisely, for the default case of $||\vec{\omega}|| = 0.108 rad/s$, APIC loses a minuscule amount of angular momentum at a rate of approximately 4.6×10^{-5} $\%/s$ for $a_\tau = 2.5 \times 10^{-2}$ (Figure 8 (b)), and roughly twice as much with a twice higher time step, i.e. $a_\tau = 5 \times 10^{-2}$ (Figure 8 (c)). AFLIP has



(a) PIC, FLIP, and FLIP0.99 and $a_\tau = 5 \times 10^{-2}$ (same results for $a_\tau = 2.5 \times 10^{-2}$)



(b) Affine- and Taylor-based cases and $a_\tau = 2.5 \times 10^{-2}$



(c) Affine- and Taylor-based cases and $a_\tau = 5 \times 10^{-2}$

Fig. 8: (Non-)Conservation of angular momentum for the rotating cube, with various motion integration strategies and time steps ($L^{tot}_{ref} \approx 7,413\ kg.m^2/s$).

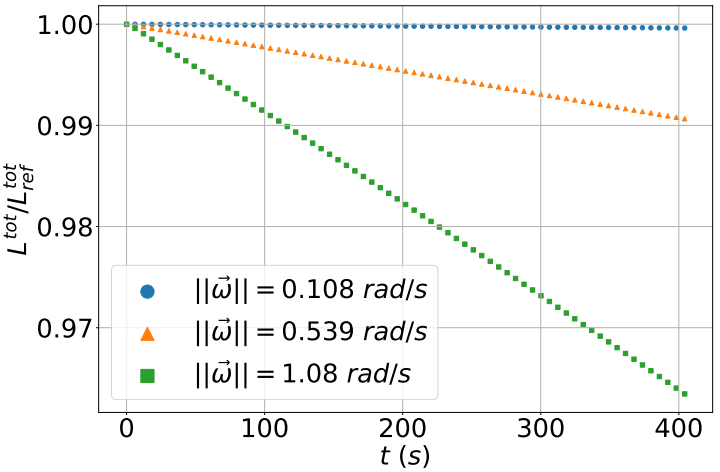


Fig. 9: Impact of $||\vec{\omega}||$ onto the (non-)conservation of the angular momentum for the rotating cube up to $t \approx 400\ s$ for APIC, with $a_\tau = 5 \times 10^{-2}$ $(7,413\ kg.m^2/s \leq L^{tot}_{ref} \leq 74,128\ kg.m^2/s)$.

a similar behaviour, with a steeper and more noisy decrease in L^{tot}, as well as a higher sensitivity to Δt. Also, one can see that TFLIP gives almost the same results as AFLIP, while TPIC differs from APIC by both its non-monotonous evolution and the amount of L^{tot} it dissipates, which is approximately 0.3% more than APIC at $t \approx 400\ s$. Various simulations conducted in [21] also showed that the TPIC conservation of some angular momentum was less than the APIC one.

For what concerns the influence of the angular velocity magnitude on the APIC results, Figure 9 shows that the faster the cube rotates, the faster its angular momentum is lost. As a matter of fact, multiplying the velocity by 5 makes the angular momentum dissipates approximately 24 times faster, and multiplying it by 10 makes it dissipate approximately 95 times faster, suggesting that the dissipation ratio evolves as the square of the angular velocity ratio. Keeping in mind the results given in Figure 8, one could accommodate this loss on L^{tot} by dividing the time step by the squared angular velocity ratio.

Although all affine augmented or Taylor-based motion integration strategies are able to accurately conserve the angular momentum, APIC stands out as the most predictable and performant in that regard. However, its incapacity to preserve the energy through an impact, as shown in Section 4.1, makes AFLIP or TFLIP emerges as the possibly optimal choice for integrating motion equations. The following section pursues the analysis on a more realistic MPM simulation setting.

5 Granular column collapse case study

The influence of the MPM motion integration strategy is now examined on a more realistic case in the form of a granular column collapse, which is a classical case study for granular materials being seen as solids prone to large displacements and often simulated with MPM [19, 32, 44, 50], among other numerical approaches [51, 52, 53, 54].

5.1 Simulation setup

5.1.1 General description

For simplicity, the problem is considered to be invariant in the out-of-plane direction, enabling a plane-strain numerical analysis with a 2D mesh geometry, likewise to [32, 44, 50, 51, 55, 56]. While a third principal stress is naturally still computed along the out-of-plane direction, the material point volume is computed in Eq. (9) as a surface and all energies will be given in J/m, i.e., normalized with respect to the out-of-plane length.

The simulation includes two steps: first, the material is let to settle vertically under gravity in a lateral displacement-constrained column; second, the material is triggered to collapse under its own weight after releasing the previous lateral constraint. While the first step basically consists in computing

a lithostatic stress state in the column which could also be directly assigned, its simulation will enable further discussion of the MPM results. Gravitational acceleration is taken to be of magnitude $g = 9.81\ m/s^2$.

5.1.2 Geometry, mesh and material points

The width of the column is considered to span over the x-axis while its height spans over the y-axis. The aspect ratio AR is defined as the column initial height L_y^{init} divided by its initial width $L_x^{init} = 1\ m$, the latter being the same for all simulations (Figure 10).

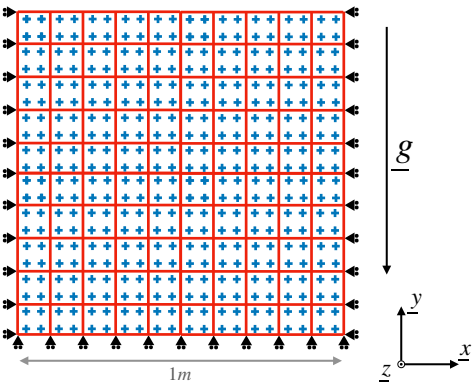


Fig. 10: Initial conditions for the granular column collapse simulation, illustrated for the specific case of $AR = 1$, $N_{mppc} = 4$ and $l_{cell} = 10\ cm$ (see text)

The mesh consists of square elements with a side of $l_{cell} \in \{10\ cm, 7.69\ cm, 5.88\ cm, 5\ cm\}$, i.e. $L_x^{init}/l_{cell} \in \{10, 13, 17, 20\}$. For the initial settling phase, the mesh cells just cover material extents:

$$W_{mesh}^{settling} = L_x^{init} \tag{57}$$

$$H_{mesh} = L_y^{init} = AR \times W_{mesh}^{settling} \tag{58}$$

with H_{mesh} the mesh height which will be constant for the whole simulation while $W_{mesh}^{settling}$ is the mesh width only during this settling phase. The second simulation phase will actually adopt a wider mesh for enabling a collapse on the right side. Taking advantage of the absence of mesh-history in MPM, the mechanical state of material points obtained after the settling phase is imported on the left-side of another mesh defined as a substantial enlargement of the settling one:

$$W_{mesh}^{collapse} = 6 \times W_{mesh}^{settling} \tag{59}$$

All cells contain initially N_{mppc} (a squared integer) material points. For the purpose of verifying quadrature rules, at least in the initial setup, those are located at the roots of Legendre's polynomials (in local coordinates), given in e.g. [57]. As an example, the roots of the second Legendre's polynomial correspond to the classical local positions of the points in each direction for $N_{mppc} = 2^2 = 4$:

$$\widetilde{s} = \pm\frac{1}{\sqrt{3}}, \quad \forall s \in \{x, y\} \tag{60}$$

In terms of boundary conditions, a nil orthogonal velocity is imposed at the left, right and bottom boundary nodes. For convenience, the left and right boundaries are referred to as walls, while the bottom boundary is referred to as a floor. A Coulomb friction condition is imposed on the floor, driven by a $\mu = 0.3$ coefficient.

Figure 10 illustrates the initial configuration of the simulation for $AR = 1$, $N_{mppc} = 4$ and $l_{cell} = 10\ cm$.

Mass density ρ	Young's modulus E	Poisson's ratio ν	Friction angle ϕ	Dilatancy angle ψ	Cohesion c	Tension cut-off T_{cut}
1,748 $kg.m^{-3}$	1.284×10^7 Pa	0.16	33.75 °	0 °	1 Pa	$\frac{c}{\tan\phi}$

Table 2: Parameters used with the Mohr-Coulomb model

5.1.3 Material parameters and numerical damping

In order to have a simple access to all energy quantities including material dissipation (see below), the collapse is simulated adopting the simplest constitutive model for frictional materials, i.e. the elastic-plastic Mohr-Coulomb model that combines Hooke's law for the elastic regime and Mohr-Coulomb perfect plasticity with a non-associated flow rule. Corresponding material parameters are calibrated (in the best possible way for this simple model) from the triaxial behavior of a real sandy soil, Camargue's sand studied in [58], and are given in Table 2.

As for the initial settling phase which can be seen as purely numerical, an elastic behaviour is chosen in order to prevent plastic deformations that would otherwise occur during the P-wave-like oscillations of the model from zero initial stresses (1 Pa, actually, for all material points in order to avoid edge-cases in the Mohr-Coulomb model) towards lithostatic equilibrium. When explictly mentioned for some of those settling simulations whose dynamics is not of interest, a fictitious Cundall's damping force [59] is also introduced in Eq. (5) in order to dissipate energy in a user-controlled way and converge to the intended lithostatic equilibrium. Such damping force is implemented as follows, being computed from the total nodal force supplemented by a damping parameter $D \geq 0$ (used with $D = 0.1$ when mentioned to be present) and oriented to be component-wise dissipative:

$$\overrightarrow{f}_{damp} = -D\|\overrightarrow{f}^i\|\overrightarrow{\mathrm{cws}}(\overrightarrow{V}^i) \tag{61}$$

$$\text{where} \qquad \text{cws}_k(\overrightarrow{u}) = \frac{u_k}{|u_k|} \qquad \text{for any vector } \overrightarrow{u} = (u_k), \quad k \in [\![1,3]\!] \tag{62}$$

5.1.4 Energy balance and other post-processing quantities

In the present usage of the Mohr-Coulomb constitutive model, the elastic energy E_{el} and the energy dissipated during plastic deformation, i.e. the plastic work W_{pl}, are computed incrementally as follows (with δ_{ij} the Kronecker's symbol):

$$d\epsilon_{ij}^{el} = \frac{1}{E}\left((1+\nu)d\sigma_{ij} - \nu d\sigma_{kk}\delta_{ij}\right) \tag{63}$$

$$dE_{el} = \sum_p \sigma_{ij} d\epsilon_{ij}^{el} v^p \tag{64}$$

$$dW_{pl} = -\sum_p \sigma_{ij}(d\epsilon_{ij} - d\epsilon_{ij}^{el}) v^p \tag{65}$$

with $E_{el}(0) = W_{pl}(0) = 0\ J/m$ in line with chosen initial conditions and $dW_{pl} < 0$ during plastic deformation by convention. For the numerical evaluation of Eq. (64), σ_{ij} is replaced by the average of its two values obtained before and after the constitutive update (i.e., at the very beginning and at the very end of a MPM iteration), which enables one to avoid finite size (of $d\epsilon_{ij}^{el}$) effects and keep an exact numerical integration by virtue of the linear relationship between σ_{ij} and ϵ_{ij}^{el}.

The energy dissipated by frictional forces W_{frict} during the collapse is also computed incrementally from:

$$dW_{frict} = \sum_i \overrightarrow{F}^i_{frict}.\overrightarrow{V}^i \Delta t \tag{66}$$

with $\overrightarrow{F}^i_{frict}$ the frictional force at grid point i and $dW_{frict} < 0$ during sliding by construction.

In the general absence of Cundall's damping, the energy balance of the system is then:

$$dE_k + dE_g + dE_{el} = dW_{pl} + dW_{frict} + dW_{MPM} \tag{67}$$

where dW_{MPM} (and its integral W_{MPM}) will quantify the numerical MPM energy error (dissipation, if negative) related with the chosen motion integration strategy, after being computed as:

$$dW_{MPM} = dE_k + dE_g + dE_{el} - (dW_{pl} + dW_{frict}) \tag{68}$$

As for the kinetic and gravitational energies, they are computed from Eq. (47) and Eq. (48) respectively. The difference in potential energy between two stable states (before and after the collapse) $|\Delta E_p^{stable}|$ will often be used as a reference value to normalize energy balance and is computed as follows:

$$|\Delta E_p^{stable}| = |\Delta E_g + \Delta E_{el}| \tag{69}$$

In addition to the above energy consideration, the time evolution of the front of the column is measured as a major insight on how the collapse unfolds, with an upper bound equal to the mesh right boundary $x = W_{mesh}^{collapse}$. Because the left edge does not move during the collapse, the latter front position is simply equal to the width of the column, L_x, and is tracked from the furthest material point along the $x-$axis and the mesh geometry (naturally considering the material to be present in a whole cell as long as at least one material point is inside):

$$L_x = \text{floor}\left(\frac{\max_{p\in\{p\}}(x^p)}{l_{cell}} + 1\right) l_{cell} \tag{70}$$

Because of this "voxelized" point of view, L_x logically increases by steps of l_{cell} as the column spreads to the right. Note that L_x is directly related to the so-called runout distance d_r:

$$d_r = L_x - L_x^{init} \tag{71}$$

Additionally, the normalized spreading length $\widetilde{L}$ and collapse time $\widetilde{t}$ are other common dimensionless quantities used to describe the dynamics of the collapse, both being computed as in [60]:

$$\widetilde{L} = \frac{d_r}{L_x^{init}} \tag{72}$$

$$\widetilde{t} = \frac{t}{\sqrt{\frac{AR \times L_x^{init}}{g}}} \tag{73}$$

Finally, in order to quantify how much the column is sheared, the second invariant ϵ_D of material point strain tensor $\boldsymbol{\epsilon}^p$ is also systematically monitored, using its classical expression:

$$\boldsymbol{\epsilon}^p_{dev} = \boldsymbol{\epsilon}^p - \frac{\text{tr}(\boldsymbol{\epsilon}^p)}{3}\boldsymbol{I_3} \tag{74}$$

$$\epsilon_D = \|\boldsymbol{\epsilon}^p_{dev}\| = \sqrt{(\epsilon_{dev})_{ij}(\epsilon_{dev})_{ij}} \tag{75}$$

l_{cell}	N_{mppc}	Δt	a_τ	Motion integration strategy	AR
10 cm	4	2.92×10^{-6} s	2.25×10^{-3}	Each of the following FLIP, PIC, APIC, FLIP0.9, AFLIP, TFLIP	1
		1.46×10^{-6} s	1.25×10^{-3}		
		3.65×10^{-7} s	3.125×10^{-4}		
		1.46×10^{-7} s	1.25×10^{-4}		
		7.29×10^{-8} s	6.25×10^{-5}		

Table 3: Parameters used to investigate the time step influence for different motion integration strategies (series S1, 30 simulations)

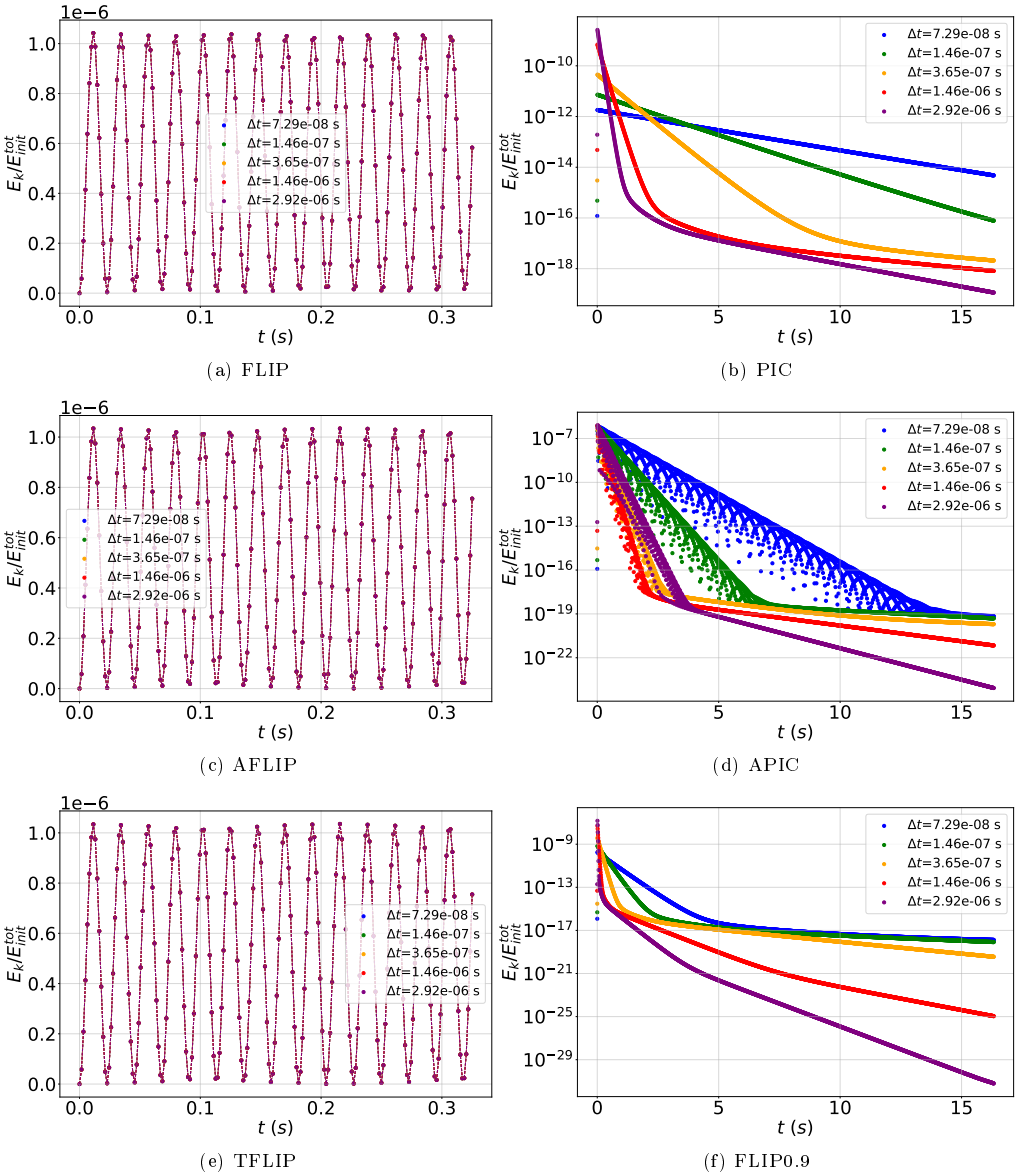
5.2 Time step influence for PIC damping

As a first interesting result, the chosen value for the time step appears to possibly have a strong influence on the settling dynamics, looking at the so-called S1 series of simulations that combine different motion integration strategies with a variable time step (Table 3).

Observing kinetic energy during this settling phase (Figure 11), one can see that for all FLIP-based strategies (Figure 11 (a), (c) and (e)) Δt has virtually no impact on the simulated dynamics: all tested values lead the column to oscillate indefinitely with the same period, in line with the conservative nature of FLIP and the elastic nature of the settling process.

However, a drastic influence of Δt appears with the PIC-based strategies (for the same Δt values being below the critical one, Figure 11 (b), (d) and (f)) where the dissipation rate of E_k is lower for smaller Δt. From this point of view, one can interpret the PIC damping as being even more artificial and numeric in nature than Cundall's damping of Eq. (61) since it is purely cumulative according to MPM iterations instead of being time-proportional to a given dissipative power as is the case for Cundall's damping which is introduced in the form of an ad-hoc additional force.

A more striking evidence of this very artificial nature of the PIC damping is presented in Figure 12 (b), where all values of W_{MPM} are nicely grouped together when plotted with respect to $t\Delta t$, demonstrating a direct dependence



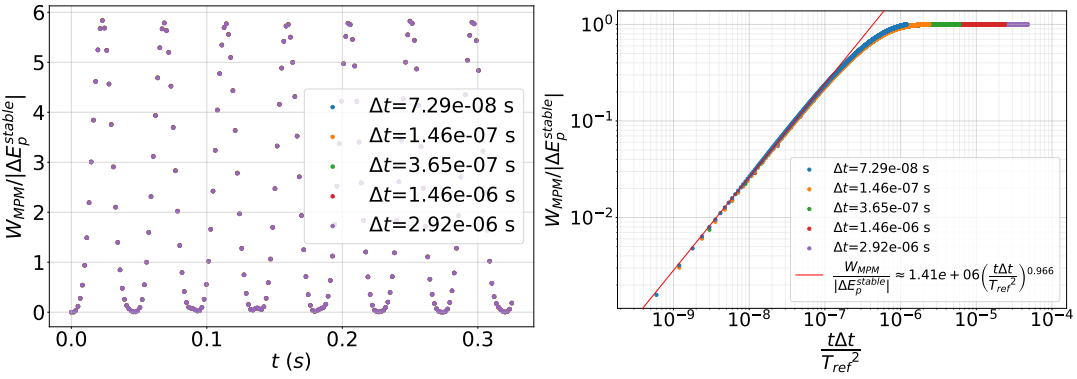
(a) FLIP (b) PIC

(c) AFLIP (d) APIC

(e) TFLIP (f) FLIP0.9

Fig. 11: Kinetic energy during the settling phase for different Δt and motion integration strategy (simulation series S1)

on Δt. As a contrast, the W_{MPM} observed for FLIP on Figure 12 (a) has no noticeable correlation to Δt.



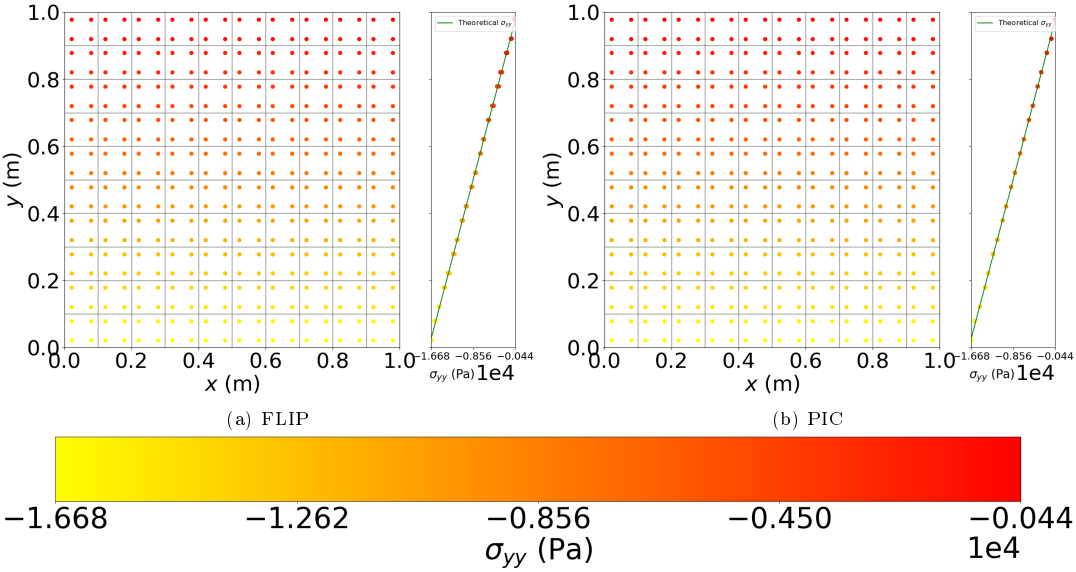
(a) FLIP (b) PIC

Fig. 12: Energy difference caused by the MPM procedure W_{MPM} during the settling (simulation series S1, $|\Delta E_p^{stable}| \approx 2.45\ J/m$)

Because FLIP-based strategies induce permanent oscillations of the column, the settling step was performed again for these simulations using Cundall's damping, see appendix C. Figure 13 (a) and (b) show that the final vertical stress field obtained with both FLIP-based (with $D = 0.1$) and PIC-based (with $D = 0$) strategies is the expected lithostatic one.

These first observations on the settling phase suggest that FLIP-based strategies are to be preferred from both a theoretical and practical point of view. Indeed, since PIC significantly decreases the kinetic energy, the column takes approximately 50 times longer (with the finest time discretization) to settle, requiring much more computational resources.

Looking then at the collapse phase where displacements are much more significant, the time evolution of the column width L_x is plotted on Figure 14 for the same simulation series S1 (Table 3). Here again, one can see that Δt does not have a significant influence on the results with FLIP-based strategies. Indeed, both the collapse dynamics and the final L_x obtained are in these cases virtually identical for different Δt. However, the final column obtained with AFLIP and TFLIP is approximately 9% lower than the one obtained



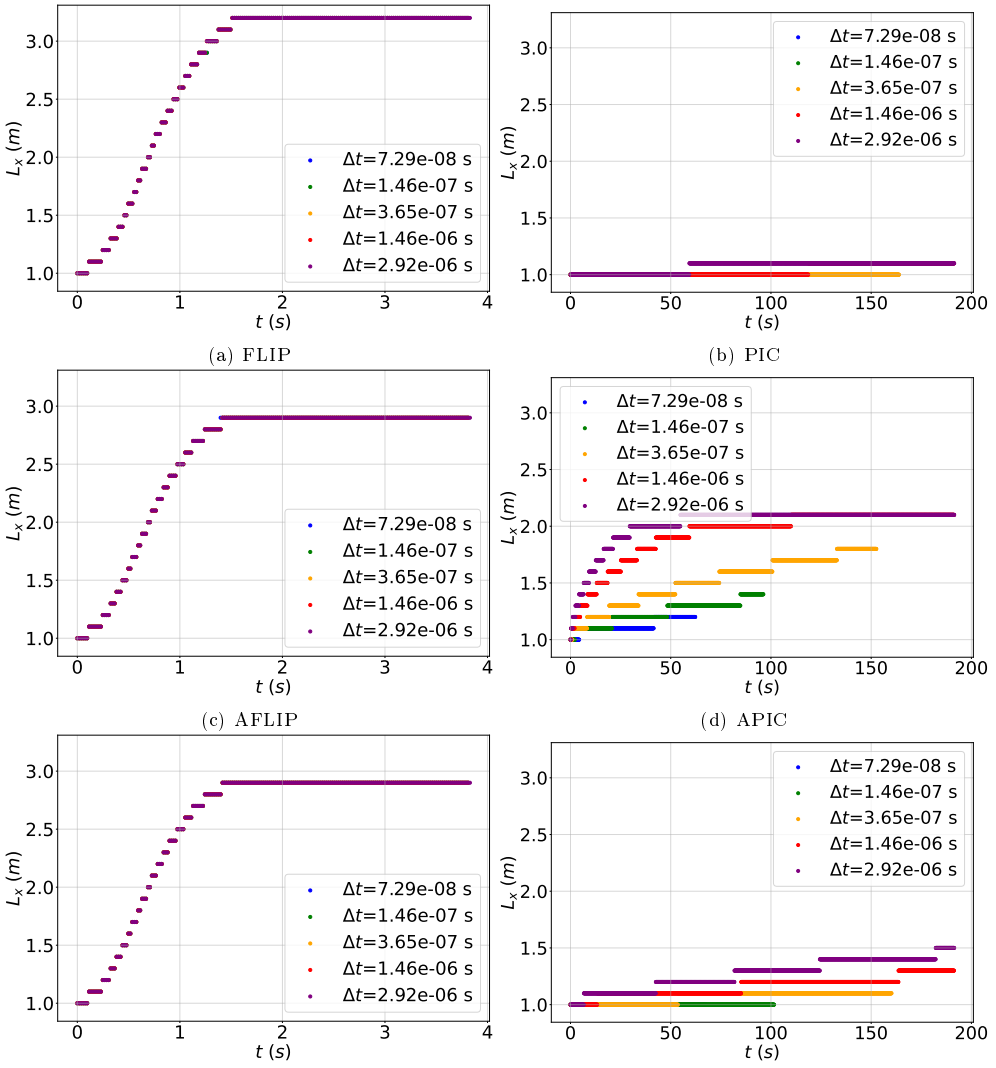
(a) FLIP (b) PIC

Fig. 13: Vertical stress at the end of the settling phase with $\Delta t = 1.46 \times 10^{-6}$ s (simulation series S1)

with FLIP. Regarding the kinetic energy E_k observed on Figure 15 (a), (c), and (e), it is also independent from Δt for all FLIP-based collapses, but one can notice that at the end a slight amount of kinetic energy (less than 0.001% of the initial potential energy) remains, although the material points do not appear to be moving. The use of Cundall's damping could solve this issue, by decreasing globally the energy level.

As for PIC-based strategies, Figures 14 (b), (d) and (f) clearly show that they induce the column to collapse at an unrealistically low rate, as it can also be observed on Figure 16 (d), where the collapse is still in an early stage 3 minutes after releasing the right constraint on the column. As a comparison, it takes approximately a second and a half with FLIP-based strategies for the 1 m-high column to completely collapse, which is much more realistic.

Similarly to the settling phase, the PIC damping observed during the collapse is dependent on the value of Δt used, with higher time steps leading to



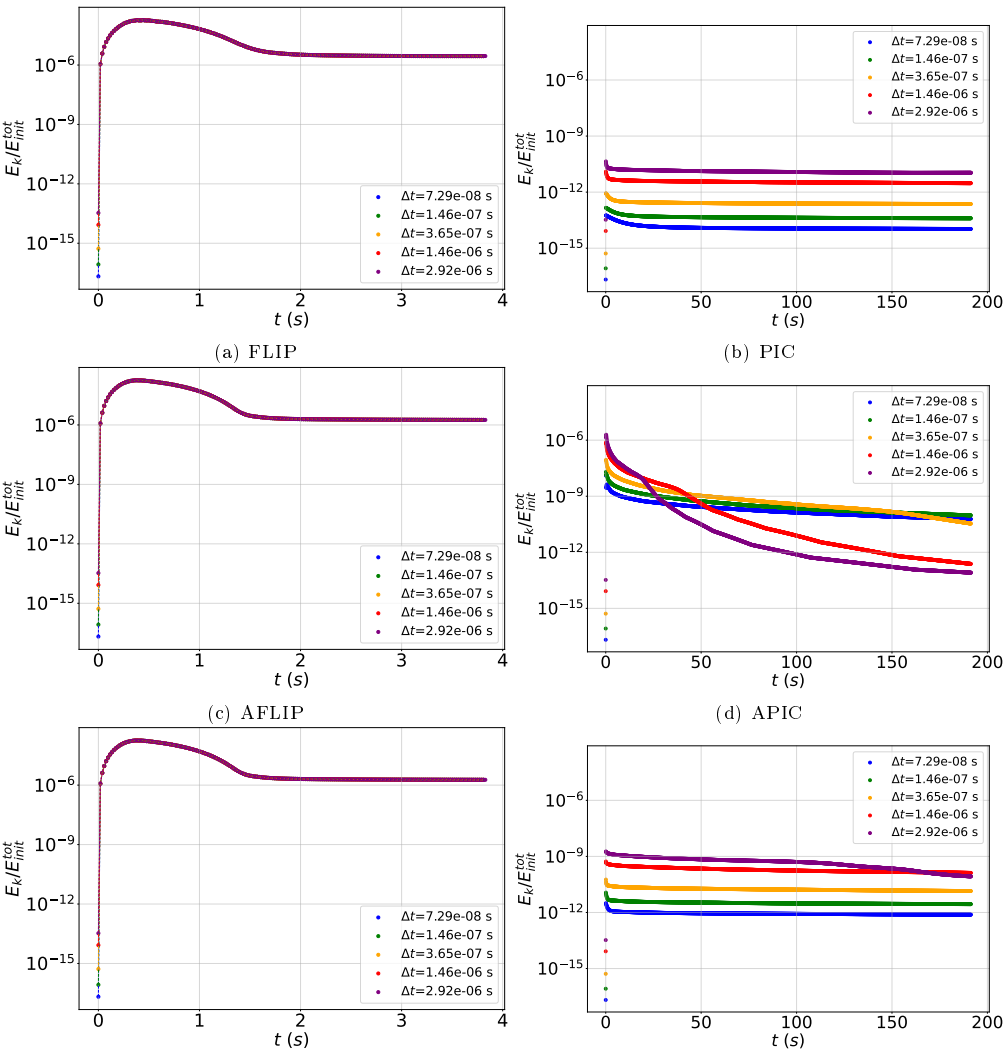
(a) FLIP (b) PIC

(c) AFLIP (d) APIC

(e) TFLIP (f) FLIP0.9

Fig. 14: Width of the column during the collapse for different Δt and motion integration strategies (simulation series S1)

faster collapses. The developed kinetic energy E_k in the PIC simulation is at least 4 million times lower than with FLIP, see Figure 15 (a) and (b), and the gap increases with low values of Δt, which require more MPM iterations to cover the same model time. Yet again, using APIC and the PIC-FLIP blend



(a) FLIP (b) PIC

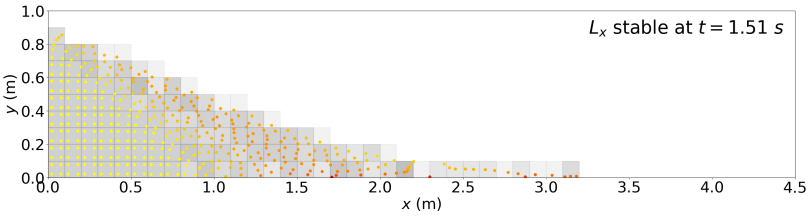
(c) AFLIP (d) APIC

(e) TFLIP (f) FLIP0.9

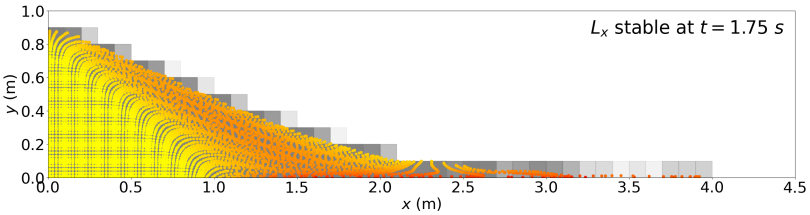
Fig. 15: Kinetic energy during the collapse for different Δt and motion integration strategies (series S1, $E_{init}^{tot} \approx 8.57 \cdot 10^3\ J/m$)

FLIP0.9 (with a non-negligible 0.1 portion of PIC) enables one to mitigate the PIC damping and describe slightly faster column collapses, but it is certainly not enough for the results to be realistic.

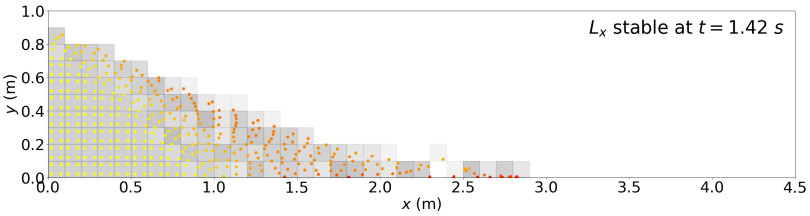
Figure 17 (a) shows that for $\Delta t \leq 1.46 \times 10^{-7}\ s$, FLIP dissipates the same small amount of energy (approximately 0.1% of the expected energy



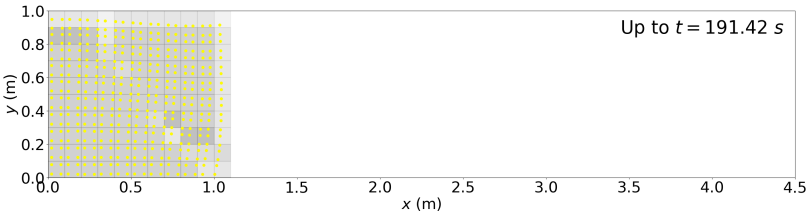
(a) With FLIP, $N_{mppc} = 4$, $l_{cell} = 10\ cm$, and $\Delta t = 7.29 \times 10^{-7}\ s$ (from series S1)



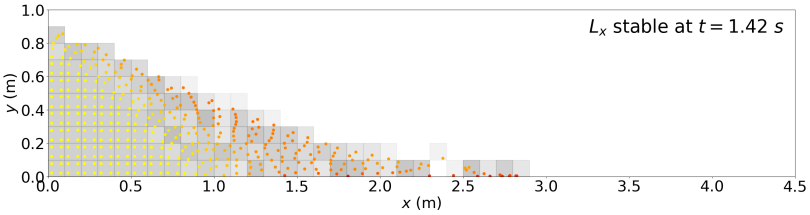
(b) With FLIP, $N_{mppc} = 49$, $l_{cell} = 10\ cm$, and $\Delta t = 7.29 \times 10^{-6}\ s$ (from series S3)



(c) With AFLIP, $N_{mppc} = 4$, $l_{cell} = 10\ cm$, and $\Delta t = 7.29 \times 10^{-7}\ s$ (from series S1)



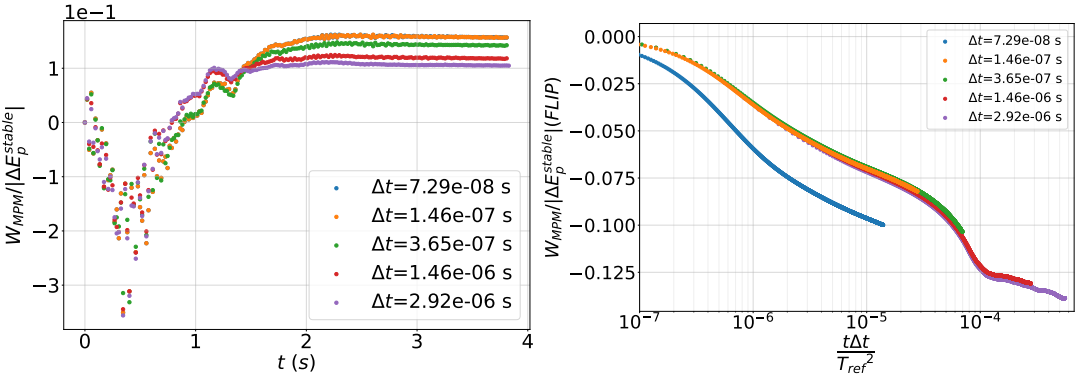
(d) With PIC, $N_{mppc} = 4$, $l_{cell} = 10\ cm$, and $\Delta t = 2.92 \times 10^{-6}\ s$ (from series S1)



(e) With TFLIP, $N_{mppc} = 4$, $l_{cell} = 10\ cm$, and $\Delta t = 1.46 \times 10^{-7}\ s$ (from series S1)

Fig. 16: Positions of material points after the collapse (except for PIC), with deviatoric strains ϵ_D in colorbar (see corresponding video in the supplementary material)

difference) at the end of the collapse. For higher values of Δt, the column gains energy as Δt increases. In Figure 17 (b), an overlap of the data similar to one obtained during the settling is found when plotting the energy lost by MPM as a function of Δt, although the result for $\Delta t = 7.29 \times 10^{-8}$ somewhat deviates from the other results. Note that in this figure, the W_{MPM} obtained with PIC is normalized by the ΔE_p^{stable} obtained with FLIP because PIC columns did not have the time to reach a stable state.



(a) FLIP (b) PIC

Fig. 17: Energy difference caused by the MPM procedure W_{MPM} during the collapse (series S1, $|\Delta E_p^{stable}| \approx 4 \times 10^3\ J/m$)

5.3 Spatial discretization influence

While the previous results were obtained using $N_{mppc} = 4$ material points per mesh element, likewise to [32], the possibility for the MPM results to converge with respect to the spatial discretization is often an open question, be it in terms of N_{mppc} or the size of a mesh element l_{cell}. Considering various simulations and/or various quantities for similar granular column collapse simulations such as shown here, convergence was for instance usually obtained for what regards l_{cell} in [50] but that was less the case in [29, 32]. Here, two

l_{cell}	N_{mppc}	Δt	a_τ	Motion integration strategy	AR
10 cm	4	1.46×10^{-6} s	1.25×10^{-3}	FLIP	1
7.69 cm		1.12×10^{-6} s			
5.88 cm		8.58×10^{-7} s			
5 cm		7.29×10^{-7} s			

Table 4: Parameters used to investigate the influence of the cell size, l_{cell} (series S2, 4 simulations)

l_{cell}	N_{mppc}	Δt	a_τ	Motion integration strategy	AR
10 cm	1	1.46×10^{-6} s	1.25×10^{-3}	FLIP	1
	4				
	9				
	16				
	25				
	36				
	49				

Table 5: Parameters used to investigate the influence of the number of particles per cell, N_{mppc} (series S3, 7 simulations)

other simulation series, S2 and S3, investigate this aspect in terms of both l_{cell} (series S2, Table 4) and N_{mppc} (S3 series, Table 5).

Regarding first the simulation of the settlement process, Figure 18 shows that neither l_{cell} nor N_{mppc} has a noticeable influence for the most part of the stabilization, for $t < 0.3$ s. After that, lower values of l_{cell} lead to lower E_k, but the value of N_{mppc} has still almost no impact on E_k. In fact, a difference can be observed between $N_{mppc} = 1$ and $N_{mppc} \geq 4$. Indeed, when using only 1 material point per mesh element, E_k is higher than for all other values of N_{mppc}, but since its value is already less than 0.1% of the maximum E_k, this variation can be considered negligible. Moreover, Figure 19 (a) confirms that even with $N_{mppc} = 1$, the vertical stress obtained after stabilization is the one expected.

As for the collapse phase, Figure 20 shows L_x and E_k for all tested values of l_{cell} and N_{mppc}, and Figure 16 (b) (or the corresponding video) shows the

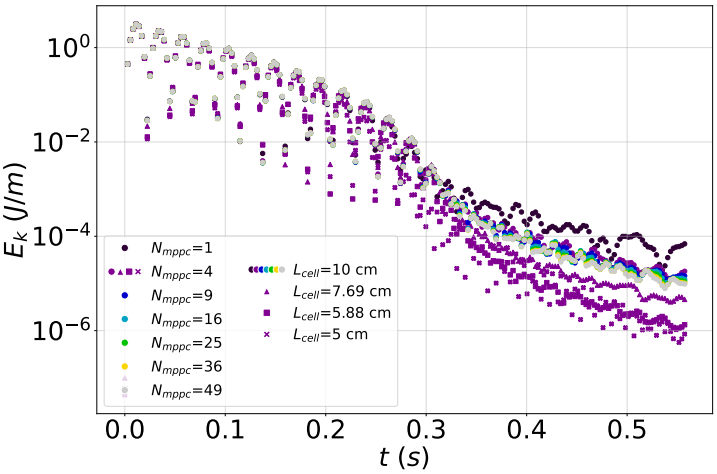
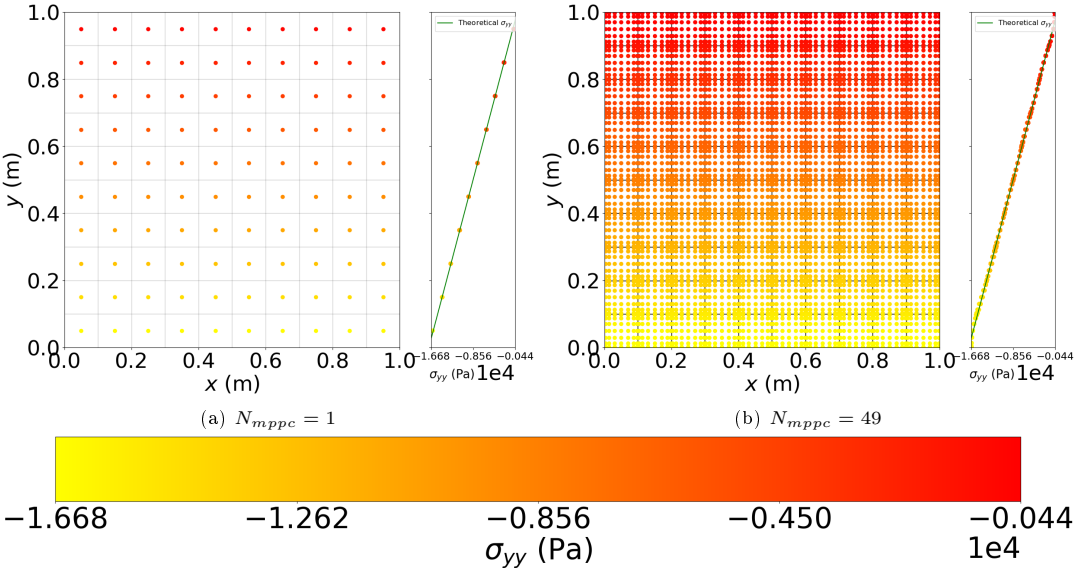


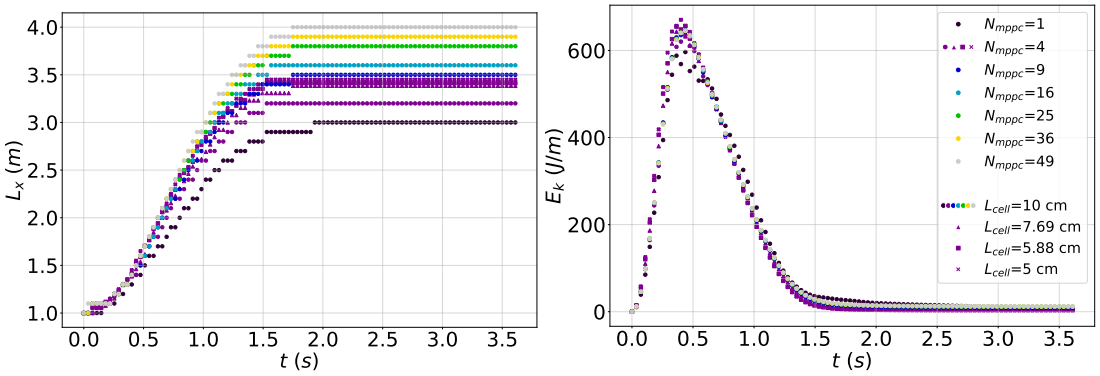
Fig. 18: E_k during the settling phase for different spatial discretization parameters (simulation series S2 and S3 with FLIP)



(a) $N_{mppc} = 1$ (b) $N_{mppc} = 49$

Fig. 19: Vertical stress at the end of the settling phase in various configurations and with $\Delta t = 1.46 \times 10^{-6}$ s (simulation series S3)

positions of all material points throughout the collapse for $N_{mppc} = 49$. Lower values of l_{cell} increase L_x by approximately 7% (see Figure 20 (a)), with only



(a) Width of the column L_x (b) Kinetic energy E_k

Fig. 20: Influence of the spatial discretization parameters during the collapse phase (series S2 and S3)

a slight impact for $l_{cell} < 10\ cm$, the mesh element size has thus no significant impact on the collapsing column. A stronger influence is here observed for what concerns N_{mppc} since the final L_x is 25% higher for $N_{mppc} = 49$ than for $N_{mppc} = 1$. Although the influence of N_{mppc} is less important for $25 \leq N_{mppc} \leq 49$, no clear convergence is observed for L_x.

This dependency to N_{mppc} is probably caused by the "voxelized" point of view inherent to the MPM, along with the extensive shear experienced by the material points on the right side of the column (see Figure 16 (a) and (b)). Indeed, if one was to attribute an initial domain to each material point, likewise to the use of GIMP in [50], at some point the ones on the right side of the column should span over several mesh elements, but this can not be taken into account in the present MPM formulation. Increasing N_{mppc} decreases the size of these domains, improving their chances to restrict in only 1 mesh element. This observation suggests that a splitting procedure similar to the one used in FEMLIP [31] could reduce the dependency on N_{mppc}. Basically, the size of the domains previously mentioned would be tracked, and a material point be split into two if its domain became too large. Nevertheless, this procedure is left

l_{cell}	N_{mppc}	Δt	a_τ	Motion integration strategy	AR
10 cm	4	1.46×10^{-6} s	1.25×10^{-3}	FLIP and AFLIP	0.6
				FLIP	1
					2.4
				FLIP and AFLIP	3

Table 6: Parameters used for comparison with results from the literature (series S4, 4 simulations)

for future work as it would require particular attention on the velocities and state parameters to attribute to each new material point in order to conserve important quantities (e.g. the momentum) in the process.

5.4 (Non-)necessity of an affine-augmented motion integration strategy for different aspect ratio and comparison with the literature

A last simulation series (S4, see Table 6) combines a variable aspect ratio AR and two different motion integration strategies among the most conservative choices FLIP and AFLIP.

Looking at the collapse dynamics in terms of spreading length, i.e. $L_x(t)$ (Figure 21), one can see that both FLIP and AFLIP provide similar results for $AR = 0.6$ and $AR = 3$. More precisely, FLIP gives a wider collapsed column by approximately 9% for $AR = 0.6$, and 5% for $AR = 3$. Considering the conclusions from Section 4, these results suggest that the collapse of a granular column up to $AR = 3$ doesn't involve much rotational motion and can thus be modelled using the simple FLIP strategy.

For further validation of the results, the spreading lengths obtained with the FLIP scheme in this same S4 series are finally compared to results from the literature in Figure 22. The latters include:

- experimental results conducted on glass beads in [61];

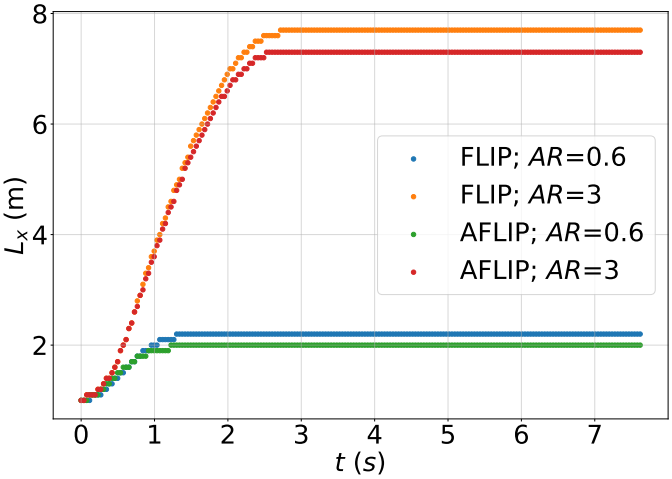


Fig. 21: Evolving width of the column during the collapse for different aspect ratios and motion integration strategies (simulation series S4)

- numerical results performed in [51] with a FEM-based hybrid Eulerian-Lagrangian method, in conjunction with the Mohr-Coulomb model with $\phi \in [25°;\ 40°]$.
- numerical results obtained in [52] using a 3D DEM model, with a viscous elasto-plastic contact law;
- numerical results from [53], obtained with the SPH method and the micromechanical 3D-H model (denoted in this paper SPHx3D-H).

While no attempt was made in the present study to define material parameters similar to those of these literature results, one should note that the latter form a consistent data set which seems to be independent of material properties and should therefore be appropriate to serve as a comparison basis for our results.

A first observation is that our MPM columns take slightly longer to reach their final length, for all values of AR. Considering that PIC-based motion integration strategies lead to even longer spreading times, as previously shown

in Figure 14, the energy dissipation observed with FLIP in Section 4.2 might be to blame.

Figure 22 also shows that the final spreading length is higher in MPM, with respect to collapses performed with other numerical models for the same AR. For instance, with $AR = 1$, the MPM simulation gives a final length approximately 38% higher than the results presented in [51], while the results from [53] are only approximately 19% lower. A similar gap is observed between our results and the DEM ones from [52]. Because both SPHx3D-H and DEM are supposed to be more accurate for granular materials than the Mohr-Coulomb model, the proximity of our results with the SPHx3D-H and DEM results is quite comforting.

According to the experimental results from [61], the real collapsing columns are clearly less wide than the MPM ones, more precisely by 34% for $AR = 3$. However, for $AR = 0.6$, this difference is restricted to approximately 5%, and the experimental $\widetilde{L}$ is higher than the MPM one for the most part of the collapse. This difference could come from the way the measurement of $\widetilde{L}$ is performed: experimentally, some grains that are isolated from the rest of the column might be excluded from the measurement, but with numerical results, all of the material present initially in the column is included in the measurement.

6 Conclusion

This paper reviewed the implications of using different motion integration strategies in MPM, whose possibility stems from the necessity to express and integrate in time kinematic fields on a double layer of spatial discretization with uncoupled grid nodes and material points. Numerical investigations, in a USF scheme by default, focused on the impact of these motion integration

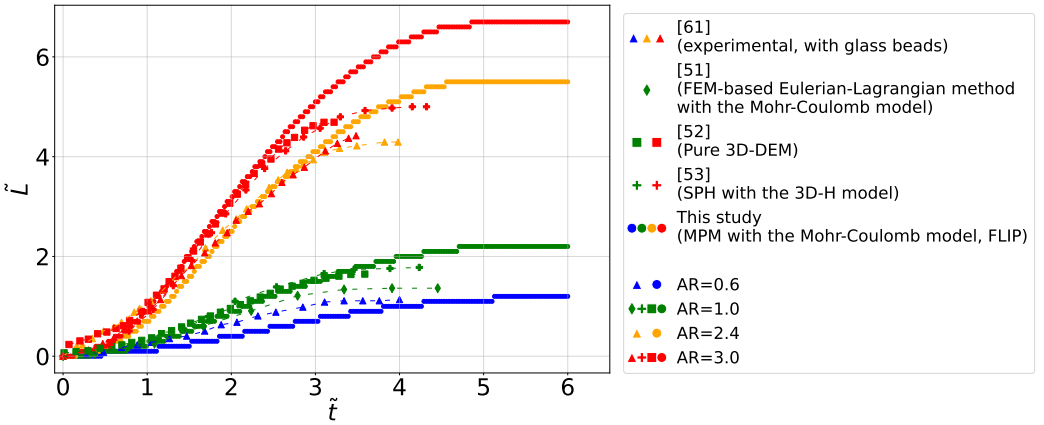


Fig. 22: Obtained collapse dynamics in terms of normalized spreading length $\widetilde{L}$ (series S4), in comparison with results from the literature

strategies for conserving energy in a number of cases of increasing complexity, up to the simulation of a granular column collapse.

Even though theoretically correct for describing the simplest rigid linear motions, PIC was recalled to dissipate the energy during an elastic deformation event in a MPM simulation and considerably dampen, i.e., slow down, the description of a granular column collapse. Moreover, this unnatural damping was shown herein to increase with lower time steps, unveiling an impossible convergence of the PIC strategy with respect to the temporal discretization.

On the other hand, it was shown that if only a translational motion and elastic deformations are involved in the simulation, the FLIP strategy was the most performant at preserving the total energy in the system and unaffected by the chosen time step, when below the divergence Courant-Friedrichs-Lewy limit.

In cases involving rotational motions, it has been recalled how both PIC and FLIP are unable to describe rigid body rotations, even though spatial discretization affects FLIP performances in this aspect, and that an affine augmentation procedure such as the APIC strategy is necessary to conserve the

angular momentum. As a matter of fact, APIC was found to conserve accurately the angular momentum in the simple case of a rigid rotating object, with the condition that the time step is low enough compared to the object's angular velocity. In that very simple case, APIC is not only more predictable than AFLIP, TPIC and TFLIP, but also more performant. However, in the general case involving deformations, APIC suffers from the unnatural damping inherited from PIC, along with its impossibility to converge with respect to the temporal discretization. The AFLIP and TFLIP strategies are thus better than APIC when rotational motions and large deformations are involved, both giving almost identical results. Ultimately, because the implementation of TFLIP is more straightforward than AFLIP, TFLIP is certainly the most suitable motion integration strategy to recommend.

A Bouncing cube with the USL scheme

This appendix gives insights on the effect of the stress update scheme on the bouncing cube simulation presented in Section 4.1. The evolution of total energy E_{tot} during the simulation is plotted for the USL scheme in Figure 23, similarly to previous Figure 3 (a) for the USF case. One can see that all motion integration strategies are strictly dissipative when using USL, even FLIP-based ones, unlike the previous USF case. This observation is consistent with the conclusions of [38], where the USL scheme was demonstrated to be strictly dissipative by formulation. One can also note that with the present USL choice, a given motion integration strategy can not be distinguished from its affine-augmented version (e.g., PIC vs APIC or FLIP vs AFLIP) in this translation regime, in some contrast with the USF case.

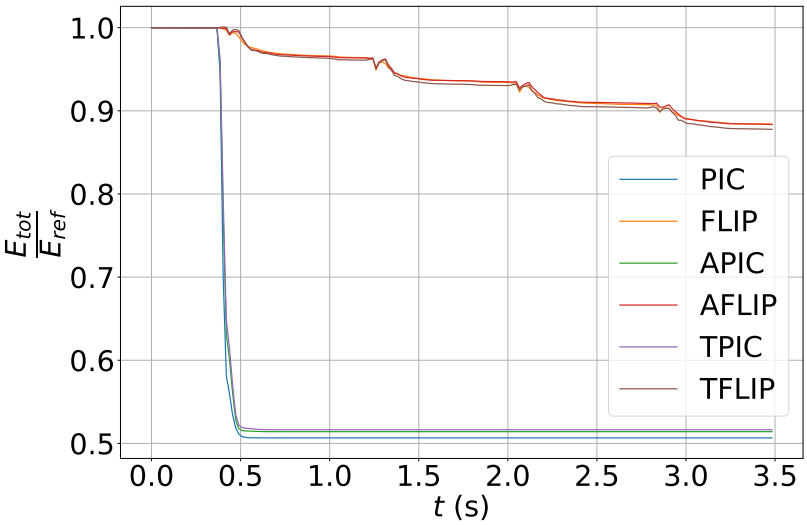


Fig. 23: Total energy for the bouncing cube example simulated with different motion integration strategies and the USL stress update scheme ($E_{ref} \approx 25.7$ kJ, $a_\tau = 5 \times 10^{-2}$)

B Reproduction of results from the literature

For validation purposes of the used MPM implementation, this section aims to reproduce two different simulations taken from the literature: a bouncing disk case conducted in [16], Section 4.1 therein, and a rotating disk simulation of [18], Section 6.1 therein.

Although a Neo-Hookean material model was used in [16] and [18], the results in this Appendix are obtained using the same hyperelastic Hooke's law as used in previous Section 4.1, assigning the Neo-Hookean linearized elastic parameters of [16] and [18] to their constant Hooke's counterparts. Also, the simulations are performed using the USL scheme as well as boundary modified cubic B-spline shape functions.

B.1 Bouncing elastic disk in comparison with [16]

The bouncing disk simulation from [16], §4.1 therein, is very similar to the one presented in previous Section 4.1, the main differences being the number of dimensions (2D in this Appendix, 3D in Section 4.1), the shape of the bouncing object (a disk in this section, a cube in Section 4.1), and the absence of gravity in the present Appendix (the movement is caused by an initial velocity instead).

The simulation is performed with the same spatial and time discretizations as [16] and three different motion integration strategies (PIC, FLIP, and APIC), although [16] only used FLIP.

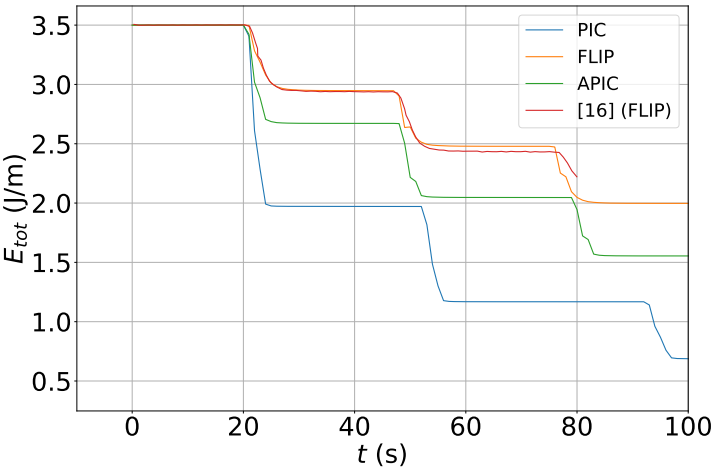


Fig. 24: Bouncing elastic disk simulation using various motion integration strategies

These results show that our MPM implementation is in accordance with the one from [16] in terms of energy conservation. Indeed, it is herein obtained, with the FLIP strategy, virtually the same E_{tot} than [16]. The minor differences observed starting from $t \approx 50\ s$ can be attributed to the more complex constitutive model used in [16]. The results obtained for PIC and APIC are

noticeably different one from another for this 2D gravitation-less simulation, while they were almost identical for the 3D cube bouncing under gravity from Section 4.1.

B.2 Rotating disk in comparison with [18]

The rotating disk simulation from [18], §6.1 therein, is similar to the one in Section 4.2 of the present study, the main differences being the number of dimensions (2D in this Appendix, 3D in Section 4.2) and the shape of the rotating object (a disk in this Appendix, a cube in Section 4.2).

For reproducing that case of a rotating disk, several values of the time step are herein tested, with a_τ ranging from 10^{-2} to 0.2 ($1.4 \times 10^{-5}\ s \leq \Delta t \leq 2.8 \times 10^{-4}\ s$), as well as three different motion integration strategies (PIC, FLIP, and APIC) and two different mesh cell sizes, between 0.03125 m and 0.0625 m for a 0.6 m disk diameter.

Figure 25 shows that both PIC and FLIP results are in agreement with the results from [18], although we obtain a slightly lower angular momentum. However, our implementation used in conjunction with PIC dissipated almost all the angular momentum as soon as the first iteration, while the PIC simulation from [18] loose its angular momentum slowly over approximately 4 s. This might be due to the use of a more advanced time integration scheme in [18], which can implicit depending on the value of a parameter λ.

Figure 26 shows that, even though the variations are small, higher time steps actually promote the conservation of angular momentum, thanks to a reduced number of MPM iterations that accumulate errors. More importantly, the size of the mesh cells is shown to have an significant impact on the conservation of the angular momentum in the case of using FLIP.

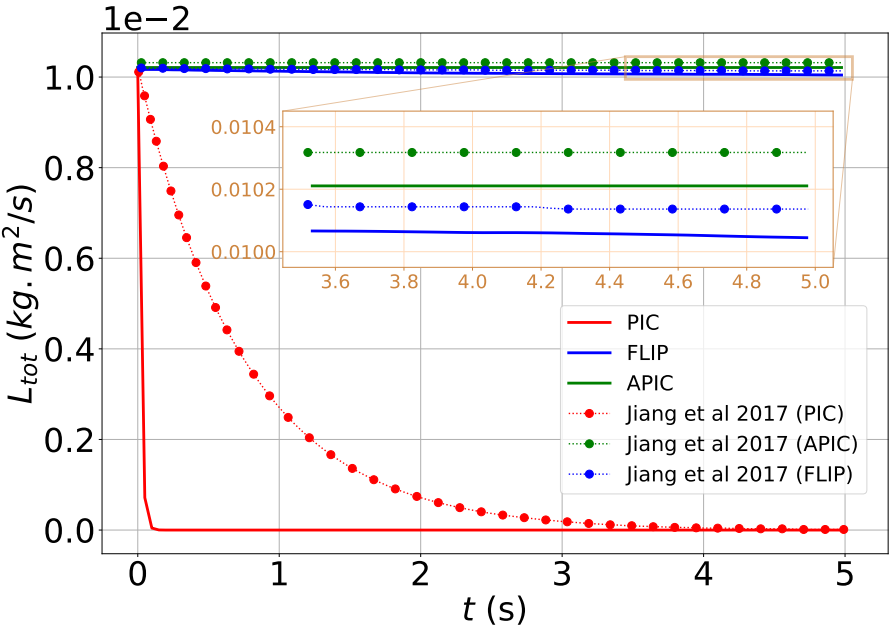
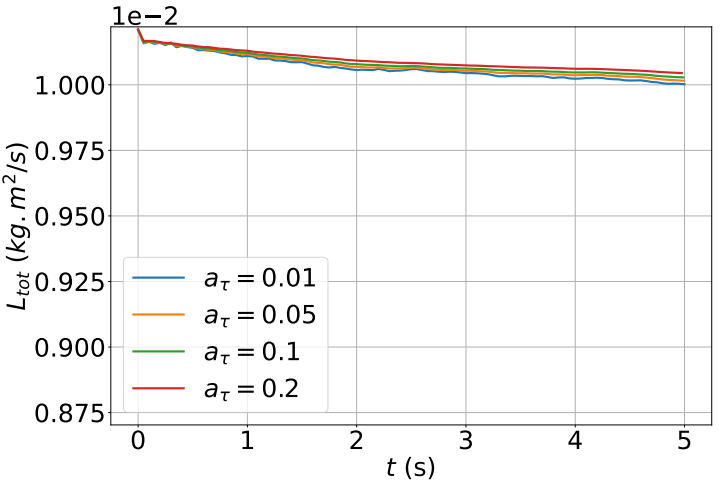


Fig. 25: Rotating disk simulation of [18] using various motion integration strategies (for $a_\tau = 0.2$)

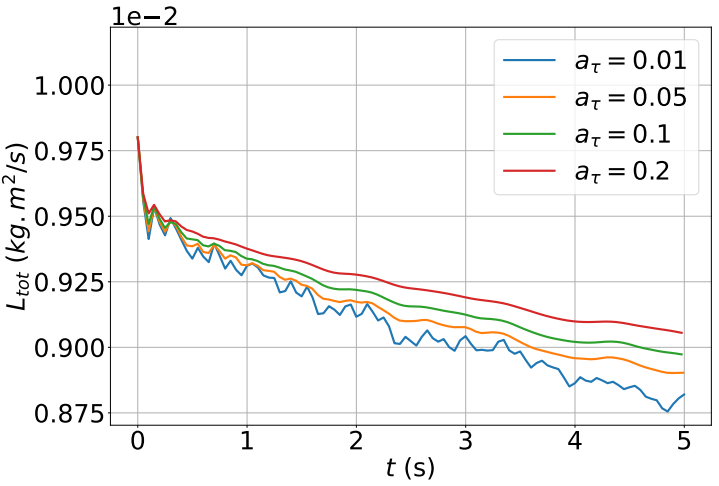
C Granular column settling with Cundall's damping for FLIP-based strategies

Although not using Cundall's damping is necessary to investigate the energy dissipation coming from the MPM procedure, during the settling phase this consideration lead the column modelled using FLIP-based strategies to perpetually oscillate. As a consequence, the settled states used to initialize the material points positions and stresses before the collapse presented in Figure 14 were obtained after using $D = 0.1$ for FLIP-based settling simulations. This appendix shows the evolution of E_k during this settling phase in Figures 27, corresponding to Figures 11.

One can notice that Cundall's damping has no influence on the non-dependance of the results on the motion integration strategy and Δt during



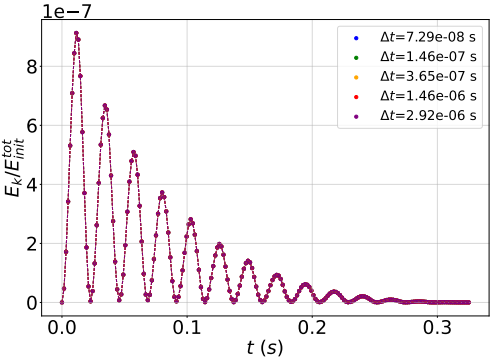
(a) $l_{cell} = 3.125 \times 10^{-2}\ m$



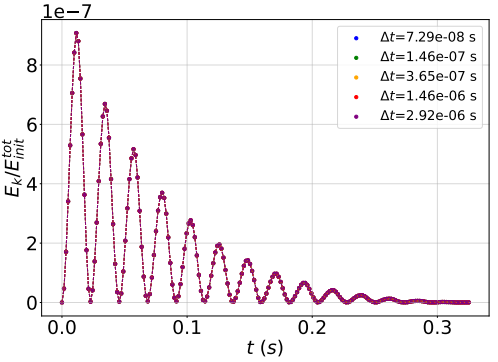
(b) $l_{cell} = 6.25 \times 10^{-2}\ m$

Fig. 26: Effect of Δt and l_{cell} on the angular momentum during the rotating disk (of 0.6 m diameter) simulation of [18] using FLIP

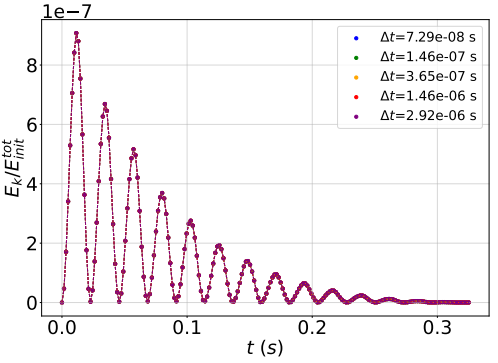
the settling phase. Indeed, FLIP, AFLIP and NFLIP all lead the column to stabilize in as much time with a same period for the pseudo-oscillations, no matter Δt.



(a) FLIP



(b) AFLIP



(c) TFLIP

Fig. 27: Kinetic energy during the settling phase for different Δt and motion integration strategy (series S1 with Cundall's damping, FLIP-based)

Statements and Declarations

Competing interests

The authors have no competing interests to declare that are relevant to the content of this article.

Data deposition

The source code of the used MPM code will be made available at https://forgemia.inra.fr/mpm-at-recover/cbgeo (currently under private access) in case of publication.

Acknowledgements

Inspiring discussions with Frédéric Dufour (Univ. Grenoble-Alpes, Grenoble INP, CNRS, 3SR) are gratefully acknowledged. Constructive and thorough reviews by John Nairn (Oregon State University) and another anonymous reviewer greatly contributed to improve the manuscript.

References

[1] A Hrennikoff. Solution of problems of elasticity by the framework method. *Journal of Applied Mechanics*, 8(4):A169–A175, 1941.

[2] R Courant. Variational methods for the solution of problems of equilibrium and vibrations. *Bulletin of the American Mathematical Society*, 49(1):1–23, 1943.

[3] Jung-Ho Cheng. Automatic adaptive remeshing for finite element simulation of forming processes. *International Journal for Numerical Methods in Engineering*, 26(1):1–18, 1988.

[4] SB Petersen and PAF Martins. Finite element remeshing: a metal forming approach for quadrilateral mesh generation and refinement. *International Journal for Numerical Methods in Engineering*, 40(8):1449–1464, 1997.

[5] Annika Trädegård, Fred Nilsson, and Sören Östlund. Fem-remeshing technique applied to crack growth problems. *Computer Methods in Applied Mechanics and Engineering*, 160(1-2):115–131, 1998.

[6] Massimiliano Cremonesi, Alessandro Franci, Sergio Idelsohn, and Eugenio Oñate. A state of the art review of the particle finite element method (PFEM). *Archives of Computational Methods in Engineering*, 27(5):1709–1735, 2020.

[7] S.R. Idelsohn, E. Oñate, and F. Del Pin. The particle finite element method: a powerful tool to solve incompressible flows with free-surfaces and breaking waves. *International Journal for Numerical Methods in Engineering*, 61(7):964–989, 2004.

[8] Francis H Harlow. The particle-in-cell method for numerical solution of problems in fluid dynamics. Technical report, Los Alamos National Lab.(LANL), Los Alamos, NM (United States), 1962.

[9] Jeremiah U Brackbill and Hans M Ruppel. FLIP: A method for adaptively zoned, particle-in-cell calculations of fluid flows in two dimensions. *Journal of Computational physics*, 65(2):314–343, 1986.

[10] Deborah Sulsky, Zhen Chen, and Howard L Schreyer. A particle method for history-dependent materials. *Computer Methods in Applied Mechanics and Engineering*, 118(1-2):179–196, 1994.

[11] Scott G Bardenhagen and Edward M Kober. The generalized interpolation material point method. *Computer Modeling in Engineering and Sciences*, 5(6):477–496, 2004.

[12] A. Sadeghirad, R. M. Brannon, and J. Burghardt. A convected particle

domain interpolation technique to extend applicability of the material point method for problems involving massive deformations. *International Journal for Numerical Methods in Engineering*, 86(12):1435–1456, 2011.

[13] Alban de Vaucorbeil, Vinh Phu Nguyen, Sina Sinaie, and Jian Ying Wu. Material point method after 25 years: Theory, implementation, and applications. *Advances in applied mechanics*, 53:185–398, 2020.

[14] Wojciech T. Sołowski, Martin Berzins, William M. Coombs, James E. Guilkey, Matthias Möller, Quoc Anh Tran, Tito Adibaskoro, Seyedmohammadjavad Seyedan, Roel Tielen, and Kenichi Soga. Chapter Two - Material point method: Overview and challenges ahead. *Advances in Applied Mechanics*, 54:113–204, 2021.

[15] Vinh Phu Nguyen, Alban de Vaucorbeil, and Stephane Bordas. *The Material Point Method Theory, Implementations and Applications.* Springer, 2023.

[16] E Love and Deborah L Sulsky. An unconditionally stable, energy–momentum consistent implementation of the material-point method. *Computer Methods in Applied Mechanics and Engineering*, 195(33-36):3903–3925, 2006.

[17] Chenfanfu Jiang, Craig Schroeder, Andrew Selle, Joseph Teran, and Alexey Stomakhin. The affine particle-in-cell method. *ACM Transactions on Graphics (TOG)*, 34(4):1–10, 2015.

[18] Chenfanfu Jiang, Craig Schroeder, and Joseph Teran. An angular momentum conserving affine-particle-in-cell method. *Journal of Computational Physics*, 338:137–164, 2017.

[19] Yun Fei, Qi Guo, Rundong Wu, Li Huang, and Ming Gao. Revisiting integration in the material point method: a scheme for easier separation and less dissipation. *ACM Transactions on Graphics (TOG)*, 40(4):1–16, 2021.

[20] PC Wallstedt and JE Guilkey. Improved velocity projection for the material point method. *Computer Modeling in Engineering and Sciences*, 19(3):223, 2007.

[21] Keita Nakamura, Satoshi Matsumura, and Takaaki Mizutani. Taylor particle-in-cell transfer and kernel correction for material point method. *Computer Methods in Applied Mechanics and Engineering*, 403:115720, 2023.

[22] Chad C. Hammerquist and John A. Nairn. A new method for material point method particle updates that reduces noise and enhances stability. *Computer Methods in Applied Mechanics and Engineering*, 318:724–738, 2017.

[23] John A Nairn and Chad C Hammerquist. Material point method simulations using an approximate full mass matrix inverse. *Computer Methods in Applied Mechanics and Engineering*, 377:113667, 2021.

[24] Alban de Vaucorbeil, Vinh Phu Nguyen, and Christopher R. Hutchinson. A total-lagrangian material point method for solid mechanics problems involving large deformations. *Computer Methods in Applied Mechanics and Engineering*, 360:112783, 2020.

[25] Marc Kohler, Andreas Stoecklin, and Alexander M. Puzrin. A MPM framework for large-deformation seismic response analysis. *Canadian Geotechnical Journal*, 59(6):1046–1060, 2022.

[26] Krishna Kumar, Jeffrey Salmond, Shyamini Kularathna, Christopher Wilkes, Ezra Tjung, Giovanna Biscontin, and Kenichi Soga. Scalable and modular material point method for large-scale simulations. *arXiv preprint arXiv:1909.13380*, 2019.

[27] Sacha Duverger and Jérôme Duriez. PyCBG, a python module for generating CB-Geo MPM input files (1.1.4), 2021.

[28] D. Burgess, D. Sulsky, and J.U. Brackbill. Mass matrix formulation of the flip particle-in-cell method. *Journal of Computational Physics*, 103(1):1–15, 1992.

[29] P.C. Wallstedt and J.E. Guilkey. An evaluation of explicit time integration schemes for use with the generalized interpolation material point method. *Journal of Computational Physics*, 227(22):9628–9642, 2008.

[30] Michael Steffen, Robert M Kirby, and Martin Berzins. Analysis and reduction of quadrature errors in the material point method (MPM). *International journal for numerical methods in engineering*, 76(6):922–948, 2008.

[31] Louis Moresi, Frédéric Dufour, and H-B Mühlhaus. A lagrangian integration point finite element method for large deformation modeling of viscoelastic geomaterials. *Journal of computational physics*, 184(2):476–497, 2003.

[32] Elliot James Fern and Kenichi Soga. The role of constitutive models in mpm simulations of granular column collapses. *Acta Geotechnica*, 11(3):659–678, 2016.

[33] Gregory Abram, Andrew Solis, Yong Liang, and Krishna Kumar. In situ visualization of regional-scale natural hazards with galaxy and material point method. *Computing in Science & Engineering*, 24(2):31–39, 2022.

[34] Weijian Liang, Shiwei Zhao, Huanran Wu, and Jidong Zhao. Bearing capacity and failure of footing on anisotropic soil: A multiscale perspective. *Computers and Geotechnics*, 137:104279, 2021.

[35] JU Brackbill. The ringing instability in particle-in-cell calculations of low-speed flow. *Journal of Computational Physics*, 75(2):469–492, 1988.

[36] M Steffen, PC Wallstedt, JE Guilkey, RM Kirby, and M Berzins. Examination and analysis of implementation choices within the material

point method (mpm). *Computer Modeling in Engineering and Sciences*, 31(2):107–127, 2008.

[37] Quoc-Anh Tran and Wojciech Sołowski. Temporal and null-space filter for the material point method. *International Journal for Numerical Methods in Engineering*, 120(3):328–360, 2019.

[38] SG Bardenhagen. Energy conservation error in the material point method for solid mechanics. *Journal of Computational Physics*, 180(1):383–403, 2002.

[39] Martin Berzins. Energy conservation and accuracy of some MPM formulations. *Computational Particle Mechanics*, pages 1–13, 2022.

[40] Chuyuan Fu, Qi Guo, Theodore Gast, Chenfanfu Jiang, and Joseph Teran. A polynomial particle-in-cell method. *ACM Transactions on Graphics (TOG)*, 36(6):1–12, 2017.

[41] Alexey Stomakhin, Craig Schroeder, Lawrence Chai, Joseph Teran, and Andrew Selle. A material point method for snow simulation. *ACM Transactions on Graphics (TOG)*, 32(4):1–10, 2013.

[42] Kun Zhang, Shui-Long Shen, Annan Zhou, and Daniel Balzani. Truncated hierarchical b-spline material point method for large deformation geotechnical problems. *Computers and Geotechnics*, 134:104097, 2021.

[43] Joel Wretborn, Rickard Armiento, and Ken Museth. Animation of crack propagation by means of an extended multi-body solver for the material point method. *Computers & Graphics*, 69:131–139, 2017.

[44] Weijian Liang and Jidong Zhao. Multiscale modeling of large deformation in geomechanics. *International Journal for Numerical and Analytical Methods in Geomechanics*, 43(5):1080–1114, 2019.

[45] John A. Nairn. Numerical simulation of orthogonal cutting using the material point method. *Engineering Fracture Mechanics*, 149:262–275, 2015.

[46] Anura3D MPM Research Community. *Anura3D Scientific Manual version 2022*, 2022.

[47] John A Nairn. Material point method calculations with explicit cracks. *Computer Modeling in Engineering and Sciences*, 4(6):649–664, 2003.

[48] Deborah Sulsky, Shi-Jian Zhou, and Howard L. Schreyer. Application of a particle-in-cell method to solid mechanics. *Computer Physics Communications*, 87(1):236–252, 1995.

[49] Stephen P. Timoshenko and J. N. Goodier. Theory of elasticity (2nd ed.). *McGraw-Hill*, 1951.

[50] W. T. Sołowski and S. W. Sloan. Evaluation of material point method for use in geotechnics. *International Journal for Numerical and Analytical Methods in Geomechanics*, 39(7):685–701, 2015.

[51] GB Crosta, S Imposimato, and D Roddeman. Numerical modeling of 2-d granular step collapse on erodible and nonerodible surface. *Journal of Geophysical Research: Earth Surface*, 114(F3), 2009.

[52] L Girolami, V Hergault, G Vinay, and A Wachs. A three-dimensional discrete-grain model for the simulation of dam-break rectangular collapses: comparison between numerical results and experiments. *Granular Matter*, 14(3):381–392, 2012.

[53] Hao Xiong, Zhen-Yu Yin, François Nicot, Antoine Wautier, Marie Miot, Félix Darve, Guillaume Veylon, and Pierre Philippe. A novel multiscale large deformation approach for modelling of granular collapse. *Acta Geotechnica*, 16(8):2371–2388, 2021.

[54] Nhu H. T. Nguyen, Ha H. Bui, and Giang D. Nguyen. Effects of material properties on the mobility of granular flow. *Granular Matter*, 22(3), 2020.

[55] Lydie Staron and EJ Hinch. Study of the collapse of granular columns using two-dimensional discrete-grain simulation. *Journal of Fluid*

Mechanics, 545:1–27, 2005.

[56] Gert Lube, Herbert E Huppert, R Stephen J Sparks, and Armin Freundt. Collapses of two-dimensional granular columns. *Physical Review E*, 72(4):041301, 2005.

[57] Arnold N Lowan, Norman Davids, and Arthur Levenson. Table of the zeros of the Legendre polynomials of order 1-16 and the weight coefficients for Gauss' mechanical quadrature formula. *Bull. Amer. Math. Soc.*, 48(12):739–743, 1942.

[58] Rodaina Aboul Hosn, Luc Sibille, Nadia Benahmed, and Bruno Chareyre. Discrete numerical modeling of loose soil with spherical particles and interparticle rolling friction. *Granular matter*, 19(1):1–12, 2017.

[59] P. A. Cundall. Distinct element models of rock and soil structure. In E. T. Brown, editor, *Analytical and Computational Methods in Engineering Rock Mechanics*, pages 129–163. George Allen and Unwin, 1987.

[60] Stefano Utili, Tao Zhao, and GT Houlsby. 3d dem investigation of granular column collapse: evaluation of debris motion and its destructive power. *Engineering geology*, 186:3–16, 2015.

[61] E Lajeunesse, JB Monnier, and GM Homsy. Granular slumping on a horizontal surface. *Physics of fluids*, 17(10):103302, 2005.