An Explicit Asynchronous Contact Algorithm for Elastic Body-Rigid Wall Interaction
— Article in Press (10.1002/nme.3266) —

Raymond A. Ryckman and Adrian J. Lew*

Department of Mechanical Engineering, Stanford University, 496 Lomita Mall, Stanford CA 94305, USA
{rryckman,lewa}@stanford.edu

SUMMARY

The use of multiple-time-step integrators can provide substantial computational savings over traditional one-time-step methods for the simulation of solid dynamics, while maintaining desirable properties, such as energy conservation. Contact phenomena generally require the adoption of either an implicit algorithm or the use of unacceptably small time steps to prevent large amount of numerical dissipation from being introduced. This paper introduces a new explicit dynamic contact algorithm that, by taking advantage of asynchronous time stepping (AVI), delivers an outstanding energy performance at a much more acceptable computational cost. We demonstrate the performance of the numerical method with several three-dimensional examples. Copyright © 2000 John Wiley & Sons, Ltd.

KEY WORDS: Contact, explicit time integration, variational methods, nonlinear dynamics

1. Introduction

Numerical methods for impact and contact between deformable and rigid bodies during transient dynamics is an area of computational mechanics that has not yet reached the maturity and robustness of others. While very complex simulations are nowadays performed, issues of robustness, accuracy, parallel scalability and computational efficiency still plague problems in which contact plays an important role. In this paper we address one particular issue, concerned with the construction of explicit contact algorithms that showcase good energy conservation properties through contact events at moderately large time steps.

Explicit in time approaches to contact between deformable bodies has been dominated by penalty methods. As described in many standard textbooks [1, 2], these methods consist of replacing the walls of the bodies by stiff potentials (often quadratic) that allow for some interpenetration between bodies. By making the potentials stiffer, the representation of the
walls is improved and the contact conditions are more accurately imposed. The main drawback of such approach is that in increasing the stiffness of the penalty potentials to improve accuracy, the time step of the simulation needs to be severely reduced for stability reasons. As such it is often challenging to find an appropriate value of the penalty parameter that strikes the necessary balance between accuracy and efficiency, and that is useful throughout an entire simulation involving material nonlinearities.

Implicit approaches to dynamic problems with contact often involve a more accurate imposition of the contact conditions, through for example Lagrange multipliers or augmented Lagrangian methods, although penalty approaches have also been proposed [3, 4]. As always, for some problems it is convenient to solve the large systems of nonlinear equation characteristic of implicit schemes instead of stepping explicitly in time.

Contact problems give rise to generally non-smooth mechanics, even in apparently simple settings. In particular, this means that often the trajectory of the mechanical system is multiply-defined, as in the well-known problem by Truesdell [5]. This is materialized in the formulation of the evolution of the dynamic equations as differential inclusions, see e.g., [6, 7, 8, 9]. Selection criteria of different forms have been proposed to remedy the lack of a unique evolution [6, 10, 5]. Finite element discretizations lead to finite dimensional mechanical systems, and often the selection criterion is tacitly prescribed by, for example, the choice of the values of the different penalty parameters.

Recently variational integrators for contact problems have been formulated [11, 12]. The resulting integrators are implicit, and exhibit outstanding energy conservation properties for very long integration times. The main drawback is that the impact time of each contact event needs to be solved for. Arguing that for problems involving deformable bodies solving for each contact event could be a computationally overwhelming task, Cirak and West in [13] proposed relaxing this condition, as penalty methods do. In doing so, they lost the precise variational nature of the algorithm, but obtained an explicit contact algorithm with very good conservation properties as well. The algorithm can be summarized as follows: a momentum-preserving variational integrator is adopted to integrate the system forward for some given initial condition. If at a given time step interpenetration is detected, a projection is defined to move the configuration of the system slightly to a penetration-free one, and a new momentum for the system is computed to respect the appropriate energy and momentum balances. This new position and momentum for the system can be interpreted as new initial conditions to be integrated forward with the variational integrator until the following contact event. A similar design concept for purely rigid body interactions can be found in [14]. Finally, a nice aspect of the work in [13] is that they proposed an intrinsic decomposition of the momentum of the system during a contact event that enables the extension of their algorithm to include a coefficient of restitution.

A drawback of the algorithm in [13] is that at each contact event an energy drift is observed. Consequently, the outstanding energy behavior of the variational integrator between contact events is deteriorated by energy loss or gain at each contact event (if the system is supposed to conserve energy). To obtain a satisfactory energy behavior very small time steps need to be adopted. The algorithm proposed here addresses this problem. The basic idea is to take advantage of the possibility of advancing each element with its own time step afforded by the Asynchronous Variational Integrators (AVI) [15, 16, 17] to take small time steps only in those elements that violate the contact constraints. As shown here, the resulting algorithm has a much better energy behavior at a small additional computational cost, only because the
time step is reduced at very few elements in the mesh during the contact event only. Since AVIs are momentum-preserving and variational, they display the nice behavior we expect to have between two contact events. We have briefly outlined this idea in the context of ballistic penetration problems in [18].

A related idea has been proposed in [19]. In this case, the authors constructed a multilayer penalty potential with increasing stiffness as the depth of penetration grows, and use AVI to avoid having to reduce their time step in the entire mesh. The algorithm is perfectly variational, and if the time steps are chosen small enough, then there is no energy drift through each contact event. Otherwise, similar energy drift as that observed in the algorithm in [13] would be observed.

To formulate the contact algorithm, we introduce a set of level set functions to define the contact boundary. Instead of projecting back to it once the system violates the contact constraint, as in [13], we use the nearby level sets of such functions as approximate contact boundaries to apply the contact conditions. Additionally, we introduce a time step refinement zone (TSRZ). Whenever an element has any of its nodes within the TSRZ, its time step is reduced so as to minimize the energy drift upon contact.

In this paper we focus our attention to the formulation of the contact algorithm for deformable bodies impacting rigid (moving) walls. This is a necessary step towards formulating the algorithm for impacts between deformable bodies. In this setting, it is simpler to study the energy behavior of the algorithm and it is possible to circumvent some of the issues introduced by the possibly multiply-defined outcomes of each contact event. Interaction between deformable bodies are conceptually similar, but a careful study of the algorithm behavior is better left for a separate paper. Additionally, they introduce a host of issues related to contact detection algorithms in an asynchronous setting, as done for example in [19]. We do consider the case of a rigid spherical projectile interacting with a deformable body.

This paper begins with Section 2 by reviewing the decomposition of the momentum in [13] for frictionless contact problems in finite dimensional system. Section 3 then further discusses the basic idea behind the integrators proposed here further, leading to a review of AVI in Section 4. Section 5 introduces the asynchronous contact algorithm. Section 6 examines the numerical performance of the algorithm, which includes a detailed evaluation of the conservation and convergence properties. In particular, we analyze a one-dimensional example to explain the reasons behind the energy drift after a contact event. We also compare the computed stresses between this method and a simple penalty-based one, and numerically evaluate the convergence rate of the algorithm. A numerical example involving the ballistic penetration of a nonlinear elastic block is shown in Section 7, for which we slightly extend the algorithm to handle the contact of deformable bodies with rigid spheres.

2. Frictionless Contact Mechanics for Finite Dimensional Mechanical Systems

We briefly recall here elementary aspects of contact mechanics for finite dimensional mechanical systems. A more detailed description can be found elsewhere, e.g., [6, 13] and references therein. We consider a finite dimensional mechanical system whose motion can be described with Cartesian coordinates \( q \in \mathbb{R}^d \). The equations of motion for this system written in these coordinates take the form

\[
M \ddot{q} = f, \tag{1}
\]
at all times, where $\mathbf{f}$ denotes all forces acting on the particles, and $\mathbf{M}$ denotes the symmetric and positive definite mass matrix of the system in these coordinates. This is the case for most mass matrices obtained as a result of a finite element discretization. For this set of coordinates the conjugate momenta are defined as

$$\mathbf{p} = \mathbf{M} \mathbf{q}.$$  \hfill (2)

The forces often satisfy $\mathbf{f}(\mathbf{q}) = -\nabla V(\mathbf{q})$, for some potential energy $V$. In this case, the finite dimensional mechanical system above is Hamiltonian, and among other things, the mechanical energy of the system

$$H(\mathbf{q}, \mathbf{p}) = \frac{1}{2} \mathbf{p}^T \cdot \mathbf{M}^{-1} \cdot \mathbf{p} + V(\mathbf{q})$$  \hfill (3)

is conserved throughout the trajectory.

A type of contact constraint we consider here is defined as a restriction of the mechanical system to have its coordinates satisfy $\mathbf{q} \in \mathcal{Q}$, where $\mathcal{Q}$ is an open subset of $\mathbb{R}^d$. The closure of $\mathcal{Q}$, $\overline{\mathcal{Q}}$, is termed the admissible region. We will assume that $\mathcal{Q}$ has a piecewise smooth boundary, with a uniquely defined outer normal $\mathbf{n}$ almost everywhere. We will also assume that $\mathcal{Q}$ can be specified implicitly through a set of functions $g_i : \mathbb{R}^d \rightarrow \mathbb{R}$, $i = 1, \ldots, n_g$, as

$$\mathcal{Q} = \{ \mathbf{q} \in \mathbb{R}^d | g_i(\mathbf{q}) < 0, \ i = 1, \ldots, n_g \}.$$  \hfill (4)

Furthermore, we will assume that

$$\mathbf{q} \in \partial \mathcal{Q} \Leftrightarrow g_i(\mathbf{q}) \leq 0 \text{ for all } i \in \{1, \ldots, n_g\} \text{ and } g_j(\mathbf{q}) = 0 \text{ for some } j.$$  

This last conditions enables us to identify points in $\partial \mathcal{Q}$ by evaluating the functions $g_i$.

For example, for a system of $N$ particles in two-dimensions with Cartesian coordinates $(x_1, y_1, x_2, y_2, \ldots, x_N, y_N)$ that need to move in the $\{y > 0\}$ region of the plane, we may define $N$ functions $g_i$ as

$$g_i(x_1, y_1, \ldots, x_N, y_N) = -y_i.$$  \hfill (5)

We will also assume that the functions $g_i$ are smooth and $\nabla g_i \neq 0$ in a neighborhood of $\partial \mathcal{Q}$.

Under the contact constraint, the evolution of the mechanical system may undergo impact events. An impact event occurs when at a time $t_c$ the system satisfies $g_j(\mathbf{q}(t_c)) = 0$ for some $j$ and $g_i(\mathbf{q}(t_c)) \geq 0$ for all $i$. It is possible to have a continuous time interval of contact events, during which the system moves along the boundary of the admissible region, known as persistent contact. The discussion below, however, focuses on isolated contact events, in which the momentum of the system is continuous in an interval of time around $t_c$, except perhaps at $t_c$. In this context, we define

$$\mathbf{p}^\pm = \lim_{t \to t_c^\pm} \mathbf{p}$$  \hfill (6)

as the momentum of the system just before (-) and just after (+) the contact event. Since at each contact event the momentum of the system may be discontinuous, we may have $\mathbf{p}^- \neq \mathbf{p}^+$. When $g_i(\mathbf{q}(t_c)) < 0$ for all $i \neq j$ and $g_j(\mathbf{q}(t_c)) = 0$ for some $j \in \{1, \ldots, n_g\}$, then the relation between the $\mathbf{p}^+$ and $\mathbf{p}^-$ is obtained from the contact conditions

$$p_{n}^+ = -p_{n}^-,$$  \hfill (7a)

$$p_{\ell}^+ = p_{\ell}^-.$$  \hfill (7b)
AN EXPLICIT ASYNCHRONOUS CONTACT ALGORITHM

\[ p_n^\pm = p^\pm \cdot M^{-1} \cdot n_j(q) \quad \text{and} \quad p_t^\pm = p^\pm - p_n^\pm n_j(q), \]

respectively, where for simplicity we denoted \( q \equiv q(t_c) \). The unit normal \( n_j(q) \) to \( \partial Q \) at \( q \), can be computed from \( g_j \) as

\[ n_j(q) = \frac{\nabla g_j(q)}{[\nabla g_j(q) \cdot M^{-1} \cdot \nabla g_j(q)]^{1/2}}. \]

These equations are graphically depicted in Fig. 1. This normal and tangential decomposition of the momentum of the system was introduced in [13].

In contrast, when \( g_j(q(t_c)) = 0 \) for more than one value of \( j \), and \( g_i(q(t_c)) < 0 \) for all remaining values of \( i \), the system has likely reached a corner of \( \partial Q \). Contact conditions at corners of \( \partial Q \) are generally not well defined, since there is not a single normal to \( \partial Q \) therein. For example, for the systems of \( N \) particles in (5), if \( q \in \overline{Q} \) has two or more particles at \( y = 0 \), then \( q \) is a corner of \( \partial Q \). For a convex admissible region \( \overline{Q} \) with a corner at \( q \in \partial Q \), the impulse imparted on the system by a collision therein can lie anywhere in the normal cone at \( q \) (see, e.g., [6]). In general, the evolution of the system after a contact event can only be defined after additional conditions are prescribed, since the balance of momentum and energy are satisfied by more than one possible outcome.

Because in the algorithm described later we encounter the system at a corner of the admissible region often, we prescribe one specific outcome for such event, or selection criterion, when describing the algorithm.

Prepared using nmeauth.cls
Symplectic or variational integrators possess properties that make them attractive for the simulation of problems in solid dynamics (cf. [20, 21]). Chief among these properties is exact linear and angular momentum conservation and excellent long-term energy conservation. Of course, this can only occur if the mechanical system has the same properties.

A simple way to regard the dynamics of the type of finite dimensional mechanical systems above is as a sequence of discrete contact events at time instants \( t_1, t_2, \ldots, t_k, \ldots \). The system evolves between times \( t_{k-1} \) and \( t_k \) according to (1). At time \( t_k \), the contact conditions (6) and (7) compute a new set of initial conditions for the evolution between \( t_k \) and \( t_{k+1} \) (same coordinates \( \mathbf{q} \), but the momentum changes from \( \mathbf{p}^- \) to \( \mathbf{p}^+ \)), and the evolution of the system then continues according to (1). This is sketched in Fig. 2.

A natural way to think about the design of time integration algorithms when contact is involved is to select any standard time-integration scheme to evolve the system between times \( t_{k-1} \) and \( t_k \), for any \( k \), and then use the contact conditions (6) and (7) to reinitialize the simulation for the next integration interval, in this case \( (t_k, t_{k+1}) \). Consequently, if we select a symplectic time-integration scheme to perform the time integration between contact events, we should expect to obtain a time-integrator with outstanding energy and momentum conservation properties.

This ideal scenario has some drawbacks. First, it requires the precise detection of each contact event. Second, while the energy between contact events is nearly exactly preserved due to the symplectic nature of the time-integration algorithm, a noticeable energy change is observed after each contact event. Such energy drift can be partially analyzed and understood based on the computation of the shadow Hamiltonian of the underlying symplectic integrator, which we illustrate with an example in §6.2. However, it also emanates from the analysis that the size of the energy drift after each collision scales with the time step of the time integrator.

This is one key motivation behind the use of an asynchronous time integrator for a deformable body spatially discretized with finite elements. Since generally only a few elements are involved in a contact event at any time, it is possible to obtain a much smaller energy drift on each contact event by reducing the size of the time steps on those elements only.
4. Asynchronous Variational Integrators (AVI)

Asynchronous variational integrators were introduced in [15, 16] as multisymplectic time integrators for deformable bodies. A parallel version of these integrators has been introduced in [22]. We briefly highlight some features below, and refer the reader to the earlier references for a thorough description.

AVI can be regarded as an extension to asynchronous time stepping of Newmark’s explicit second-order method (also known as central differences or velocity Verlet). This last algorithm is given by the following update equations that return \( (\mathbf{q}^{i+1}, \mathbf{p}^{i+1}) \) if \( (\mathbf{q}^i, \mathbf{p}^i) \) is given:

\[
\begin{align*}
\mathbf{p}^{i+1/2} &= \mathbf{p}^i + \frac{\Delta t}{2} \mathbf{f}^i \\
\mathbf{q}^{i+1} &= \mathbf{q}^i + \Delta t \mathbf{M}^{-1} \mathbf{p}^{i+1/2} \\
\mathbf{p}^{i+1} &= \mathbf{p}^{i+1/2} + \frac{\Delta t}{2} \mathbf{f}^{i+1}
\end{align*}
\]  

(10a), (10b), (10c)

where \( \Delta t = t_{i+1} - t_i \) is the time step, and for simplicity the vector of forces \( \mathbf{f}(\mathbf{q}) \) was assumed to depend on the coordinates of the particle only. Of course, other types of forces, such as dissipative forces, can be included in a standard way. With conservative forces only, this is a variational integrator that conserves linear and angular momentum, and nearly exactly preserves the value of the energy for very long times.

AVI extends this algorithm to multiple time steps by relying on an additive decomposition of the forces, namely,

\[
\mathbf{f}(\mathbf{q}) = \mathbf{f}_1(\mathbf{q}) + \ldots + \mathbf{f}_N(\mathbf{q})
\]  

(11)

For example, for a deformable body spatially discretized with finite elements, \( \mathbf{f}_i \) are the internal forces computed with element \( i \). Each one of these forces is then assigned a possibly different time step (subjected to stability considerations), and the order in which each one of them needs to be computed is determined based on their time steps. Additionally, the mass matrix \( \mathbf{M} \) is assumed to be diagonal, or in the case of finite elements, it is lumped. Equations (10) are then applied to advance the positions and velocities from the time step of one force to the next time step among all forces. More precisely, each node \( a \) that belongs to an element \( K \) whose internal forces are computed at time \( t_K \) is updated at this time according to

\[
\begin{align*}
\mathbf{q}^{i_a}_{i_a} &= \mathbf{q}^{i_a-1}_{i_a} + \Delta t_{i_a} \mathbf{M}^{-1}_{i_a} \mathbf{p}^{i_a-1/2}_{i_a} \\
\mathbf{p}^{i_a}_{i_a} &= \mathbf{p}^{i_a-1/2}_{i_a} + \frac{\Delta t_{i_K}}{2} \mathbf{f}^i_{K} \\
\mathbf{p}^{i_a+1/2}_{i_a} &= \mathbf{p}^{i_a-1/2}_{i_a} + \Delta t_{K} \mathbf{f}^i_{K}
\end{align*}
\]  

(12a), (12b), (12c)

where \( \mathbf{M}_{i_a} \) is the diagonal entry in the mass matrix for node \( a \), \( \mathbf{q}^{i_a}_{i_a} \) denotes the coordinates of node \( a \) at time \( t_{i_K} \) or its \( i_a \)-th update, \( \Delta t_{i_a} \) is time interval between the \( i_a \)-th and \( (i_a - 1) \)-th updates of node \( a \), and \( \mathbf{p}^{i_a-1/2}_{i_a} \) is the momentum of the node in between two updates. Together with \( \mathbf{q}^{i_a-1}_{i_a} \), \( \mathbf{p}^{i_a-1/2}_{i_a} \) is assumed to be known at the beginning of the update. Additionally, \( \mathbf{p}^{i_a}_{i_a} \) denotes the momentum of the node at the \( i_a \)-th update of node \( a \). The internal forces \( \mathbf{f}_{K} \) are computed with the updated positions at time \( t_K \) of all nodes in element \( K \), and \( \mathbf{f}^i_{K} \) denotes the internal forces acting on the momenta of node \( a \). Time step \( \Delta t_{i_K} \) is computed as the time interval between \( t_K \) and the last time the internal forces for the same element were computed. A more detailed description of the algorithm can be found in [23].
In the context of deformable bodies, this reduces to advancing each element with its own time step, computing the internal forces on that element at each time step, and imparting an impulse on each one of its nodes computed from these internal forces. The evolution of the elements in time for a one-dimensional bar is sketched in Fig. 3. Each horizontal line above an element is an internal force computation. After each one of these computations, the slope of the trajectories of the nodes in spacetime changes, since the impulse exerted by the element has changed their velocities. Causality is respected by making sure that all operations are performed in the right order. We generally do it through a priority queue, although other alternatives are possible.

![Figure 3. Example of a one-dimensional, three-element mesh discretizing a deformable body that is advanced in time with AVI. The evolution in time of the reference configuration is shown on the left, while that of the deformed configuration is on the right. The horizontal lines represent each elemental update, while the dashed lines represent the trajectories of the nodes.](image)

Therefore, in addition to the outstanding momentum and energy conservation properties, AVI has the desirable feature that the time step of each element can be selected independently. This is in contrast with synchronous explicit Newmark’s methods where the smallest element in the mesh generally determines the time step for all elements in the simulation. The stability of asynchronous methods is a complex issue that is not fully resolved yet. For example, combinations of arbitrarily small time steps can generate instabilities, as discussed in [23]. In practice, however, for meshes that do not have very sharp spatial changes on element sizes or material constants, selecting each individual time step to be a fraction of the local CFL condition renders stable integrators. More precisely, the time step for an element with diameter $h$ is set to be $\Delta t = fh/c$, where $c$ is the maximum wave speed of the material and $f \in (0, 1)$ is called the time factor and sets the time step to be a fraction of the CFL limit.
5. Explicit Asynchronous Contact Algorithm

In the following we introduce an asynchronous contact algorithm based on the ideas discussed in Sections 3 and 4. Our contact algorithm consists of essentially two main ideas:

1. The consideration of level sets of \( g_i \) near the level set \( \{ q \in \mathbb{R}^d \mid g_i(q) = 0 \} \) to have a collection of possible boundaries of the admissible region near the exact one. Instead of having a fixed admissible region \( Q \), we consider the possibility of the system moving into coordinates outside of \( Q \) during a fraction of the time step, adopting as the boundary of the admissible region whatever level set of each \( g_i \) the system falls on at that time. This idea lowers the order of convergence of the algorithm (and could make some contact events to be utterly missed in some cases). However, it is a computationally efficient approach because it circumvents the need to find the exact time at which a contact event happens, in the context of an explicit algorithm. A discussion about the convenience of this choice is provided in [13]. In contrast to [13], we do not project the configuration of the system back on \( \partial Q \), although in some occasions it could be convenient. For example, when working with thin shells.

2. The creation of a zone near the boundary of the admissible region in which the time step of elements with a node inside it is reduced, termed the time step refinement zone (TSRZ). This zone serves to minimize the significance of the violations of the contact constraint introduced by the first idea. More importantly, however, this is what makes the energy behavior of the algorithm substantially better. We are able to do this because with AVI we can choose the time step for each element in the mesh.

We discuss the details of these ideas below.

5.1. Construction of the admissible region

Given a deformable body spatially discretized with finite elements, and an open set \( \mathcal{R} \in \mathbb{R}^3 \) termed the rigid wall, we define the admissible region \( Q \) with

\[
g_a(q) = -d(q_a; \mathcal{R})
\]

for each node \( a \) in the mesh, where \( q_a \) denotes the Cartesian coordinates of node \( a \), and \( d : \mathbb{R}^3 \rightarrow \mathbb{R} \) is the signed distance function of a point in \( \mathbb{R}^3 \) to the rigid wall \( \mathcal{R} \). The value of \( d(q_a; \mathcal{R}) \) is negative if \( q_a \in \mathcal{R} \), zero if \( q_a \in \partial \mathcal{R} \), and positive otherwise. Its magnitude is equal to the distance of \( q_a \) to its closest point in \( \mathcal{R} \). In this way, \( g_a \) is equal to zero on \( \partial Q \), negative in \( Q \), and positive elsewhere. With this definition, we have \( n_g \) equal to the number of nodes in the mesh.

5.2. Time Step Refinement Zone

The second central element of our contact algorithm is using the possibility of adopting independent time steps on each element of AVI to improve the accuracy of the numerical results without vastly increasing the computational cost. This property not only prevents elements with poor aspect ratios or local refinements from affecting the time step of the entire mesh, but it also allows for time step refinement in regions where additional accuracy is required. One such case is for contact, where insufficiently small time steps can induce energy drifts at each contact event, as well as deep penetrations of the system into the inadmissible region.
This is particularly relevant for high velocity contact, where it becomes necessary to prevent elements from inverting, as well as the associated onset of instabilities. Locally refining the time step in the region of the contact interface is a nice application of AVI.

The method proposed here takes full advantage of this property by creating a Time Step Refinement Zone (TSRZ). The TSRZ is defined as

$$TSRZ = \{ \mathbf{x} \in \mathbb{R}^3 | d(\mathbf{x}; \mathcal{R}) < C_{TSRZ} \}$$

(14)

for some $C_{TSRZ} \geq 0$. The calculation of (13) already involves computing $d(\mathbf{q}_a; \mathcal{R})$ for any node $a$, so the computation of the TSRZ does not involve any additional calculation, at least in the context of deformable and rigid body contact.

The way the TSRZ works is that whenever an element has any of its nodes within the TSRZ, the time factor $f$ on the element is reduced to a fraction of its value, decreasing in this way the time step size. The time step size of an element is changed after the elemental update that follows the entrance of the first node of the element in the TSRZ, and remains that way as long as the above condition is satisfied. The full time step of the element is restored at the elemental update that follows the last node of the element leaving the TSRZ. A sketch of these ideas is shown in Fig. 4.

The size of the TSRZ defined by $C_{TSRZ}$ and the time factor within it should be sized so as to prevent nodes from entirely bypassing the zone. Given foreknowledge of the expected maximum velocity attainable by a node, the TSRZs are designed to prevent that. A few iterations might be needed to tune this value for a given problem, so as to avoid defining a very large TSRZ. In many problems, however, this value can be robustly set a priori.

5.3. Contact Detection and Momentum Reflection

As mentioned earlier, we adopt a strategy in which we do not detect the exact time at which our system reaches the boundary of the admissible region. Instead, we allow the system to attain values outside the admissible region, and define a new admissible region boundary wherever the system fails.

The contact detection and momentum reflection strategy are described next. Following the notation in Sec. 4, after the elemental update of element $K$ according to (12a) and (12b), we check whether

$$g_a(\mathbf{q}_a^i) = -d(\mathbf{q}_a^i, \mathcal{R}) \geq 0 \quad \text{and} \quad (p_a^{i-a})_n = p_a^{i-a} \cdot \mathbf{M}_a^{-1} \cdot \mathbf{n}_a(\mathbf{q}_a^i) > 0,$$

(15)

for each node $a$ in $K$, where we have abused notation and used $\mathbf{q}_a$ as the sole argument for $g_a$. Here

$$\mathbf{n}_a(\mathbf{q}_a) = \frac{\nabla g_a(\mathbf{q}_a)}{[\nabla g_a(\mathbf{q}_a) \cdot \mathbf{M}_a^{-1} \cdot \nabla g_a(\mathbf{q}_a)]^{1/2}}.$$  

(16)

Then, each node $a$ in $K$ for which conditions (15) are met has its momentum changed according to

$$\mathbf{p}_a^{i-a} = \mathbf{p}_a^{i-a} - 2(p_a^{i-a})_n \mathbf{n}_a(\mathbf{q}_a^i).$$

(17)

Equation (17) is a consequence of contact conditions (7).

A few remarks are appropriate. Notice first that one way of regarding this algorithm is as defining different layers parallel to the rigid wall, and every time a node is advanced due to an elemental update and finds one of these layers, it bounces back from it. Since we are allowing
the nodes to penetrate the wall, we need the second condition in (15), so that we do not reflect the momentum of a node that is still inside the wall but is already moving towards escaping from it.

In terms of the more abstract perspective introduced in §2, we could regard this algorithm as having instantaneously defined the admissible region for the nodes in element \( K \) at time \( t_i \) as the closure of
\[
Q^K(t_i) = \{ \mathbf{q} \in \mathbb{R}^d \mid g_a(\mathbf{q}_a) < g_a(\mathbf{q}_i^a), a \in K \}.
\]
In this case we have taken advantage of the fact that \( g_a \) depends only on the value of \( \mathbf{q}_a \) itself, since it would not be possible to write a similar equation in the most general case. With this perspective, it the follows that when two or more nodes of \( K \) are inside the wall, then the system is making contact at a corner of of \( Q^K(t_i) \). As such, the outcome of the system after contact is not precisely defined, and a selection criteria needs to be adopted to decide, see e.g. [10, 5].

In our case the choice of the selection criterion is very natural, since each node reflects back from the surface by conserving the kinetic energy it brought with it. In particular, this makes the order in which the collision of each node is treated irrelevant. This is not the case if instead of a rigid wall the nodes make contact with, for example, a rigid body with finite mass. We discuss more about this in §7, when we present one such numerical example.
5.4. Asynchronous Contact Algorithm

A detailed description of the algorithm for asynchronous time-stepping with contact is provided in Algorithm 1, which gathers all the ideas discussed in §5.1, §5.2 and §5.3. For the sake of clarity, some evident optimizations are not reflected in the algorithm. For example, the internal force computation with each element in lines 21 and 26 need not be repeated, but can be computed only once each time the elements is popped from the priority queue. Similarly, depending on the computational cost of evaluating $d(q; R)$ and its gradient, it might be convenient to perform this elsewhere and not where, for simplicity, is indicated in the algorithm. In Algorithm 1 we denote by $q_K$ the set of positions of all nodes in element $K$.

This algorithm bears many similarities to the synchronous algorithm proposed in [13]. A notable difference of the algorithm in [13] with the one here is that the second condition in (15) is not needed, since the authors map $(q_i^-, p_i^-)$ to $(q_i^+, p_i^+)$ for each node $a$ by setting $q_i^+$ to be a projection of $q_i^-$ on $\partial Q$. They then compute $p_i^+$ using the normal $n(q_i^+)$ in (7) and (8). For a smooth enough $g$, both normals $n(q_i^-)$ and $n(q_i^+)$ become progressively similar as the time step is reduced. The projection used in [13] can be applied to our asynchronous algorithm. This projection repositions nodes inside the inadmissible region directly to the contact surface. However, we did not observe any improvement as a result of including this projection in the algorithm.

An undesirable aspect of the asynchronous contact algorithm is that the discrete trajectories might not change continuously with the time steps of each element. Nevertheless, the differences between computed trajectories vanish as uniformly smaller time steps in all elements are adopted. This is a result of the nonlinearity of the contact conditions (15). Consider for example two elements $K_1$ and $K_2$ sharing a node $a$. By keeping the time step of $K_2$ constant and slightly changing that of $K_1$, the relative order in which elemental updates of the two elements happen can change. For concreteness, assume that the time steps of the two elements are such that both are updated at some time $t$. In the absence of contact, updating first $K_1$ and $K_2$, or first $K_2$ and then $K_1$, results in exactly the same momentum for node $a$ after time $t$. Instead, if condition (15) is satisfied at time $t$, then the two alternative ways of ordering $K_1$ and $K_2$ to be updated will almost certainly render different outcomes for the momentum of $a$ after time $t$. The difference in the momentum of $a$ between the alternatives is $O(\Delta t_{K_1}) + O(\Delta t_{K_2})$, so it vanishes as smaller times steps are adopted. This non-commutativity of the elemental updates ultimately affects the rate of convergence of the algorithm, rendering it at most first order in time.

For small enough time steps, the violations of the contact constraint are akin to what is encountered in adopting a penalty method, with the difference that it takes the penalty approach many time steps to eventually revert the direction of the normal component of the momentum, while here it is done in a single time step. Additionally, in this case we do not need to define what the penalty parameter(s) should be. However, a nice feature of a penalty approach is that, if a small enough time step is chosen, the resulting mechanical system is perfectly Hamiltonian and hence by adopting a symplectic integrator like (10) the energy through each contact event is nearly perfectly conserved. As we shall see in the numerical examples, the approach we adopt here does introduce a drift of the energy at each contact event, the size of which can be controlled by reducing the time step size (a similar behavior would be observed if a penalty approach was adopted and the time step was not small enough). A nice advantage of our approach over a penalty one, however, is that by considering smaller
and smaller time steps the solution converges to the exact one*, see §6.4. This is unlike penalty methods, where the time step refinement is not enough to converge. The penalty parameter should be properly scaled with the time step to attain convergence.

Figure 5. Sketch of one of the ideas behind the proposed contact algorithm. At time $t_{i-1}$ the system is within the admissible region, but in the following time step the system has violated the contact constraint. Instead of projecting back to the $g_a = 0$ surface, or backtracking to find the precise time at which contact happens, we adopt as the instantaneous contact boundary the level set of $g_a$ that goes through the position of the system at time $t_i$. Contact conditions are applied with respect to this new boundary and the simulation is continued with a new momentum for the system at time $t_i$. In this case, the system is still not in the admissible region at time $t_{i+1}$, but since it is moving towards it, no contact conditions are applied until time $t_{i+2}$. The system is back in the admissible region at time $t_{i+3}$.

6. Performance

In the following, we demonstrate the performance of the proposed algorithm and highlight its main properties through numerical examples. All simulations were performed by choosing a nonlinear elastic model for the continuum, and discretizing it with piecewise affine finite elements. The mass matrix was lumped by equidistributing the mass of each element among its nodes, see e.g. [24]. For completeness, the Hamiltonian (3) for this system is

$$H(q, p) = \sum_{\text{all nodes } a} \frac{1}{2} m^{-1}_a p_a \cdot p_a + \sum_{\text{all elements } K} \int_K W \left( \sum_{a \in K} q_a \nabla N_a \right) dV, \quad (19)$$

*When a single solution can be defined
Algorithm 1 Asynchronous Contact Algorithm

1: Input: $t^0, T, \{q^0_a, p^0_a\}$ for every node $a$
2: Output: $\{t^+_a, q^+_a, p^+_a\}$ for all nodes $a$ and all times $t^+_a < t_{\text{final}}$

Require: $q^0 \in Q$

3: \{$\text{Initialization}$\}
4: $q_a \leftarrow q^0_a$, $p_a \leftarrow p^0_a$, $t_a \leftarrow t^0$, for all nodes $a$
5: for all element $K$ do
6: \{$\text{Advance in time}$\}
7: while $Q$ not empty do
8: Pop top element $(\tau^K, K)$ from priority queue $P$
9: for all node $a \in K$ do
10: $q_a \leftarrow q_a + (\tau^K - t_a)M^{-1}_a p_a$
11: $t_a \leftarrow \tau^K$
12: end for
13: if $\min_{a \in K} g_a(q) < C_{\text{TSRZ}}$ then
14: $\tau^K \leftarrow \tau^K + \Delta t_{\text{TSRZ}}$
15: else
16: $\tau^K \leftarrow \tau^K + \Delta t_K$
17: end if
18: if $\tau^K < t_{\text{final}}$ then
19: Push $(\tau^K, K)$ into priority queue $P$
20: end if
21: for all nodes $a \in K$ do
22: Output $(t_a, q_a, p_a)$
23: $p_a \leftarrow p_a + \frac{\Delta t_K}{2} f^K_a(q_K)$
24: end for
25: $q_a \leftarrow q_a + (t_a - t^0)M^{-1}_a p_a$
26: $t_a \leftarrow t_a$
27: end for
28: end while

Prepared using mmeauth.cls
Algorithm 2: Apply Contact Conditions Algorithm

1: Input: \((K, q, p)\)
2: Output: \(\{p_a\}_{a \in K}\)
3: 
4: for all nodes \(a \in K\) do \{Check contact\}
5: \(g_a \leftarrow -d(q_a; R)\)
6: if \(g_a \geq 0\) then \{Possible contact\}
7: \(\nabla g_a \leftarrow -\nabla d(q_a, R)\)
8: \(\nabla g_a^{\text{norm}} \leftarrow [\nabla g_a \cdot M_a^{-1} \cdot \nabla g_a]^{1/2}\)
9: \(p_a^- \leftarrow p_a \cdot M_a^{-1} \cdot \nabla g_a / \nabla g_a^{\text{norm}}\)
10: if \(p_a^- > 0\) then \{Contact conditions satisfied\}
11: \(p_a \leftarrow p_a - 2p_a^- \nabla g_a / \nabla g_a^{\text{norm}}\)
12: end if
13: end if
14: end for

where \(m_a\) is the mass of node \(a\), and \(N_a\) is its shape function. Here \(W\) is the strain energy density function, in this case set to either that of a neo-Hookean material model extended to the compressible range

\[
W(F) = \frac{\lambda}{2} \log(\det F)^2 - \mu \log(\det F) + \frac{\mu}{2} F : F,
\]

or that of an isotropic linear elastic material

\[
W(F) = \frac{\lambda}{2} \text{Trace}(F - I)^2 + \frac{\mu}{2} (F - I) : (F - I),
\]

with material constants \(\lambda\) and \(\mu\). Notice that for these systems

\[
f_K = -\frac{\partial}{\partial q_a} \int_K W \left( \sum_{b \in K} q_b N_b \right) dV.
\]

The mechanical energy in (19) should be constant as the body deforms. In the following sections we look at how well this is reproduced by the algorithm, compute convergence curves for a model problem, and compare the stresses computed in this way with those stemming from a penalty approach. Unless otherwise stated, all the following examples were computed by setting \(\lambda = 0\), \(\mu = 1/2\) and \(\rho = 1\), which gives a maximum sound speed \(c = 1\).

### 6.1. Conservation Properties

We showcase next the main benefit of the proposed asynchronous contact algorithm: the possibility of resolving contact events by adopting smaller time steps in those elements that may have contact interactions only. This benefit is directly manifested in a substantial reduction of the drift in the energy of the system before and after a contact event.

Since the energy of the system is not constant but rather oscillates around a nearly constant value for very long times, we compute the drift as the average value in time over a long enough time interval for the average to converge. We also compute its standard deviation over the same time interval, to measure the magnitude of the oscillations around the mean value.
6.1.1. Energy under Single Impact  We consider a nonlinear elastic cylinder impacting a rigid flat wall parallel to one of its circular caps. Snapshots of the cylinder are shown in Fig. 6, in which the time step adopted for each element during an asynchronous simulation is shown through the colored contours. Notice that elements near the flat wall are within the TSRZ, and hence have a much smaller time step.

We simulate this problem in two ways. In the first case, we adopt the same time step for each element, equal to a fraction $f$ of the minimum value of the CFL limit throughout the mesh. We denote it the synchronous case. The simulated energy evolution through the collision and bounce back from the wall are shown in Fig. 7(a) for $f = 0.1$ and $f = 0.01$. Being the collision perfectly elastic, the energy of the system should be conserved throughout the simulation. This is clearly the case before and after the collision, a characteristic of our symplectic integrator. The remarkable feature in this example is, however, the fact that as $f$ becomes smaller the energy drift (or loss in this case) shrinks with it.

Introducing the asynchronicity does not substantially change the energy behavior, as shown in Fig. 7(b). In this second case all elements outside the TSRZ are simulated with $f = 0.1$. The TSRZ was chosen here so that $C_{TSRZ}$ is smaller than one element size away from the wall. In one of the simulations the elements in the TSRZ have also been simulated with $f = 0.1$, so that effectively there is no TSRZ. The time steps in this case are only mildly different than those in Fig. 7(a) with $f = 0.1$, but asynchronous. The energy drift through the collision is, as mentioned earlier, essentially the same, with the asynchronous case displaying slightly larger oscillations of the energy after the collision is over. The difference however, is found in the computational time, with the synchronous case taking 93% more time than the asynchronous one.

More remarkable results are found when we change the time factor in the TSRZ to $f = 0.01$, as shown in Fig. 7(b). The energy drift in this case is much smaller than in the last case, and roughly the same to that obtained in the synchronous case by adopting $f = 0.01$ for all elements in the mesh, Fig. 7(a). The asynchronous algorithm with $f = 0.01$ in the TSRZ only is approximately 20 times faster than the synchronous algorithm with the same time factor throughout the mesh, rendering the same energy drift after the collision. Changing the time factor from $f = 0.1$ to $f = 0.01$ increased the computational cost of the asynchronous algorithm by approximately 19% (measured as the number of elemental updates), but rendered a much smaller energy drift.

It is somewhat remarkable that the energy drifts in Figs. 7(a) and 7(b) seem to depend exclusively on the time steps of the elements that enter in contact with the wall. We investigated this some more. We first examined the effect of changing the time factor in those elements outside the TSRZ, while leaving the time factor in element in the TSRZ fixed. The results shown in Fig. 8(a) demonstrate that the energy drift after the collision is over essentially does not depend on the time step outside the TSRZ. The size of the oscillations in energy do increase markedly with it. In contrast, by fixing the time factor outside the TSRZ and changing the one inside it, we find that the energy drift decays linearly with the time step inside the TSRZ, and that the magnitude of the energy oscillations after impact remains unchanged, see Fig. 8(b).

A more comprehensive perspective on the effect of changing the time factors within and without the TSRZ is shown in Fig. 9. Therein we plotted the mean value and standard deviation of the energy drift after the collision is over. Remarkably, the figure shows that the energy drift seems to depend essentially on the time step inside the TSRZ only, while the magnitude
Figure 6. A coarse (3D) mesh for a cylinder impacting a flat wall. The time instant of each snapshot grows to the right. The time step for each element is indicated by the contour plot, highlighting their variation throughout the mesh and within the TSRZ.

(a) The same time step was selected for all elements in the mesh, equal to a fraction \( f \) of the minimum of the CFL limit among all elements in the mesh. Simulations for \( f = 0.1 \) and \( f = 0.01 \) are shown, as indicated in the legend.

(b) Asynchronous time stepping with a time factor \( f = 0.1 \) for all elements outside the TSRZ. Elements in the TSRZ have \( f = 0.1 \) or \( f = 0.01 \), as indicated in the legend.

Figure 7. Evolution of the mechanical energy of an elastic cylinder impacting into a rigid wall. The simulations used affine tetrahedral elements with standard mass lumping. The maximum sound speed in the undeformed cylinder is 1, and the initial velocity of the cylinder with respect to the wall is 0.1.
of the oscillations after impact (measured by the standard deviation) seem to depend only on
the time factor in the elements outside the TSRZ, with an approximate first-order dependence
in both cases. Quadratic tetrahedral elements have been used to compute Figure 9, to show
that these results are similar for both types of widely used elements.

(a) Changing the time step outside the TSRZ, as indicated by the time factor in the legend.

(b) Changing the time step in the TSRZ, as indicated by the time factor in the legend.

Figure 8. Effect of changing the time steps on the energy evolution in the simulation of the elastic
cylinder impacting against a flat rigid wall of Fig. 7.

Several of the advantages mentioned during the algorithm description have been illustrated
here: the energy level is well-conserved both before and after the impact; and the time step in
the TSRZ reduces the energy drift through the collision. This last benefit is gained by changing
the time step for only a relatively small number of elements, resulting in substantially better
energy behavior without a large increase in the computational cost.

6.1.2. Energy under Multiple Impacts The case of repeated impacts of an elastic sphere
against multiple walls of a cube serves to illustrate the consistent energy behavior of the
algorithm. In this example, the sphere starts by traveling along a line forming angles of 45
degrees with two walls of the cube, and orthogonal to their common edge. The sphere initially
impacts these two walls simultaneously. It then bounces back and impacts the walls adjacent to
the opposite edge of the cube. Due to the non-symmetric spatial discretization of the sphere, at
later times the sphere impacts the two walls adjacent to each edge at different times. Eventually
the sphere leaves the edge entirely, impacting only a single wall at each impact, see Fig. 10 for
some snapshots. Somewhat surprisingly, figure 11 shows that the energy drift resulting from
each impact for this example happens to always be negative, a feature that we did not observe
in all cases we tested. Each impact against walls can be detected in this plot as a sharp drop
in the energy value. Single wall interactions have roughly half the energy effect of the initial

---

1The mass lumping scheme for quadratic elements consisted in assigning 1/32 of an element mass to each node
at a vertex, and 7/48 to each node at the middle of an edge.
Figure 9. Dependence on the time factors within and without the TSRZ of the mean value and standard deviation of the energy drift after the collision is over, for the elastic cylinder impacting a rigid flat wall. Both the mean energy drift and the standard deviation are normalized with the initial total energy. The simulations were performed with quadratic tetrahedral elements, and the ratio formed by the velocity of the cylinder with respect to the flat wall over the bulk sound speed is 0.1.

Simultaneous impacts against two walls, which can be clearly observed in the last energy drop in Fig. 11.

Figure 10. Impact of an elastic sphere against the walls of a rigid cube in a simulation that includes 16 separate impacts against walls.
Figure 11. Evolution of the mechanical energy of an elastic sphere bouncing inside a rigid cube, with a total of 16 collision events. Only 3% of the energy is lost throughout this simulation, which can be made smaller by reducing the time step in the TSRZ. The inset shows the full energy scale.

A worthwhile observation to make is that in the absence of friction, no rolling of the sphere occurs. During sustained contact, any motion along the boundary takes the form of sliding.

6.2. Energy Behavior After Contact

Under some restrictions on the Hamiltonian and the numerical method that are non-essential for the forthcoming discussion, the trajectories computed by a symplectic integrator with a fixed time step $\Delta t$ almost exactly conserve the value of a time-step-dependent Hamiltonian $\tilde{H}_{\Delta t}$ for very long times. This is known as the shadow or modified Hamiltonian, and satisfies that

$$\tilde{H}_{\Delta t}(\mathbf{q}, \mathbf{p}) = H(\mathbf{q}, \mathbf{p}) + O(\Delta t^k),$$

(23)

where $k$ is the order of the integrator, for all $(\mathbf{q}, \mathbf{p})$ in at least a neighborhood of the exact trajectory and for a small enough $\Delta t$, see e.g. [25, 26]. This is the reason for the good energy conservation properties of symplectic schemes: the shadow Hamiltonian is nearly exactly conserved, and the value of the exact Hamiltonian is close to it, so the latter is often observed to at most oscillate around some constant value in time, and the size of these oscillations diminishes with $\Delta t$.

As discussed in §3, a natural way to think about the time integration algorithm introduced here is as evolving the system between any two contact events with a symplectic algorithm, and regarding the change in momentum introduced by the contact conditions (7) as simply the computation of the initial conditions for the evolution until the next contact event. Therefore, since symplectic algorithms essentially conserve energy for very long times, and the contact conditions (7) guarantee that

$$H(\mathbf{q}, \mathbf{p}^-) = H(\mathbf{q}, \mathbf{p}^+)$$

(24)
AN EXPLICIT ASYNCHRONOUS CONTACT ALGORITHM

at each contact event, it could be expected that the energy of the system is conserved after multiple contact events. However, as the results in §6.1.1 and §6.1.2 show, this is not the case, even for a symplectic method with a single time step. This energy drift can be understood by noticing that in general

$$\bar{H}_{\Delta t}(q, p^-) \neq \bar{H}_{\Delta t}(q, p^+)$$.

(25)

Consequently, after the contact event the discrete trajectory is likely to essentially continue along a different level set of $\bar{H}_{\Delta t}$.

We illustrate this with a simple example for which $\bar{H}_{\Delta t}$ can be computed exactly. To this end, we consider a system made of two equal masses $m = 1$ joined by a linear spring with stiffness $k = 1$ restricted to move along one spatial dimension. Its Hamiltonian is

$$H(q_1, q_2, p_1, p_2) = \frac{1}{2}(p_1^2 + p_2^2) + \frac{1}{2}(q_2 - q_1)^2,$$

(26)

where $q_i$ and $p_i$ are the displacement and momentum of mass $i = 1, 2$. The central differences method for such system reads

$$\begin{bmatrix} q_1^{n+1} \\ q_2^{n+1} \\ p_1^{n+1} \\ p_2^{n+1} \end{bmatrix} = \begin{bmatrix} q_1^n \\ q_2^n \\ p_1^n \\ p_2^n \end{bmatrix} + \Delta t \begin{bmatrix} \frac{\Delta t}{2} & 0 & \frac{\Delta t}{2} & 0 \\ 0 & \frac{\Delta t}{2} & 0 & \frac{\Delta t}{2} \\ \frac{\Delta t}{2} & 0 & \frac{\Delta t}{2} & 0 \\ 0 & \frac{\Delta t}{2} & 0 & \frac{\Delta t}{2} \end{bmatrix} \begin{bmatrix} q_2^n - q_1^n \\ q_1^n - q_2^n \\ q_2^{n+1} - q_1^{n+1} \\ q_1^{n+1} - q_2^{n+1} \end{bmatrix}.$$ 

(27a)

$$\begin{bmatrix} p_1^{n+1} \\ p_2^{n+1} \end{bmatrix} = \begin{bmatrix} p_1^n \\ p_2^n \end{bmatrix} + \Delta t \begin{bmatrix} 0 & 1 & 0 & \frac{\Delta t}{2} \\ -\frac{\Delta t}{2} & 0 & \frac{\Delta t}{2} & 0 \\ 0 & -\frac{\Delta t}{2} & 0 & \frac{\Delta t}{2} \\ -\frac{\Delta t}{2} & 0 & \frac{\Delta t}{2} & 0 \end{bmatrix} \begin{bmatrix} q_2^n - q_1^n \\ q_1^n - q_2^n \\ q_2^{n+1} - q_1^{n+1} \\ q_1^{n+1} - q_2^{n+1} \end{bmatrix}.$$ 

(27b)

This method can be regarded as a map on phase space by setting $z = [q_1 \ q_2 \ p_1 \ p_2]^T$, so that

$$z^{n+1} = R(\Delta t)z^n,$$

(28)

where

$$R(\Delta t) = \begin{bmatrix} 1 - \frac{\Delta t^2}{2} & \frac{\Delta t^2}{2} & \Delta t & 0 \\ \frac{\Delta t^2}{2} & 1 - \frac{\Delta t^2}{2} & 0 & \Delta t \\ \frac{1}{2} \Delta t (\Delta t^2 - 2) & \Delta t - \frac{\Delta t^3}{2} & 1 - \frac{\Delta t^2}{2} & \frac{\Delta t^2}{2} \\ \Delta t - \frac{\Delta t^3}{2} & \frac{1}{2} \Delta t (\Delta t^2 - 2) & \frac{\Delta t^2}{2} & 1 - \frac{\Delta t^2}{2} \end{bmatrix}.$$ 

(29)

Under the assumption that the integrator is stable, which requires $\Delta t^2 < 2$, the shadow Hamiltonian is then given by

$$\bar{H}_{\Delta t}(z) = \frac{1}{2}z^TA(\Delta t)z,$$

(30)

where the symmetric matrix $A(\Delta t)$ is such that

$$R(\Delta t) = \exp(\Delta tJA(\Delta t)),$$

(31)

with

$$J = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \end{bmatrix}$$

(32)

as the canonical symplectic form. Notice that the equations of motion under the Hamiltonian (30) have the form $\dot{z} = JA(\Delta t)z$, so if $z(0) = z^n$ then $z(\Delta t) = \exp(\Delta tJA(\Delta t))$, exactly as in
The computation of $A(\Delta t)$ can be found in, for example, [25, pp. 110]. As a result, the shadow Hamiltonian is

$$\hat{H}_{\Delta t}(q_1, q_2, p_1, p_2) = \frac{1}{4}(p_1 + p_2)^2 + \frac{1}{4}[(p_1 - p_2)^2 + (q_1 - q_2)^2(2 - \Delta t^2)] \arccos \left(1 - \frac{\Delta t^2}{2 - \Delta t^2} \right), \tag{33}$$

where $\arccos : [-1, 1] \rightarrow [0, \pi]$. Notice that for $\Delta t = 0$ we recover the exact Hamiltonian (26).

Next, without loss of generality, assume that the positions of the two masses are computed as $x_1 = q_1$ and $x_2 = 1 + q_2$, where $x$ is a coordinate along the real line, and that there is a rigid wall at $x = 0$, so the particles can only be found in $\{x \geq 0\}$. This can be identically expressed by setting

$$g_1(q_1, q_2) = -q_1, \quad g_2(q_1, q_2) = -q_2 - 1 \tag{34}$$

If at some time step we have $(q_1, q_2, p_1^-, p_2^-)$ with $q_1 < 0$, $1 + q_2 > 0$ and $p_1^- < 0$, then the conditions (15) for $a = 1$ are satisfied. It then follows from (17) that $p_1^+ = -p_1^-$ and $p_2^+ = p_2^-$. The change in the shadow Hamiltonian due to the contact algorithm is then

$$\Delta \hat{H}_{\Delta t} = \hat{H}_{\Delta t}(q_1, q_2, p_1^+, p_2^+) - \hat{H}_{\Delta t}(q_1, q_2, p_1^-, p_2^-) = p_1^- p_2^- \left( \frac{\arccos \left(1 - \frac{\Delta t^2}{2 - \Delta t^2} \right)}{\Delta t \sqrt{2 - \Delta t^2}} - 1 \right). \tag{35}$$

It can be checked that the sign of $\Delta \hat{H}_{\Delta t}$ is identical to that of $p_1^- p_2^-$. Consequently, the value of the shadow Hamiltonian may decrease, increase or remain invariant depending on the momenta of the two masses. It also follows from (35) that in this case $\Delta \hat{H}_{\Delta t} = O(\Delta t^2)$, so that the energy drift through each contact event decreases with the time step. Figure 12 schematically shows the change of the shadow Hamiltonian in this example. This example shows that (25) does effectively occur.

Notice next that the results of integrating with AVI in Fig. 9 showed the energy drift decreasing linearly with the time step in the TSRZ, in contrast with $\Delta \hat{H}_{\Delta t} = O(\Delta t^2)$. The introduction of the asynchronicity and the first-order error made due to the non-commutativity of the elemental updates, as described in §5.4, seem to be the main responsible for the slower decay of the energy drift.

Changing the time step to enter or leave the TSRZ is also expected to induce a change in the value of $\hat{H}_{\Delta t}$, as it emanates from this simple example. We have found such changes in the example in §6.1.1 for both cases, asynchronous and synchronous (in which we change all times steps simultaneously). The energy drift induced by changing the time step was always significantly smaller than that induced by the application of the contact conditions.

### 6.3. Stress Computation

One of the appealing features of the method proposed here, as well as of [13], is that there are no penalty parameters to tune. One of the many ways in which this choice directly affects the solution is by looking at the value of the stresses that result from a contact event. In this example we compare the stresses computed with the method introduced here with those obtained with different choices of a penalty parameter. In particular, the example illustrates the sensitivity of the stresses to the value of the penalty parameter, which is of course expected. But most importantly, this example illustrates another key advantage of the method adopted to impose the contact conditions here.
Figure 12. Evolution of a sample trajectory for the one-dimensional motion of two masses joined by a linear spring. The horizontal axis serves to indicate both the particle positions and the shadow Hamiltonian $\tilde{H}_\Delta t$, while the vertical axis indicates time. The left part of the plot shows various snapshots of the nodal positions before and after contact events at times $t_1$ and $t_2$. In a contact event the value of the shadow Hamiltonian may grow or decrease, depending on the momenta of the masses, but its magnitude decreases at least quadratically with the time step size.

To this end, we consider a two-dimensional plane-strain problem that has a one-dimensional analytical solution, see Fig. 13. The square block has edge length $L = 1$ and is made of an isotropic linear elastic material with mass density $\rho = 1$ and material constants $\lambda = 0$ and $\mu = 1/2$. The sound speed for volumetric waves here is $c = 1$. Initially the block is undeformed and traveling with a spatially homogeneous speed $V_0 = 0.1$ in the negative $x$-direction towards a flat wall parallel to the $y$-direction. The problem becomes essentially one-dimensional by preventing the points along the two edges of the block parallel to the $x$-direction from moving in the $y$-direction. We are interested in the Cauchy stress field for this problem at a time instant $t > t_c$ where $t_c = 0$ is the time at which the block first collides with the wall, but before the generated shock wave reaches the other end of the block, i.e., $t < L/c$. The component of the Cauchy stress tensor of interest at such time is

$$\sigma_{xx}(x, y, t) = \begin{cases} 0 & x > ct \\ -\rho c V_0 & x < ct. \end{cases}$$  \hspace{1cm} (36)$$

For this type of spatial discretization, shock waves in the continuum result in highly oscillatory velocity and stress fields. This is purely a numerical artifact and reflects the exact dynamics of the dynamical system that results from introducing the finite element discretization. Although not optimal, a standard remedy is to introduce a viscosity commensurate with the mesh size, which does not affect the stress values far away from the shock front, see e.g. [27]. In this example we introduce a Newtonian viscosity term. Since such term does not depend solely on the element’s deformed shape but also on its deformation rate, we need a slight expansion of the description in §4. In this case, for an element $K$ the nodal forces at time $t_K$ in (12) are computed by adding the elastic force in (22), as done in all previous examples, and the viscous...
Constrained in y-direction

Figure 13. Model problem to examine the algorithmic convergence rate and the computation of the stress. The linear elastic square block on the left moves with a velocity $V_0$ towards the wall, as shown on the left. To make the problem essentially one-dimensional, the two horizontal edges of the block are prevented from moving in the vertical direction. Once the collision with the wall happens, a shock wave travels through the block, as illustrated in the figure on the right.

For the force

$$f_{\eta,a}^K = \int_K P^v \cdot \nabla N_a \, dV,$$

with

$$P^v = 2\eta \det(F) \left( \text{sym} \dot{F} F^{-1} \right)_{\text{dev}} F^{-T}.$$

Here $\eta$ is the Newtonian viscosity coefficient, “sym” and “dev” indicate the symmetric and deviatoric part of a tensor, and the deformation gradient and its time derivative are approximated with

$$F = \sum_{b \in K} q_b \nabla N_b \quad \dot{F} = \frac{F - F_{\text{last}}}{\Delta t_K},$$

respectively, where the nodal positions $\{q\}_{b \in K}$ are evaluated at time $t_K$, and $F_{\text{last}}$ is the approximation of the deformation gradient computed at the last elemental update, i.e., at time $t_K - \Delta t_K$. Other, more accurate, formulations of this viscous terms are possible as well, but this one is enough for the purpose of this example.

We computed the value of $\sigma_{xx}$ in the bar at a time $t > t_c$ with our method, without including any TSRZ. The results are shown in Fig. 14, and they reproduce the expected value.
of $\sigma_{xx} = -0.1$ predicted by (36) away from the shock. The same behavior is observed for smaller meshes and smaller time steps.

Figure 14. Computed values of $\sigma_{xx}$ at a time instant after the initial collision with the rigid wall. The value of $\sigma_{xx}$ computed from (36) would be $-0.1$, as reproduced by the algorithm in this paper.
We performed the same computation by replacing the wall by a penalty force of the form

\[
f_{K,\text{penalty}} = -\frac{1}{\epsilon^2} \int_K \min\left(0, \sum_{a \in K} x_a N_a \right) \, dV, \tag{40}
\]

where \(x_a\) is the \(x\)-coordinate of node \(a\) in the set of axes in Fig. 13, and \(\epsilon\) is the penalty parameter. This force is computed through quadrature at the nodes, and is added to the elastic force in (12) every time element \(K\) is updated. The results obtained with the penalty formulation for different values of \(\epsilon\) are shown in Fig. 15. These were computed by adopting the same time factor and same mesh than those used with the method in this paper. As expected, for small values of \(\epsilon\), \(\epsilon = 1, 0.1, 0.01\), neither the stress distribution nor the value of the stress near the wall are anywhere close to what is expected. For \(\epsilon = 0.01, 0.001\) the value of \(\sigma_{xx}\) is close to its expected one. The distribution of stress is also fairly good, except perhaps near the wall, where there is a spread of values that diminishes as the value of the penalty parameter is reduced. This could affect the computation of the contact force between the bodies. The method is unstable for \(\epsilon = 0.0001\) for the time step chosen. Better strategies to impose the contact conditions by penalty approaches, such as [19] might ameliorate or eliminate these features.

6.4. Convergence Properties

A nice feature of this algorithm is that, when the solution of the contact problem is uniquely defined, then the algorithm is expected to converge to it as the time step is made smaller. This is in contrast with penalty approaches, which can only attain convergence by suitably tailoring the penalty parameter with the times step.

As showcased in [16], AVI is expected to have second-order convergence in the values of displacement and velocities at a given time instant. However, because of the relaxed penetration and the non-commutativity of the elemental updates discussed in §5.4, the convergence rate when contact is present can only be expected to be first order. This is in fact what we observed numerically, and show next.

To this end, we considered the two-dimensional problem in §6.3 over a finite element mesh with 469 node and 856 elements. While the exact solution is available for the exact problem, this is not the case for the dynamics of the nodes once the finite element mesh has been introduced. Consequently, we computed a highly refined solution by adopting the same time factor \(f = 0.0001\) for every element in the mesh (there was no TSRZ). We used it as a proxy for the exact solution over the chosen finite element mesh, and hence errors in displacement and velocities were measured against it. In this way, the convergence rate of the algorithm can be numerically examined. The results are shown in Fig. 16. The error is measured by computing the \(L^2(\Omega)\)-norm of the difference between each numerical solution and the proxy for the exact solution at time \(t=1.52\). Here \(\Omega\) is the domain of the square excluding those elements that make contact with the wall, since the velocity oscillates noticeably therein resulting in non-converging error plots for the velocity field. Once these elements are excluded from the computation, the results are very cleanly first order in both displacement and velocities.
7. Numerical Example
A straightforward application of the ideas in this paper is to impact problems between rigid and elastic bodies. As we describe below, it only requires small modifications to the method in §5.4. In particular, for rigid spherical projectiles no additional integration of the body’s rotational motion is needed.

In the following we show a simulation of a rigid spherical projectile impacting a soft polymeric block. Simulating this problem effectively requires extremely small length scales to handle the high-velocity impact, as well as large time scales to deal with the full duration of the penetration and the later ringing of the simulant block. For simplicity we consider the block to be a nonlinear elastic neo-Hookean material, we consider a pre-existing guide hole in the

Figure 15. Dependence of the $\sigma_{xx}$ distribution with the value of the penalty parameter.
Figure 16. Convergence curve as a function of the time factor $f$ for a fixed-mesh. The $L^2$-norm of the difference between each solution and the proxy for the exact one (see text) for both displacement and velocity fields are shown on the top and bottom figure, respectively. In both cases, first order convergence was observed.
admissible region we still have one function $g_a$ for each node $a$ defined as

$$g_a(q) = \begin{cases} -\infty & \text{if } a = a_{sph} \\ R - \|q_a - q_{a_{sph}}\|, & \text{otherwise} \end{cases} \quad (41)$$

where $q_{a_{sph}}$ is the position of the center of mass of the spherical projectile and $R$ its radius, while $\|\mathbf{v}\|$ denotes the length of vector $\mathbf{v}$. Consequently, $g_a$ is not longer a function of only $q_a$, as in §5.3, but also the center of mass of the sphere.

The necessary modifications in Algorithm 2 are better described by simply restating the algorithm, which is done in Algorithm 3. In this case, each time that the contact conditions are satisfied for a given $g_a$, then the momenta of both node $a$ and the center of mass of the sphere are modified. These changes are computed according to (8) and (9). As remarked in §5.4, the presentation of Algorithms 3 was constructed to emphasize simplicity and not efficiency. A number of simple optimizations can be immediately performed at the time of their implementation.

Notice that the contact events and resulting momentum changes for each node of an element are treated sequentially, as we did for the rigid wall. We always deal with the collision of a single node and the sphere, even though there may be more than one node colliding with the sphere simultaneously. As with the rigid wall, this is in fact a selection criterion. A key difference is that here the order in which the collisions of each node with the sphere are computed matters, since different answers are obtained for different orderings of the nodes. Fortunately, the importance of these differences quickly diminishes as the mesh is refined, since the mass of each node becomes very small compared to that of the sphere.

We remark that this observation is not particular to this algorithm, but is intrinsic to the finite dimensional mechanical system obtained after introducing the finite element mesh. Similar issues would appear in the algorithm in [13], or by adopting different penalty parameters for the different interactions between the sphere and the nodes, or by the change of the relative masses of the nodes impacting the sphere as a result of perturbations of the mesh.

In order to simulate this problem at a scale similar to that of the experimental tests, the elastic block needs to be meshed with a relatively large number of elements. This example has 6,222,500 degrees of freedom with 11,866,368 linear tetrahedral elements partitioned onto 256 processors. We therefore ran this case in a message-passing-based parallel environment following the parallel AVI implementation in [22]. The only change is the addition of the contact algorithm, which can be easily done by restricting all elements which might have contact events with the rigid body in a given time interval to a single processor, the contact processor, so that all contact events are dealt with locally, see Fig. 18. This strategy prevents having to update the spherical body dynamics across multiple processors. This partition is changed at regular intervals to account for the changing position of the sphere. More specifically, we included in the contact processors all elements which could be in the TSRZ induced by the sphere as it travels through the guide hole during each stage of the simulation, as well as some rings of elements around them. The size of the partition in the contact processor is determined by considering load balancing issues. Fortunately, since the TSRZ is rather small relative to the mesh size, this strategy works well for a wide range of mesh sizes, although it stops being efficient for a large enough mesh size. Notice that only a few elements relative to the entire mesh have a smaller time factor.
Algorithm 3 Apply Contact Conditions Algorithm (for contact with a rigid sphere)

1: Input: \((K, q, p)\)
2: Output: \(\{p_a\}_{a \in K}\)
3: 
4: for all nodes \(a \in K\) do  
5: \(\text{if } a \neq a_{\text{sph}} \text{ then} \)
6: \(g_a \leftarrow R - \|q_a - q_{a_{\text{sph}}}\|\)
7: \(\text{if } g_a \geq 0 \text{ then} \)
8: \(dg_a \leftarrow -(q_a - q_{a_{\text{sph}}})/\|q_a - q_{a_{\text{sph}}}\|\) \{Compute \(\partial g_a/\partial q_a\} \)
9: \(\nabla g_a^{\text{norm}} \leftarrow \left[ dg_a \cdot (M^{-1} - M^{-1}_{a_{\text{sph}}}) \cdot dg_a \right]^{1/2} \)
10: \(p_n \leftarrow p_a \cdot M^{-1} \cdot dg_a/\nabla g_a^{\text{norm}} + p_{a_{\text{sph}}} \cdot M^{-1}_{a_{\text{sph}}} \cdot (-dg_a)/\nabla g_a^{\text{norm}}\) \{Contact conditions satisfied\}
11: \(\text{if } p_n > 0 \text{ then} \)
12: \(p_a \leftarrow p_n - 2p_n \cdot \nabla g_a^{\text{norm}}\)
13: \(p_{a_{\text{sph}}} \leftarrow p_{a_{\text{sph}}} - 2p_n (-dg_a)/\nabla g_a^{\text{norm}}\)
14: \(\text{end if} \)
15: \(\text{end if} \)
16: \(\text{end if} \)
17: \(\text{end for} \)

To define the TSRZ we chose \(C_{TSRZ} = 5 \times 10^{-4}\) m, so an element has a smaller time step if any of its vertices is at a distance from \(q_{\text{sph}}\) smaller than \(C_{TSZ}\). The time factor within the TSRZ was 0.01, while away from it was 0.05.

Figure 17. Sketch of the numerical example. A block made of a nonlinear elastic material is impacted by a rigid sphere traveling in the horizontal direction. For this example, a guide hole was introduced to circumvent the need to model and compute the way in which the block may fail. Dimensions are in meters.

The simulation takes place in the absence of gravity. The material properties of the block are \(\lambda = 31\) MPa, \(\mu = 3.45\) MPa and \(\rho = 850\) kg/m\(^3\) and the dimensions of the block, the sphere and the guide hole are indicated in Fig. 17. The mass of the sphere is 10g, and the contact with the block is assumed to be frictionless. The block is initially at rest, and the sphere travels
Figure 18. (a) Sketch of a possible partition of a two-dimensional domain to have in the same processor all elements which the sphere may contact during the initial stages of the impact. (b) Sample partition into 16 processors. Each color corresponds to a different processor. The simulations in Figs. 19 and 20 used a partition into 256 processors.

towards the guide hole with a speed 100m/s parallel to it. Some viscosity has been added to eliminate the oscillations in the velocity as a result of the shocks generated in the block. Snapshots of the resulting simulations at different times are shown in Figs. 19 and 20.

8. Final Remarks

The asynchronous contact algorithm introduced enables the reduction of the energy drift at each contact event by adopting smaller time steps in those elements in contact. While it would be desirable to have exact energy conservation, and even better perhaps, a variational integrator for contact which is computationally efficient, this algorithm is very attractive for a wide variety of practical scenarios. This is because in most contact problems a shock wave is generated, and today’s shock capturing methods involve some form of energy dissipation to ameliorate the otherwise present oscillations. In this context, the amount of energy dissipated during each contact event will generally be smaller than that needed to capture the shock, and can be tailored through the time step in the TSRZ otherwise.

9. Acknowledgement

This research was made with government support under and awarded by DoD, Air Force Office of Scientific Research, National Defense Science and Engineering Graduate (NDSEG) Fellowship, 32 CFR 168a.

REFERENCES

Figure 19. Z-axis velocity contours shortly after impact. All three images occur at the same instant, but are snapshots at different distances from the block to illustrate the mesh sizes used and the relative dimensions of the block, the guiding hole, and the sphere. The quarter of the block closest to the camera has been blanked to show the contours through the block.

5. Truesdell C. A simple example of an initial-value problem with any desired number of solutions. *Instituto Lombardo Accademia di Scienze Lettere Rendiconti* 1974; 108:301–304.
Figure 20. Snapshots of the z-axis velocity contours at three different times of the simulation, from an initial projectile velocity of 100 m/s. The last snapshot is when the sphere is effectively at rest.


