ESAIM: M2AN
Vol. 40, N° 5, 2006, pp. 815–841
DOI: 10.1051/m2an:2006035

ESAIM: Mathematical Modelling and Numerical Analysis
www.edpsciences.org/m2an

SYMPLECTIC LOCAL TIME-STEPPING IN NON-DISSIPATIVE DGTD METHODS APPLIED TO WAVE PROPAGATION PROBLEMS

Serge Piperno[1]

Abstract. The Discontinuous Galerkin Time Domain (DGTD) methods are now popular for the solution of wave propagation problems. Able to deal with unstructured, possibly locally-refined meshes, they handle easily complex geometries and remain fully explicit with easy parallelization and extension to high orders of accuracy. Non-dissipative versions exist, where some discrete electromagnetic energy is exactly conserved. However, the stability limit of the methods, related to the smallest elements in the mesh, calls for the construction of local-time stepping algorithms. These schemes have already been developed for N-body mechanical problems and are known as symplectic schemes. They are applied here to DGTD methods on wave propagation problems.

Mathematics Subject Classification. 65L05, 65L20, 65M12, 65M60, 78M10.

Received: July 28, 2005.

1. Introduction

The accurate modeling of systems involving electromagnetic waves, in particular through the resolution of the time-domain Maxwell equations on space grids, remains of strategic interest for many emerging technologies (optical waveguides, furtivity, weapon technologies, etc.). Although the explicit, energy-conserving Finite Difference Time-Domain (FDTD) method proposed by Yee [33] is still prominent, it lacks two important features to be easily applicable in industrial contexts. First, the use of structured or block-structured grids is a huge constraint when a complex geometry is to be analyzed. Second, the accuracy or the efficiency of FDTD methods are limited when fully curvilinear coordinates are used.

Many different types of methods have been proposed in order to handle complex geometries and heterogeneous configurations by dealing with unstructured tetrahedral meshes. One can mention Finite Element Time-Domain (FETD) methods, which have been accelerated using accurate mass lumping [9, 20], mimetic methods [19], or Finite Volume Time-Domain (FVTD) methods [4, 27, 30], which all fail in being at the same time efficient, easily extendible to high orders of accuracy, stable, and energy-conserving. The global conservation of the electromagnetic energy, which is one particular aspect of Yee's original method, is also achieved for FETD methods or for FVTD methods based on totally centered numerical fluxes [27], coupled with a centered implicit time-scheme or an explicit leapfrog time-scheme.

The Discontinuous Galerkin methods enjoy an impressive favor nowadays and are now used in many and various applications [5], taking advantage of their ability to achieve a high order of accuracy by simply choosing

Keywords and phrases. Waves, acoustics, Maxwell's system, Discontinuous Galerkin methods, symplectic schemes, energy conservation, second-order accuracy.

[1] Cermics, project-team caiman, École des Ponts, ParisTech, INRIA, France. serge.piperno@cermics.enpc.fr

suitable basis functions (spectral elements [22], Lagrange high-order polynomials on tetrahedra [12–14]) or to handle complicated geometries and meshes (including locally-refined [2] and non-conformal grids [32]). The existing software are mostly based on upwind fluxes and multi-step low-storage Runge-Kutta time-schemes, which lead to robust and stable, but slightly dissipative Discontinuous Galerkin Time-Domain (DGTD) methods. However, centered fluxes coupled with an explicit leapfrog time-scheme lead to a convergent, stable, and energy-conserving DGTD method [10]. This property, which is important for long-term computations, cannot be exactly obtained with DGTD methods based on upwind fluxes [4, 22, 31], although upwind fluxes lead to more robust codes, particularly for frequency-domain computations [14].

At the same time, the most popular methods for computational electromagnetics, including FDTD, FVTD or DGTD methods, cannot deal very easily with configurations involving small devices or details in the geometry. The use of FDTD methods for these configurations would require fine cartesian grids, quickly becoming unmanageable for small details in three-dimensional problems. Numerical methods based on finite element meshes (FVTD, FETD, DGTD) would be able to handle locally refined unstructured grids, but the time-integration would remain a concern: implicit time-schemes are expensive, while explicit time-schemes have a stability constraint on the time-step directly related to the smallest elements in the mesh.

In this paper, we propose an original strategy to overcome these difficulties in the particular context where an energy conservation property is seeked for. The idea is to introduce locally implicit time-integration (the time scheme is implicit in some parts of the domain and explicit everywhere else) or explicit local time-stepping. Algorithmic solutions have already been proposed in the literature. High-order localized time-integration techniques, in particular based on recently developed additive Runge-Kutta methods [21], are now available [3]. If some additional property is seeked for, the situation is more complex. For example, for the transient solution of hyperbolic conservation laws where several algorithms have been proposed [8, 26], it is not easy to verify a maximum principle and to maintain high order accuracy. In our context, where a non-dissipative spatial discretization is used in order to achieve exact energy conservation, efficient local time-stepping is also difficult to built: a solution for wave propagation with Lagrange multipliers has been proposed [7] and enhanced [1], which leads however to the solution of a fixed, interface-sized linear system at each time iteration. Another totally explicit solution was built, which seems to be only consistent in average and then first-order accurate [25].

The algorithm proposed in this paper is directly inspired from the theory of symplectic integrators developed for the numerical time integration of dynamical Hamiltonian systems. Such methods have been successfully used in the fields of astronomy and molecular dynamics where numerical accuracy and energy conservation are very important over large time integration periods [29]. Using traditional methods, very small time steps are sometimes needed to maintain roughly constant energy throughout a long simulation. On the contrary, the preservation of the symplectic structure is known to lead to improved conservation of energy in long-term simulations. As the Maxwell's equations can be written as an infinite-dimensional Hamiltonian system of PDEs, people are now considering the use of symplectic schemes for the time discretization in time-domain simulations [15, 16, 28].

We consider in this paper the application of some particular symplectic schemes to the finite-dimensional system obtained after space-discretization using a Discontinuous Galerkin method based on totally centered fluxes, with a particular attention to configurations where different scales in the grid are present. In Section 2, we recall the basic features of Discontinuous Galerkin space-discretizations of first-order Maxwell's equations in the time domain, based on totally centered numerical fluxes. In Section 3, we recall some basic results concerning symplectic schemes for Hamiltonian systems, with a particular emphasis on partially-implicit schemes and multi-scale time-schemes. In Section 4, we present two symplectic approaches in the particular context of DGTD methods for Maxwell's equations. The second-order accurate algorithms are presented in full details, and elementary stability properties are proved (energy conservation, boundedness of solutions). Numerical results in two space dimensions are presented in Section 5 and conclusions and further research and development directions are summarized in Section 6.

2. Discontinuous Galerkin method for Maxwell's system

We consider the Maxwell's equations in three space dimensions for heterogeneous anisotropic linear media with no source. The electric permittivity tensor $\bar{\bar{\varepsilon}}(x)$ and the magnetic permeability tensor $\bar{\bar{\mu}}(x)$ are varying in space and both symmetric positive definite (with uniform strictly positive lower and upper bounds). The electric field $\vec{E}$ and the magnetic field $\vec{H}$ verify

$$\bar{\bar{\varepsilon}}\partial_t\vec{E} = \vec{\text{curl}}\,\vec{H}, \quad \bar{\bar{\mu}}\partial_t\vec{H} = -\vec{\text{curl}}\,\vec{E}, \tag{1}$$

where the symbol ∂_t denotes a time derivative. These equations are set and solved on a bounded polyhedral domain Ω of $\mathbb{R}^3$. For the sake of simplicity, a metallic boundary condition is set everywhere on the domain boundary $\partial\Omega$, *i.e.* $\vec{n}\times\vec{E} = \vec{0}$ (where $\vec{n}$) is the unitary outwards normal). We assume we dispose of a partition of a polyhedral domain Ω_h (approximating the regular or Lipschitz-continuous domain of interest Ω) into a finite number of polyhedra (each one having a finite number of faces). For each polyhedral element $\mathcal{T}_i$, V_i denotes its volume, and $\bar{\bar{\varepsilon}}_i$ and $\bar{\bar{\mu}}_i$ are respectively the local electric permittivity and magnetic permeability tensors of the medium, which could be varying inside the element $\mathcal{T}_i$. We call face between two finite elements their intersection, whenever it is a polyhedral surface. We denote by $\mathcal{F}_h$ the union of faces and by $\mathcal{F}_h^{\text{int}} = \mathcal{F}_h/\partial\Omega_h$ the union of internal faces (common to two finite elements). For each internal face $a_{ik} = \mathcal{T}_i \bigcap \mathcal{T}_k$, we denote by S_{ik} the measure of a_{ik} and by $\vec{n}_{ik}$ the unitary normal, oriented from $\mathcal{T}_i$ towards $\mathcal{T}_k$. The same definitions are extended to metallic boundary faces (in the intersection of the domain boundary $\partial\Omega_h$ with a finite element), the index k corresponding to a fictitious element outside the domain. Finally, we denote by $\mathcal{V}_i$ the set of indices of the neighboring elements of the $\mathcal{T}_i$ (having a face in common). We also define the perimeter P_i of $\mathcal{T}_i$ by $P_i = \sum_{k\in\mathcal{V}_i} S_{ik}$. We recall the following geometrical property for all elements: $\sum_{k\in\mathcal{V}_i} S_{ik}\vec{n}_{ik} = 0$.

Following the Discontinuous Galerkin approach, the electric and magnetic fields inside each finite element are seeked for as linear combinations $(\vec{\mathbf{E}}_i, \vec{\mathbf{H}}_i)$ of linearly independent basis vector fields $\vec{\varphi}_{ij},\ 1 \le j \le d_i$, where d_i denotes the local number of scalar degrees of freedom inside $\mathcal{T}_i$. We denote by $\mathcal{P}_i = \text{Span}(\vec{\varphi}_{ij},\ 1 \le j \le d_i)$. The approximate fields $(\vec{\mathbf{E}}_h, \vec{\mathbf{H}}_h)$, defined by $(\forall i,\ \vec{\mathbf{E}}_{h|\mathcal{T}_i} = \vec{\mathbf{E}}_i,\ \vec{\mathbf{H}}_{h|\mathcal{T}_i} = \vec{\mathbf{H}}_i)$ are allowed to be completely discontinuous across element boundaries. Because of this complete discontinuity, a global variational formulation cannot be obtained. However, dot-multiplying (1) by any given vector field $\vec{\varphi} \in \mathcal{P}_i$, integrating over each single element $\mathcal{T}_i$ and integrating by parts, yields

$$\left\{\begin{array}{l} \displaystyle\int_{\mathcal{T}_i} \vec{\varphi}\cdot\bar{\bar{\varepsilon}}_i\partial_t\vec{\mathbf{E}} = -\int_{\partial\mathcal{T}_i}\vec{\varphi}\cdot(\vec{\mathbf{H}}\times\vec{n}) + \int_{\mathcal{T}_i}\vec{\text{curl}}\,\vec{\varphi}\cdot\vec{\mathbf{H}}, \\ \displaystyle\int_{\mathcal{T}_i} \vec{\varphi}\cdot\bar{\bar{\mu}}_i\partial_t\vec{\mathbf{H}} = \int_{\partial\mathcal{T}_i}\vec{\varphi}\cdot(\vec{\mathbf{E}}\times\vec{n}) - \int_{\mathcal{T}_i}\vec{\text{curl}}\,\vec{\varphi}\cdot\vec{\mathbf{E}}. \end{array}\right. \tag{2}$$

In equations (2), we now replace the exact fields $\vec{\mathbf{E}}$ and $\vec{\mathbf{H}}$ by the approximate fields $\vec{\mathbf{E}}_h$ and $\vec{\mathbf{H}}_h$ in order to evaluate volume integrals. For integrals over $\partial\mathcal{T}_i$, some additional approximations have to be done since the approximate fields are discontinuous through element faces. We choose to use completely centered fluxes, *i.e.* $\forall i, \forall k \in \mathcal{V}_i$, $\vec{\mathbf{E}}_{|a_{ik}} \simeq (\vec{\mathbf{E}}_i + \vec{\mathbf{E}}_k)/2$, $\vec{\mathbf{H}}_{|a_{ik}} \simeq (\vec{\mathbf{H}}_i + \vec{\mathbf{H}}_k)/2$. The metallic boundary condition on a boundary face a_{ik} (k in the element index of the fictitious neighboring element) is dealt with *weakly*, in the sense that traces of fictitious fields $\vec{\mathbf{E}}_k$ and $\vec{\mathbf{H}}_k$ are used for the computation of numerical fluxes for the boundary element $\mathcal{T}_i$. In the present case, where all boundaries are metallic, we simply take $\vec{\mathbf{E}}_{k|a_{ik}} = -\vec{\mathbf{E}}_{i|a_{ik}}$ and $\vec{\mathbf{H}}_{k|a_{ik}} = \vec{\mathbf{H}}_{i|a_{ik}}$. Replacing surface integrals using centered fluxes in (2) and re-integrating by parts yields

$$\left\{\begin{array}{l} \displaystyle\int_{\mathcal{T}_i} \vec{\varphi}\cdot\bar{\bar{\varepsilon}}_i\partial_t\vec{\mathbf{E}} = \frac{1}{2}\int_{\mathcal{T}_i}(\vec{\text{curl}}\,\vec{\varphi}\cdot\vec{\mathbf{H}} + \vec{\text{curl}}\,\vec{\mathbf{H}}\cdot\vec{\varphi}) - \frac{1}{2}\sum_{k\in\mathcal{V}_i}\int_{a_{ik}}\vec{\varphi}\cdot(\vec{\mathbf{H}}_k\times\vec{n}_{ik}), \\ \displaystyle\int_{\mathcal{T}_i} \vec{\varphi}\cdot\bar{\bar{\varepsilon}}_i\partial_t\vec{\mathbf{H}} = -\frac{1}{2}\int_{\mathcal{T}_i}(\vec{\text{curl}}\,\vec{\varphi}\cdot\vec{\mathbf{E}} + \vec{\text{curl}}\,\vec{\mathbf{E}}\cdot\vec{\varphi}) + \frac{1}{2}\sum_{k\in\mathcal{V}_i}\int_{a_{ik}}\vec{\varphi}\cdot(\vec{\mathbf{E}}_k\times\vec{n}_{ik}). \end{array}\right. \tag{3}$$

We can rewrite this formulation in terms of scalar unknowns. Inside each element, the fields are recomposed according to $\vec{\mathbf{E}}_i = \sum_{1\le j\le d_i} E_{ij}\ \vec{\varphi}_{ij}$, $\vec{\mathbf{H}}_i = \sum_{1\le j\le d_i} H_{ij}\ \vec{\varphi}_{ij}$. Let us denote by $\mathbf{E}_i$ and $\mathbf{H}_i$ respectively the columns $(E_{il})_{1\le l\le d_i}$ and $(H_{il})_{1\le l\le d_i}$. The equations (3) can be rewritten as:

$$\begin{cases} M_i^{\epsilon}\partial_t\mathbf{E}_i = \quad K_i\mathbf{H}_i - \displaystyle\sum_{k\in\mathcal{V}_i} S_{ik}\mathbf{H}_k, \\ M_i^{\mu}\partial_t\mathbf{H}_i = -K_i\mathbf{E}_i + \displaystyle\sum_{k\in\mathcal{V}_i} S_{ik}\mathbf{E}_k, \end{cases} \tag{4}$$

where the mass matrices M_i^{ϵ}, M_i^{μ}, and the rigidity matrices K_i are symmetric matrices of size d_i (the mass matrices being positive definite), given by

$$\begin{aligned} (M_i^{\epsilon})_{jl} &= \int_{\mathcal{T}_i} {}^t\vec{\varphi}_{ij}\bar{\bar{\epsilon}}_i\vec{\varphi}_{il},\ 1\le j,l\le d_i, \\ (M_i^{\mu})_{jl} &= \int_{\mathcal{T}_i} {}^t\vec{\varphi}_{ij}\bar{\bar{\mu}}_i\vec{\varphi}_{il},\ 1\le j,l\le d_i, \\ (K_i)_{jl} &= \frac{1}{2}\int_{\mathcal{T}_i}\left({}^t\vec{\varphi}_{ij}\,\vec{\mathrm{curl}}\vec{\varphi}_{il} + {}^t\vec{\varphi}_{il}\,\vec{\mathrm{curl}}\vec{\varphi}_{ij}\right), \end{aligned}$$

and for any interface a_{ik}, the $d_i\times d_k$ rectangular matrix S_{ik} is given by

$$1\le j\le d_i,\ 1\le l\le d_k,\ (S_{ik})_{jl} = \frac{1}{2}\int_{a_{ik}} \vec{\varphi}_{ij}\cdot(\vec{\varphi}_{kl}\times\vec{n}_{ik}). \tag{5}$$

Finally, if all electric (resp. magnetic) unknowns are regrouped inside column vectors $\mathbb{E}$ (resp. $\mathbb{H}$) of size $d=\sum_i d_i$, then the space discretized system (4) can be rewritten as

$$\begin{cases} \mathbb{M}^{\epsilon}\partial_t\mathbb{E} = \quad \mathbb{K}\mathbb{H} - \mathbb{A}\mathbb{H} - \mathbb{B}\mathbb{H}, \\ \mathbb{M}^{\mu}\partial_t\mathbb{H} = -\mathbb{K}\mathbb{E} + \mathbb{A}\mathbb{E} - \mathbb{B}\mathbb{E}, \end{cases}$$

where we have the following definitions and properties:

- $\mathbb{M}^{\epsilon}$, $\mathbb{M}^{\mu}$ and $\mathbb{K}$ are $d\times d$ block diagonal matrices with diagonal blocks equal to M_i^{ϵ}, M_i^{μ}, and K_i respectively. Therefore $\mathbb{M}^{\epsilon}$ and $\mathbb{M}^{\mu}$ are symmetric positive definite, and $\mathbb{K}$ is symmetric; one can recall that the matrices M_i^{ϵ} and M_i^{μ} being block diagonal, time integration with an explicit time-scheme leads to an almost completely explicit algorithm;
- $\mathbb{A}$ also is a $d\times d$ block sparse matrix, whose non-zero blocks are equal to S_{ik} when $k\in\mathcal{V}_i$ is not fictitious (a_{ik} then is an internal face of the grid). Since $\vec{n}_{ki} = -\vec{n}_{ik}$, it can be checked from (5) that $(S_{ik})_{jl} = (S_{ki})_{lj}$, and then $S_{ki} = {}^tS_{ik}$; then $\mathbb{A}$ is symmetric;
- $\mathbb{B}$ is a $d\times d$ block diagonal matrix, whose non-zero diagonal blocks are equal to S_{ik} when a_{ik} is an metallic boundary face of the grid. In that case, $(S_{ik})_{jl} = -(S_{ik})_{lj}$, and $S_{ik} = -{}^tS_{ik}$; then $\mathbb{B}$ is skew-symmetric (${}^t\mathbb{B} = -\mathbb{B}$).

One finally obtains that the Maxwell's equations, discretized using discontinuous Galerkin finite-elements with centered fluxes and arbitrary local accuracy and basis functions can be written, in function of the matrix $\mathbb{S} = \mathbb{K} - \mathbb{A} - \mathbb{B}$, in the general form:

$$\begin{cases} \mathbb{M}^{\epsilon}\partial_t\mathbb{E} = \mathbb{S}\mathbb{H}, \\ \mathbb{M}^{\mu}\partial_t\mathbb{H} = -{}^t\mathbb{S}\mathbb{E}, \end{cases} \quad (\mathbb{M}^{\epsilon},\ \mathbb{M}^{\mu}\ \text{symmetric positive definite}). \tag{6}$$

The general form of the system of ordinary differential equations obtained preserves an energy. Indeed, for any solution of (6), the quantity $\mathcal{E} \equiv \frac{1}{2}\left({}^t\mathbb{E}\mathbb{M}^{\epsilon}\mathbb{E} + {}^t\mathbb{H}\mathbb{M}^{\mu}\mathbb{H}\right)$ is exactly conserved.

3. Symplectic schemes for Hamiltonian systems

Symplectic integrators include a variety of different time-discretization schemes designed to preserve the global symplectic structure of the phase space for a Hamiltonian system. These integrators are well established for finite-dimensional Hamiltonian systems (see [24] for several references), but their extension to infinite-dimensional PDEs has not been very extensive. The most part of applications of symplectic schemes have been devoted to N-body mechanical systems. However, the number of applications of symplectic schemes in the context of computational electromagnetics is currently growing [16, 28]. Indeed, the Maxwell's equations can be written as an infinite-dimensional Hamiltonian system of PDEs. In order to fully exploit the properties of symplectic schemes, the most commonly used technique to design "symplectic" numerical methods consists in

(1) discretizing the Maxwell's equations with the numerical method at hand;
(2) considering the finite-dimensional system of ODEs obtained has an input for symplectic methods (another approach uses a symplectic time-integrator of the continuous system [24]).

However, in very few cases only, the discretization of Maxwell's equations actually leads to a Hamiltonian system of ODEs. This is the case for some discretizations like Finite Differences [16] or Finite Elements [28], and time accuracies up to fourth order have been obtained in both cases using symplectic schemes. This is also the case for Discontinuous Galerkin discretizations based on totally centered fluxes, as was proved in the previous section.

In N-body mechanical systems for instance, fixed step-size numerical integration leads to difficulties when particles are very close: the global solution is not far from a singularity (of the Newtonian potential in N-body mechanical systems) and accuracy should be achieved by reducing the time step. A solution to get rid of these small time-steps is provided by the implicit midpoint rule, which can also be used with an adaptive time-steps if it is required. However, in some nonlinear cases, it can be much more expensive than explicit methods, like the leapfrog-Verlet method. In the computational electromagnetics community, the leapfrog time-scheme is widely used and would take the following form for the time-integration of (6):

$$\begin{cases} \mathbb{M}^{\epsilon}\dfrac{\mathbb{E}^{n+1}-\mathbb{E}^{n}}{\Delta t} = \mathbb{S}\mathbb{H}^{n+\frac{1}{2}}, \\ \mathbb{M}^{\mu}\dfrac{\mathbb{H}^{n+\frac{3}{2}}-\mathbb{H}^{n+\frac{1}{2}}}{\Delta t} = -{}^{t}\mathbb{S}\mathbb{E}^{n+1}. \end{cases} \tag{7}$$

Remark. A well-known result concerning the leapfrog scheme is that the following quadratic form $\mathcal{E}^n$ of numerical unknowns $\mathbb{E}^n$ and $\mathbb{H}^{n+\frac{1}{2}}$ is exactly conserved:

$$\mathcal{E}^n = {}^{t}\mathbb{E}^n\mathbb{M}^{\epsilon}\mathbb{E}^n + {}^{t}\mathbb{H}^{n+\frac{1}{2}}\mathbb{M}^{\mu}\mathbb{H}^{n-\frac{1}{2}}.$$

The quadratic form $\mathcal{E}^n$ (it is actually a quadratic form of $\mathbb{E}^n$ and $\mathbb{H}^{n+\frac{1}{2}}$ after having developed $\mathbb{H}^{n-\frac{1}{2}}$ in function of $\mathbb{E}^n$ and $\mathbb{H}^{n+\frac{1}{2}}$!) is positive definite if Δt is small enough (for example $\Delta t \left\| \sqrt{\mathbb{M}^{\mu}}^{-1}\mathbb{S}\sqrt{\mathbb{M}^{\epsilon}}^{-1} \right\| < 2$).

The Verlet method used for N-body systems is exactly equivalent, but would be written in a more obviously reversible way, as:

$$\begin{cases} \mathbb{M}^{\mu}\dfrac{\mathbb{H}^{n+\frac{1}{2}}-\mathbb{H}^{n}}{\Delta t/2} = -{}^{t}\mathbb{S}\mathbb{E}^{n}, \\ \mathbb{M}^{\epsilon}\dfrac{\mathbb{E}^{n+1}-\mathbb{E}^{n}}{\Delta t} = \mathbb{S}\mathbb{H}^{n+\frac{1}{2}}, \\ \mathbb{M}^{\mu}\dfrac{\mathbb{H}^{n+1}-\mathbb{H}^{n+\frac{1}{2}}}{\Delta t/2} = -{}^{t}\mathbb{S}\mathbb{E}^{n+1}. \end{cases} \tag{8}$$

The leapfrog writing leads to an equivalent, cheaper two-step algorithm. The Verlet writing allows for the computations of fields at the same time stations. Moreover, it seems the reversible writing leads to many quite easy enhancements in the scheme. For example, an adaptive Verlet method allows for a stable, energy-conserving, leapfrog-type integration with a varying time-step [18]. Higher-order accurate extensions are available as generalizations of the Verlet method [17] and some of these extensions have already been applied to Maxwell's equations [16, 28]. Fast, multi-scale, regularized integrators are available for Kepler motion or atomic dynamics [23]. Last but not least, two very promising possibilities can be imagined in the context of the present paper, where the local refinement of the unstructured grid could motivate the use of local time-stepping for the resolution of the system of ODEs (6):

- while the energy-preserving coupling of the leapfrog method and on the implicit midpoint-rule remains a non-obvious question, the coupling of the Verlet method with the midpoint rule is quite easy, and will be presented in the next section; it would allow, for example, the time-integration of Maxwell's equations with a locally implicit time-scheme;
- the totally explicit integration of symplectic systems with different time-steps (*i.e.* local time-stepping) is already available [11]. It is globally second-order accurate and symplectic, thus leading to the conservation of some approximate energy. This kind of algorithm (totally explicit, globally second-order accurate, energy preserving, stable) has been seeked for for many years. A simplified version will be presented in the next section too. A very elegant solution with Lagrange multipliers has been proposed [7] and enhanced [1], which leads however to the solution of a fixed linear system at each iteration. Another totally explicit solution was built, which seems to be only consistent in average and then first-order accurate [25].

4. Symplectic schemes designed for locally refined meshes

In this section, we present two particular symplectic algorithms designed for the particular case where a DGTD method is used for the time-domain solution of Maxwell's equations on unstructured meshes where some geometrical details or flaws in the mesh generator lead to locally refined grids. In that case, the classical explicit leapfrog time discretization, which is very efficient and simple, has stability constraints which reduces the possible time-step to an upper bound directly proportional to the smallest edge of the mesh, which can be unmanageable. Two solution algorithms are proposed herein. In the first one, the idea is to use a midpoint rule in a limited number of elements where the stability constraint is too severe. In the second one, the algorithm proposed is fully explicit and local time-stepping is introduced. In both cases, the algorithms obtained are stable and non-dissipative, *i.e.* some discrete electromagnetic energy is exactly conserved.

4.1. **A locally-implicit symplectic scheme**

We first consider a case where the set of elements has been partitioned into two classes: one made of particularly small elements and the other one gathering all other elements. We assume this partition has been done once and for all, before the beginning of the time-domain simulation and is based for example on geometrical and physical criteria. At this stage, there is no need of a particular assumption on the connectivity of the set of "small" or "large" elements. The "small" elements will be handled using an implicit midpoint rule, while all other elements will be time-advanced using a Verlet method.

Using notations inspired from domain decomposition algorithms, we denote with an "*e*" (resp. "*i*") subscript unknowns and matrices related to the explicit (resp. implicit) subdomain. Unknowns are reordered such that explicit elements and unknowns are numbered first, *i.e.*

$$\mathbb{E} = \begin{pmatrix} \mathbb{E}_e \\ \mathbb{E}_i \end{pmatrix}, \ \mathbb{H} = \begin{pmatrix} \mathbb{H}_e \\ \mathbb{H}_i \end{pmatrix},$$

and the block-diagonal matrices $\mathbb{M}^\epsilon$, $\mathbb{M}^\mu$, $\mathbb{K}$ and $\mathbb{B}$ are decomposed as

$$\mathbb{M}^\epsilon = \begin{pmatrix} \mathbb{M}^\epsilon_e & \mathbb{O}_d \\ \mathbb{O}_d & \mathbb{M}^\epsilon_i \end{pmatrix}, \mathbb{M}^\mu = \begin{pmatrix} \mathbb{M}^\mu_e & \mathbb{O}_d \\ \mathbb{O}_d & \mathbb{M}^\mu_i \end{pmatrix}, \mathbb{K} = \begin{pmatrix} \mathbb{K}_e & \mathbb{O}_d \\ \mathbb{O}_d & \mathbb{K}_i \end{pmatrix}, \mathbb{B} = \begin{pmatrix} \mathbb{B}_e & \mathbb{O}_d \\ \mathbb{O}_d & \mathbb{B}_i \end{pmatrix},$$

where $\mathbb{M}^{\epsilon}_{e/i}$ and $\mathbb{M}^{\mu}_{e/i}$ are symmetric positive definite, $\mathbb{K}_{e/i}$ are symmetric, and $\mathbb{B}_e$ are skew-symmetric. The non-block diagonal matrix $\mathbb{A}$, corresponding to interfaces fluxes is decomposed into

$$\mathbb{A} = \begin{pmatrix} \mathbb{A}_{ee} & \mathbb{A}_{ei} \\ \mathbb{A}_{ie} & \mathbb{A}_{ii} \end{pmatrix},$$

where $\mathbb{A}_{ee}$ and $\mathbb{A}_{ii}$ are symmetric and $\mathbb{A}_{ei} = {}^t\mathbb{A}_{ie}$. Finally, defining the two symmetric matrices $S_e = \mathbb{K}_e - \mathbb{A}_{ee} + \mathbb{B}_e$ and $S_i = \mathbb{K}_i - \mathbb{A}_{ii} + \mathbb{B}_i$, the system of ordinary differential equations (6) can be rewritten as

$$\left\{\begin{array}{l} \mathbb{M}^{\epsilon}_e \partial_t \mathbb{E}_e = \mathbb{S}_e \mathbb{H}_e - \mathbb{A}_{ei}\mathbb{H}_i, \\ \mathbb{M}^{\mu}_e \partial_t \mathbb{H}_e = -{}^t\mathbb{S}_e \mathbb{E}_e + \mathbb{A}_{ei}\mathbb{E}_i, \end{array}\right.$$

$$\left\{\begin{array}{l} \mathbb{M}^{\epsilon}_i \partial_t \mathbb{E}_i = \mathbb{S}_i \mathbb{H}_i - \mathbb{A}_{ie}\mathbb{H}_e, \\ \mathbb{M}^{\mu}_i \partial_t \mathbb{H}_i = -{}^t\mathbb{S}_i \mathbb{E}_i + \mathbb{A}_{ie}\mathbb{E}_e. \end{array}\right.$$

We propose the following implicit-explicit algorithm: starting from unknowns at time $t^n = n\Delta t$, we perform the three following sub-steps:

(1) we time-advance of $\Delta t/2$ the explicit domain with a pseudo-forward-Euler scheme;
(2) we time-advance of Δt the implicit domain with the implicit midpoint rule;
(3) we time-advance of $\Delta t/2$ the explicit domain again with the reversed pseudo-forward-Euler scheme.

The whole algorithm reads:

$$\begin{array}{l} \left\{\begin{array}{l} \mathbb{M}^{\mu}_e \dfrac{\mathbb{H}_e^{n+\frac12} - \mathbb{H}_e^n}{\Delta t/2} = -{}^t\mathbb{S}_e \mathbb{E}_e^n + \mathbb{A}_{ei}\mathbb{E}_i^n, \\ \mathbb{M}^{\epsilon}_e \dfrac{\mathbb{E}_e^{n+\frac12} - \mathbb{E}_e^n}{\Delta t/2} = \mathbb{S}_e \mathbb{H}_e^{n+\frac12} - \mathbb{A}_{ei}\mathbb{H}_i^n, \end{array}\right. \\ \left\{\begin{array}{l} \mathbb{M}^{\epsilon}_i \dfrac{\mathbb{E}_i^{n+1} - \mathbb{E}_i^n}{\Delta t} = \mathbb{S}_i \dfrac{\mathbb{H}_i^n + \mathbb{H}_i^{n+1}}{2} - \mathbb{A}_{ie}\mathbb{H}_e^{n+\frac12}, \\ \mathbb{M}^{\mu}_i \dfrac{\mathbb{H}_i^{n+1} - \mathbb{H}_i^n}{\Delta t} = -{}^t\mathbb{S}_i \dfrac{\mathbb{E}_i^n + \mathbb{E}_i^{n+1}}{2} + \mathbb{A}_{ie}\mathbb{E}_e^{n+\frac12}, \end{array}\right. \\ \left\{\begin{array}{l} \mathbb{M}^{\epsilon}_e \dfrac{\mathbb{E}_e^{n+1} - \mathbb{E}_e^{n+\frac12}}{\Delta t/2} = \mathbb{S}_e \mathbb{H}_e^{n+\frac12} - \mathbb{A}_{ei}\mathbb{H}_i^{n+1}, \\ \mathbb{M}^{\mu}_e \dfrac{\mathbb{H}_e^{n+1} - \mathbb{H}_e^{n+\frac12}}{\Delta t/2} = -{}^t\mathbb{S}_e \mathbb{E}_e^{n+1} + \mathbb{A}_{ei}\mathbb{E}_i^{n+1}, \end{array}\right. \end{array} \tag{9}$$

Remark. This algorithm is obviously reversible. It can be sequentially read as made of five operations, the central one corresponding to the implicit midpoint-rule for the "implicit" subdomain. One can verify that, if the two subdomains are disconnected (*i.e.* $\mathbb{A}_{ei} = \mathbb{O}_d$), this algorithm reduces to the juxtaposition of the Verlet-method for the "explicit" subdomain and the midpoint-rule for the "implicit" subdomain.

Stability. The stability of the algorithm (9) can be shown using an energy approach. We have the following result:

Lemma 4.1. *The following quadratic form $\mathcal{E}^n$ of numerical unknowns $\mathbb{E}_e^n$, $\mathbb{E}_i^n$, $\mathbb{H}_e^n$, and $\mathbb{H}_i^n$ is exactly conserved (i.e. $\mathcal{E}^{n+1} = \mathcal{E}^n$) through a time step of algorithm* (9)*:*

$$\mathcal{E}^n = \mathcal{E}_e^n + \mathcal{E}_i^n + \mathcal{E}_c^n \ \textit{with} \ \left\{ \begin{array}{l} \mathcal{E}_e^n = {}^t\mathbb{E}_e^n \mathbb{M}_e^\epsilon \mathbb{E}_e^n + {}^t\mathbb{H}_e^{n+\frac{1}{2}} \mathbb{M}_e^\mu \mathbb{H}_e^{n-\frac{1}{2}}, \\ \mathcal{E}_i^n = {}^t\mathbb{E}_i^n \mathbb{M}_i^\epsilon \mathbb{E}_i^n + {}^t\mathbb{H}_i^n \mathbb{M}_i^\mu \mathbb{H}_i^n, \\ \mathcal{E}_c^n = -\frac{\Delta t^2}{4} {}^t\mathbb{H}_i^n {}^t\mathbb{A}_{ei} \left(\mathbb{M}_e^\epsilon\right)^{-1} \mathbb{A}_{ei} \mathbb{H}_i^n. \end{array} \right. \tag{10}$$

Proof. Simple calculations yield that the variation of the "explicit" energy $\mathcal{E}_e^n$ through the time step is given by:

$$\mathcal{E}_e^{n+1} = \mathcal{E}_e^n + 2\Delta t \left({}^t\mathbb{H}_e^{n+\frac{1}{2}} \mathbb{A}_{ei} \frac{\mathbb{E}_i^n + \mathbb{E}_i^{n+1}}{2} - {}^t\frac{\mathbb{E}_e^n + \mathbb{E}_e^{n+1}}{2} \mathbb{A}_{ei} \frac{\mathbb{H}_i^n + \mathbb{H}_i^{n+1}}{2} \right).$$

Similarly, the variation of the "implicit" energy $\mathcal{E}_i^n$ through the time step is given by:

$$\mathcal{E}_i^{n+1} = \mathcal{E}_i^n + 2\Delta t \left({}^t\frac{\mathbb{H}_i^n + \mathbb{H}_e^{n+1}}{2} \mathbb{A}_{ie} \mathbb{E}_e^{n+\frac{1}{2}} - {}^t\frac{\mathbb{E}_i^n + \mathbb{E}_i^{n+1}}{2} \mathbb{A}_{ie} \mathbb{H}_e^{n+\frac{1}{2}} \right).$$

Recalling that $\mathbb{A}_{ei} = {}^t\mathbb{A}_{ie}$, one gets

$$\mathcal{E}_e^{n+1} + \mathcal{E}_i^{n+1} = \mathcal{E}_e^n + \mathcal{E}_i^n - \Delta t\, {}^t\left(\mathbb{E}_e^n - 2\mathbb{E}_e^{n+\frac{1}{2}} + \mathbb{E}_e^{n+1}\right) \mathbb{A}_{ei} \frac{\mathbb{H}_i^n + \mathbb{H}_i^{n+1}}{2}.$$

Subtracting the second equation of (9) to the fifth one, one gets

$$\mathbb{M}_e^\epsilon \frac{\mathbb{E}_e^{n+1} - 2\mathbb{E}_e^{n+\frac{1}{2}} + \mathbb{E}_e^n}{\Delta t/2} = \mathbb{A}_{ei}\left(\mathbb{H}_i^n - \mathbb{H}_i^{n+1}\right).$$

Reporting this result in the equation above leads to

$$\mathcal{E}_e^{n+1} + \mathcal{E}_i^{n+1} = \mathcal{E}_e^n + \mathcal{E}_i^n + \frac{\Delta t^2}{4} {}^t\left(\mathbb{H}_i^{n+1} - \mathbb{H}_i^n\right) {}^t\mathbb{A}_{ei} \left(\mathbb{M}_e^\epsilon\right)^{-1} \mathbb{A}_{ei} \left(\mathbb{H}_i^n + \mathbb{H}_i^{n+1}\right).$$

The matrix ${}^t\mathbb{A}_{ei}\left(\mathbb{M}_e^\epsilon\right)^{-1}\mathbb{A}_{ei}$ being symmetric, one finally gets

$$\begin{array}{rcl} \mathcal{E}_e^{n+1} + \mathcal{E}_i^{n+1} & = & \mathcal{E}_e^n + \mathcal{E}_i^n + \frac{\Delta t^2}{4} \left({}^t\mathbb{H}_i^{n+1}\, {}^t\mathbb{A}_{ei} \left(\mathbb{M}_e^\epsilon\right)^{-1} \mathbb{A}_{ei} \mathbb{H}_i^{n+1} - {}^t\mathbb{H}_i^n\, {}^t\mathbb{A}_{ei} \left(\mathbb{M}_e^\epsilon\right)^{-1} \mathbb{A}_{ei} \mathbb{H}_i^n \right) \\ & = & \mathcal{E}_e^n + \mathcal{E}_i^n + \mathcal{E}_c^n - \mathcal{E}_c^{n+1}, \end{array}$$

which concludes the proof. □

Remark. One can easily show that the explicit-implicit coupled algorithm (9) is stable for Δt small enough (for Δt small enough, the total energy $\mathcal{E}_e^n$ is a positive definite quadratic form of unknowns). A more closer investigation is required to determine if a sufficient condition on Δt for having a stable coupled scheme is that the explicit Verlet scheme alone is stable.

Remark. It is interesting to notice that $\mathcal{E}_e^n$ and $\mathcal{E}_c^n$ can be recombined into a more compact expression:

$$\mathcal{E}_e^n + \mathcal{E}_c^n = {}^t\left(\mathbb{E}_e^n + \frac{\Delta t}{2}\left(\mathbb{M}_e^\epsilon\right)^{-1}\mathbb{A}_{ei}\mathbb{H}_i^n\right) \mathbb{M}_e^\epsilon \left(\mathbb{E}_e^n - \frac{\Delta t}{2}\left(\mathbb{M}_e^\epsilon\right)^{-1}\mathbb{A}_{ei}\mathbb{H}_i^n\right) + {}^t\mathbb{H}_e^{n+\frac{1}{2}} \mathbb{M}_e^\mu \mathbb{H}_e^{n-\frac{1}{2}}.$$

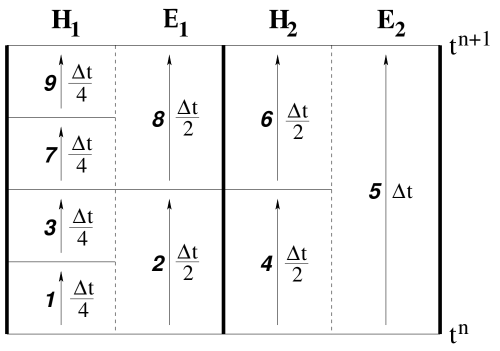

FIGURE 1. Algorithm $R^2(\Delta t)$: the nine sub-steps are detailed from **1** to **9**.

4.2. **A multi-scale fully-explicit symplectic scheme**

The fully explicit algorithm proposed in this section is directly inspired from the one introduced by Hardy *et al.* [11]. In this paper, the authors propose a second-order accurate symplectic integration scheme for N-body problems with multiple time stepping, *i.e.* the atoms or bodies are time-advanced simultaneously with different time steps. In their papers, the authors consider the general case where successive classes of bodies have corresponding time steps being multiple of the next one, the choice of powers of 2 being probably the most efficient in general. We present here a less general version, with time steps given as $\Delta t/2^k$ where Δt is the global time step of the algorithm. To make things clear, we assume that:

(1) the set of elements has been partitioned into N classes;
(2) this partition has been done once and for all, before the beginning of the time-domain simulation and is based for example on geometrical or physical criteria;
(3) the global time step of the algorithm is Δt; for $1 \leq k \leq N$, elements of the class k will be time-advanced using the Verlet method with the local time step $\Delta t/2^{N-k}$; thus the larger elements should lie in class N and the smallest in class 1.

4.2.1. *Recursive definition of the algorithm*

Let us denote by $R^N(\tau)$ the algorithm for advancing in time N classes over the time interval $\tau > 0$. We define $R^N(\tau)$ in a recursive way. We decide that the algorithm $R^1(\tau)$ with only one class is exactly the Verlet method (8) with $\Delta t = \tau$. For any $N \geq 1$, if $R^N(\tau)$ is well defined, we define $R^{N+1}(\tau)$ by:

(1) start with all unknowns at time $t^n = n\Delta t$;
(2) advance all elements with class $k \leq N$ with $R^N(\Delta t/2)$; if required, use values at time t^n for unknowns in elements of class $N+1$;
(3) advance all elements with class $k = N+1$ with the Verlet method (*i.e.* $R^1(\Delta t)$); if required, use values at time $t^n + \Delta t/2$ for unknowns in elements of class $k \leq N$;
(4) advance all elements with class $k \leq N$ with $R^N(\Delta t/2)$; if required, use values at time t^{n+1} for unknowns in elements of class $N+1$;
(5) all unknowns at time $t^{n+1} = t^n + \Delta t$ have been computed.

Remark. The reader can check that this algorithm does not require any additional storage and remains completely explicit. It is reversible, symplectic, second-order accurate and conserves an energy [11]. We again refer to [11] where the authors have also proposed accelerations for the computation of forces or fluxes in nonlinear N-body mechanical systems.

4.2.2. *The algorithm $R^2(\Delta t)$*

Let us consider the case where $N = 2$ (we recall the case $N = 1$ is exactly the Verlet method (8)). As in the previous section, the subscripts $k \in \{1, 2\}$ denote the class of the elements and replace the subscripts e and i. Elements are reordered and matrices are substructured as in the explicit-implicit coupling case. The algorithm $R^2(\Delta t)$ is described in Figure 1.

It can be thoroughly developed as follows:

$$\left\{\begin{array}{ll}
\textbf{Step 1.} & \mathbb{M}_1^\mu \dfrac{\mathbb{H}_1^{n+\frac14} - \mathbb{H}_1^n}{\Delta t/4} = -{}^t\mathbb{S}_1\mathbb{E}_1^n + \mathbb{A}_{12}\mathbb{E}_2^n, \\
\textbf{Step 2.} & \mathbb{M}_1^\epsilon \dfrac{\mathbb{E}_1^{n+\frac12} - \mathbb{E}_1^n}{\Delta t/2} = \mathbb{S}_1\mathbb{H}_1^{n+\frac14} - \mathbb{A}_{12}\mathbb{H}_2^n, \\
\textbf{Step 3.} & \mathbb{M}_1^\mu \dfrac{\mathbb{H}_1^{n+\frac12} - \mathbb{H}_1^{n+\frac14}}{\Delta t/4} = -{}^t\mathbb{S}_1\mathbb{E}_1^{n+\frac12} + \mathbb{A}_{12}\mathbb{E}_2^n, \\
\textbf{Step 4.} & \mathbb{M}_2^\mu \dfrac{\mathbb{H}_2^{n+\frac12} - \mathbb{H}_2^n}{\Delta t/2} = -{}^t\mathbb{S}_2\mathbb{E}_2^n + \mathbb{A}_{21}\mathbb{E}_1^{n+\frac12}, \\
\textbf{Step 5.} & \mathbb{M}_2^\epsilon \dfrac{\mathbb{E}_2^{n+1} - \mathbb{E}_2^n}{\Delta t} = \mathbb{S}_2\mathbb{H}_2^{n+\frac12} - \mathbb{A}_{21}\mathbb{H}_1^{n+\frac12}, \\
\textbf{Step 6.} & \mathbb{M}_2^\mu \dfrac{\mathbb{H}_2^{n+1} - \mathbb{H}_2^{n+\frac12}}{\Delta t/2} = -{}^t\mathbb{S}_2\mathbb{E}_2^{n+1} + \mathbb{A}_{21}\mathbb{E}_1^{n+\frac12}, \\
\textbf{Step 7.} & \mathbb{M}_1^\mu \dfrac{\mathbb{H}_1^{n+\frac34} - \mathbb{H}_1^{n+\frac12}}{\Delta t/4} = -{}^t\mathbb{S}_1\mathbb{E}_1^{n+\frac12} + \mathbb{A}_{12}\mathbb{E}_2^{n+1}, \\
\textbf{Step 8.} & \mathbb{M}_1^\epsilon \dfrac{\mathbb{E}_1^{n+1} - \mathbb{E}_1^{n+\frac12}}{\Delta t/2} = \mathbb{S}_1\mathbb{H}_1^{n+\frac34} - \mathbb{A}_{12}\mathbb{H}_2^{n+1}, \\
\textbf{Step 9.} & \mathbb{M}_1^\mu \dfrac{\mathbb{H}_1^{n+1} - \mathbb{H}_1^{n+\frac34}}{\Delta t/4} = -{}^t\mathbb{S}_1\mathbb{E}_1^{n+1} + \mathbb{A}_{12}\mathbb{E}_2^{n+1}.
\end{array}\right. \tag{11}$$

Energy conservation and stability. The stability of the algorithm (11) can be shown using the theory of symplectic schemes. Hence, it does not yield in general an explicit expression of the energy which is conserved. Such an expression can be obtained using a not so classical energy approach. However, the computations are tedious and the generalizations to more complex versions $R^N(\Delta t)$ with $N > 2$ seems a difficult task. We begin with the following lemma on a sub-scaled reversible scheme built on implicit midpoint rules:

Lemma 4.2. *Consider the following midpoint-rule-based sub-scaled scheme:*

$$\left\{\begin{array}{rcl}
M_X X^{n+\frac12} &=& M_X X^n + \frac{\Delta t}{2}\left(A_X \frac{X^n + X^{n+\frac12}}{2} + BY^n\right), \\
M_Y Y^{n+1} &=& M_Y Y^n + \Delta t\left(A_Y \frac{Y^n + Y^{n+1}}{2} - {}^tBX^{n+\frac12}\right), \\
M_X X^{n+1} &=& M_X X^{n+\frac12} + \frac{\Delta t}{2}\left(A_X \frac{X^{n+\frac12} + X^{n+1}}{2} + BY^{n+1}\right),
\end{array}\right.$$

where X and Y denote vectors in $\mathbb{R}^d$ (the sizes could be different) and M_X, M_Y, A_X, A_Y, and B are $d \times d$ square matrices with the additional assumptions that M_X and M_Y are symmetric positive definite matrices, and ${}^tA_X = -A_X$, ${}^tA_Y = -A_Y$. Then, *(i) the following symmetric quadratic form is exactly conserved:*

$$\mathcal{E}^n = {}^t\begin{pmatrix} X^n \\ Y^n \end{pmatrix} \begin{pmatrix} M_X - \frac{\Delta t^2}{16} {}^t A_X M_X^{-1} A_X & \frac{\Delta t^2}{8} A_X M_X^{-1} B \\ \frac{\Delta t^2}{8} {}^t B M_X^{-1t} A_X & M_Y - \frac{\Delta t^2}{4} {}^t B M_X^{-1} B \end{pmatrix} \begin{pmatrix} X^n \\ Y^n \end{pmatrix}.$$

(ii) for Δt small enough, this quadratic form is positive definite, therefore the scheme is stable (solutions $(X^n, Y^n)_{n\in\mathbb{N}}$ are bounded).

(iii) a sufficient stability condition on Δt is that

$$\rho_{max}\begin{pmatrix} {}^t A_X M_X^{-1} A_X & -2 A_X M_X^{-1} B \\ -2 {}^t B M_X^{-1t} A_X & 4 {}^t B M_X^{-1} B \end{pmatrix} \Delta t^2 \ < \ 16 \ \min\left(\rho_{min}(M_X), \rho_{min}(M_Y)\right),$$

where $\rho_{min}(M)$ (resp. $\rho_{max}(M)$) denotes the smallest (resp. largest) eigenvalue of a real symmetric matrix M.

Proof. Let us introduce the following simplifying notations: $\bar{X} \equiv (X^n + X^{n+1})/2$, $\bar{Y} \equiv (Y^n + Y^{n+1})/2$, $\delta X \equiv X^{n+1} - X^n$, $\bar{Y} \equiv (Y^n + Y^{n+1})/2$. Simple calculations based on the equations describing the scheme yield:

$$\begin{aligned} M_X \delta X &= \Delta t A_X \bar{X} + \Delta t B \bar{Y} - \frac{\Delta t^2}{16} A_X M_X^{-1} A_X \delta X - \frac{\Delta t^2}{8} A_X M_X^{-1} B \delta Y, \\ M_Y \delta Y &= \Delta t A_Y \bar{Y} - \Delta t {}^t B \bar{X} + \frac{\Delta t^2}{8} {}^t B M_X^{-1} A_X \delta X + \frac{\Delta t^2}{4} {}^t B M_X^{-1} B \delta Y. \end{aligned}$$

This can be rewritten into

$$\begin{pmatrix} M_X - \frac{\Delta t^2}{16} {}^t A_X M_X^{-1} A_X & \frac{\Delta t^2}{8} A_X M_X^{-1} B \\ \frac{\Delta t^2}{8} {}^t B M_X^{-1t} A_X & M_Y - \frac{\Delta t^2}{4} {}^t B M_X^{-1} B \end{pmatrix} \begin{pmatrix} \delta X \\ \delta Y \end{pmatrix} = \Delta t \begin{pmatrix} A_X & B \\ -{}^t B & A_Y \end{pmatrix} \begin{pmatrix} \bar{X} \\ \bar{Y} \end{pmatrix}.$$

The matrix in the left hand side being symmetric and the one in the right and side being skew-symmetric, the proof of (i) follows simply. For Δt small, the energy matrix is close to the block-diagonal matrix (M_X, M_Y) which is positive definite, then for Δt small enough, the proposed quadratic form is also positive definite. More precisely, the condition given in (iii) is a sufficient condition for that. Then the energy provides a norm, and for this norm the solutions $(X^n, Y^n)_{n\in\mathbb{N}}$ are bounded, which ends the proof. □

We now propose some manipulations of the nine different steps in algorithm R^2 given in (11). They are summed up in the next lemma.

Lemma 4.3. *The three groups of three steps of algorithm R^2 given in* (11) *can be rewritten in the following form:*

$$\begin{aligned}
&\textbf{Steps 1.2.3.} \iff \mathbb{M}_1^{[\frac{\Delta t}{2}]} \mathbb{F}_1^{n+\frac12} = \mathbb{M}_1^{[\frac{\Delta t}{2}]} \mathbb{F}_1^n + \frac{\Delta t}{2}\ \mathbb{P}_1 \frac{\mathbb{F}_1^n + \mathbb{F}_1^{n+\frac12}}{2} + \frac{\Delta t}{2}\ \mathbb{Q}_1 \mathbb{F}_2^n, \\
&\textbf{Steps 4.5.6.} \iff \mathbb{M}_2^{[\Delta t]} \mathbb{F}_2^{n+1} = \mathbb{M}_2^{[\Delta t]} \mathbb{F}_2^n + \Delta t\ \mathbb{P}_2 \frac{\mathbb{F}_2^n + \mathbb{F}_2^{n+1}}{2} + \Delta t\ \mathbb{Q}_2 \mathbb{F}_1^{n+\frac12}, \\
&\textbf{Steps 7.8.9.} \iff \mathbb{M}_1^{[\frac{\Delta t}{2}]} \mathbb{F}_1^{n+1} = \mathbb{M}_1^{[\frac{\Delta t}{2}]} \mathbb{F}_1^{n+\frac12} + \frac{\Delta t}{2}\ \mathbb{P}_1 \frac{\mathbb{F}_1^{n+\frac12} + \mathbb{F}_1^{n+1}}{2} + \frac{\Delta t}{2}\ \mathbb{Q}_1 \mathbb{F}_2^{n+1},
\end{aligned}$$

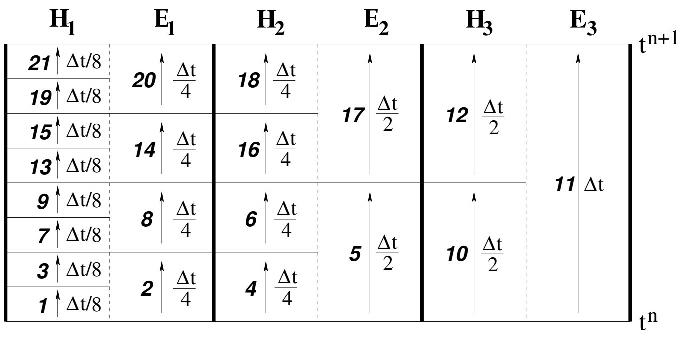

FIGURE 2. Algorithm $R^3(\Delta t)$: the twenty-one sub-steps are detailed from **1** to **21**.

where we have used the following notations: $\forall\theta \in \{1,2\}$, $\forall\tau$,

$$\mathbb{P}_\theta = \begin{pmatrix} 0 & S_\theta \\ -{}^tS_\theta & 0 \end{pmatrix}, \; \mathbb{F}^n_\theta = \begin{pmatrix} \mathbb{E}^n_\theta \\ \mathbb{H}^n_\theta \end{pmatrix}, \; \mathbb{M}^{[\tau]}_\theta = \begin{pmatrix} \mathbb{M}^\epsilon_\theta - \frac{\tau^2}{4} S_\theta \left(\mathbb{M}^\mu_\theta\right)^{-1} {}^tS_\theta & 0 \\ 0 & \mathbb{M}^\mu_\theta \end{pmatrix},$$

$$\mathbb{Q}_1 = \begin{pmatrix} 0 & -A_{12} \\ A_{12} & 0 \end{pmatrix}, \quad \mathbb{Q}_2 = \begin{pmatrix} 0 & -A_{21} \\ A_{21} & 0 \end{pmatrix}.$$

Proof. The proof is pure calculation. For the sake of simplicity, we give here the main stages corresponding to steps 4.5.6. Starting from the expressions in (11) for steps 4.5.6, the idea is to get rid of $\mathbb{H}_2^{n+\frac{1}{2}}$ by adding and subtracting the two equations corresponding to steps 4 and 6. We have:

$$\begin{aligned} \mathbb{M}^\mu_2 \left(\mathbb{H}^{n+1}_2 - \mathbb{H}^n_2\right) &= -\Delta t \; {}^t\mathbb{S}_2 \frac{\mathbb{E}^n_2 + \mathbb{E}^{n+1}_2}{2} + \Delta t \; \mathbb{A}_{21}\mathbb{E}_1^{n+\frac{1}{2}}, \\ \mathbb{H}_2^{n+\frac{1}{2}} &= \frac{\mathbb{H}^n_2 + \mathbb{H}^{n+1}_2}{2} + \frac{\Delta t}{4} \left(\mathbb{M}^\mu_2\right)^{-1} \left({}^t\mathbb{S}_2\mathbb{E}^{n+1}_2 + \mathbb{E}^n_2\right), \end{aligned} \tag{12}$$

and, using this result inside the equation corresponding to step 5, we get

$$\mathbb{M}^\epsilon_2 \frac{\mathbb{E}^{n+1}_2 - \mathbb{E}^n_2}{\Delta t} = \mathbb{S}_2 \frac{\mathbb{H}^n_2 + \mathbb{H}^{n+1}_2}{2} + \frac{\Delta t}{4} \mathbb{S}_2 \left(\mathbb{M}^\mu_2\right)^{-1} \left({}^t\mathbb{S}_2\mathbb{E}^{n+1}_2 + \mathbb{E}^n_2\right) - \mathbb{A}_{21}\mathbb{H}_1^{n+\frac{1}{2}}. \tag{13}$$

Finally, grouping (12) and (13) leads to the result of the lemma for steps 4.5.6. □

Remark. We just proved that the algorithm R^2 given in (11) can be seen as a particular occurrence of the the midpoint-rule-based sub-scaled scheme of Lemma 4.2, with the following values:

$$\left\{ \begin{array}{l} X^n = \mathbb{F}^n_1, \\ Y^n = \mathbb{F}^n_2, \end{array} \right. \quad \left\{ \begin{array}{l} A_X = \mathbb{P}_1, \\ A_Y = \mathbb{P}_2, \end{array} \right. \quad \left\{ \begin{array}{l} M_X = \mathbb{M}_1^{[\frac{\Delta t}{2}]}, \\ M_Y = \mathbb{M}_2^{[\Delta t]}, \end{array} \right. \quad B = \mathbb{Q}_1,$$

and we have verified that M_X, M_Y are symmetric positive definite matrices (for Δt small enough), A_X and A_Y are skew-symmetric and $\mathbb{Q}_2 = -{}^t\mathbb{Q}_1$. This leads to the following energy conservation and stability theorem:

Theorem 4.4. *The algorithm* R^2 *given in* (11) *conserves an energy and is stable (the solutions computed for given initial values are bounded) if* Δt *is small enough.*

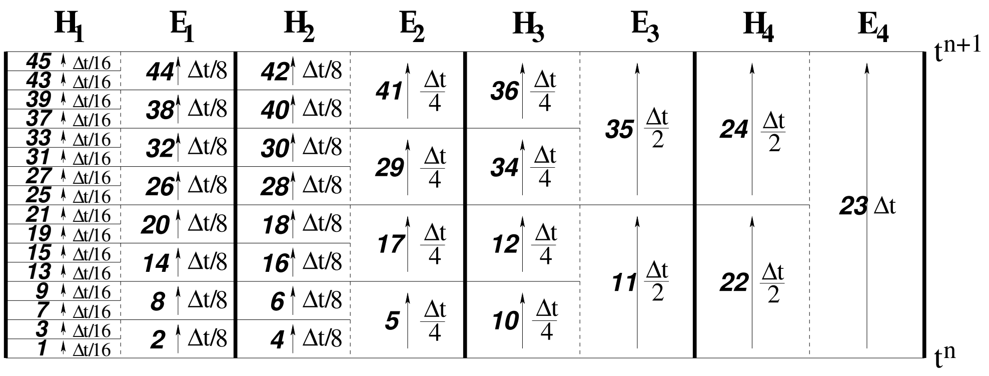

FIGURE 3. Algorithm $R^4(\Delta t)$: the forty-five sub-steps are detailed from **1** to **45**.

Proof. The proof is explained in the preceding remark. For Δt small enough, more precisely, if $\mathbb{M}_1^{[\frac{\Delta t}{2}]}$ and $\mathbb{M}_2^{[\Delta t]}$ are positive definite, and if the condition on Δt given in Lemma 4.2 is satisfied for the above values for the matrices, then the algorithm conserves an energy which is a positive definite quadratic norm of the unknowns. The set of conditions on Δt writes

$$\begin{cases} \mathbb{M}_1^\epsilon - \dfrac{\Delta t^2}{16}\mathbb{S}_1\left(\mathbb{M}_1^\mu\right)^{-1}{}^t\mathbb{S}_1 \text{ is positive definite} \\ \mathbb{M}_2^\epsilon - \dfrac{\Delta t^2}{4}\mathbb{S}_2\left(\mathbb{M}_2^\mu\right)^{-1}{}^t\mathbb{S}_2 \text{ is positive definite} \\ \rho_{max}\begin{pmatrix} {}^tA_X M_X^{-1} A_X & -2A_X M_X^{-1} B \\ -2{}^tB M_X^{-1t} A_X & 4{}^tB M_X^{-1} B \end{pmatrix} \Delta t^2 \; < \; 16\ \min\left(\rho_{min}(M_X), \rho_{min}(M_Y)\right), \end{cases}$$

which is verified for Δt small enough. □

4.2.3. *The algorithms $R^3(\Delta t)$ and $R^4(\Delta t)$*

We consider the case where $N = 3$ (resp. $N = 4$). Again, the subscripts $k \in \{1,2,3\}$ (resp. $k \in \{1,2,3,4\}$) denote the class of the elements, elements are reordered and matrices are substructured as previously. The algorithm $R^3(\Delta t)$ and $R^4(\Delta t)$ are described in Figures 2 and 3 respectively.

5. Numerical results

The locally implicit algorithm of Section 4.1 has not been implemented yet. This section is devoted to numerical results obtained with the local time-stepping algorithm of Section 4.2.

We consider here the homogeneous Maxwell equations in two space dimensions and in the TE case. The unknown fields are E_x, E_y, and H_z and satisfy the following equations and reflecting boundary condition:

$$\begin{cases} \partial_t E_x = \partial_y H_z, \\ \partial_t E_y = -\partial_x H_z, \\ \partial_t H_z = \partial_y E_x - \partial_x E_y, \end{cases} \qquad \text{with } E_x n_y - E_y n_x = 0 \text{ on } \Gamma.$$

These equations are equivalent to the acoustics equations in homogeneous medium, where the unknowns are the pressure perturbation p and the velocity perturbation (horizontal u and vertical v components), $p \leftrightarrow H_z$, $u \leftrightarrow E_y$, $v \leftrightarrow -E_x$, and the perfectly reflecting condition of electromagnetic waves corresponds to a perfect slip boundary condition for acoustics $\vec{u}\cdot\vec{m} = 0$.

TABLE 1. Characteristics of mesh generated on the unit square for the accuracy study.

Mesh	# Vertices	# Elements	h_{min}	h_{max}
M4	23	32	0.214	0.336
M3	81	132	0.094	0.188
M2	279	500	0.048	0.097
M1	1038	1962	0.023	0.052

The first part of this section is devoted to the numerical investigation of the impact of local time-stepping on the accuracy. The following sections report numerical results obtained on toy problems where detailed structures are involved.

5.1. **Impact of local time-stepping on the accuracy**

In this section, we investigate the impact of local-time stepping on the global accuracy of the numerical solution. Let us recall the known results on the accuracy of the DGTD method used in this paper. Let is introduce the largest element diameter h in the mesh. Using the method of lines (*i.e.* before time discretization and after Discontinuous Galerkin space discretization), the semi-discretized method based on $\mathbb{P}_k$ polynomials inside all elements yields an error in h^k if totally centered fluxes are used [10] and in h^{k+1} if upwind fluxes are used [13]. Although upwind fluxes lead to slightly dissipative but more robust schemes, it is well known they do not cope well with leapfrog-type time schemes (the use of RKDG-scheme is a possible way to get high accuracy in both time and space [6], but the global stability of algorithms using locally adapted time steps is still to be proved). At the same time, a study on the accuracy of Discontinuous Galerkin methods with variable spatial discretization (for instance a polynomial degree k being spatially variable) could be conducted. The aim of this paper being to concentrate on local time-stepping, we intend to assess the accuracy of the time-discretization algorithm, assuming the "fixed-k" $\mathbb{P}_k$ Discontinuous Galerkin spatial discretization is used.

We consider a totally reflecting unit square cavity for which acoustic eigenmodes are given by

$$\begin{cases} p = \omega \ \cos(\pi n_x x)\cos(\pi n_y y)\cos(\omega t), \\ u = \pi n_x \ \sin(\pi n_x x)\cos(\pi n_y y)\sin(\omega t), \\ v = \pi n_y \ \cos(\pi n_x x)\sin(\pi n_y y)\sin(\omega t), \end{cases}$$

with $\omega^2 = \pi^2(n_x^2 + n_y^2)$. We consider the $(1,1)$-mode (*i.e.* $n_x = n_y = 1$). For all computations, the initial approximate solution is obtained via element-wise L^2-projection on $\mathbb{P}_k$ polynomials over each element, a sufficiently accurate quadrature formula being used (we have chosen a quadrature exact for polynomials of degree up to 15). At the end of each computation (final time $T = 1$), we compute the L^2-norm of the difference between the discontinuous approximate solution and the element-wise L^2-projection on $\mathbb{P}_k$ of the exact solution (the difference between this projection and the very smooth exact solution is in $O(h^{k+1})$).

In order to discriminate space and time discretization errors, we have chosen to use unstructured quasi-uniform grids. We have built four unstructured but regular grids M4, M3, M2, and M1 (each one being twice finer than the previous one). The characteristics of these meshes are given in Table 1.

We first verified the convergence of the $\mathbb{P}_k$-DGTD method based on totally centered fluxes and the Verlet time-integration scheme. We obtained the errors (in L^2-norm) reported in Table 2. The time step Δt used is derived from the Courant number $\nu \equiv \Delta t/h_{min}$, and the $\mathbb{P}_k$-DGTD is numerically proved to be stable for $\nu \leq \nu_k$ (with $\nu_0 \simeq 0.768$, $\nu_1 \simeq 0.252$, $\nu_2 \simeq 0.133$, $\nu_3 \simeq 0.085$, $\nu_4 \simeq 0.0584$, etc.). For all computations, the Courant number was chosen as a fraction of $\tilde{\nu_k} = 0.9\ \nu_k$. In a general setting, the L^2-norm of the error should be bound by $e \leq c_k h_{max}^k + K_t \Delta t^2$, where c_k and K_t should be mesh- and time-step-independent constants. In our context where h_{max}/h_{min} is bound, we get $e \leq c_k h^k + c_t h^2 \nu^2$ (where we have simply used $h = h_{max}$ and

TABLE 2. L^2-norm of the error at time $T = 1$, using the Verlet method.

M4	$k = 1$	$k = 2$	$k = 3$	$k = 4$
$\nu = \tilde{\nu_k}/1$	4.71×10^{-2}	4.64×10^{-3}	5.24×10^{-5}	5.66×10^{-5}
$\nu = \tilde{\nu_k}/2$	4.69×10^{-2}	4.64×10^{-3}	5.21×10^{-5}	4.06×10^{-5}
$\nu = \tilde{\nu_k}/4$	4.69×10^{-2}	4.63×10^{-3}	5.20×10^{-5}	4.00×10^{-5}
$\nu = \tilde{\nu_k}/8$	4.69×10^{-2}	4.63×10^{-3}	5.21×10^{-5}	3.99×10^{-5}
$\nu = \tilde{\nu_k}/16$	4.69×10^{-2}	4.64×10^{-3}	5.21×10^{-5}	3.99×10^{-5}
$\nu = \tilde{\nu_k}/32$	4.69×10^{-2}	4.64×10^{-3}	5.20×10^{-5}	3.99×10^{-5}
$\nu = \tilde{\nu_k}/64$	4.69×10^{-2}	4.64×10^{-3}	5.20×10^{-5}	3.99×10^{-5}

M3	$k = 1$	$k = 2$	$k = 3$	$k = 4$
$\nu = \tilde{\nu_k}/1$	2.30×10^{-2}	1.11×10^{-3}	7.78×10^{-5}	1.04×10^{-5}
$\nu = \tilde{\nu_k}/2$	2.30×10^{-2}	1.11×10^{-3}	7.62×10^{-5}	3.36×10^{-6}
$\nu = \tilde{\nu_k}/4$	2.30×10^{-2}	1.11×10^{-3}	7.62×10^{-5}	2.32×10^{-6}
$\nu = \tilde{\nu_k}/8$	2.30×10^{-2}	1.11×10^{-3}	7.62×10^{-5}	2.27×10^{-6}
$\nu = \tilde{\nu_k}/16$	2.30×10^{-2}	1.11×10^{-3}	7.62×10^{-5}	2.27×10^{-6}
$\nu = \tilde{\nu_k}/32$	2.30×10^{-2}	1.11×10^{-3}	7.62×10^{-5}	2.27×10^{-6}
$\nu = \tilde{\nu_k}/64$	2.30×10^{-2}	1.11×10^{-3}	7.62×10^{-5}	2.27×10^{-6}

M2	$k = 1$	$k = 2$	$k = 3$	$k = 4$
$\nu = \tilde{\nu_k}/1$	1.15×10^{-2}	2.42×10^{-4}	1.12×10^{-5}	2.96×10^{-6}
$\nu = \tilde{\nu_k}/2$	1.15×10^{-2}	2.42×10^{-4}	9.71×10^{-6}	7.50×10^{-7}
$\nu = \tilde{\nu_k}/4$	1.15×10^{-2}	2.42×10^{-4}	9.65×10^{-6}	2.24×10^{-7}
$\nu = \tilde{\nu_k}/8$	1.15×10^{-2}	2.42×10^{-4}	9.65×10^{-6}	1.35×10^{-7}
$\nu = \tilde{\nu_k}/16$	1.16×10^{-2}	2.42×10^{-4}	9.65×10^{-6}	1.31×10^{-7}
$\nu = \tilde{\nu_k}/32$	1.16×10^{-2}	2.42×10^{-4}	9.65×10^{-6}	1.31×10^{-7}
$\nu = \tilde{\nu_k}/64$	1.16×10^{-2}	2.42×10^{-4}	9.65×10^{-6}	1.31×10^{-7}

M1	$k = 1$	$k = 2$	$k = 3$	$k = 4$
$\nu = \tilde{\nu_k}/1$	5.76×10^{-3}	4.90×10^{-5}	1.71×10^{-6}	6.01×10^{-7}
$\nu = \tilde{\nu_k}/2$	5.76×10^{-3}	4.90×10^{-5}	1.21×10^{-6}	1.50×10^{-7}
$\nu = \tilde{\nu_k}/4$	5.76×10^{-3}	4.90×10^{-5}	1.19×10^{-6}	3.81×10^{-8}
$\nu = \tilde{\nu_k}/8$	5.76×10^{-3}	4.90×10^{-5}	1.19×10^{-6}	1.13×10^{-8}
$\nu = \tilde{\nu_k}/16$	5.76×10^{-3}	4.90×10^{-5}	1.19×10^{-6}	6.81×10^{-9}
$\nu = \tilde{\nu_k}/32$	5.76×10^{-3}	4.90×10^{-5}	1.19×10^{-6}	6.61×10^{-9}
$\nu = \tilde{\nu_k}/64$	5.76×10^{-3}	4.90×10^{-5}	1.19×10^{-6}	6.59×10^{-9}

with c_t is also a mesh- and time-step-independent constant). In view of the results in Table 2, many remarks can be made:

- for $k = 1$, the spatial errors prevail ($\forall\ \nu \leq 1$) and the convergence according to $e \sim c_1 h$ is confirmed;
- for $k = 2$, the spatial and time errors have the same order but the space errors seem (in this particular case) dominant. The convergence according to $e \sim O(h^2)$ is also confirmed;

TABLE 3. Study of the time accuracy of Verlet method (based on the L^2-norm of the error at time $T = 1$); for a given mesh, the error behaves like $O(\Delta t^2)$.

$k = 4$	M1	M2	M3
$e(\tilde{\nu_k}/1) - e(\nu \to 0)$	5.95×10^{-7}	2.83×10^{-6}	8.13×10^{-7}
$e(\tilde{\nu_k}/2) - e(\nu \to 0)$	1.44×10^{-7}	6.19×10^{-7}	1.09×10^{-7}
$e(\tilde{\nu_k}/4) - e(\nu \to 0)$	3.15×10^{-8}	9.30×10^{-8}	5.00×10^{-8}

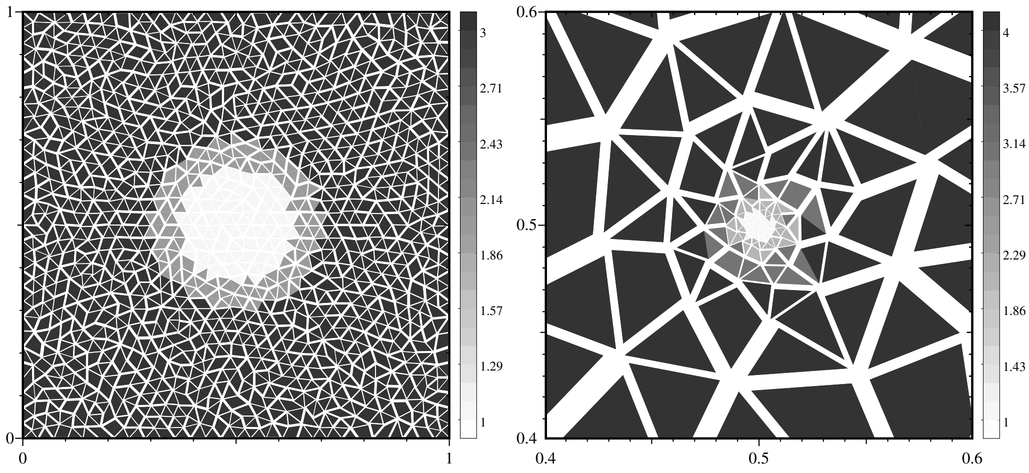

FIGURE 4. Time-step classes on the regular mesh (left) and the irregular mesh (right) of the unit square.

- for $k = 3$ and $k = 4$, the observed limit for e when $\nu \to 0$ is close to the expected $O(h^k)$;
- for $k > 2$, the time errors prevail when $\nu \sim \nu_k$. The value of $e(\nu) - e(\nu \to 0)$ actually behaves like $O(\nu^2)$, *i.e.* $O(\Delta t^2)$ (it appears more clearly for $k = 4$). We have given in Table 3 the successive values for $e(\nu) - e(\nu \to 0)$ for $k = 4$ and different ν.

In a second series of computations, we have used the $R^3(\Delta t)$ algorithm on the same meshes in order to compare easily the errors obtained to those obtained with the classical $R^1(\Delta t)$ Verlet method. In order to actually use different classes of elements, we have artificially reduced the admissible time step in given zones: the global time step used Δt is the same as for the corresponding computation using the Verlet method, but the time step is $\Delta t/2$ if the distance to the center is less than 0.1333 and $\Delta t/4$ if it is smaller than 0.2. The time step classes are shown on the regular mesh of the square in Figure 4 (left).

The L^2-norms of the error at time $T = 1$ are given in Table 4 for the meshes M1 and M2. The results are similar to those obtained with the standard Verlet method. The algorithm is second-order accurate in time: the errors due to time discretization are masked for $k = 1$, and the variation of these errors is clearly in $O(\Delta t^2)$, with a larger constant than for the Verlet method though.

It is interesting to note that, in general, the error obtained with the $R^3(\Delta t)$ algorithm is larger than the one obtained with the Verlet method with Δt. This means that the local time-stepping should not be used as a way to obtain a better accuracy (although more computations are performed than with the Verlet method for the same macro time step Δt). It should rather be seen as a way to accelerate the computation. This conclusion should also be moderated by the fact that the local-time stepping zone is quite large in the test cases considered.

Finally, we have made additional computations showing the accuracy obtained with local time-stepping in a typical case where the mesh is locally – and only locally – refined. We have used the mesh of the square shown on the right part of Figure 4. We have given in Table 5 the L^2-norm of the error at time $T = 10$ for

TABLE 4. L^2-norm of the error at time $T = 1$, using the $R^3(\Delta t)$ algorithm.

M2	$k = 1$	$k = 2$	$k = 3$	$k = 4$
$\nu = \tilde{\nu_k}/1$	1.16×10^{-2}	2.56×10^{-4}	6.55×10^{-5}	4.80×10^{-5}
$\nu = \tilde{\nu_k}/2$	1.16×10^{-2}	2.43×10^{-4}	1.87×10^{-5}	1.19×10^{-5}
$\nu = \tilde{\nu_k}/4$	1.15×10^{-2}	2.42×10^{-4}	1.04×10^{-5}	2.98×10^{-6}
$\nu = \tilde{\nu_k}/8$	1.15×10^{-2}	2.42×10^{-4}	9.71×10^{-6}	7.54×10^{-7}
$\nu = \tilde{\nu_k}/16$	1.16×10^{-2}	2.42×10^{-4}	9.65×10^{-6}	2.24×10^{-7}
$\nu = \tilde{\nu_k}/32$	1.16×10^{-2}	2.42×10^{-4}	9.65×10^{-6}	1.38×10^{-7}
$\nu = \tilde{\nu_k}/64$	1.16×10^{-2}	2.42×10^{-4}	9.65×10^{-6}	1.31×10^{-7}

M1	$k = 1$	$k = 2$	$k = 3$	$k = 4$
$\nu = \tilde{\nu_k}/1$	5.76×10^{-3}	6.33×10^{-5}	3.18×10^{-5}	2.34×10^{-5}
$\nu = \tilde{\nu_k}/2$	5.76×10^{-3}	5.01×10^{-5}	7.98×10^{-6}	5.81×10^{-6}
$\nu = \tilde{\nu_k}/4$	5.76×10^{-3}	4.91×10^{-5}	2.29×10^{-6}	1.45×10^{-6}
$\nu = \tilde{\nu_k}/8$	5.76×10^{-3}	4.90×10^{-5}	1.28×10^{-6}	3.63×10^{-7}
$\nu = \tilde{\nu_k}/16$	5.76×10^{-3}	4.90×10^{-5}	1.20×10^{-6}	9.08×10^{-8}
$\nu = \tilde{\nu_k}/32$	5.76×10^{-3}	4.90×10^{-5}	1.19×10^{-6}	2.35×10^{-8}
$\nu = \tilde{\nu_k}/64$	5.76×10^{-3}	4.90×10^{-5}	1.19×10^{-6}	8.63×10^{-9}

TABLE 5. L^2-norm of the error at time $T = 10$ on a locally refined mesh.

Verlet method	Δt	error	cpu
$k = 1$	3.21×10^{-4}	1.62×10^{-2}	32
$k = 2$	1.70×10^{-4}	2.42×10^{-4}	131
$k = 3$	1.08×10^{-4}	9.44×10^{-6}	428
$k = 4$	7.45×10^{-5}	1.35×10^{-7}	1210

$R^4(\Delta t)$	Δt	error	cpu
$k = 1$	2.57×10^{-3}	1.63×10^{-2}	10
$k = 2$	1.36×10^{-3}	2.43×10^{-4}	35
$k = 3$	8.67×10^{-4}	1.19×10^{-5}	104
$k = 4$	5.96×10^{-4}	3.90×10^{-6}	284

computations using $\mathbb{P}_k$ polynomials with $1 \leq k \leq 4$, with $\nu = \tilde{\nu_k}$ and with both the Verlet method and the $R^4(\Delta t)$ algorithm.

For $k < 4$, the influence of local time-stepping on the global error is not important. However, it is perceptible for $k = 4$. A way to improve these results should consist in using local time stepping along with more than second-order accurate symplectic time-schemes. A related problem of constructing more than second-order accurate local time-stepping schemes remains open.

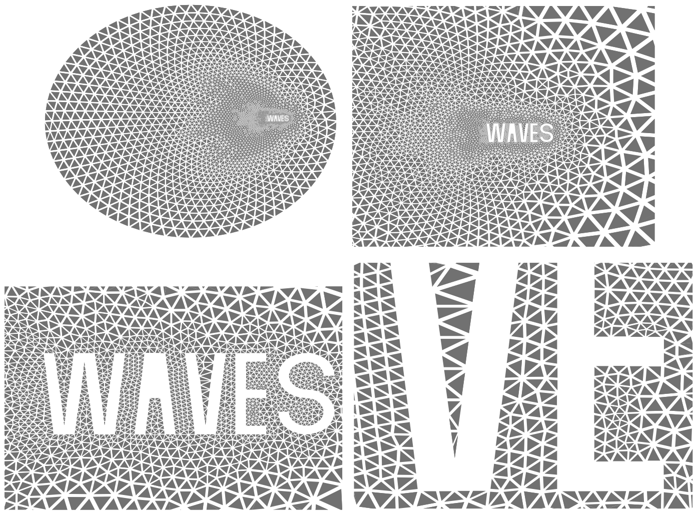

FIGURE 5. Unstructured triangular mesh for the "WAVES" toy problem.

TABLE 6. Comparison of CPU times and gain between algorithms R^7, R^1, and a classical leapfrog implementation (7).

Algorithm	R^7(3.54 ms)	R^1(6.06 μs)	leapfrog (7, 6.06 μs)
CPU time	2412	11 820	7880
Gain (*vs.* leapfrog)	3.68	0.67	1

5.2. **Test-case with a typical unstructured mesh**

We have imagined a first toy problem where the propagation of waves in an homogeneous medium is confined in a completely reflecting cavity. In order to have different scales in the geometry, the cavity has be designed the following way:

- the cavity is an ellipse (2 m $\times$ 1.6 m);
- inside the cavity, a small geometrical detail is located on the right focus (the detail is the word "WAVES"); the characteristic size for the whole detail is nearly 0.1 m, with small elements like the thickness of the letters smaller than 0.01 m; the boundary of the detail is also perfectly reflecting;
- the initial condition is a p/H_z pulse (for the acoustic equations) located at the other focus, such that the solution should refocus exactly on the other focus and scatter on the detail.

We have generated an unstructured mesh using a commercial mesh generator (with no indication on the sizes of elements, except for the domain boundaries which were meshed according to the local geometrical characteristic length). The mesh obtained contains 3883 vertices and 7258 elements. Successive zooms of the mesh are shown in Figure 5.

One can see that the size of the elements in the mesh obtained is varying quite smoothly. One can also notice that the mesh generator has produced triangular elements with bad aspect ratio, which will lead locally to small admissible time steps.

We report here the computed results of two simulations up to time $T = 8$ s with the $\mathbb{P}_4$-DGTD method introduced in Section 2 (the fields are described with polynomials of degree at most 4 inside elements). For both computations, some local admissible time step Δt_i is computed inside each element $\mathcal{T}_i$. It is directly proportional to the smallest height of the element. Then,

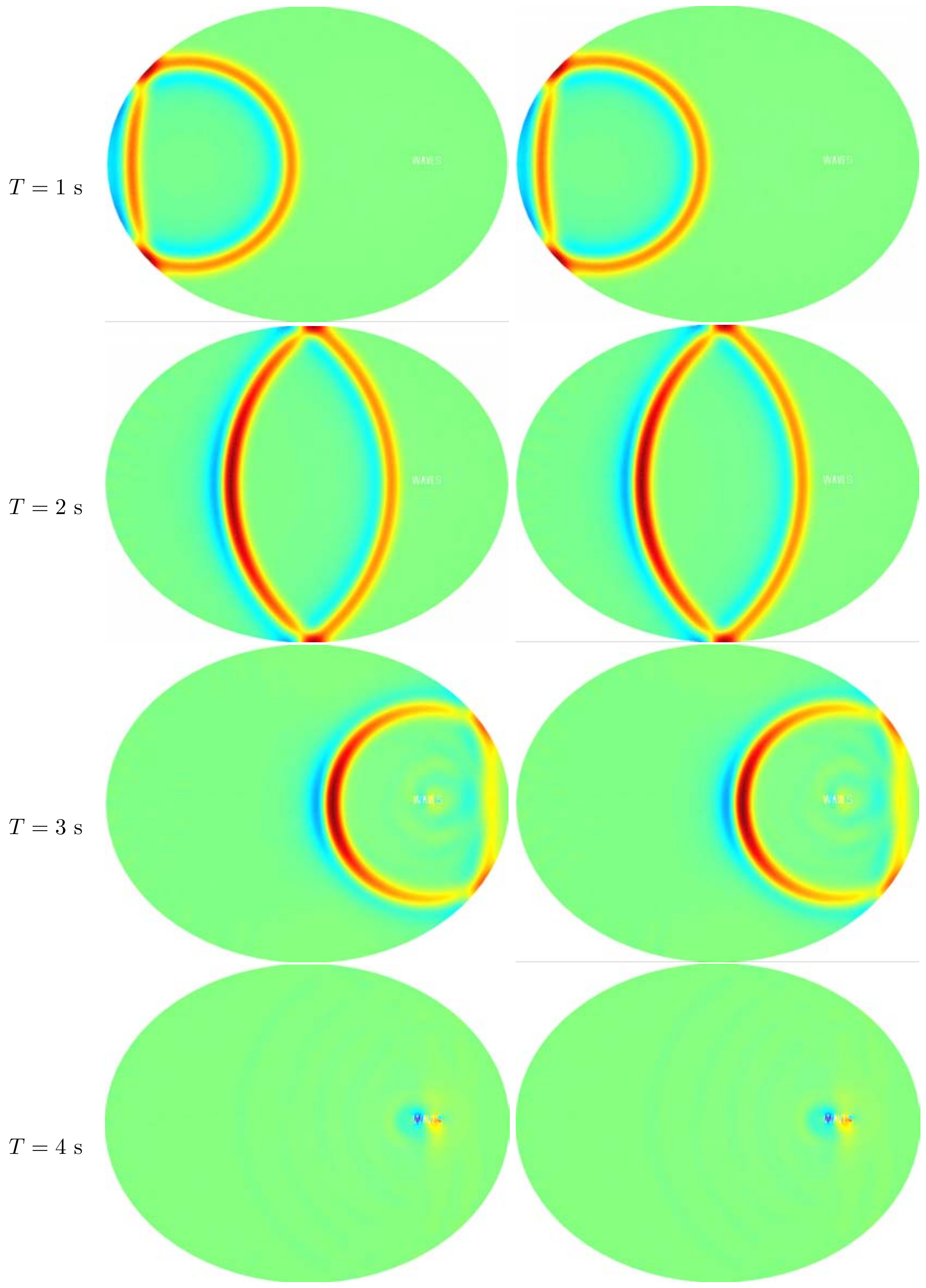

FIGURE 6. WAVES-cavity: algorithm $R^7(3.54$ ms) (left) *vs.* algorithm $R^1(6.06\ \mu$s) (right).

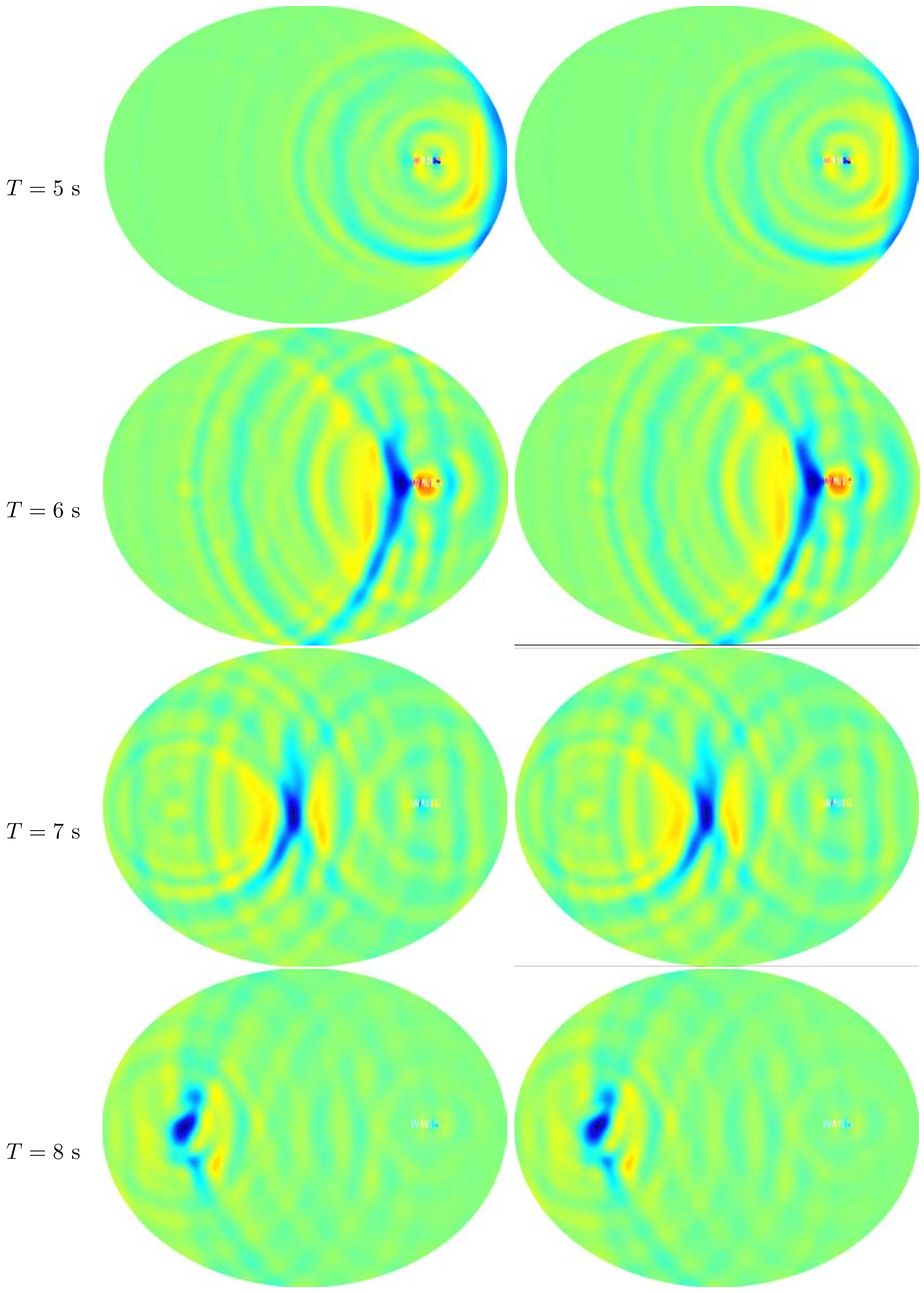

FIGURE 7. WAVES-cavity: algorithm R^7(3.54 ms) (left) *vs.* algorithm R^1(6.06 μs) (right).

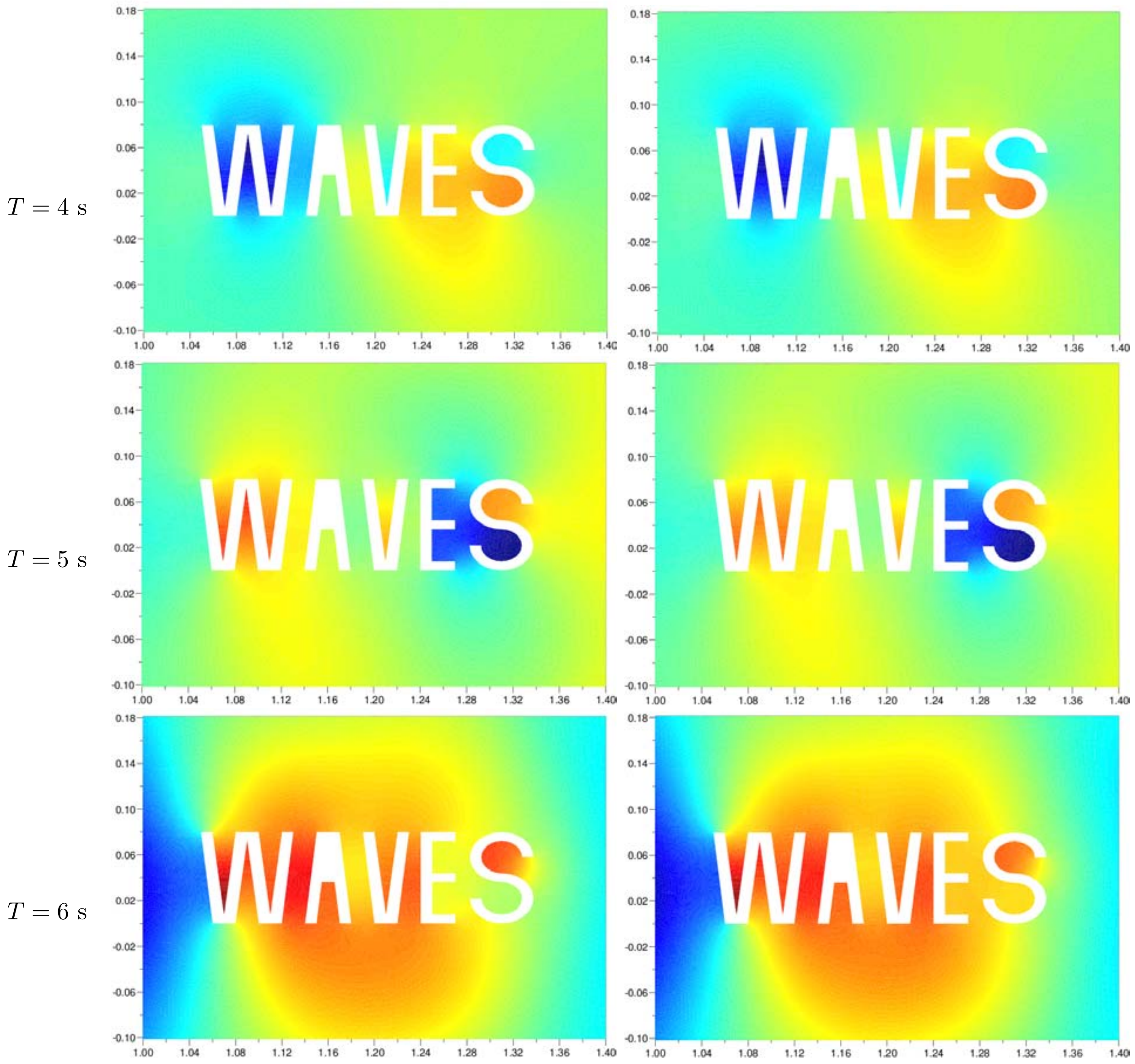

FIGURE 8. WAVES-cavity (zoom): algorithm R^7(3.54 ms) (left) *vs.* algorithm R^1(6.06 μs) (right).

- in a "reference" computation, we have used the algorithm $R^1(\Delta t)$ with $\Delta t \equiv \min_i(\Delta t_i)$;
- in a "multi-scale" computation we have used the algorithm $R^7(\Delta t)$ with $\Delta t \equiv \max_i(\Delta t_i)/1.999$: the number of 7 classes was reached simply because, in this particular mesh, $2^6 \min_i(\Delta t_i) < \max_i(\Delta t_i) < 2^7 \min_i(\Delta t_i)$. For each element, the class c_i of the element was set such that $2^{c_i-7} < \Delta t_i/\Delta t < 2^{c_i-6}$ (thus the element with the largest Δt_i – *i.e.* $2\Delta t > \Delta t_i > \Delta t$ – is of class $c_i = 7$). The reader must realize that the time step actually used in the smallest elements (with $c_i = 1$) is $\Delta t/64$.

Results for the p/H_z field obtained with both simulations are shown at integer times in Figures 6 and 7. Zooms on the WAVES-detail (p/H_z field) obtained with both simulations are shown in Figure 8. Contours of u/E_y and $v/-E_x$ components are shown near the WAVES-detail in Figure 9. Singularities near corners are partially obtained. Finally, we must compare the CPU times obtained for both computations. The CPU times (obtained

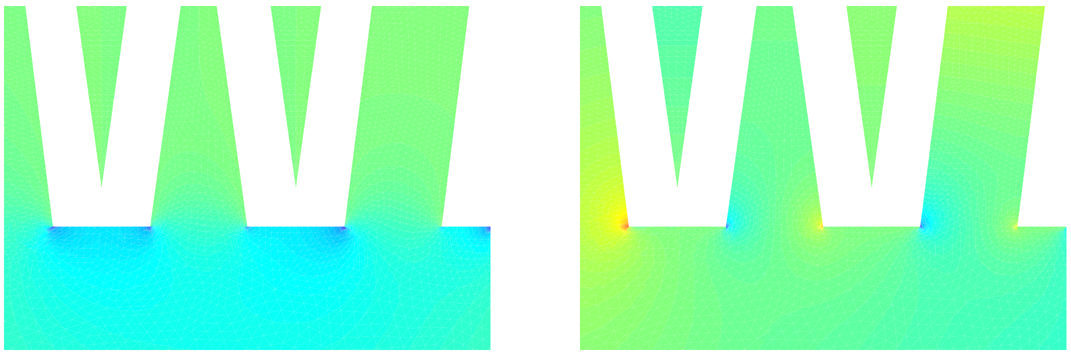

FIGURE 9. WAVES-cavity (zoom): u/E_y (left) and $v/-E_x$ (right) near the WAVES-detail, obtained with algorithm $R^7(3.54 \text{ ms})$.

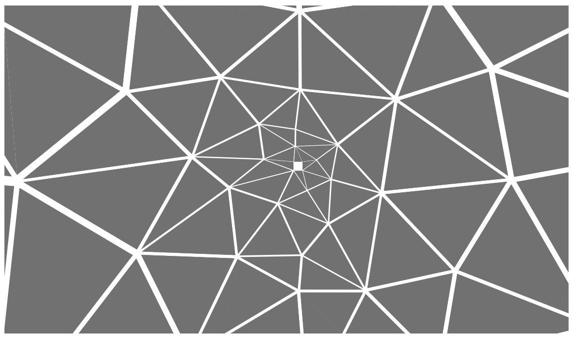

FIGURE 10. Unstructured triangular mesh for the square inclusion problem.

TABLE 7. Comparison of CPU times and gain between algorithms R^{10}, R^1, and a classical leapfrog implementation (7).

Algorithm	$R^{10}(2.6 \text{ ms})$	$R^1(6.16\ \mu\text{s})$	leapfrog (7, 6.16 μs)
CPU time	527	29 040	19 360
Gain (*vs.* leapfrog)	36.7	0.67	1

on a linux PC with 3.4 Ghz Pentium IV processor) are given in Table 6. We have added a third column with the CPU time which should have been obtained with a classical leapfrog implementation (7) instead of (8), *i.e.* only two-thirds of the CPU time of algorithm R^1. For this particular case, the computational time is reduced by a factor 3.68. This reduction is strongly related to the distribution of element sizes over the mesh. In the present case, the automatically generated mesh has a smoothly varying element-size, and the gain obtained is quite typical (it is reasonable to think that this gain is not far from a lower bound). The aim of the second computation is to show cases where the gain might be much larger.

Table 8. Comparison of CPU times and gain between algorithms R^{11}, R^1, and a classical leapfrog implementation (7).

Algorithm	R^{11}(2.6 ms)	R^1(3.54 μs)	leapfrog (7, 3.54 μs)
CPU time	7958	58 212	38 808
Gain (*vs.* leapfrog)	4.88	0.67	1

5.3. **Test-case with a strongly refined unstructured mesh**

We consider a second test-case where the mesh has been designed to be strongly refined in a limited area. We have considered the same elliptic domain as previously, but the reflecting inclusion is now a square of size 0.2 mm centered at the right focus of the ellipse (the mesh is conforming and refined inside a square of size 0.01 m also centered at the focus). The mesh obtained contains 1017 vertices and 1958 elements. The mesh partitioning leads to ten classes of elements, *i.e.* the smallest elements are time-advanced 512 times more often than the largest elements. A zoom of the mesh near the square is shown in Figure 10. Contours of the fields obtained with the algorithm R^{10}(2.6 ms) are shown in Figure 11.

Since the mesh is quite coarse, we have used in this section the $\mathbb{P}_5$-DGTD (the fields are described with polynomials of degree at most 5 inside elements). The CPU times obtained with the different time schemes considered are given in Table 7.

For this particular case, the computational time is reduced by a factor near 37! This reduction is due to the fact that 45% of elements are time-advanced only every global time-step, 43% twice more often, and only 10% four times often or more. This gain is not surprising, since the existence of small elements is not a concern for the multi-scale algorithm while it leads to a proportionally smaller time step for the classical algorithm, *i.e.* a conversely growing computational time.

5.4. **Test-case with a detailed structure**

We consider a last test-case with a more complex geometry. The elliptic domain encloses again a reflecting inclusion centered at the right focus of the ellipse. The device is a circular array of 0.2 mm square, set at a distance equal to 1 mm from the focus. The mesh obtained contains 1176 vertices and 2254 elements. The mesh partitioning leads to eleven classes of elements, *i.e.* the smallest elements are time-advanced 1024 times more often than the largest elements. A zoom of the mesh near the square is shown in Figure 12. Contours of the fields obtained with the algorithm R^{11}(2.6 ms) are shown in Figure 13.

We have also used in this section the $\mathbb{P}_5$-DGTD (the fields are described with polynomials of degree at most 5 inside elements). The CPU times obtained with the different time schemes considered are given in Table 8.

For this particular case, the computational time is reduced by a factor near 5. This smaller reduction is due to the fact that 80% of elements are time-advanced at most 4 times per global time-step, but 11% of elements are time-advanced at most 512 times per global time-step (the refined zone of the mesh is more significant).

6. Conclusion

In this paper, we have presented two symplectic algorithms which are able to perform a reversible, energy-conserving, second-order accurate, stable, and adaptive time-integration of the Maxwell's equations after discretization on unstructured meshes using the Discontinuous Galerkin method. The main conclusion is that, if totally centered numerical fluxes are to be used, in order to have no numerical dissipation at all, local time-stepping can overcome the stability limit set by the leapfrog time-scheme.

This kind of algorithm can be particularly valuable if the mesh is distorted or locally refined, *i.e.* the mesh is refined in a very limited area, for example around a geometrical detail. Two ways have been proposed in this paper. The first one relies on a simple implicit/explicit coupled algorithm. It has not been implemented but is the most promising for configurations where the unstructured mesh at hand has very small elements and

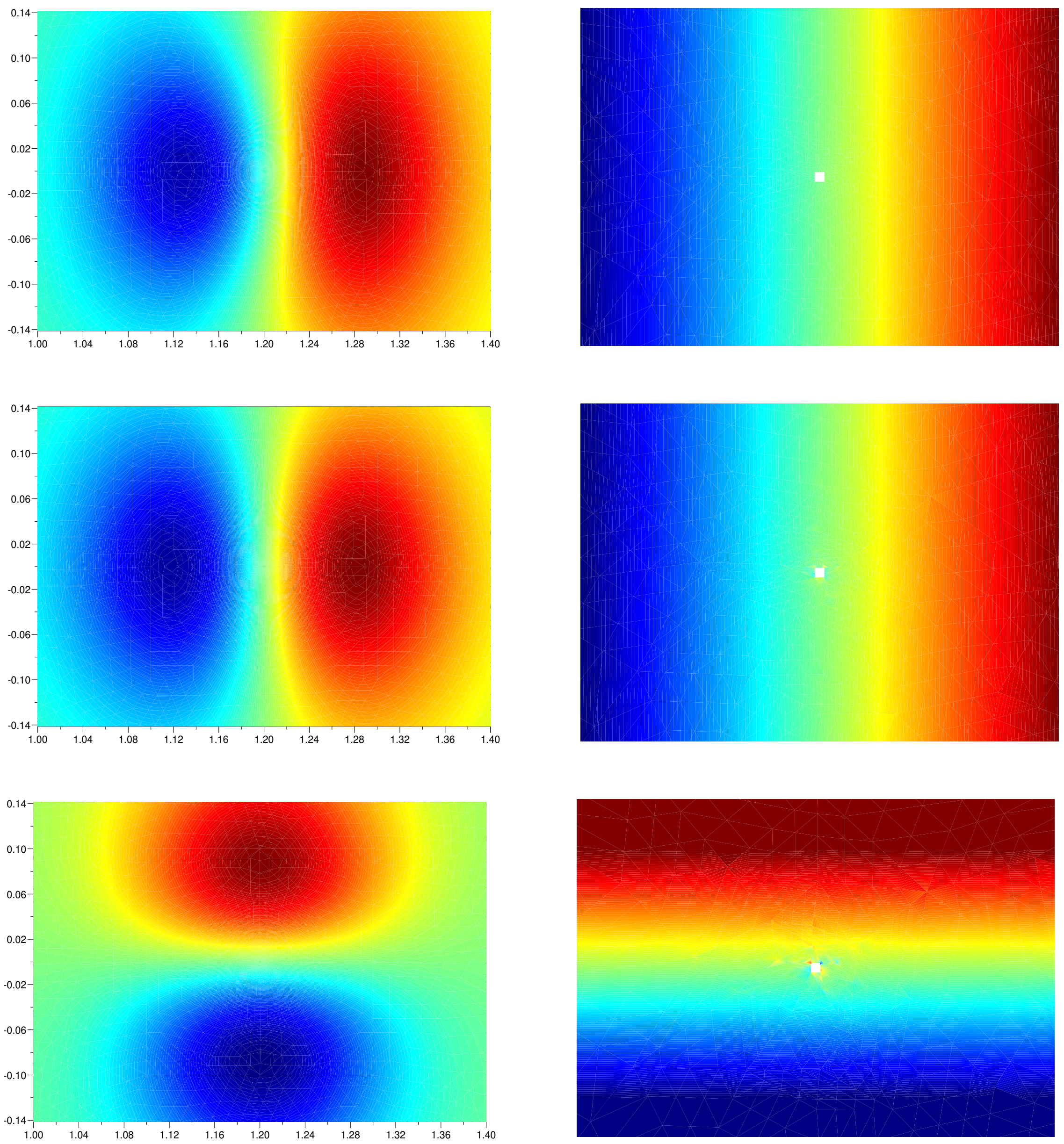

FIGURE 11. Square inclusion: p/H_z (top), u/E_y (middle), and $v/-E_x$ (bottom) near the inclusion, obtained with algorithm R^{10}(2.6 ms) at $t = 4$ s (extremal values for contours on the right have been adapted).

is difficult to restore. Another totally explicit algorithm, with no additional storage, has been proposed, and leads to very efficient implementations, at least in two space dimensions.

Further works will deal with the implementation of the locally implicit scheme, and with implementations in three space dimensions, the latter being quite straightforward because the algorithms can be seen as time-step

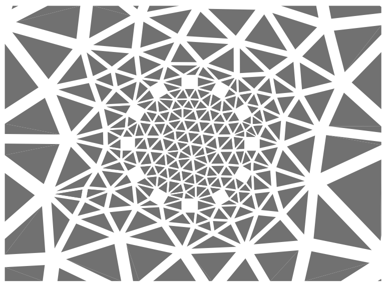

FIGURE 12. Unstructured triangular mesh near the circular array.

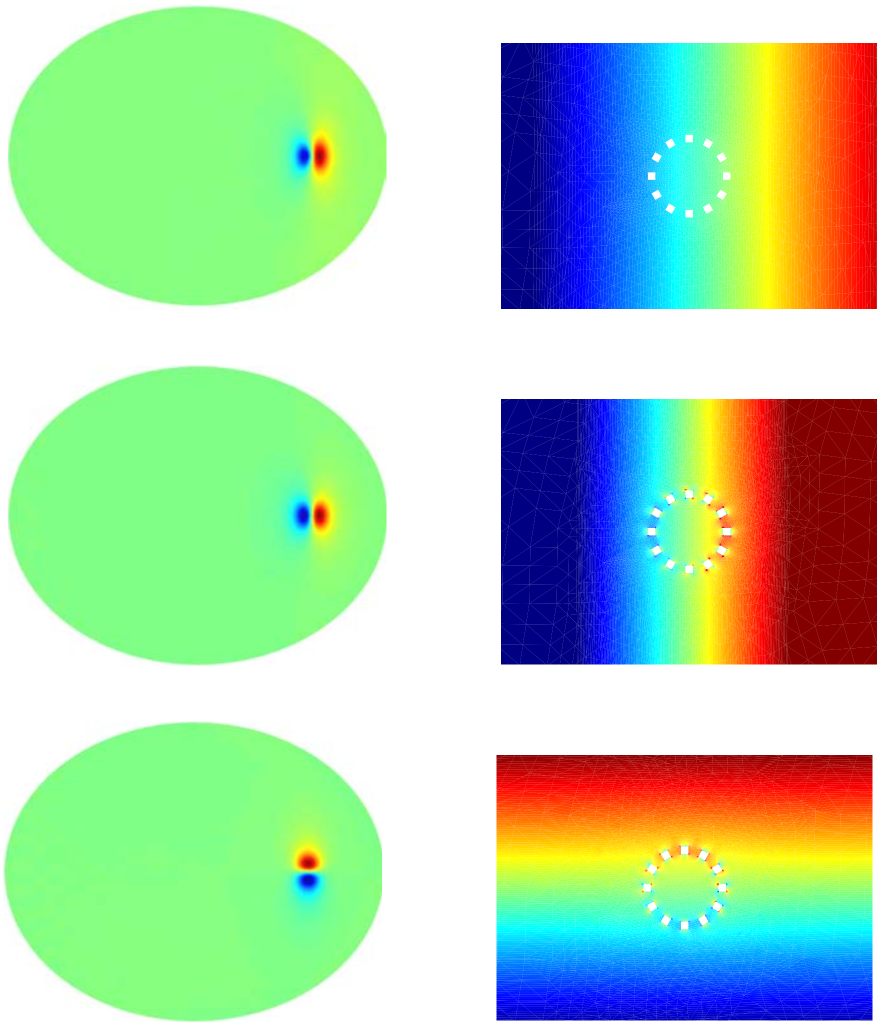

FIGURE 13. Square inclusion: p/H_z (top), u/E_y (middle), and $v/-E_x$ (bottom) near the inclusion, obtained with algorithm $R^{11}(2.6\text{ ms})$ at $t = 4$ s (extremal values for contours on the zooms have been adapted).

reorganizations only. The main difficult task will certainly consist in obtaining an efficient parallel implementation of these local time-stepping algorithm. In particular, mesh partitioning and message passing have to be optimized. Another promising research direction concerns the use of more than second-order accurate symplectic time schemes.

References

[1] E. Bécache, P. Joly and J. Rodríguez, Space-time mesh refinement for elastodynamics. Numerical results. *Comput. Method. Appl. M.* **194** (2005) 355–366.

[2] N. Canouet, L. Fezoui and S. Piperno, A new Discontinuous Galerkin method for 3D Maxwell's equations on non-conforming grids, in *Proc. Sixth International Conference on Mathematical and Numerical Aspects of Wave Propagation*, G.C. Cohen *et al.* Ed., Springer, Jyväskylä, Finland (2003) 389–394.

[3] C. Chauviere, J.S. Hesthaven, A. Kanevsky and T. Warburton, High-order localized time integration for grid-induced stiffness, in *Proc. Second M.I.T. Conference on Computational Fluid and Solid Mechanics*, Cambridge, MA (2003).

[4] J.-P. Cioni, L. Fezoui, L. Anne and F. Poupaud, A parallel FVTD Maxwell solver using 3D unstructured meshes, in *Proc. 13th annual review of progress in applied computational electromagnetics*, Monterey, California (1997) 359–365.

[5] B. Cockburn, G.E. Karniadakis, C.-W. Shu Eds., *Discontinuous Galerkin methods. Theory, computation and applications* **11** *Lect. Notes Comput. Sci. Engrg.*, Springer-Verlag, Berlin (2000).

[6] B. Cockburn and C.-W. Shu, Runge-Kutta discontinuous Galerkin methods for convection-dominated problems. *J. Sci. Comput.* **16** (2001) 173–261.

[7] F. Collino, T. Fouquet and P. Joly, Conservative space-time mesh refinement methods for the FDTD solution of Maxwell's equations. *J. Comput. Phys.* **211** (2006) 9–35.

[8] C. Dawson and R. Kirby, High resolution schemes for conservation laws with locally varying time steps. *SIAM J. Sci. Comput.* **22** (2001) 2256–2281.

[9] A. Elmkies and P. Joly, Éléments finis d'arête et condensation de masse pour les équations de Maxwell: le cas de dimension 3. *C. R. Acad. Sci. Paris Sér. I Math.* **325** (1997) 1217–1222.

[10] L. Fezoui, S. Lanteri, S. Lohrengel and S. Piperno, Convergence and stability of a discontinuous Galerkin time-domain method for the 3D heterogeneous Maxwell equations on unstructured meshes. *ESAIM: M2AN* **39** (2005) 1149–1176.

[11] D.J. Hardy, D.I. Okunbor and R.D. Skeel, Symplectic variable step size integration for N-body problems. *Appl. Numer. Math.* **29** (1999) 19–30.

[12] J. Hesthaven and C. Teng, Stable spectral methods on tetrahedral elements. *SIAM J. Sci. Comput.* **21** (2000) 2352–2380.

[13] J. Hesthaven and T. Warburton, Nodal high-order methods on unstructured grids. I: Time-domain solution of Maxwell's equations. *J. Comput. Phys.* **181** (2002) 186–221.

[14] J. Hesthaven and T. Warburton, High-order nodal discontinuous Galerkin methods for the maxwell eigenvalue problem. *Philos. Trans. Roy. Soc. London Ser. A* **362** (2004) 493–524.

[15] T. Hirono, W.W. Lui and K. Yokoyama, Time-domain simulation of electromagnetic field using a symplectic integrator. *IEEE Microwave Guided Wave Lett.* **7** (1997) 279–281.

[16] T. Hirono, W.W. Lui, K. Yokoyama and S. Seki, Stability and numerical dispersion of symplectic fourth-order time-domain schemes for optical field simulation. *J. Lightwave Tech.* **16** (1998) 1915–1920.

[17] T. Holder, B. Leimkuhler and S. Reich, Explicit variable step-size and time-reversible integration. *Appl. Numer. Math.* **39** (2001) 367–377.

[18] W. Huang and B. Leimkuhler, The adaptive Verlet method. *SIAM J. Sci. Comput.* **18** (1997) 239–256.

[19] J.M. Hyman and M. Shashkov, Mimetic discretizations for Maxwell's equations. *J. Comput. Phys.* **151** (1999) 881–909.

[20] P. Joly and C. Poirier, A new second order 3D edge element on tetrahedra for time dependent Maxwell's equations, in *Proc. Fifth International Conference on Mathematical and Numerical Aspects of Wave Propagation*, A. Bermudez, D. Gomez, C. Hazard, P. Joly, J.-E. Roberts Eds., SIAM, Santiago de Compostella, Spain (2000) 842–847.

[21] C.A. Kennedy and M.H. Carpenter, Additive Runge-Kutta schemes for convection-diffusion-reaction equations. *Appl. Numer. Math.* **44** (2003) 139–181.

[22] D.A. Kopriva, S.L. Woodruff and M.Y. Hussaini, Discontinuous spectral element approximation of Maxwell's equations, in *Discontinuous Galerkin methods. Theory, computation and applications* **11** *Lect. Notes Comput. Sci. Engrg.* B. Cockburn, G.E. Karniadakis, C.-W. Shu Eds., Springer-Verlag, Berlin (2000) 355–362.

[23] B. Leimkuhler, Reversible adaptive regularization: perturbed Kepler motion and classical atomic trajectories. *Philos. Trans. Roy. Soc. London Ser. A* **357** (1999) 1101–1134.

[24] X. Lu and R. Schmid, Symplectic discretization for Maxwell's equations. *J. Math. Computing* **25** (2001) 1–21.

[25] S. Piperno, Fully explicit DGTD methods for wave propagation on time-and-space locally refined grids, in *Proc. Seventh International Conference on Mathematical and Numerical Aspects of Wave Propagation*, Providence, RI (2005) 402–404.

[26] J.-F. Remacle, K. Pinchedez, J.E. Flaherty and M.S. Shephard, An efficient local time stepping-discontinuous Galerkin scheme for adaptive transient computations. Technical report 2001-13, Rensselaer Polytechnic Institute (2001).

[27] M. Remaki, A new finite volume scheme for solving Maxwell's system. *COMPEL* **19** (2000) 913–931.

[28] R. Rieben, D. White and G. Rodrigue, High-order symplectic integration methods for finite element solutions to time dependent Maxwell equations. *IEEE Trans. Antennas Propagation* **52** (2004) 2190–2195.

[29] J.M. Sanz-Serna and M.P. Calvo, *Numerical Hamiltonian Problems*, Chapman and Hall, London, UK (1994).

[30] J. Shang and R. Fithen, A comparative study of characteristic-based algorithms for the Maxwell equations. *J. Comput. Phys.* **125** (1996) 378–394.

[31] T. Warburton, Application of the discontinuous Galerkin method to Maxwell's equations using unstructured polymorphic *hp*-finite elements, in *Discontinuous Galerkin methods. Theory, computation and applications* **11** *Lect. Notes Computat. Sci. Engrg.*, B. Cockburn, G.E. Karniadakis, C.-W. Shu Eds., Springer-Verlag, Berlin (2000) 451–458.

[32] T. Warburton, Spurious solutions and the Discontinuous Galerkin method on non-conforming meshes, in *Proc. Seventh International Conference on Mathematical and Numerical Aspects of Wave Propagation*, Providence, RI (2005) 405–407.

[33] K.S. Yee, Numerical solution of initial boundary value problems involving Maxwell's equations in isotropic media. *IEEE Trans. Antennas Propagation* **16** (1966) 302–307.