fluids

MDPI

Article

Second-Order Time-Accurate ALE Schemes for Flow Computations with Moving and Topologically Changing Grids

Daniel Costero and Federico Piscaglia *

Department of Aerospace Science and Technology (DAER), Politecnico di Milano, 20156 Milano, Italy; daniel.costero@polimi.it

* Correspondence: federico.piscaglia@polimi.it; Tel.: +39-0223998620

Abstract: In computations of unsteady flow problems by the arbitrary Lagrangian–Eulerian (ALE) method, the introduction of the grid velocity in the transport terms of the governing equations is not a sufficient condition for conservativeness if topology changes in the dynamic mesh are present and the number of mesh cells changes. We discuss an extension to second-order time differencing schemes (Implicit Euler and Crank–Nicolson) in the finite volume framework, to achieve second-order time-accuracy of the solution. Numerical experiments are given to illustrate the effectiveness of the presented method.

Keywords: mesh motion; dynamic mesh; ALE method; automatic topological changes; discrete geometric conservation; aw; DGCL; second-order time schemes; layer addition/removal; OpenFOAM

Citation: Costero, D.; Piscaglia, F. Second-Order Time-Accurate ALE Schemes for Flow Computations with Moving and Topologically Changing Grids. *Fluids* **2023**, *8*, 177. https://doi.org/10.3390/fluids8060177

Academic Editors: D. Andrew S. Rees and Nilanjan Chakraborty

Received: 24 April 2023
Revised: 1 June 2023
Accepted: 2 June 2023
Published: 8 June 2023

1. Introduction

Many interesting scientific and engineering problems in computational fluid dynamics (CFD) involve the coupling of high-speed fluid flows with body-fitted meshes with moving and possibly deforming boundaries [1–4]. With large displacements, the flow problem is often formulated in an arbitrary Lagrangian–Eulerian (ALE) scheme [5–8] and is discretized on a moving grid. An ALE integrator is constructed by combining the same time integrator adopted in static grids together with a procedure to include the velocity of the moving grid points. In finite volume (FV) Eulerian solvers based on the co-located grid arrangement, the transposition in time-varying coordinates of the conservation equations is achieved by substituting all advection velocities by their relative (to grid movement) counterparts [9]. Despite that the ALE method is applied to problems with prescribed boundary motion and large deformations [10–13], it is well known that it does not necessarily preserve the order of time-accuracy of its fixed counterpart. To make the rezone calculation conservative of mass, momentum, and energy, the ALE method requires that an additional constraint is fulfilled, namely the geometric conservation law (GCL). This was first discovered by Trulio [5,6], who applied it to solve one-dimensional problems by a finite difference method. In [14], the GCL was applied to the finite volume (FV) method, while its application in arbitrarily moving geometries was presented in [15]. A mathematical analysis has demonstrated the role of the GCL for its time–space accuracy order [16,17], while in [18], it is proved that the discrete GCL (DGCL) is a sufficient condition for some schemes to satisfy the maximum principle for passive species. The failure to satisfy the discretized geometric conservation Law (DGCL) on moving meshes introduces errors in the form of artificial mass sources [14]. In [19,20], it is shown that satisfaction of the DGCL is a necessary condition for any temporal discretization scheme to preserve on moving grids the non-linear stability properties of its fixed-grid counterpart but also that satisfaction of the DGCL is not a sufficient condition for a temporal discretization scheme to preserve its order of time-accuracy on moving grids, as established on fixed grids [21]. When mesh topology modifications are combined with ALE schemes, the fulfillment of the GCL is

not trivial: grid points are added or removed, and the mesh resolution and connectivity dynamically change to accommodate large prescribed boundary deformations (re-zone phase). Additionally, the solution interpolation (remap) must be conservative and accurate and must preserve the monotonicity of the solution. Topology changes may involve the addition/deletion of cell faces without affecting the number of the control volumes (CV) (as it happens when different mesh regions are attached/detached through non-conformal interfaces [22]), or it can also modify the number of CVs, as in adaptive methods, namely dynamic cell layering [13,23], adaptive mesh refinement (AMR) and remeshing. In a broad sense, the simplest example of topology change is remeshing [24], which consists of a remap of the fluid-dynamic solution onto a different mesh at prescribed times, when the mesh quality is assumed to become critical. This approach is simple to perform, because the solver solution is decoupled by the mesh generation, which is usually carried out during the pre-processing; it may become quite inconvenient if several grids must be generated to prevent invalid elements from appearing, because it can be very costly. In general, any local re-meshing or error-driven adaptive refinement may be considered as an example of topological change, which implies recomputing the mesh connectivity. The overhead due to topology changes is strictly implementation-dependent and it is never negligible. When adaptive methods are applied on deforming grids, the solution transfer between meshes with different topologies may be non-trivial.

The extension of ALE schemes to variable topology grids was investigated in [25,26], where the edge-swapping techniques were exploited to modify the grid without changing the number of grid nodes. In [3,27,28], methods based on explicit interpolation of the solution from the old to the new grid are proposed for ALE schemes used with grid connectivity changes. This may originate problems in enforcing conservativeness and monotonicity and in complicating the implementation of multi-step time integration algorithms [29]. An alternative strategy consists of interpreting the insertion or deletion of a new element as a series of fictitious continuous deformations of the finite volumes associated with the nodes involved in the modification. When applied to three-dimensional problems in the Finite Volume framework, this approach was often labeled as inflation layer meshing, as in [10,12]. This is possible to achieve by a repair step after the re-zone phase where mass is re-distributed between neighboring cells to ensure local bound preservation [30,31] or, if first order time-accurate methods are used, by considering the velocity from the newly added faces to sweep new volumes during the re-zone phase [31–34]. No studies are available about the fulfillment of the DGCL with the ALE scheme when second time-accurate implicit discretization methods are applied with topology changes that involve a variation in the number of cells in the dynamic grid [35].

1.1. Motivation of This Research

The research described in this paper is motivated by the need to improve the temporal accuracy of the solution in simulations where moving meshes are combined with dynamic addition or deletion of cell volumes, as it happens when adaptive mesh refinement (AMR) or dynamic cell layering [13] are triggered. Examples of applications to CFD cases are the calculation of the aerodynamics of moving wings, the calculation of moving pistons in rapid compressing machines, or the simulation of hull hydrodynamics. Increasing the temporal accuracy is extremely important in cases where mixing is present, as small errors in the solution will propagate during the simulation.

1.2. Goals and Highlights

The goal of the present work is to develop an efficient methodology to achieve second-order time accuracy when new volumes are added or deleted into a dynamic grid. In particular, the second-order backward Euler (SOBE) and the Crank–Nicolson (CN) schemes were considered. Numerical analysis was performed on one-dimensional cases, namely: (a) the uniformly accelerated piston, for which an analytical solution is available [36] and (b) a three-dimensional cavity case. Comparisons of the proposed work with other mesh

moving techniques that do not involve topology changes were performed to illustrate the accuracy and conservativeness of the methodology. The verification of the method is demonstrated on a one-dimensional test case for which the analytical solution is available and on a three-dimensional test case where the addition and the removal of cell layers is performed over one preferred direction in confined regions of the mesh. The polyhedral mesh support makes topological changes easier to handle, because the solver is always presented with a valid mesh [13]. The code was implemented as an open-source C++ code in the OpenFOAM® technology.

1.3. Paper Structure

The remainder of this paper is organized as follows. In Section 2, the formulation of the governing equations in a dynamic grid is reviewed, followed by an explanation of their spatial discretization in Section 3. In Section 4, the handling of topology changes in dynamics grids for ALE schemes is explained, followed by the discretization of the temporal derivatives in Section 5. The focus is put on topology changes that involve a change in the total number of cells in the grid, following a two-step procedure to decouple the topology change from the mesh movement. In order to use second-order temporal schemes in the presence of topology changes, an *equivalent ghost state* is introduced in Section 6, applying it to the SOBE and CN schemes in Sections 6.1 and 6.2, respectively. Some clarifications regarding its application to adaptive mesh refinement (AMR) are introduced in Section 6.3. To validate the proposed methodology, two numerical experiments are computed. First, a one-dimensional uniformly accelerated piston case is presented in Section 7, where the solutions are compared to the analytical solution and to a different mesh motion strategy without topology changes. Then, a three-dimensional cavity case is presented in Section 8, where a region of the mesh was moved periodically inside the domain. Conclusions are finally drawn in Section 9. The numerical tool employed in the tests is included in a set of dynamic C++ libraries that were used in combination with the OpenFOAM® FV code as released by the OpenFOAM Foundation in the development version (commit 9ea6e2) [37].

2. Governing Equations for Fluid Transport in Time-Varying Domains

In the ALE framework, the conservation equation in integral form for a generic variable ϕ over a time-changing domain $\tilde{\Omega}(t) \subset \mathbb{R}^3$ reads:

$$\frac{\partial}{\partial t}\int_{\tilde{\Omega}(t)} \rho\phi \,\mathrm{d}V + \int_{\partial\tilde{\Omega}(t)} \rho[(\boldsymbol{U}-\boldsymbol{U}_b)\cdot\boldsymbol{n}]\phi \,\mathrm{d}S - \int_{\partial\tilde{\Omega}(t)} \Gamma_\phi \nabla\phi\cdot\boldsymbol{n}\,\mathrm{d}S = \int_{\tilde{\Omega}(t)} s_\phi \,\mathrm{d}V \tag{1}$$

where ρ is the density, $\boldsymbol{U}$ is the flow velocity, $\boldsymbol{U}_b$ is the velocity of the cell faces, Γ_ϕ is the diffusion coefficient and s_ϕ is any source/sink term of ϕ. From (1), continuity and momentum equations of a compressible flow can be obtained by replacing $\phi = 1$ and $\phi = \boldsymbol{U}$, respectively:

$$\frac{\partial}{\partial t}\int_{\tilde{\Omega}(t)} \rho \,\mathrm{d}V + \int_{\partial\tilde{\Omega}(t)} \rho(\boldsymbol{U}-\boldsymbol{U}_b)\cdot\boldsymbol{n}\,\mathrm{d}S = 0 \tag{2}$$

$$\frac{\partial}{\partial t}\int_{\tilde{\Omega}(t)} \rho\boldsymbol{U} \,\mathrm{d}V + \int_{\partial\tilde{\Omega}(t)} \rho\boldsymbol{U}[(\boldsymbol{U}-\boldsymbol{U}_b)\cdot\boldsymbol{n}]\,\mathrm{d}S - \int_{\tilde{\Omega}(t)} \nabla\cdot\boldsymbol{\sigma}\,\mathrm{d}V = -\int_{\tilde{\Omega}(t)} \nabla p \,\mathrm{d}V + \int_{\tilde{\Omega}(t)} \boldsymbol{F}\,\mathrm{d}V \tag{3}$$

with $p = \rho/RT$ as the thermodynamic pressure, $\boldsymbol{\sigma}$ as the resultant surface stress tensor and $\boldsymbol{F}$ as the resultant external volumetric forces. System closure is achieved by the so-called constitutive laws, whose formulation depends on the properties of the continuous medium, and, for compressible flows, by the energy equation.

3. Variable Positioning and Spatial Discretization

In this work, the finite volume method (FVM) is used, calculating the value of the intensive variables in the center of the cells with a collocated variable arrangement. Spatial discretization of the primary variables was computed as in the following:

- Diffusive term (Laplacian) of a quantity Ψ, e.g., $\nabla \cdot (\Gamma \nabla \Psi)$:

$$\int_V \nabla \cdot (\Gamma \nabla \Psi)\, dV = \int_S (\Gamma \nabla \Psi)_f \cdot \boldsymbol{n} dS \simeq \sum_f \Gamma_f \boldsymbol{S}_f \cdot (\nabla \Psi)_f = \sum_f \Gamma_f |\boldsymbol{S}_f| \nabla_n \Psi_f \tag{4}$$

where $\nabla_n \Psi_f$ is the surface normal gradient of Ψ. The subscript f in (4) indicates the cell-to-face interpolated quantities. Linear cell-to-face interpolation was applied: for irregular polyhedral meshes, interpolation is generalized by defining a weight w for each face:

$$\boldsymbol{\Psi}_f = w \boldsymbol{\Psi}_P + (1 - w) \boldsymbol{\Psi}_N \tag{5}$$

where Ψ_f is the face-interpolated quantity. Subscripts P and N indicate values at the centers of two neighboring cells. The surface gradient of a quantity Ψ is decomposed into an orthogonal part and a (non-orthogonal) correction:

$$\nabla_n \Psi_f^n = \underbrace{\alpha (\Psi_P^n - \Psi_N^n)}_{\text{implicit}} + \underbrace{(\mathbf{n}_f - \alpha \mathbf{d}) \cdot (\overline{\nabla_n \Psi})_f^{n-1}}_{\text{correction (explicit)}} \tag{6}$$

where $\alpha = \frac{1}{\boldsymbol{n}_f \cdot \boldsymbol{d}}$ and $\overline{\nabla_n \Psi_f}$ are the uncorrected normal gradient from the two values of the two cells sharing the face. The explicit part is computed from (5) as:

$$\nabla \Psi_f^{n-1} = w \nabla \Psi_P^{n-1} + (1 - w) \Psi_N^{n-1} \tag{7}$$

where $w = 0.5$ in this work; the normal gradient is computed as:

$$(\overline{\nabla_n \Psi})_f^{n-1} = \nabla \Psi_f^{n-1} \cdot \boldsymbol{n}_f^{n-1}. \tag{8}$$

- Gradient terms: these were discretized by the Green–Gauss theorem:

$$\nabla \Psi_P = \frac{1}{V_P} \sum_f \Psi_f \boldsymbol{S}_f \tag{9}$$

where V_P is the volume of the polyhedral cell P, and $\boldsymbol{S}_f$ is the surface vector of the f-th face of the cell.

- Non-linear terms (convective terms): the convective term in the momentum balance is linearized with the Picard approach: the mass flux ϕ is treated explicitly, and the non-linear term is approximated by:

$$\phi u_{j,rel} \simeq \phi^{n-1} u_j^n - \varphi_{M,\text{moving}}^{n-1} \tag{10}$$

the index $n-1$ in (10) denotes that the values are taken from the result of the previous time step. A technique for momentum-based interpolation of mass fluxes on cell faces [38] is used to mimic staggered-grid discretization to prevent checkerboard effects. Using the divergence theorem, the convective terms are rewritten as:

$$\int_V \nabla \cdot (\boldsymbol{u}\boldsymbol{u}) \simeq \sum_f \phi_f (u_f - u_b) \tag{11}$$

The velocity u_f is interpolated with the same approach presented in (5), while a second-order central differencing scheme is used for the fluxes.

- Conservative remap: in a dynamic grid, the position of the cell centers changes from one time step to the next. Linear interpolation is used for mapping cell-centered quantities from the old to the new mesh, to favor the convergence rate of the solver:

$$\phi(\boldsymbol{x}^n, t^n) = \phi(\boldsymbol{x}^{n-1}, t^{n-1}) + \nabla\phi(\boldsymbol{x}^{n-1}, t^{n-1}) \cdot (\boldsymbol{x}^n - \boldsymbol{x}^{n-1}) \tag{12}$$

However, remapping of the fields defined over the faces of the CVs cannot be applied to extensive quantities, as it strongly influences the conservation (and the convergence rate) of the p-U algorithm. To ensure conservation, fields in the CV faces are interpolated from the values in the CV centers, and a Helmholtz-like equation is then solved to ensure that the remapped state is fully conservative.

4. Finite Volume ALE Scheme for Dynamic Meshes with Topology Changes

In a co-located variable arrangement, all the primitive variables are assigned to the cell centroids, while face-centered variables are obtained by interpolation, denoted by the operator $[\![\cdot]\!]_f$. The mass fluxes, surface integrals from (3), are evaluated by either lower- or higher-order interpolation of velocity components at the CV faces. However, in order to avoid decoupling of pressure and velocity when using co-located variable arrangement, the interpolated pressure contribution to the cell face velocity is corrected [38]:

$$\frac{\partial}{\partial t}\rho\,\phi V + \sum_f \rho_f\,\phi_f\Big(\varphi_f - \varphi_{M,f}\Big) - \sum_f \Gamma_\phi \nabla\phi_f = s_\phi V \tag{13}$$

having defined the cell face flux φ_f:

$$\varphi_f = [\![\boldsymbol{U}]\!]_f \cdot \boldsymbol{n}_f\, S_f \tag{14}$$

where S_f is the face area and $\boldsymbol{n_f}$ its normal unity vector. $\varphi_{M,f}$ is the corresponding mesh flux due to point motion (see [9]), and it is expressed as:

$$\varphi_{M,f} = \boldsymbol{U}_{b,f} \cdot \boldsymbol{n}_f\, S_f \tag{15}$$

This represents the flux induced by the movement of the CV faces. If a face moves with the same velocity than the fluid, the mass flux through the CV face will be zero. If this is true for all the CV faces, the same fluid remains inside the CV, and it becomes a control mass: a Lagrangian description of fluid motion is achieved. On the other hand, if a CV face does not move, its mesh flux will be zero, and (1) simplifies to the traditional equation for static domains. Solving the Navier–Stokes equations in moving boundary problems is not sufficient to ensure the correct solution of the problem. If the mesh changes with time, an additional equation has to be solved to ensure that the numerical procedure followed to account for the movement of the mesh does not introduce any error, typically in the form of spurious mass sources. This equation is called the geometric conservation law (GCL) and establishes a relationship between the time derivative of the cell volumes and the calculation of the fluxes due to CV face motion:

$$\frac{d}{dt}\int_{\tilde{\Omega}(t)} \mathrm{d}V - \int_S \boldsymbol{U}_\mathrm{b} \cdot \mathrm{d}\boldsymbol{S} = 0 \tag{16}$$

The GCL equation can be seen as a conservation equation for the cell volume, stating that the variation in the volume of a cell must be equal to the sum of the volume swept by each CV face. It is equivalent to a conservation law of a fluid with uniform velocity and, in continuous form, does not add any constrain to the problem as it is implicitly satisfied. If the semi-discrete counterpart (DGCL) of (16) is considered, it follows:

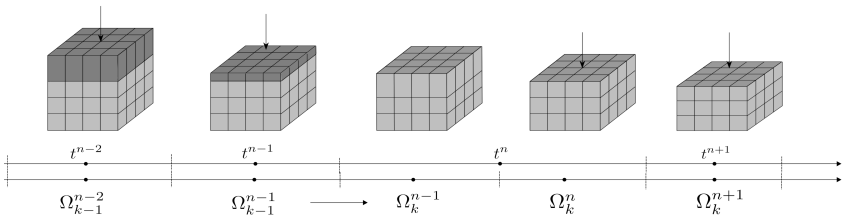
$$\left.\frac{dV}{dt}\right|_{t=n-1}^{t=n} - \sum_f \varphi_{M,f} = 0 \tag{17}$$

a restriction on the numerical schemes arises. More concretely, the calculation of mesh fluxes is defined by the temporal discretization. In problems formulated with the arbitrary Lagrangian–Eulerian framework, the velocity of the mesh points can be imposed independently of the fluid velocity. If the displacement of the points becomes too large, the mesh may quickly become distorted and produce some invalid elements in the mesh. A possible solution for this case would be to change the grid topology, changing the number of entities of the grid (points, faces or cells) or the connectivity between them. A discussion on different topology changes, including connectivity variations, sliding interfaces, element redefinition, can be found in [13,23,34]. In the FV formulation, the interest falls on how the topology change affects the calculation of volume integrals and face fluxes. In the present work, only the topological changes involving a variation in the number of grid cells are considered, because of their connection to the time discretization. In the following, Ω will represent the discrete approximation of the domain volume $\tilde{\Omega}(t)$, i.e., the subdivision of $\tilde{\Omega}(t)$ in a finite number of polyhedral, non-overlapping control volumes (CV) or cells V_i such that $\{V_i\} = \Omega$. Each polyhedral CV is delimited by an arbitrary number of faces $\{f_j\} = \partial V_i$, so that adjacent cells share the same faces. The shorthand Ω_k^n will be used to identify the discretization Ω, whose topology is k, at time t^n. The value of a quantity ϕ in the cell i belonging to the discretization Ω_k^n will be expressed as $\phi_{i,k}^n$. If the topology change involves a variation in the number of grid cells, a remapping has to be made before solving the governing equations in the updated mesh. In a co-located variable arrangement, the commonly applied mesh-to-mesh mapping of intensive variables between Ω_{k-1}^n and Ω_k^n is no longer trivial, as the one-to-one correspondence of cells is missing.

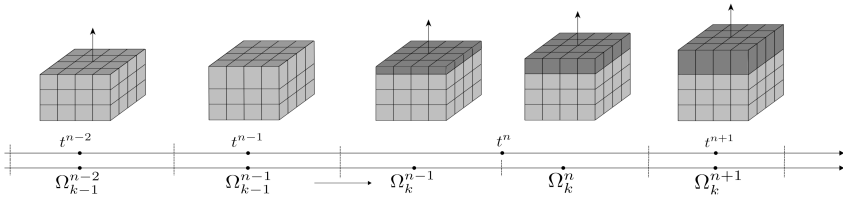
To perform a topology change in a dynamic grid, a two-step procedure was followed, decoupling the topology change from the mesh motion. First, the topology change is performed on a static grid ($\Omega_{k-1}^{n-1} \to \Omega_k^{n-1}$), mapping the solution from the cell centers of the initial topology to the new ones ($\phi_{i,k-1}^{n-1} \to \phi_{j,k}^{n-1}$). Then, the value of the fluxes in the faces are computed by interpolation of the values in the cell centers, and conservativeness is restored by solving a Helmholtz-like equation. Similarly, an *equivalent state* is obtained: all the physical fields at t^{n-1} are now expressed in a grid with a different topology. From this equivalent state, the mesh can be moved following the traditional procedure, as no topology change is involved ($\Omega_k^{n-1} \to \Omega_k^n$):

$$\Omega_{k-1}^{n-1} \to \Omega_k^n = \underbrace{(\Omega_{k-1}^{n-1} \to \Omega_k^{n-1})}_{\text{topology change}} + \underbrace{(\Omega_k^{n-1} \to \Omega_k^n)}_{\text{mesh motion}} \tag{18}$$

This procedure is illustrated in Figure 1 for the particular case of dynamic cell layering, even though the methodology applies to any cell shape, mesh motion or topology change. In Figure 1a, cell volumes are removed while the mesh is moved; in Figure 1b, cell volumes are added during motion in the time interval $\Delta t = t^n - t^{n-1}$.



(**a**) Volume removal



(**b**) Volume addition

Figure 1. Two-step execution of dynamic cell layering: (**a**) cell compression is applied by the upper moving boundary, and removal of cell layers is triggered at $t^{n-1} \rightarrow t^{n}$; (**b**) the upper boundary expands the neighboring cell, and the layer addition at $t^{n-1} \rightarrow t^{n}$ is triggered. Ω_k^n is the FV discretization of the domain, n is the global temporal index and k the index referring to the mesh topology.

5. Temporal Discretization with Topology Changes

The procedure described in Section 4 is well established in CFD solvers supporting mesh motion and topology changes [10,13,35,37], when the first-order implicit Euler scheme is applied for time differencing in a generic flow transport equation:

$$\frac{\partial}{\partial t}\int_{\tilde{V}} \phi \, \mathrm{d}V \approx \frac{V_k^n \, \phi_k^n - V_{k-1}^{n-1} \, \phi_{k-1}^{n-1}}{\Delta t} \tag{19}$$

The transported variables ϕ_{k-1}^{n-1} across the topology change must be computed to ensure conservation with the ALE scheme. This requires particular attention as new volumes to the mesh are added (transition $\Omega_{k-1}^{n-1} \rightarrow \Omega_k^{n-1}$ in Figure 1b). This can be achieved in two ways:

- With cell inflation: it is assumed that the cell faces at t^{n-1} are duplicated to generate new zero-volume cells, which are then inflated to form the new cells at Ω_k^{n-1}:

$$V_k^{n-1} = 0 \tag{20}$$

  and the local topology change (volume addition and mesh motion) is accounted for by the mesh fluxes of the newly added faces from position $\Omega_k^{n-1} \rightarrow \Omega_k^n$, which are computed as:

$$\sum_f \varphi_{M,f_i} = \frac{\Delta \boldsymbol{x}_{f_i} \cdot \boldsymbol{S}_f}{\Delta t} \tag{21}$$

  where $\Delta \boldsymbol{x}_{f_i}$ is the mesh flux corresponding to the volume swept by each face from state Ω_k^{n-1} to state Ω_k^n.
- Without cell inflation: if the newly added faces are assumed to be inserted into their final positions at state Ω_k^{n-1}, their mesh fluxes will be zero:

$$\varphi_{M,f_i} = 0 \tag{22}$$

  and the local topology change will be accounted for in the equation by a conservative remapping of the cell quantities between two different grids. The same reasoning can be applied if a static face is being removed.

After this step, mesh fluxes are computed between $\Omega_k^{n-1} \rightarrow \Omega_k^n$, as in any deforming mesh. Once the mesh fluxes are updated, direct mapping of the primary variables is applied onto the updated topology, while fluxes are adjusted to ensure conservativeness. With first-order temporal schemes, the described steps ensure the fulfillment of the DGCL during mesh motion with topology changes. On the other hand, if second-order accurate temporal schemes are used, primary variables and volumes at the old–old time t^{n-2} are required (ϕ_k^{n-2}). However, transported variables $\phi_{i,k-1}^{n-2}$ are only available at t^{n-2} on the old topology Ω_{k-1}^{n-2}: a way of handling $\phi_{k-1}^{n-2} \rightarrow \phi_k^{n-2}$ is required.

6. Second-Order Temporal Discretization with Dynamic Mesh Refinement

With second-order time schemes and in the presence of topology changes involving the addition of control volumes (namely adaptive mesh refinement and dynamic addition of cell layers [13]), all the primary variables and the cell volumes at time t^{n-2} must be estimated on the updated mesh topology k: ϕ_k^{n-2} (see Figure 1b). In the following, this state will be referred to as the *equivalent ghost state* Ω_k^{n-2}. This equivalent ghost state is only needed for the cells being added or deleted and their neighboring ones. The rest of the mesh does not become influenced by the local topology change, which reduces the computational time, as very few operations have to be performed in a very limited number of cells. While adaptive mesh refinement and the dynamic addition of cell layers are different techniques, the same theory applies to ensure that the DGCL is properly conserved with second-order time differencing schemes. In the following, it will be discussed how to apply second-order backward Euler (SOBE) and the Crank–Nicolson (CN) differencing schemes to discretize the temporal derivatives in the presence of dynamic, topologically changing grids.

6.1. Second-Order Backward Euler Scheme (SOBE)

The second-order backward Euler is a two-step Adams–Moulton method [39,40], where a linear combination of the current-time and the old-time derivatives is used in the LHS:

$$\frac{3}{2}\frac{V^n\phi^n - V^{n-1}\phi^{n-1}}{\Delta t} - \frac{1}{2}\frac{V^{n-1}\phi^{n-1} - V^{n-2}\phi^{n-2}}{\Delta t} = V^n\mathcal{F}(\phi^n) \tag{23}$$

That yields the following expression for the transient term:

$$\frac{\partial}{\partial t}\int_{\tilde{\Omega}(t)} \phi \, \mathrm{d}V \approx \frac{1}{\Delta t}\left(\frac{3}{2}V^n\phi^n - 2V^{n-1}\phi^{n-1} + \frac{1}{2}V^{n-2}\phi^{n-2}\right) \tag{24}$$

The SOBE method is formally second-order and unbounded [41]. The DGCL for the SOBE takes the form:

$$\frac{\frac{3}{2}V_k^n - 2V_k^{n-1} + \frac{1}{2}V_k^{n-2}}{\Delta t} = \frac{3}{2}\frac{(V_k^n - V_k^{n-1})}{\Delta t} - \frac{1}{2}\frac{(V_k^{n-1} - V_k^{n-2})}{\Delta t} = \sum_f \varphi_{M,f}^n \tag{25}$$

If the static cell volumes are added without inflation, it holds that $V_k^n = V_k^{n-1} = V_k^{n-2}$ and mesh fluxes on the newly added faces are zero. If static cell volumes are deleted without inflation, the removed faces will have nil mesh flux, and the volumes can be added to the value of the deforming ones. The DGCL will be naturally satisfied, and no spurious mass sources will appear in the solution. Despite that a uniform time step was considered in the equations, the application of the equations to variable time steps does not impact the validity of the methodology.

6.2. Crank–Nicolson Time-Differencing Scheme (CN)

With dynamic topologically changing grids, the Crank–Nicolson scheme (see Appendix A) must be reformulated to account for the variation in cell volumes during the

different time steps. In particular, the transient term of the N-S equations can be therefore rewritten as:

$$\frac{\partial}{\partial t}\int_{\tilde{\Omega}(t)} \phi \, \mathrm{d}V \approx \frac{(1+\theta)}{\Delta t} \cdot \left[V^n \phi^n - V^{n-1}\phi^{n-1}\right] - \theta\left[\left(\frac{\partial \phi}{\partial t}\right) V\right]^{n-1} \tag{26}$$

with the previous time derivative being evaluated as:

$$\left[\left(\frac{\partial \phi}{\partial t}\right) V\right]^{n-1} = \frac{(1+\theta_0)}{\Delta t_0} \cdot \left[V^{n-1}\phi^{n-1} - V^{n-2}\phi^{n-2}\right] - \theta_0 \left[\left(\frac{\partial \phi}{\partial t}\right) V\right]^{n-2} \tag{27}$$

where the subscript 0 refers to the value of the coefficients at the previous time step. Similarly to the SOBE, the CN scheme also requires the value of the variables at t^{n-2}; thus, a similar procedure can be followed: old–old fields must be computed on the updated topology (equivalent ghost state, Ω_k^{n-2}). With the CN scheme, there is also the need to compute $\frac{\partial \phi}{\partial t}\Big|_k^{n-2}$, which expresses the variation in the field ϕ_k in the time interval $\Delta t^{n-2} = t^{n-2} - t^{n-3}$ in the grid with the new topology. This, in turn, requires recursively knowing the values of the derivatives from previous times until the beginning of the simulation. Therefore, introduction of a topology change will inevitably create a discontinuity in the solution. With local mesh refinements, only a limited amount of cells will change their volumes. As the evaluation of the time derivatives with the Crank–Nicolson scheme depends on the evolution of the cell volumes, the quantity $\frac{\partial \phi}{\partial t}\Big|_k^{n-2}$ also needs to be remapped onto the currently updated topology. If cell refinement is triggered, the estimation of $\frac{\partial \phi}{\partial t}\Big|_k^{n-2}$ is not trivial: a direct mapping of the field from the old topology does not ensure conservativeness, while the calculation of that term would require reconstructing the history of the field on the recent topology reconstructed from the beginning of the simulation. This is clearly extremely demanding to carry out in some cases, and it is impossible to carry out with complex geometries. A possible solution to the problem consists of setting $\frac{\partial \phi}{\partial t}\Big|_k^{n-2} = 0$ in (27) when a topology change is triggered, for the cells that are deforming, right after remapping. This simplification translates to the DGCL, as follows:

$$\frac{(1+\theta)}{\Delta t} \cdot \left[V_k^n - V_k^{n-1}\right] - \frac{(1+\theta^0)}{\Delta t^0} \cdot \left[V_k^{n-1} - V_k^{n-2}\right] + \theta^0 V_k^{n-2} \cancelto{0}{\frac{\partial(\phi)_k^{n-2}}{\partial t}} = \sum_f \varphi_{M,f}^n = \varphi_{M,\text{moving}}^n \tag{28}$$

where $\varphi_{M,\text{moving}}^n$ corresponds to the mesh flux of the moving face, whose value is not affected by the topology change. This implies an imbalance between the mesh flux and the value of V_k^{n-2}, which provokes that the DGCL is no longer satisfied. However, V_k^{n-2} was corrected to fulfill the DGCL with the approximations introduced, such that:

$$\widetilde{V}_k^{n-2} = V_k^{n-1} - \frac{\Delta t^0}{\Delta t}\left(\frac{1+\theta}{1+\theta^0}\right) \cdot \left[V_k^n - V_k^{n-1}\right] + \frac{\Delta t^0}{1+\theta^0} \cdot \varphi_{M,\text{moving}}^n \tag{29}$$

The corrected $\widetilde{V}_k^{n-2}$ of the cells in the dynamic layer will not fill the entire domain completely and uniquely. However, as the topology of the equivalent ghost state is not being updated, this simplification is equivalent to changing just the value of $\widetilde{V}_k^{n-2}$ in the cells belonging to the dynamic layer and will not have any consequences in the following time steps. In other words, $\widetilde{V}_k^{n-2}$ is used to solve the governing equations in the newly added cells and compensates for the simplification introduced. In this way, all the information required to calculate the time derivative is available, and the current solution at t^n can be computed.

6.3. Adaptive Mesh Refinement

Thus far, the focus was put on topological changes involving the addition or removal of layers of cells. However, this technique also applies with adaptive mesh refinement (AMR), where only a single cell may be refined or unrefined according to some user-defined criteria. In static grids, no additional constraints are required, and the normal procedure can be used, as there is no need to ensure the geometric conservation Law. Similar reasoning can be applied if the entire grid moves rigidly, without any cell being deformed.

The focus then is to apply refinement or unrefinement to a deforming cell. The procedure is similar to the one explained for a layer of cells in Figure 1, if only one face of the cell moves along one of the logical axes of a structured grid. If all the added or removed faces during the refinement are static or deform along their own plane, such that their mesh flux is nil, the same procedure can be applied directly without any further consideration. The DGCL will be solved, and the *equivalent ghost state* will again be calculated.

7. One-Dimensional Uniformly Accelerated Piston Test Case

In the following section, the uniformly accelerated piston (UAP) test case [36], for which the analytical solution is available, was selected as the numerical experiment to validate the theory proposed in this work. In the experiment, the wave propagation of the compressible flow inside a cylinder of infinite length is forced by the uniformly accelerated motion of one of its ends (the piston). A schematic of the problem setup is shown in Figure 2. When the piston starts moving with constant acceleration, a pressure wave moving with a velocity c_0 over a gas at rest is formed, where c_0 represents the undisturbed speed of sound. If the piston has positive acceleration, a compression wave is formed; otherwise, a rarefaction wave is observed. The region of the cylinder where $x > c_0 t$ is at rest, as it has not been influenced by the pressure wave yet. On the piston surface, the velocity of the gas is the same as the piston: $v_p = \pm\, at$ in $x = \pm \frac{a\, t^2}{2}$, with $x(t = 0) = 0$ and $v_p(t = 0) = 0$.

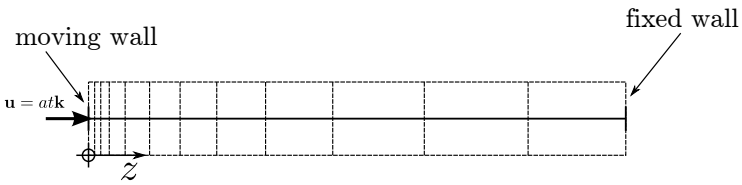


Figure 2. Uniformly accelerated piston—numerical experiment. A wall boundary (piston) moves along an infinite cylinder with uniform acceleration toward a fixed wall. With the reference frame used, positive acceleration is assumed for compression.

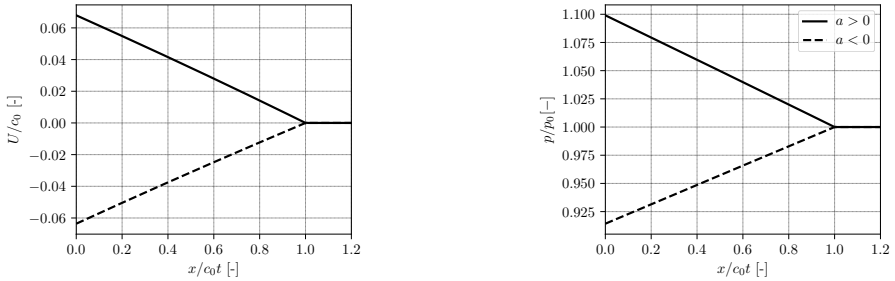
If any effect of the side walls is neglected, gas velocity, pressure and temperature in the region between the piston and $x < c_0 t$ can be calculated as:

$$U(x,t) = \begin{cases} -\frac{1}{\gamma}\left(c_0 + \frac{\gamma+1}{2}at\right) + \frac{1}{\gamma}\sqrt{\left(c_0 + \frac{\gamma+1}{2}at\right)^2 - 2a\gamma(c_0 t - x)} & \text{if } x \le c_0 t \\ 0 & \text{if } x > c_0 t \end{cases} \tag{30}$$

$$p(x,t) = \begin{cases} p_0 \left(1 \pm \frac{\gamma-1}{2}\frac{U}{c_0}\right)^{\frac{2\gamma}{\gamma-1}} & \text{if } x \le c_0 t \\ p_0 & \text{if } x > c_0 t \end{cases} \tag{31}$$

$$T(x,t) = \begin{cases} T_0 \left(1 \pm \frac{\gamma-1}{2}\frac{U}{c_0}\right)^2 & \text{if } x \le c_0 t \\ T_0 & \text{if } x > c_0 t \end{cases} \tag{32}$$

where γ is the ratio of specific heats. In a similar way, the density and speed of sound can be calculated using the ideal gas model. In Figure 3, analytical velocity and pressure profiles across the wave front in the axis of the cylinder are plotted for positive and negative acceleration, where its linear evolution is shown.



(**a**) (**b**)

Figure 3. Analytical solution of velocity (**a**) and pressure (**b**) profiles along the axis of the uniformly accelerated piston. The moving boundary is located at $x = 0$.

In both cases, the duration of the experiment is limited to a time interval $[0, t_{\text{lim}}]$ to avoid the formation of a shock wave during compression or a void region during expansion. t_{lim} can be expressed as:

$$t_{\text{lim}} = \begin{cases} \dfrac{2c_0}{(\gamma+1)|a|} & \text{if} \quad a > 0 \\ \dfrac{2c_0}{(\gamma-1)|a|} & \text{if} \quad a < 0 \end{cases} \tag{33}$$

Without any loss of generality, Equations (30)–(32) can be applied to a cylinder of finite length L: in this case the aforementioned solution will be valid either until the end of the cylinder is reached or until a shock wave is formed. Therefore, it can be written:

$$t_{\text{lim}} = \begin{cases} \min\left[\dfrac{L}{c_0}, \dfrac{2c_0}{(\gamma+1)|a|}\right] & \text{if} \quad a > 0 \\ \min\left[\dfrac{L}{c_0}, \dfrac{2c_0}{(\gamma-1)|a|}\right] & \text{if} \quad a < 0 \end{cases} \tag{34}$$

This analytical solution will be used as a benchmark of the numerical simulations to account for the error introduced by the calculations with specific focus on velocity, pressure and temperature, as it is related to the mass conservation of the problem.

7.1. Case Setup and Simulation Strategy

A one-dimensional FV mesh was used to discretize the UAP geometry of Figure 2. Grid refinement is applied toward the moving wall with a ratio $\Delta x_{\text{max}}/\Delta x_{\text{min}} = 10$. The moving wall physically translates with velocity $\boldsymbol{u} = at\,\boldsymbol{i}$, with $\boldsymbol{i}$ as the unit vector aligned with the piston axis; the same velocity is set on the moving face to enforce the non-permeability constraint. A free-slip boundary condition is applied on the side walls, while a no-slip boundary condition is applied to the upper wall. The total simulation time is set to avoid the pressure wave from reaching the fixed wall. Zero-diffusion on all boundaries is applied for pressure and temperature. The flow is assumed to be laminar. Pressure–velocity coupling is achieved by a transient compressible solver [13]. Two different strategies for mesh motion are compared (Figure 4), namely: (a) mesh motion based on cell stretching—the displacement of each mesh point is inversely proportional to its distance from the moving wall; thus, all cells undergo a deformation and change their size, while the topology of the mesh does not change; (b) rigid motion of a cell zone with dynamic layering [13] (see Figure 5)—only one cell is deforming, while dynamic addition/removal of cells is applied on the cells located at $x/L \approx 0.03$ when $t = 0$. This distance was chosen to be far enough

from the piston and to avoid any influence of the boundary, but to allow the cells to be crossed by the evolving wave. Cell layers are dynamically added or removed when the deformation of the cell involves more than 25% of its thickness. The cell count was 10,000 using the aforementioned 1:10 size ratio between domain ends. Adaptive time stepping was disabled in this set of simulations, to favor a more accurate comparison among results obtained by different dynamic mesh handlings. In simulations with topology changes, cell addition (during expansion) and cell removal (during compression) were triggered at any time step; in this way, the effect of the numerics used to handle topology changes could be accounted for in the quantification of either the time step and the global error. During the simulations, the size of the cell added during mesh refinement ensures keeping the mesh as uniform as possible, to minimize the non-uniformity error. A table containing the most important parameters of the setup can be found in Table 1.

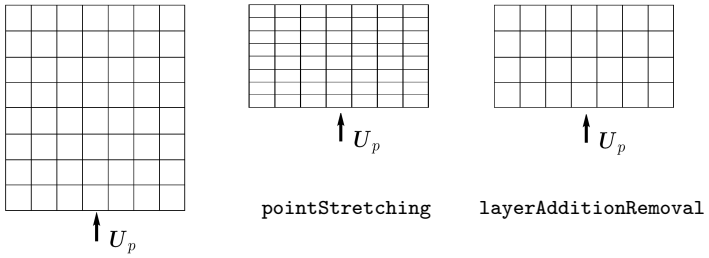


Figure 4. Uniformly accelerated piston (UAP) comparison between different dynamic mesh handling strategies. (**Left**) discretized domain; (**center**) cell stretching; (**right**) cell layering.

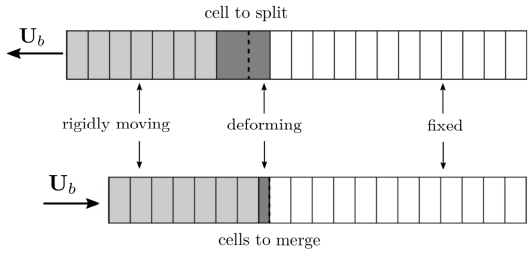


Figure 5. Basic principles of the dynamic addition/removal of cell layers.

For each configuration tested, the point-wise error was computed as:

$$e(x,t) = \frac{f(x,t) - \tilde{f}(x,t)}{\tilde{f}(x,t)} \tag{35}$$

where $f(x,t)$ is one of $\{U(x,t), p(x,t), T(x,t)\}$ computed by the CFD solver, and $\tilde{f}(x,t)$ is the same function evaluated using the analytical formula. The order of accuracy was computed with the normwise error $\|e\|_1(t)$:

$$\|e\|_1(t) = \|f(x,t) - \tilde{f}(x,t)\|_1 \tag{36}$$

All quantities were calculated at the first time step after the first topology change is triggered. This choice was made in regard to quantifying as accurately as possible the properties of the topological change and avoiding any correction of the error that may be performed by later iterations. In addition, a longer simulation was carried out in order to check if the total mass is conserved after the addition/removal of a large number of cells. The results from simulations based on dynamic mesh handling with topological changes were compared with the analytical solution and with the results provided by simulations based on a cell-stretching strategy. As outlined in Figure 4, the use of cell layering in dynamic mesh handling helps to preserve the same discretization independently

on the position of the moving boundary. This helps to maintain the initial mesh quality (uniformity, resolution, non-orthogonality) constant during the whole simulation.

Table 1. Summary of the setup parameters for the numerical experiments.

Variable	Values
Cell motion strategy	Cell stretching, layer A/R
Time scheme	Euler, SOBE, CN
N_{cells}	10,000
Δt (μs)	0.125, 0.25, 0.5, 1, 2

7.2. Code Verification

If the piston moves with positive acceleration (boundary moving inward, Figure 5 bottom), domain cells undergo a compression; this compression is distributed throughout the mesh if cell stretching is applied (deformed by a rate that is inversely proportional to their distance from the moving end), or it is concentrated in the volume between the translating and the fixed regions, where a removal of cells is eventually triggered, in case of mesh motion based on topology changes. With negative acceleration (boundary moving outward, Figure 5 top), cells expand throughout the mesh (cell stretching) or a refinement is applied to the few cells that are deforming. If cell layers are added or removed, the cell count globally changes.

In Figures 6 and 7, the velocity, pressure and temperature profiles along the domain, together with their relative error, are plotted for the compressing and expanding piston. In the graphs, the top row includes the results obtained by cell stretching, while the results obtained with layer addition or removal (A/R) are reported on the bottom. Curves for different values of the time step integration are compared against the analytical solution [36]. The vertical dotted line represents the x-coordinate where layer A/R is triggered at the time that the data are sampled. The results from the simulations are in good agreement with the analytical solution. Small discrepancies are observed at the wave front ($x/(c_0 t) = 1$). With this effect being present either with cell stretching or with cell layering, it is clearly not due to the mesh handling strategy used; it is rather a consequence of the non-conservative form of the governing equations that are solved in a segregated fashion for velocity $\boldsymbol{U}$ and h instead of $(\rho \boldsymbol{U})$ and (ρh). A table summarizing the maximum value of error introduced by the topology change in all the schemes for each time step can be found in Table 2, showing how the absolute value of this error is very small in all the cases. Finally, the formulations of the proposed temporal differencing schemes provide an error with cell layering that is comparable to the error produced by the cell stretching strategy; the relative error is lower than 10^{-6}% for all the tested cases. This confirms once more the proper operation of the methodology proposed in this work.

In the case of the Crank–Nicolson (CN) scheme, the results for the velocity, pressure and velocity fields for dynamic layering and cell stretching are shown in Figures 6c and 7c. In all the cases, the main error is again located in the front wave, and no error created by the topology change can be seen. The error with respect to the cell stretching strategy is around 1×10^{-2}% for the velocity field in the compressing piston, which is two orders of magnitude bigger than for the SOBE scheme. This is expected, as some simplifications were performed to create the equivalent state. However, this value is still very small and negligible with respect to other sources of error. In addition, it can be seen that the error for the compressing piston is bigger than for the expanding one. This may seem counterintuitive because in layer addition, an approximation was performed in two layers of cells. However, the volume that these cells occupy is smaller than the volume occupied in the equivalent state when performing layer removal, thus reducing the total error of the solution.

Table 2. Maximum relative error introduced in the velocity field by the topology change during cell removal (left) and cell addition (right).

	Layer Removal			Layer Addition		
Δt	**Euler**	**SOBE**	**CN**	**Euler**	**SOBE**	**CN**
1.25×10^{-7}	3.19×10^{-4}	3.75×10^{-4}	2.10×10^{-4}	2.27×10^{-4}	3.76×10^{-4}	4.49×10^{-4}
2.50×10^{-7}	5.07×10^{-4}	7.37×10^{-5}	3.12×10^{-4}	8.81×10^{-4}	7.70×10^{-5}	3.18×10^{-4}
5.00×10^{-7}	8.16×10^{-4}	3.50×10^{-5}	3.24×10^{-3}	2.24×10^{-4}	4.26×10^{-5}	6.03×10^{-4}
1.00×10^{-6}	2.35×10^{-3}	6.55×10^{-5}	8.85×10^{-4}	8.88×10^{-4}	6.85×10^{-5}	8.21×10^{-4}
2.00×10^{-6}	1.15×10^{-3}	6.22×10^{-4}	3.11×10^{-3}	2.34×10^{-3}	7.56×10^{-4}	1.60×10^{-3}

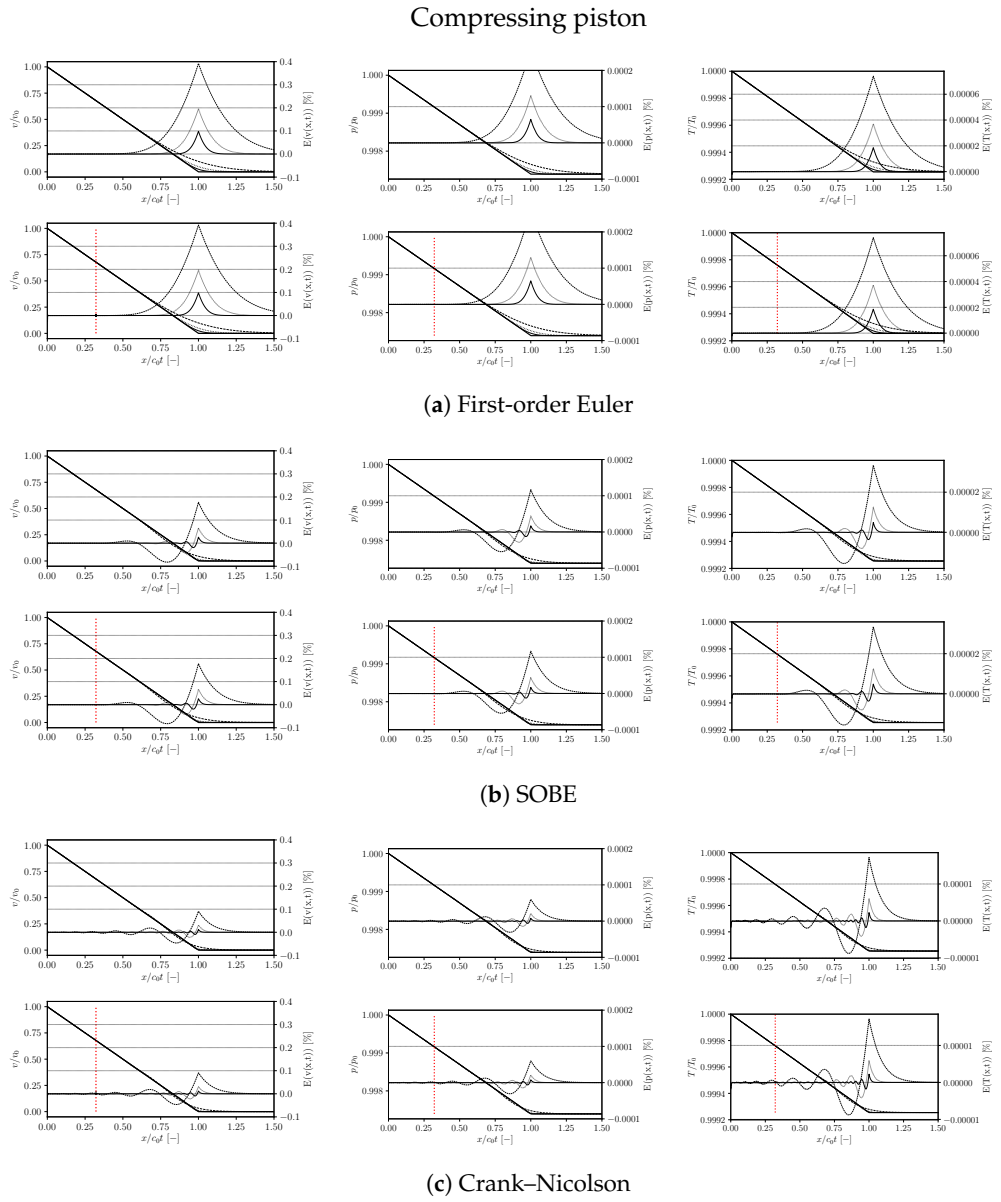


Figure 6. Solution and relative error (%) with cell stretching (upper row) and dynamic layering (bottom row) with different time steps: - - - 2 µs, ····· 0.5 µs and —— 0.125 µs; for the Euler, SOBE and CN time schemes. ····· represents the position of the dynamic layer. For each subfigure, from left to right, errors in velocity, pressure and temperature fields, respectively.

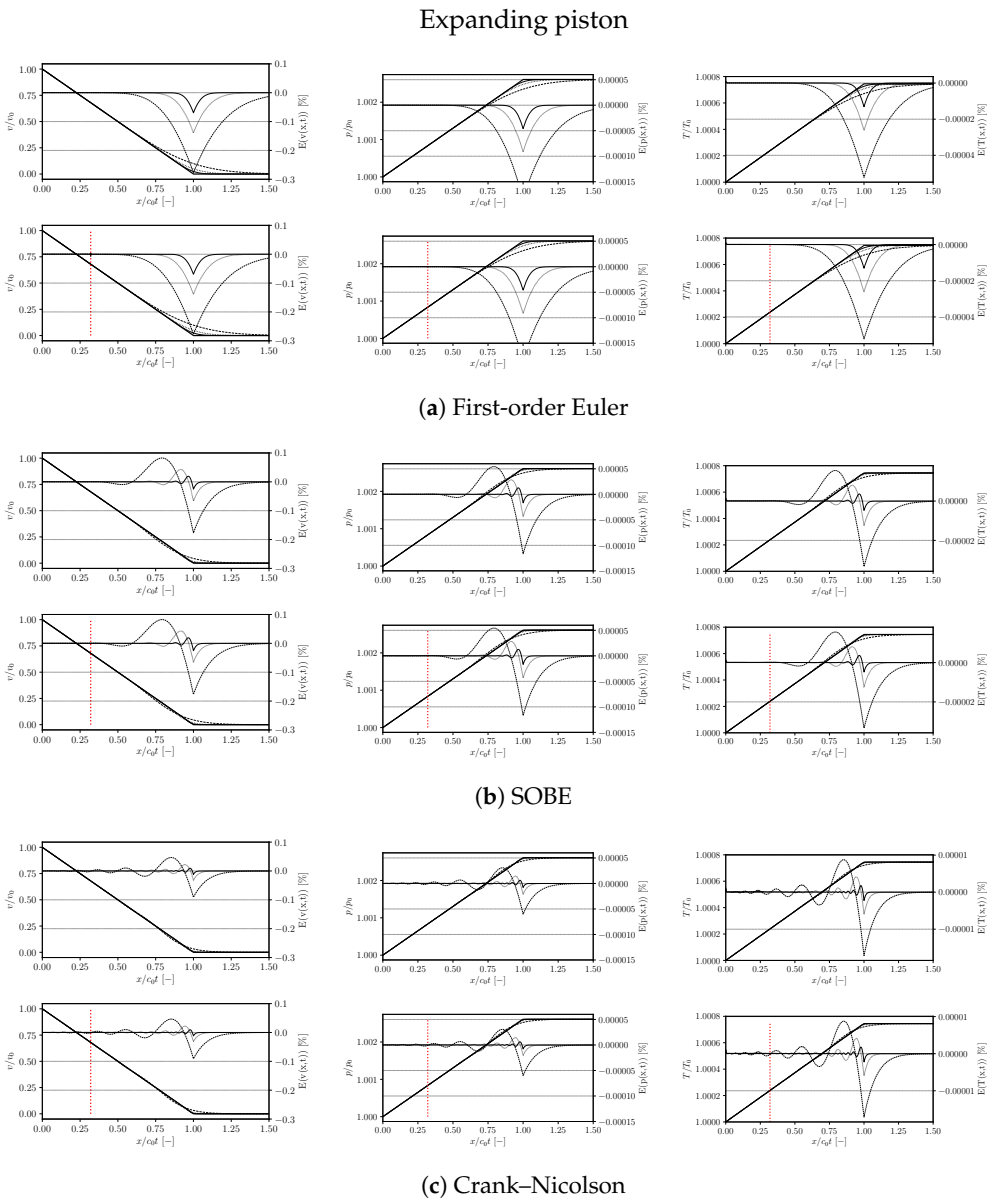


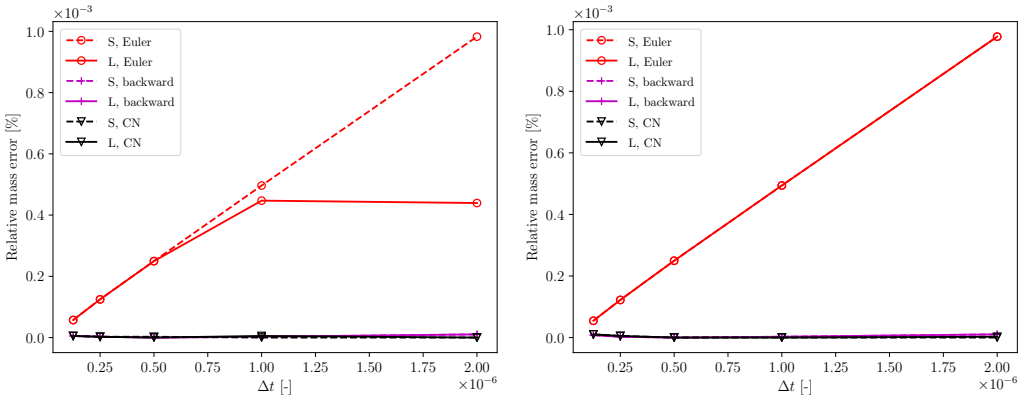
Figure 7. Solution and relative error (%) with cell stretching (upper row) and dynamic layering (bottom row) with different time steps: – – – 2 µs, ····· 0.5 µs and —— 0.125 µs; for the Euler, SOBE and CN time schemes. ····· represents the position of the dynamic layer. For each subfigure, from left to right, errors in velocity, pressure and temperature fields, respectively.

7.3. Mass Conservation

The global conservation error can be estimated by computing the mass imbalance between the start and end of the simulation:

$$E_{\text{mass}} = \left| \frac{\int_\Omega \rho \, \mathrm{d}V\big|_{t=t_N} - \int_\Omega \rho \, \mathrm{d}V\big|_{t=t_0}}{\int_\Omega \rho \, \mathrm{d}V\big|_{t=t_0}} \right| \tag{37}$$

and it is reported in Figure 8a,b for the compressing and expanding pistons, respectively. To compute this value, longer simulations were performed where the number of cells in the mesh changed by at least 100, both in compression and expansion. In all cases, the mass conservation error is very low, always with a relative error below 1×10^{-3}%. For the Euler scheme, the error decreases linearly, whereas it is always very small for the SOBE and CN schemes, both with layer A/R and with cell stretching (in the order of 1×10^{-9}). These results confirm that the proposed methodology does not create any spurious mass sources. This result was expected for Euler and SOBE due to the fact that the DGCL was satisfied explicitly, ensuring that the geometry is always conserved. However, the simplifications made with the CN scheme are very small and do not introduce any spurious mass source in the solution. In addition, these results indicate that the proposed methodology for conservative mapping is able to ensure mass conservation across topology changes both with first- and second-order temporal schemes.

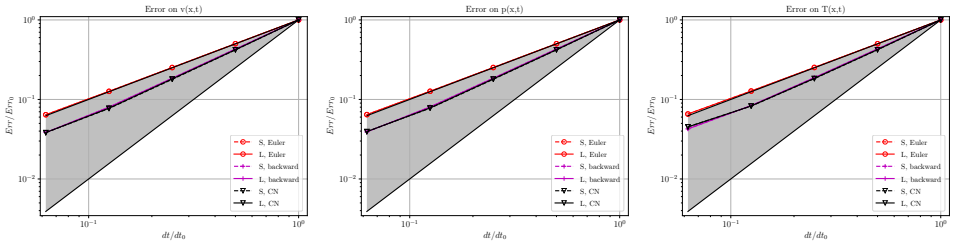


(**a**) (**b**)

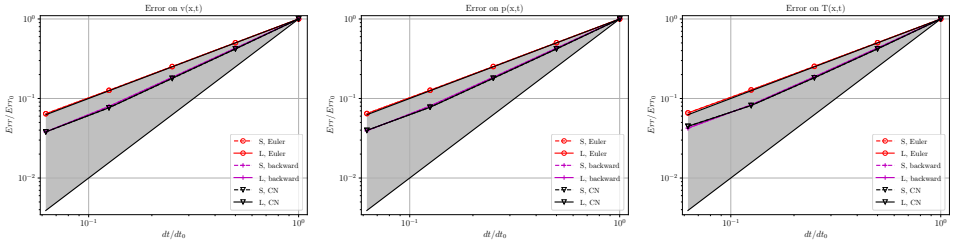
Figure 8. Relative mass error. (**a**) Mass conservation with compressing piston. (**b**) Mass conservation with expanding piston.

7.4. Temporal Order of Accuracy

Thus far, it has been shown that the error introduced in the solution when performing topological changes is very small. In addition, the effect of the methodology in the temporal order of accuracy was also checked. In Figure 9, the order of accuracy of the three numerical schemes with dynamic layering and cell stretching strategies is shown for the velocity, pressure and temperature fields, with the piston both compressing and expanding. First, in all the studied cases, both strategies of mesh motion (dynamic cell layering and mesh stretching) achieved the same order of accuracy. This probes that the error introduced by the topological change does not modify the convergence rate of the solver. For the Euler scheme, an order of 0.99 is recovered for the velocity, pressure and temperature fields, while an order of 1.28 is achieved for both of the second-order schemes. The observed convergence is determined by the relative importance of three error sources: the intrinsic truncation error in the time derivatives $\frac{\partial}{\partial t}$, the error introduced if the DGCL is unfulfilled, and the error due to a lack of momentum conservation. The latter, as in the previous case, is not reduced by a smaller Δt, thus impairing the global accuracy order that is well below the value of 2 on all test cases.



(**a**)

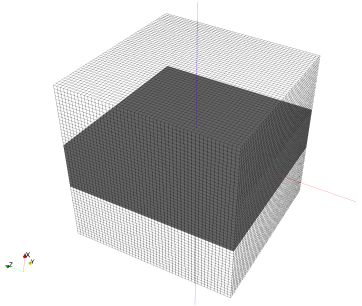


(**b**)

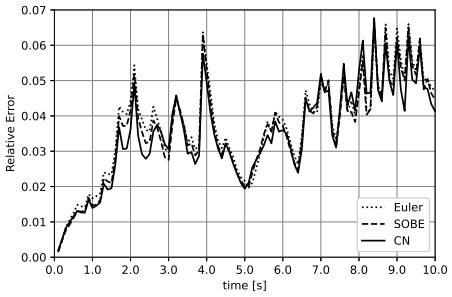
Figure 9. Temporal order of accuracy for Euler, SOBE and CN schemes. (**a**) Compressing piston. (**b**) Expanding piston.

8. Lid-Driven Cavity Test Case

A three-dimensional lid driven cavity case at Re = 5000 and an orthogonal uniform mesh serve as a test to check the proposed method. No turbulence is present in the case tested [42], and the 200 × 200 × 200 mesh is uniform and perfectly orthogonal; thus, no other effect could have an influence on the final solution. At $t = 10$ s, the stationary state is fully reached. A region of the grid made of six horizontal layers of cells located next to the upper part of the domain (Figure 10a) oscillates periodically with a sinusoidal displacement of amplitude 0.045 m and frequency 0.05 Hz, spanning all domains. The neighboring layer of cells, both above and below this region, will deform to accommodate the movement of the cells (Figure 10a). When a threshold value of the cell volume is reached, all dynamic cell layering is applied. For the mesh flux, a first-order upwind scheme was chosen, as it provides the exact solution. The case was chosen because its solution tends to steady state. If the internal grid moves after the steady state is reached, any "unsteady effect" eventually observed is due to the numerics related to the grid motion (e.g., GCL). For this reason, this is a perfect case for validation.



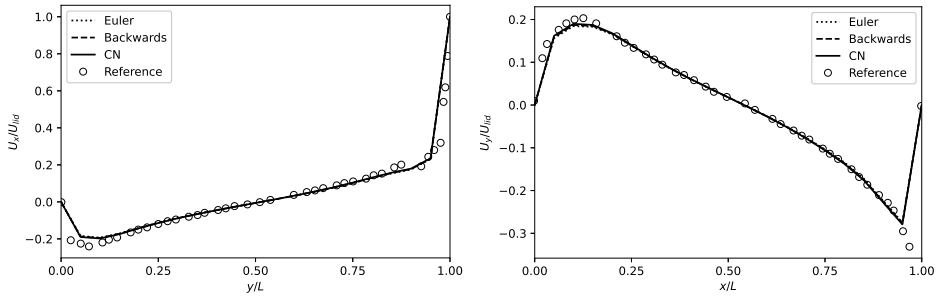
(**a**) Test geometry



(**b**) Relative error

Figure 10. Three-dimensional lid-driven cavity: simulation setup. The dark region of cells oscillates periodically with a sinusoidal displacement of amplitude 0.045 m and frequency 0.05 Hz over the z-axis of the global reference frame. Normalized velocity profiles are computed over two centerlines over the x (U_x, blue) and the z direction (U_z, red). (**a**) Representative computational grid: the number of cells was coarsened to improve clarity in the visualization; (**b**) percent error computed by Equation (38).

Normalized velocity profiles were computed over two center-lines along the x (U_x, blue) and the z directions (U_z, red). Comparisons between reference solutions on a static grid using a second-order implicit backward differencing scheme were performed against dynamic simulations (Figure 11a,b). As apparent, the error introduced by the mesh motion in the flow field is negligible.



(**a**) (**b**)

Figure 11. Lid-driven cavity test case. Normalized velocity profiles computed over two center-lines over the x and the z directions (see Figure 10a). (**a**) $U_x(y)/U_{lid}$ on the vertical line. (**b**) $U_y(x)/U_{lid}$ on the horizontal line.

To quantify the error introduced by the topological changes, the following (percent) relative error was computed:

$$E_x(t) = \frac{\overline{|U_x^{dynamic}(\hat{z},t) - U_x^{static}(z,t)|}}{U_x^{static}(z,t)} \cdot 100 \tag{38}$$

In Equation (38), the superscript dynamic shows that quantities are computed that account for the oscillating region (Figure 10a). The percent relative error is shown in Figure 10b. The results in the dynamic grid were evaluated at $z(\hat{t})$, whose distribution changes with time following the mesh motion. In order to compare the results, the fields were interpolated over the original positions of the grid, where the static solution is computed. The error of the different temporal schemes is reported in Figure 10b. The evolution of the velocity magnitude field at times 1, 2, 3, 4 and 5 s is shown in Figure 12 for the temporal schemes tested (Euler, SOBE and CN). In the figures, graphs on the upper rows refer to simulations on a static grid, while calculations on the dynamic grid are reported in the bottom row. The grid points representing the cell centers are superimposed to the velocity flow field and differ in number and positions between the static and the dynamic grid. For all three schemes tested, results on the static and the dynamic grid are in very good agreement, proving that the proposed methodology does not introduce any error in the solution. The generation and evolution of the vortex is correctly captured in the whole domain and, once it is formed, the mesh motion does not distort it. Both the location and intensity of the vortex are also maintained across the topology changes. Finally, mass conservation was also examined for this test case. The relative mass error was computed using Equation (37) for the three temporal schemes. The results show that the SOBE and CN schemes introduce more error in the solution than the Euler scheme. However, the value of this error is below 2.5×10^{-3}% for all the schemes tested.

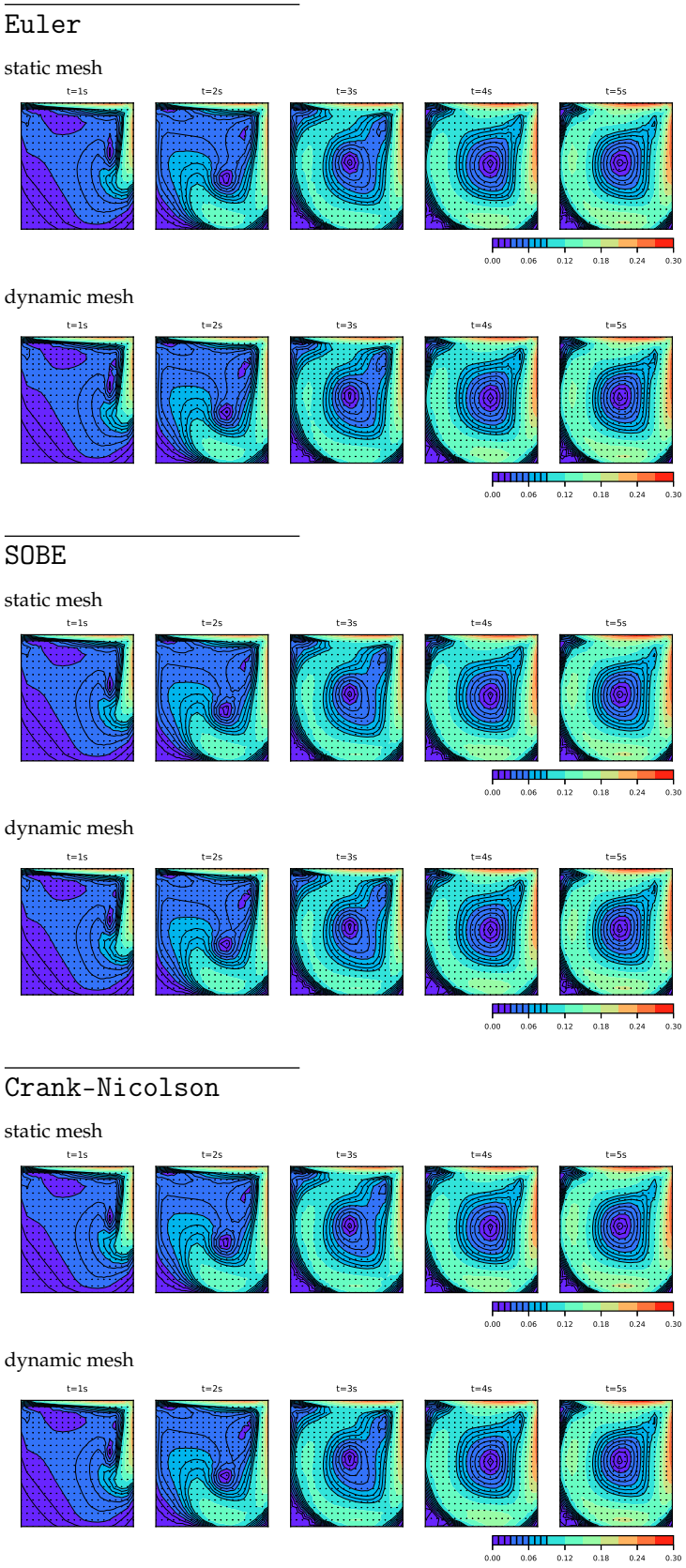


Figure 12. Lid-driven cavity test case. Comparison between the evolution of the velocity flow field in time over a central cutting plane at t = 1, 2, 3, 4, 5 s. Points representing the cell centers are superimposed to the velocity flow field and differ in number and positions between the static and the dynamic grid.

9. Conclusions

This study introduces a novel methodology to maintain second-order temporal accuracy when a mesh undergoes a topology change that alters the number of cells. The lack of available fields from two previous time steps in the new topology prevents direct remapping. To overcome this challenge, the methodology presented in this work employs the geometric conservation law (GCL) to reconstruct the old–old time fields in the new topology, which are necessary to advance the solution in time using second-order time schemes such as the second-order backward Euler (SOBE) and the Crank–Nicolson (CN). Verification on the uniformly accelerated piston one-dimensional test case, for which an analytical solution exists, was proposed. The results demonstrated that the introduced error due to the topology changes was negligible and comparable to the error of moving mesh techniques preserving the grid topology. This was also confirmed by the validation on a three-dimensional lid-driven cavity test case where the generation and convection of a vortex was properly captured. It was finally demonstrated that the proposed method preserves the temporal order of accuracy with topology changes.

Author Contributions: The authors contributed equally to this work. Conceptualization, methodology, writing—original draft, editing, software: D.C. and F.P.; resources, supervision, writing—review: F.P. All authors have read and agreed to the published version of the manuscript.

Funding: This project received funding from the European Union's Horizon 2020 Research and Innovation Programme under the Marie Skłodowska–Curie grant agreement no. 861002.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

ALE	Arbitrary Lagrangian Eulerian
DGCL	Discrete Geometry Conservation Law
SOBE	Second-Order Backward Euler
CN	Crank–Nicolson
AMR	Adaptive Mesh Refinement

Appendix A. Crank–Nicolson Time-Differencing Scheme (CN)

The Crank–Nicolson (CN) [43] temporal scheme is a second-order accurate and bounded scheme. However, it may present oscillatory behavior, which is the reason why it is usually weighted with the first-order Euler scheme. Let us consider the case of a traditional transport equation:

$$\frac{\partial \phi}{\partial t} = \mathcal{F}(\phi) \tag{A1}$$

where $\mathcal{F}(\phi)$ represents the spatial operator. One could divide the time step into two halves and use the implicit Euler scheme to solve the first half:

$$\frac{\phi^{n-\frac{1}{2}} - \phi^{n-1}}{\frac{\Delta t}{2}} = \mathcal{F}(\phi^{n-\frac{1}{2}}) \tag{A2}$$

and the explicit Euler scheme to solve the second one:

$$\frac{\phi^{n} - \phi^{n-\frac{1}{2}}}{\frac{\Delta t}{2}} = \mathcal{F}(\phi^{n-\frac{1}{2}}) \tag{A3}$$

Both schemes provide only first-order accuracy. However, if they are added, one can obtain:

$$\frac{\phi^{n} - \phi^{n-1}}{\Delta t} = \mathcal{F}(\phi^{n-\frac{1}{2}}) = \frac{1}{2}\left[\mathcal{F}(\phi^{n}) + \mathcal{F}(\phi^{n-1})\right] \tag{A4}$$

The temporal term is treated as the implicit Euler scheme, whereas the spatial term $\mathcal{F}(\phi)$ is treated in a semi-implicit manner. However, if Equation (A1) evaluated at t^{n-1} is substituted in Equation (A4), one can obtain:

$$\frac{\phi^n - \phi^{n-1}}{\Delta t} = \frac{1}{2}\left[\mathcal{F}(\phi^n) + \frac{\partial \phi^{n-1}}{\partial t}\right] \tag{A5}$$

where $\frac{\partial \phi^{n-1}}{\partial t}$ is known from the previous time step and corresponds to the evolution of ϕ between t^{n-2} and t^{n-1}. Re-arranging the equation, one can write:

$$2 \cdot \frac{\phi^n - \phi^{n-1}}{\Delta t} - \frac{\partial \phi^{n-1}}{\partial t} = \mathcal{F}(\phi^n) \tag{A6}$$

This formulation requires computing $\frac{\partial \phi^{n-1}}{\partial t}$ instead of $\mathcal{F}(\phi^{n-1})$. The time derivative at the previous time step $\frac{\partial \phi^{n-1}}{\partial t}$ is not readily available, but it can be easily calculated with the values of ϕ^{n-2} as:

$$\frac{\partial \phi^{n-1}}{\partial t} = 2 \cdot \frac{\phi^{n-1} - \phi^{n-2}}{\Delta t^{n-1}} - \frac{\partial \phi^{n-2}}{\partial t} \tag{A7}$$

where the values of $\frac{\partial \phi^{n-2}}{\partial t}$ are stored in memory of the previous time step using this same equation.

In order to increase the stability of the scheme, the method is usually blended with an implicit Euler scheme. This weighting/blending coefficient is called the "off-centering coefficient" $\theta \in [0; 1]$. Introducing θ into the previous equations, it follows:

$$(1+\theta) \cdot \frac{\phi^n - \phi^{n-1}}{\Delta t} - \theta \frac{\partial \phi^{n-1}}{\partial t} = \mathcal{F}(\phi^n) \tag{A8}$$

where $\theta = 1$ recovers the original CN, while the Euler scheme applies with $\theta = 0$. The value of the off-centering coefficient is defined by the user.

References

1. Baum, J.D.; Luo, H.; Löhner, R. A new ALE adaptive unstructured methodology for the simulation of moving bodies. In Proceedings of the Archives of Computational Methods in Engineering, Reno, NV, USA, 10–13 January 1994.
2. Compère, G.; Remacle, J.F.; Jansson, J.; Hoffman, J. A mesh adaptation framework for dealing with large deforming meshes. *Int. J. Numer. Methods Eng.* **2010**, *82*, 843–867. [CrossRef]
3. Hassan, O.; Sørensen, K.A.; Morgan, K.; Weatherill, N.P. A method for time accurate turbulent compressible fluid flow simulation with moving boundary components employing local remeshing. *Int. J. Numer. Methods Fluids* **2007**, *53*, 1243–1266. [CrossRef]
4. Staten, M.L.; Owen, S.J.; Shontz, S.M.; Salinger, A.G.; Coffey, T.S. *Proceedings of the 20th International Meshing Roundtable*; Chapter A Comparison of Mesh Morphing Methods for 3D Shape Optimization; Springer: Berlin/Heidelberg, Germany, 2012; pp. 293–311. [CrossRef]
5. Trulio, J.G.; Trigger, K.R. *Numerical Solution of the One-Dimensional Hydrodynamic Equations in an Arbitrary Time-Dependent Coordinate System*; Technical report; Report UCLR-6522; University of California Lawrence Radiation Laboratory: Berkeley, CA, USA, 1961.
6. Trulio, J.G. *Report No. AFWL- TR-66-19*; Technical report; Air Force Weapons Laboratory: Kirtland Air Force Base, NM, USA, 1961.
7. Hirt, C. *An Arbitrary Lagrangian-Eulerian Computing Technique*; Springer: Berlin/Heidelberg, Germany, 1971; Volume 8, Chapter 1, pp. 350–355. [CrossRef]
8. Hirt, C.; Amsden, A.; Cook, J. An arbitrary Lagrangian-Eulerian computing method for all flow speeds. *J. Comput. Phys.* **1974**, *14*, 227–253. [CrossRef]
9. Ferziger, J.; Perić, M.; Street, R.L. *Computational Methods for Fluid Dynamics*, 4th ed.; Springer: Berlin/Heidelberg, Germany, 2020. [CrossRef]
10. Amsden, A.; O'Rourke, P.J.; Butler, T.D. *KIVA-II: A Computer Program for Chemically Reactive Flows with Sprays*; LA 11560-MS; Los Alamos National Laboratory: Los Alamos, NM, USA, 1989.
11. Jasak, H.; Weller, H.G.; Nordin, N. In-cylinder CFD simulation using a C++ object-oriented toolkit. In *Proceedings of the SAE Technical Paper 2004-01-0110*; SAE International: Warrendale, PA, USA, 2004; p. 20. [CrossRef]
12. Jasak, H.; Tukovic, Z. Automatic Mesh Motion for the Unstructured Finite Volume Method. *Trans. FAMENA* **2007**, *30*, 1–18.

13. Montorfano, A.; Piscaglia, F.; Onorati, A. An Extension of the Dynamic Mesh Handling with Topological Changes for LES of ICE in OpenFOAM. In *Proceedings of the SAE Technical Paper 2015-01-0384*; SAE International: Warrendale, PA, USA, 2015; p. 20. [CrossRef]
14. Demirdžić, I.; Perić, M. Space conservation law in finite volume calculations of fluid flow. *Int. J. Numer. Methods Fluids* **1988**, *8*, 1037–1050. [CrossRef]
15. Demirdžić, I.; Perić, M. Finite volume method for prediction of fluid flow in arbitrarily shaped domains with moving boundaries. *Int. J. Numer. Methods Fluids* **1990**, *10*, 771–790. [CrossRef]
16. Guillard, H.; Farhat, C. On the significance of the geometric conservation law for flow computations on moving meshes. *Comput. Methods Appl. Mech. Eng.* **2000**, *190*, 1467–1482. [CrossRef]
17. Nobile, F. Numerical Approximation of Fluid-Structure Interaction Problems with Application to Haemodynamics. Ph.D. Thesis, Ecole Polytechnique Federale de Lausanne (EPFL), Lausanne, Switzerland, 2001.
18. Farhat, C.; Geuzaine, P.; Grandmont, C. The Discrete Geometric Conservation Law and the Nonlinear Stability of ALE Schemes for the Solution of Flow Problems on Moving Grids. *J. Comput. Phys.* **2001**, *174*, 669–694. [CrossRef]
19. Geuzaine, P.; Grandmont, C.; Farhat, C. Design and analysis of ALE schemes with provable second-order time-accuracy for inviscid and viscous flow simulations. *J. Comput. Phys.* **2003**, *191*, 206–227. . [CrossRef]
20. Farhat, C.; Geuzaine, P. Design and analysis of robust ALE time-integrators for the solution of unsteady flow problems on moving grids. *Comput. Methods Appl. Mech. Eng.* **2004**, *193*, 4073–4095. . [CrossRef]
21. Tukovic, Z.; Jasak, H. A moving mesh finite volume interface tracking method for surface tension dominated interfacial fluid flow. *Comput. Fluids* **2012**, *55*, 70–84. . [CrossRef]
22. Piscaglia, F.; Montorfano, A.; Onorati, A. *Development of Fully-Automatic Parallel Algorithms for Mesh Handling in the OpenFOAM-2.2.x Technology*; SAE Technical Paper 2013-24-0027; SAE International: Warrendale, PA, USA, 2013. [CrossRef]
23. Piscaglia, F.; Montorfano, A.; Onorati, A. A Moving Mesh Strategy to Perform Adaptive Large Eddy Simulation of IC Engines in OpenFOAM. In Proceedings of the International Multidimensional Engine Modeling User's Group Meeting 2014, the Detroit Downtown Courtyard by Marriott Hotel, Detroit, MI, USA, 7 April 2014; p. 20. Available online: https://piscaglia.aero.polimi.it/wp-content/uploads/2020/02/IMEM2014.pdf (accessed on 1 June 2023).
24. Anderson, A.; Zheng, X.; Cristini, V. Adaptive unstructured volume remeshing–I: The method. *J. Comput. Phys.* **2005**, *208*, 616–625. [CrossRef]
25. Loubère, R.; Maire, P.H.; Shashkov, M.; Breil, J.; Galera, S. ReALE: A reconnection-based arbitrary-Lagrangian–Eulerian method. *J. Comput. Phys.* **2010**, *229*, 4724–4761. [CrossRef]
26. Loubère, R.; Maire, P.H.; Shashkov, M. ReALE: A Reconnection Arbitrary-Lagrangian–Eulerian method in cylindrical geometry. *Comput. Fluids* **2011**, *46*, 59–69.
27. Löhner, R. Three-dimensional fluid-structure interaction using a finite element solver and adaptive remeshing. *Comput. Syst. Eng.* **1990**, *1*, 257–272. [CrossRef]
28. Alauzet, F. A changing-topology moving mesh technique for large displacements. *Eng. Comput.* **2014**, *30*, 175–200. [CrossRef]
29. Wang, R.; Keast, P.; Muir, P. A comparison of adaptive software for 1D parabolic PDEs. *J. Comput. Appl. Math.* **2004**, *169*, 127–150. [CrossRef]
30. . Barlow, A.J. A compatible finite element multi-material ALE hydrodynamics algorithm. *Int. J. Numer. Methods Fluids* **2008**, *56*, 953–964. [CrossRef]
31. Kucharik, M.; Shashkov, M.; Wendroff, B. An efficient linearity-and-bound-preserving remapping method. *J. Comput. Phys.* **2003**, *188*, 462–471. [CrossRef]
32. Margolin, L.; Shashkov, M. Second-order sign-preserving conservative interpolation (remapping) on general grids. *J. Comput. Phys.* **2003**, *184*, 266–298. [CrossRef]
33. Garimella, R.; Kucharik, M.; Shashkov, M. An efficient linearity and bound preserving conservative interpolation (remapping) on polyhedral meshes. *Comput. Fluids* **2007**, *36*, 224–237. [CrossRef]
34. Re, B.; Dobrzynski, C.; Guardone, A. An interpolation-free ALE scheme for unsteady inviscid flows computations with large boundary displacements over three-dimensional adaptive grids. *J. Comput. Phys.* **2017**, *340*, 26–54. [CrossRef]
35. Piscaglia, F. Developments in Transient Modeling, Moving Mesh, Turbulence and Multiphase Methodologies in OpenFOAM. In Proceedings of the Keynote Lecture at the 4th Annual OpenFOAM User Conference, Cologne, Germany, 11–13 October 2016.
36. Landau, L.D.; Lifshitz, E.M. *Physique Théorique–Tome VI Mécanique del Fluides*; Éditions de Moscou: Moscow, Russia, 1971.
37. The OpenFOAM® Foundation. Available online: http://www.openfoam.org/dev.php (accessed on 1 June 2023).
38. Rhie, C.; Chow, W. A numerical study of the turbulent flow past an isolated airfoil with trailing edge separation. *AIAA J.* **1983**, *21*, 1525–1532. [CrossRef]
39. Patankar, S. *Numerical Heat Transfer and Fluid Flow*; Hemisphere: New York, NY, USA, 1980.
40. Peinado, J.; Ibáñez, J.; Arias, E.; Hernández, V. Adams–Bashforth and Adams–Moulton methods for solving differential Riccati equations. *Comput. Math. Appl.* **2010**, *60*, 3032–3045. [CrossRef]
41. Hirsch, C. *Numerical Computation of Internal and External Flows*; Elsevier: Amsterdam, The Netherlands, 2007. [CrossRef]

42. Martínez, J.; Piscaglia, F.; Montorfano, A.; Onorati, A.; Aithal, S. Influence of momentum interpolation methods on the accuracy and convergence of pressure–velocity coupling algorithms in OpenFOAM®. *J. Comput. Appl. Math.* **2017**, *309*, 654–673. [CrossRef]
43. Crank, J.; Nicolson, P. A practical method for numerical evaluation of solutions of partial differential equations of the heat-conduction type. *Math. Proc. Camb. Philos. Soc.* **1947**, *43*, 50–67. [CrossRef]