D. Isola, A. Guardone, G. Quaranta
Finite-Volume Solution of Two-Dimensional Compressible Flows over Dynamic Adaptive Grids
Journal of Computational Physics, Vol. 285, 2015, p. 1-23
doi:10.1016/j.jcp.2015.01.007

Finite-volume solution of two-dimensional compressible flows over dynamic adaptive grids

D. Isola, A. Guardone*, G. Quaranta

Department of Aerospace Science & Technology, Politecnico di Milano
Via La Masa, 34, 20156 Milano, Italy

Abstract

A novel Finite Volume (FV) technique for solving the compressible unsteady Euler equations is presented for two-dimensional adaptive grids over time dependent geometries. The interpretation of the grid modifications as continuous deformations of the underlying discrete finite volumes allows to determine the solution over the new grid by direct integration of the governing equations within the Arbitrary Lagrangian-Eulerian (ALE) framework, without any explicit interpolation step. The grid adaptation is performed using a suitable mix of grid deformation, edge-swapping, node insertion and node removal techniques in order to comply with the displacement of the boundaries of the computational domain and to preserve the quality of the grid elements. Both steady and unsteady simulations over adaptive grids are presented that demonstrate the validity of the proposed approach. The adaptive ALE scheme is used to perform high-resolution computations of the steady flow past a translating airfoil and of the unsteady flow of a pitching airfoil in both the airfoil and the laboratory reference, with airfoil displacement as large as 200 airfoil chords. Grid adaptation is found to be of paramount importance to preserve the grid quality in the considered problems.

Keywords: ALE formulation, grid adaptation, unsteady, finite-volume

1. Introduction

Two natural description of the motion exists in continuum mechanics: the Lagrangian and the Eulerian one [1]. A third, hybrid approach is the arbitrary Lagrangian-Eulerian description of the fluid motion which combines the advantages of the other two classical approaches and possibly reduces their respective drawbacks [2, 3].

In their earlier applications ALE algorithms where used to extend the capabilities of Lagrangian based solver in tackling solid mechanics problems with

*Corresponding author
Email address: alberto.guardone@polimi.it (A. Guardone)

large deformations. Those type of algorithm usually can be schematically described by a three steps procedure. A Lagrangian/Eulerian phase during which the equations of motion are explicitly updated [4]. A rezone during which the grid quality is improved thanks to grid regularization [5] or by geometry-based node placement [6]. A remap phase, where the solution is interpolated from the old grid to the new one. This last operation is the most critical as it must be conservative, must preserve the monotonicity of the solution and should be as accurate as possible. Many approaches exist, a popular one performs an interpolation using the volumes swept by the elements during the rezone phase as weighs [7] and can be followed by a repair step to prevent under/overshoots [8]. Other techniques combine low-order inter cell fluxes with some portion of higher-order fluxes, in a flux-limiter fashion, e.g. the Flux-Corrected Remapping [9]. More recently, a smoothing procedure was proposed to eliminate the re-mapping error in ALE schemes [10]. Different approaches exists where the solution update is coupled with the remap phase. Indeed in many applications of interest the physics equations are recast in the ALE framework implicitly accounting for the grid movement [11, 12]. Remapping algorithm were successfully applied to the simulation of multi-material problems [13, 14, 15, 49].

Recasting Eulerian schemes in the ALE framework is fairly straightforward and usually requires minor modifications to the algorithm [16], however particular care has to be taken to preserve the time accuracy [17, 12, 18, 19]. In fact, a naïve extension of fixed-grid methods to flows in moving domains does not preserve numerical accuracy and may possibly lead to numerical instabilities [17]. Therefore, care is to be taken in both the evaluation of the local grid velocities and the definition of the geometric quantities, which cannot be chosen independently [20]. Thomas and Lombard [21] proposed to supply the discrete statement of the problem with the additional constraint of reproducing a uniform flow field exactly. This condition, known as the Geometric Conservation Law (GCL), is demonstrated to be sufficient to achieve a first-order time accuracy [22] but it is neither necessary nor sufficient for higher order accuracy [12]. Moreover, satisfying the GCL is a necessary and sufficient condition to guarantee the nonlinear stability of the integration scheme [23]. An updated review of the literature on the subject can be found in [24].

In their standard formulations, ALE methods are usually limited in their action by the occurrence of invalid elements, which poses limitations to the maximum allowable displacement. Moreover, as pointed out in [29, 28, 27], if re-mapping or explicit interpolation over different grids is applied to account for the grid topology alteration in time, difficulties arises in including multi-step time integration algorithm. For very large displacement of the boundaries, it is possible to locally change the topology of the grid without modifying the number of nodes [25, 26, 27], although preserving the grid quality and the desired spacing is not straightforward.

In a previous paper the authors presented a node-centered finite-volume ALE solver for grids undergoing edge-swapping [27], where the modifications occurring in the shape of the finite volumes due to the changes in the topology are recast in a continuous fashion. Such approach makes it possible to compute

the solution at the subsequent time level simply by integrating the governing equations and avoiding the need of an explicit remap phase. In the present work the same finite-volume solver is extended to the case of grid refinement and coarsening. Similarly to the case of edge swapping, the insertion or deletion of a node is described in terms of continuous deformation of the finite volumes associated to the computational grid. Therefore, when a vertex is inserted a new finite volume appears while it disappears when a vertex is removed.

The key idea is to give an interpretation of the changes in the topology that occur in the time lapse from t^n and t^{n+1} as continuous deformation of the finite volumes performed within the same time interval. The area swept by the interfaces is split into two separate contributions: the deformation one, arising from the continuous (in time) grid movement and distortion and the adaptation one, in which additional numerical fluxes are included to account for the changes in the topology. The proposed approach avoids the introduction of any explicit interpolation step between grids with different topologies and instead makes use of the ALE approach, as it is commonly done when only grid deformation is used. Admittedly, the application of ALE mapping is equivalent to an interpolation step; however, its application does not require any special treatment to ensure appropriate accuracy, conservativeness and preservation of function signs. Since cross-grid interpolation is avoided [28, 29], the implementation of multi-step high-order schemes for time integration, e.g. BDF schemes, is straightforward, as discussed in [27].

Future extensions to viscous, thermal conducting fluid are expected to be straightforward, since the ALE formulation do not modify the viscous and thermal conductivity contributions [2, 3]. Care must be taken however in the proximity of the body surface, where very stretched boundary-layer grids made of non-simplicial elements are commonly used. Note that ALE schemes implementing shock-capturing techniques, including the present one, can be easily extended to deal with multi-material interface by e.g. the shock-capturing method of Abgrall [30] and following modifications [31, 32]. As an alternative, thanks to the large-displacement capability of the present scheme, the material interface can be explicitly advected within the ALE formulation and boundary-conforming grids can be used to represent it [33, 34]. Extension to multi-material flows will be the focus of future research activities.

The present paper is structured as follows. The grid update strategy is briefly described in section 2. In section 3 the edge-based ALE solver is described for the case of a non-adaptive grid. A brief description of the time integration procedures is also given in 3.2. In section 4 are presented the modifications to the scheme required to account for the occurrence of edge-swapping, node insertion or deletion. In section 4.5, implementation details are reported. In section 5, numerical experiments are reported. In sections 5.1 and 5.2 the proposed scheme is applied to the pseudo-steady case of a translating airfoil and the unsteady case of a translating and oscillating airfoil, respectively. Computations are carried out in both the airfoil and the laboratory reference frame to demonstrate the suitability of the present approach.

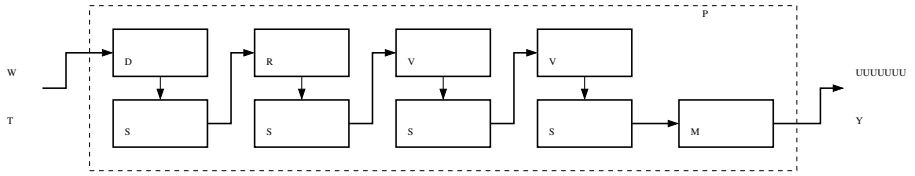


Figure 1: Example of adaptation procedure. The deformation (DEF) and the insertion/deletion procedures (REF/COA) are followed by an edge-swapping (SWP) cycle to restore the highest possible quality without change in the vertices position. At the end regularization (SMO) is carried out.

2. Grid update strategy

In the present work, grid adaptation techniques are used to preserve a high quality of the mesh and to enforce the desired element spacing distribution in numerical simulations of compressible flows involving large displacements of the boundaries. To this purpose, a suitable mix of techniques is adopted to displace the nodes and to locally modify the topology of the elements. All the selected methods are quite standard with few modifications and are reported in the present section only for completeness.

At the end of each time interval the adaptation procedure (AP) illustrated in figure 1 is carried out, which is a combination of five steps as follows: displacement of the boundary and the internal nodes in order to comply with the movement of the boundaries (DEF); triangulation update by means of edge-swapping (SWP); node insertion (REF) and deletion (COA); grid regularization (SMO). At the end of the adaptation step the updated grid complies with the new location of the boundaries, the solution is updated and the cycle is repeated.

Grid deformation is required for the grid to be conformal to the new position of the boundaries and also to ensure that the overall quality of the grid elements is preserved to limit numerical errors. The grid deformation algorithm used in this work is based on the deformable continuum analogy [50] and guarantees very high-quality meshes for small displacements of the boundaries. To allow for very large deformations of the grid, however, edge-swapping is necessary to increase the quality of the grid after the application of the deformation operator. The ability of the swap operator in improving the quality of triangular or tetrahedral grids is well assessed [52] and it allows to significantly extend the maximum allowed displacement [27]. To decide whether an edge must be swapped a comparison between the original and the swapped configuration is performed, with the goal to increase the local minimum quality measure [53, 27].

To locally control the grid spacing node insertion and removal techniques are adopted. Several kinds of element refinement techniques can be found in literature, even for simple triangular grids [54]. In the present work multiple pattern are adopted in order to ensure the highest level of grid quality every time a triangle is refined.

Whenever an element is checked for subdivision, the refinement pattern is

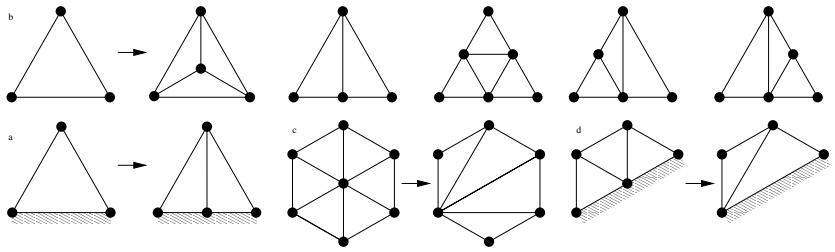


Figure 2: Refinement patterns: (a) refinement pattern for domain elements. From left to right, single node insertion in the center of mass, single node insertion along one edge, triple node insertion along the edges and double node insertion on the edges. (b) Node refinement for elements lying along the boundary, the triple and the double node insertion can be adopted as well. (c) Coarsening of a domain node. (d) Coarsening of a boundary node.

chosen among those sketched in figure 2 (a) in order to maximize the quality of the worst new element. After all the elements have been checked for refinement, a second refinement cycle is performed to avoid hanging nodes. The shape of the boundaries is described thanks to spline interpolation of the initial geometry, therefore once a node is added along the boundary the spline function is evaluated at the appropriate value of the parametric coordinate. The refinement pattern adopted with boundary elements is not limited to the one shown in figure 2 (b), which serves here as an example only.

In figure 2 (c) and (d) the node removal procedure is sketched for both domain and boundary nodes. Once a vertex is flagged for removal, the edges connected to that node are destroyed and the connectivity inside the resulting grid cavity is reconstructed as shown. The node of the cavity from which the connectivity is reconstructed is chosen also in this case to maximize the quality of the worst element of the reconstructed pattern. In the (unlikely) case where all of the possible reconstruction patterns generate invalid elements with negative area, the node is not removed. In order to guarantee the highest element quality through the grid, the node insertion/deletion procedures are followed by a edge-swapping one as illustrated in figure 1.

The goal of grid refinement and coarsening is usually to minimize the numerical errors by controlling the dimension of the elements while limiting the number of nodes, which in unsteady simulations may grow very fast if derefinement is not used. Here, the desired grid spacing is not driven by an error estimator but instead is obtained from a given size function $\mathcal{A}(\boldsymbol{x}, t) \in \mathbb{R}$, which depends on the spatial coordinate $\boldsymbol{x}$ and on the time t. In most cases of aerodynamic interest the smaller elements are gathered close to the solid walls, thus here the prescribed area decays with the square of to the distance from the boundaries, such that the edge length decays linearly. Therefore, for a generic boundary b the desired spacing function is

$$\mathcal{A}_b(\boldsymbol{x}, t) = c_{1,b} + c_{2,b} \min_{\boldsymbol{x}_b \in \partial\Omega_b(t)} ||\boldsymbol{x} - \boldsymbol{x}_b|| + c_{3,b} \min_{\boldsymbol{x}_b \in \partial\Omega_b(t)} ||\boldsymbol{x} - \boldsymbol{x}_b||^2 \qquad (1)$$

where $c_{1,b}$, $c_{2,b}$ and $c_{3,b}$ are coefficients chosen by the user and $\partial\Omega_b$ is the set of

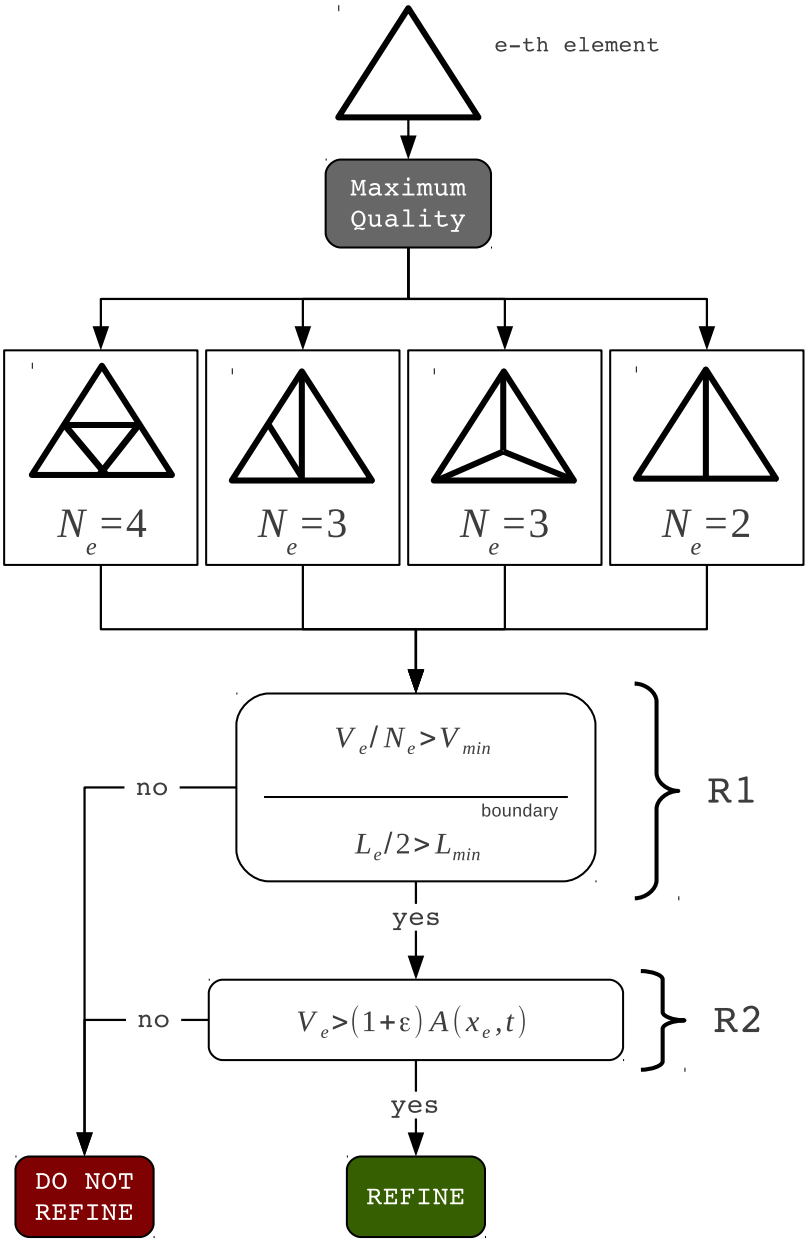


Figure 3: Refinement strategy based on the distance from the boundaries. The pattern that ensure the highest quality is chosen. N_m is the number of elements to be created. `R1`: To avoid too small triangles/segments, the refinement is forbidden if the estimated area/length of new elements is smaller than the minimum allowed. `R2`: The element is marked for refinement if its area is bigger than refinement threshold in that location.

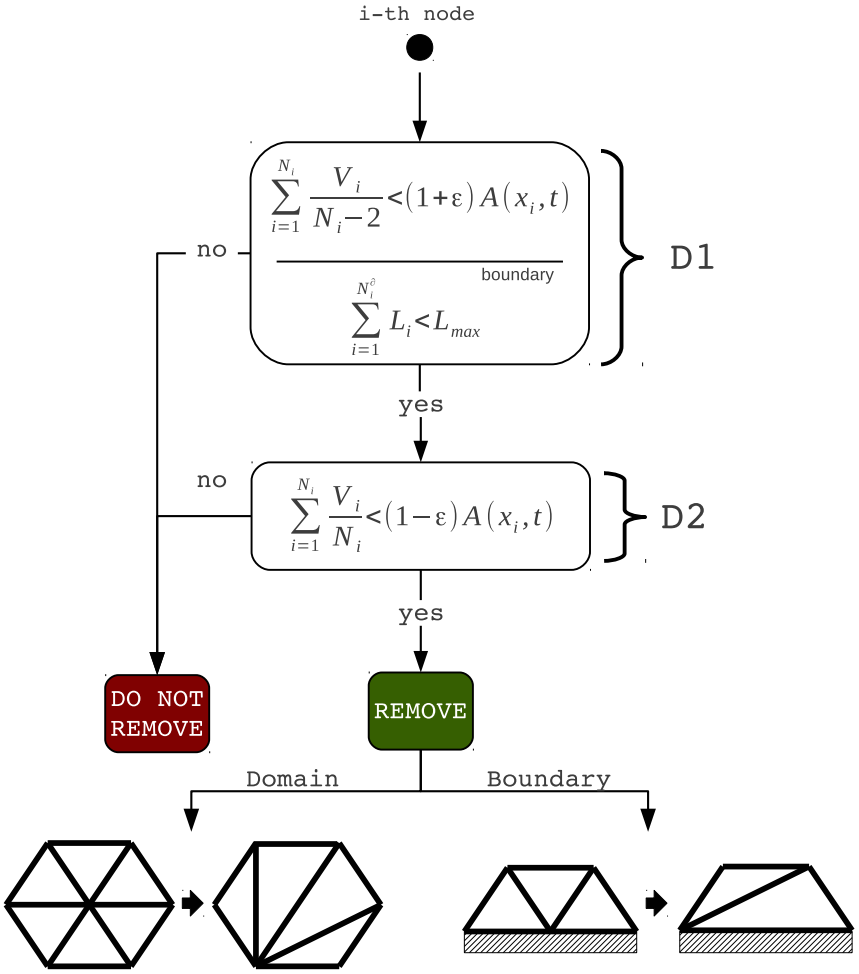


Figure 4: Node removal strategy based on the distance from the boundaries. N_i is the number of elements that belong to $\mathcal{E}_i$. `D1`: To avoid the creation of elements that would be refined in a subsequent iteration, the node is not removed if the predicted average area in the coarse configuration is bigger than maximum area imposed by the dimension function. `D2`: The node is deleted if the average area of the elements surrounding the i-th node is smaller than the coarsening threshold.

points that belongs to the b-th boundary. The overall imposed size function is therefore chosen as

$$\mathcal{A}(\boldsymbol{x}, t) = \min_{b \in \mathcal{B}} \mathcal{A}_b(\boldsymbol{x}, t)\,.$$

Note that the dependence of $\mathcal{A}_b$, thus $\mathcal{A}(\boldsymbol{x}, t)$, on the time is implicit, given the dependence of the boundary points $\boldsymbol{x}_b$ on t.

The flowchart in figure 3 describes the criteria used to mark an element for refinement at a given time t. After choosing the optimal refinement pattern as described above, the element is flagged for refinement if conditions R1 and R2 are simultaneously met. Condition R1 is met when the area of the refined elements is larger than the imposed minimum one and the edges the refined elements that lay on the boundaries (if any) are larger than the minimum imposed. Condition R2 is met when the area of an element is larger than the imposed value, i.e.

$$V_e(t) > (1 + \varepsilon)\mathcal{A}(\boldsymbol{x}_e, t),$$

where $\boldsymbol{x}_e$ is the center of mass of the e-th element and $\varepsilon = 0.1 \div 0.3$ is a suitable tolerance.

The flowchart of figure 4 describes the criteria chosen to mark a node for removal at a certain time t, i.e. if conditions D1 and D2 are simultaneously met. Condition D1 is met when the estimated average area of the triangles formed after the removal of the i-th node and the retriangulation is lower than the imposed area plus a tolerance, i.e.

$$\sum_{e \in \mathcal{E}_i} \frac{V_e(t)}{\dim(\mathcal{E}_i) - 2} < (1 + \varepsilon)\mathcal{A}(\boldsymbol{x}_i, t)$$

with $\mathcal{E}_i$ the set of elements for which the i-th node is a vertex. This helps avoiding the refinement elements that were created by previously removed nodes. Indeed, after the removal of the i-th node the empty patch is subsequently triangulated. The average area of the new elements of forming the patch is thus

$$\bar{V}_{patch} = \sum_{e \in \mathcal{E}_i} \frac{V_e(t)}{\dim(\mathcal{E}_i) - 2}$$

with $\mathcal{E}_i$ the set of elements for which the i-th node is a vertex. If $\bar{V}_{patch}$ is smaller than the refinement threshold (plus a tolerance) it is likely that the newly generated elements will be refined in the subsequent adaptation step. If this occurs condition D1 is met and the i-th node is not removed. This helps avoiding "infinite" refinement/coarsening loops.

Condition D2 is met when the area of the elements surrounding the i-th node is on average smaller than the desired area minus a tolerance, i.e.

$$\sum_{e \in \mathcal{E}_i} \frac{V_e(t)}{\dim(\mathcal{E}_i)} < (1 - \varepsilon)\mathcal{A}(\boldsymbol{x}_i, t).$$

As shown in figure 1, in order to further improve the grid quality after the application of the above techniques, grid regularization is performed [51]. In

the present work a simple, barycentric iterative Laplacian smoothing is applied to the grid coordinates [44]. A well known issue of non-weighted smoothing techniques is the fact that when convergence is achieved the final grid features a uniformly distributed spacing. For such reason the use of grid smoothing should be carefully dosed and only three loops over the grid nodes are performed.

3. Edge-based ALE solver for compressible flows

The governing equations for a two-dimensional compressible inviscid fluid flow enforce the balance of mass, momentum and total energy over a given control volume, see e.g. [38]. In the ALE framework they read

$$\frac{d}{dt}\int_{\mathcal{C}(t)} \mathsf{u}\, d\boldsymbol{x} + \oint_{\partial\mathcal{C}(t)} \big[\mathbf{f}(\mathsf{u}) - \mathsf{u}\,\boldsymbol{v}\big]\cdot\boldsymbol{n}\, ds = 0, \qquad \forall \mathcal{C}(t) \subseteq \Omega(t), \tag{2}$$

where $\boldsymbol{x} \in \Omega(t) \subseteq \mathbb{R}^2$ is the position vector and $t \in \mathbb{R}^+$ is the time. The vector $\mathsf{u} : \Omega \times \mathbb{R}^+ \to \mathbb{R}^4$ collects of the conservative variables, i.e.

$$\mathsf{u} = (\rho, \boldsymbol{m}, E^{\mathrm{t}})^{\mathrm{T}},$$

where ρ is the density, $\boldsymbol{m}$ is the momentum vector and E^{t} is total energy per unit volume. The flux function $\mathbf{f} \in \mathbb{R}^4 \times \mathbb{R}^4$ of Eq. (2) is defined as

$$\mathbf{f}(\mathsf{u}) = \big(\boldsymbol{m},\ \boldsymbol{m}\otimes\boldsymbol{m}/\rho + P(\mathsf{u})\ \boldsymbol{I}^2,\ \big[E^{\mathrm{t}} + P(\mathsf{u})\big]\,\boldsymbol{m}/\rho\big)^{\mathrm{T}}, \tag{3}$$

where $\boldsymbol{I}^2$ is the 2×2 identity matrix and P is the pressure. In the present study, the standard polytropic ideal gas model for air is adopted where the ratio of specific heats is equal to 1.4, which corresponds to air in standard conditions. In (2), $\mathcal{C}(t) \in \Omega(t)$ is an arbitrary control volume moving with velocity $\boldsymbol{v}(\boldsymbol{x}, t) \in \mathbb{R}^2$. The outward unit vector normal to the boundary $\partial\mathcal{C}$ is indicated as $\boldsymbol{n}(s, t) \in \mathbb{R}^2$ and it is a function of the curvilinear coordinate s along $\partial\mathcal{C}$ and of the time t. The dot product between the flux function and the normal vector of Eq. (2) is defined as $\mathbf{f}\cdot\boldsymbol{n} = \mathsf{f}_x n_x + \mathsf{f}_y n_y$. Eq. (2) is to be made complete by specifying suitable initial and boundary conditions on the boundary $\partial\Omega$, see e.g. [35].

Evaluating equation (2) for an uniform flow field delivers the following equation

$$\frac{d}{dt}\int_{\mathcal{C}(t)} d\boldsymbol{x} = \oint_{\partial\mathcal{C}(t)} \boldsymbol{v}\cdot\boldsymbol{n}\, ds, \tag{4}$$

a condition that is usually referred to as the Geometric Conservation Law [36, 24] and that involves the grid geometry only. The GCL is an additional constraint to be satisfied to maintain the accuracy and the stability of the numerical scheme [24, 11].

3.1. Edge-based finite volume solver

A node-centered finite volume discretization of equation (2) is now obtained over a given conformal triangulation of the computational domain Ω, composed of finite volumes $\mathcal{C}_i$. A detailed description of the discretization is presented in [44].

For a general approximation of the numerical fluxes, one finally obtains

$$\frac{d}{dt}\left[V_i\,\mathsf{u}_i\right] = \sum_{k\in\mathcal{K}_{i,\neq}} \Phi(\mathsf{u}_i,\mathsf{u}_k,\nu_{ik},\hat{\boldsymbol{\eta}}_{ik},\eta_{ik}) + \Phi^{\partial}(\mathsf{u}_i,\nu_i,\hat{\boldsymbol{\xi}}_i,\xi_i). \tag{5}$$

where u_i is the cell-averaged value of the solution, V_i is the area of $\mathcal{C}_i$ and Φ and Φ^{∂} are suitable integrated numerical fluxes, representing respectively the flux across the cell interfaces and the boundaries. In equation (5), $\boldsymbol{\eta}_{ik}$, $\boldsymbol{\xi}_i$ and ν_{ik}, ν_i are respectively the integrated outward normals and interface velocities, defined as

$$\boldsymbol{\eta}_{ik} = \int_{\partial\mathcal{C}_{ik}} \boldsymbol{n}_i, \qquad \nu_{ik} = \int_{\partial\mathcal{C}_{ik}} \boldsymbol{v}\cdot\boldsymbol{n}_i, \qquad \boldsymbol{\xi}_i = \int_{\partial\mathcal{C}_i\cap\partial\Omega} \boldsymbol{n}_i \qquad \nu_i = \int_{\partial\mathcal{C}_i\cap\partial\Omega} \boldsymbol{v}\cdot\boldsymbol{n}_i\,, \tag{6}$$

with $\eta_{ik} = |\boldsymbol{\eta}_{ik}|$, $\hat{\boldsymbol{\eta}}_{ik} = \boldsymbol{\eta}_{ik}/\eta_{ik}$, $\xi_i = |\boldsymbol{\xi}_i|$ and $\hat{\boldsymbol{\xi}}_i = \boldsymbol{\xi}_i/\xi_i$. Finally $\mathcal{K}$ is the set of all nodes of the triangulation and $\mathcal{K}_{i,\neq}$ is the set of the indexes of the cells sharing a portion of their boundary with $\mathcal{C}_i$, $\mathcal{C}_i$ excluded.

Slip boundary conditions at solid surfaces and far-field boundary conditions are enforced here in a weak form, namely, by evaluating the boundary flux (5) in a suitable boundary state variable [37, 42].

The finite volume approach in (5) is adopted to compute the interface integrals appearing in the Geometric Conservation Law. Thus, Eq. (4) becomes

$$\frac{dV_i(t)}{dt} = \nu_i(t) + \sum_{k\in\mathcal{K}_{i,\neq}} \nu_{ik}(t). \tag{7}$$

A natural way to satisfy Eq. (7) is to split the derivative of the cell area into contribution pertaining to the so called Interface Volumes. The areas swept by the different parts of the interfaces satisfy the following

$$\nu_{ik} = \frac{dV_{ik}}{dt} \qquad \text{and} \qquad \nu_i = \frac{dV_{i,\partial}}{dt}, \tag{8}$$

where $V_{ik}(t)$ and $V_{i,\partial}(t)$ are the areas swept by the interfaces $\partial\mathcal{C}_{ik}(t)$ and $\partial\mathcal{C}_i(t)\cap\partial\Omega(t)$, respectively. The above definition of the integrated normal velocities allows to automatically satisfy the GCL constraint and is therefore the most natural choice [11, 22]. The differential relations (8), which for an assigned grid motion allows one to compute the interface velocities $\nu_{ik}(t)$ and $\nu_i(t)$, complements the Ordinary Differential Equation (ODE) system (5). For a comprehensive description of the edge-based Euler solver and of how the grid metrics are computed over unstructured grids the reader is referred to [44].

3.2. Time integration of the ALE equations

The fully discrete form of the ALE system (5) in the simple case of constant grid topology is reported here with the purpose of introducing basic concepts related to the time integration of the governing equation in a moving reference. Time integration of the ODE system (5) is carried out using the so-called Backward Differences Formulæ scheme [45]

$$\begin{cases} \displaystyle\sum_{q=-1}^{p} a_q V_i^{n-q} \mathsf{u}_i^{n-q} = \Big[\sum_{k \in \mathcal{K}_{i,\neq}} \Phi(\mathsf{u}_i, \mathsf{u}_k, \nu_{ik}, \hat{\boldsymbol{\eta}}_{ik}, \eta_{ik})^{n+1} \\ \qquad\qquad + \Phi^{\partial}(\mathsf{u}_i, \nu_i, \hat{\boldsymbol{\xi}}_i, \xi_i)^{n+1} \Big] \Delta t, \qquad i \in \mathcal{K} \\ \displaystyle\sum_{q=-1}^{p-1} \alpha_q \Delta V_{ik}^{n-q} = \nu_{ik}^{n+1} \, \Delta t, \qquad k \in \mathcal{K}_{i,\neq} \\ \displaystyle\sum_{q=-1}^{p-1} \alpha_q \Delta V_{\ell,\partial}^{n-q} = \nu_{\ell}^{n+1} \, \Delta t, \qquad \ell \in \mathcal{K}_{\partial} \end{cases} \tag{9}$$

where the superscript indicate the time level and a_q and α_q are the BDF coefficients. In system (9), all quantities are assumed to be known at time level n and all grid-dependent quantities are computed from the known positions of the grid nodes at time level $n+1$.

The Geometric Conservation Law (7) is made discrete using the same BDF scheme adopted to discretize the governing equation, i.e.

$$\sum_{q=-1}^{p-1} \alpha_q \frac{\Delta V_i^{n-q}}{\Delta t} = \nu_i^{n+1} + \sum_{k \in \mathcal{K}_{i,\neq}} \nu_{ik}^{n+1}. \tag{10}$$

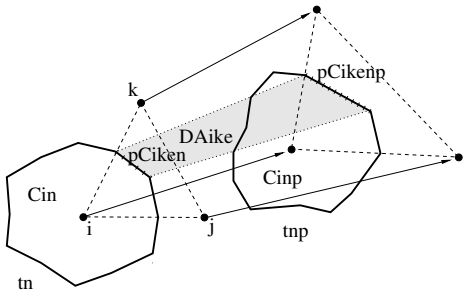
Condition (10) is usually referred to as the Discrete Geometric Conservation Law (DGCL) [46].

The GCL-compliant interface velocity ν_{ik}^{n+1} is computed from the values of ΔV_{ik}^{n+1} that represent the area swept by the interface $\partial \mathcal{C}_{ik}$ during the time step from t^n to t^{n+1}. The area swept by the interfaces during the movement is shown in figure 3.2. For a detailed description on how the swept areas are computed the reader is referred to [27, 44].

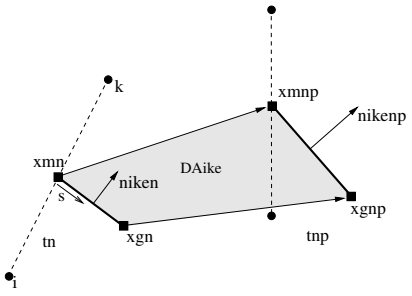
The nonlinear system is solved here for u^{n+1} by means of a modified Newton method, in which the Jacobian of the integrated flux function is approximated by that of the first-order scheme, and by resorting to a dual time-stepping technique [47].

4. ALE scheme for adaptive grids

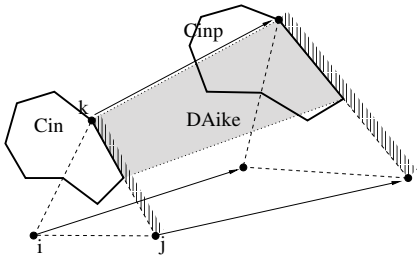
The implicit finite-volume scheme introduced in section 3.2 is extended to the case of adaptive grids. If no changes in the topology occurs, i.e. the grid is simply distorted, the change in position and shape of the finite volumes is easily taken into account by the ALE formulation as shown in section 3.2. However



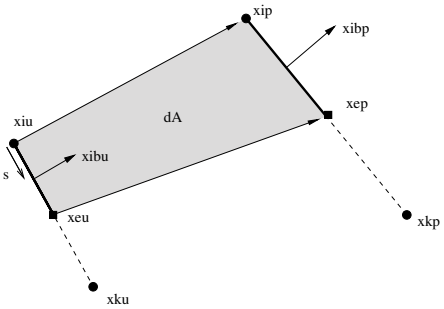
(a) Deformation of the i-th domain cell



(b) Swept area and domain metric vectors



(c) Deformation of the i-th boundary cell



(d) Swept area and boundary metrics

Figure 5: Area swept by the domain and boundary interface $\partial\mathcal{C}_{ik,e}$ and $\partial\mathcal{C}_{i,e}$, respectively, during the interval $[t^n, t^{n+1})$

the modification in the shape of the cells caused by the application of adaptation techniques is accounted for separately.

For clarity, the baseline concepts are first introduced for the case of grids with variable connectivity only (no refinement/derefinement) and then the method is extended to the case of a variable number of nodes.

4.1. ALE scheme with variable connectivity

Following [27], the finite volume ALE solver is first generalized to deal with grids with variable connectivity, i.e. undergoing edge swapping only. Because of edge swapping, the local connectivity varies with time, i.e. $\mathcal{K}_{i,\neq}(t)$, while the total number of points belonging to $\mathcal{K}$ does not change. Therefore, system (9) becomes

$$\begin{cases} \displaystyle\sum_{q=-1}^{p} a_q V_i^{n-q}\, \mathsf{u}_i^{n-q} = \Big[\boldsymbol{\Phi}^{\partial}(\mathsf{u}_i,\nu_i,\hat{\boldsymbol{\xi}}_i,\xi_i)^{n+1} + \sum_{k\in\mathcal{K}_{i,\neq}^{n+1}} \boldsymbol{\Phi}(\mathsf{u}_i,\mathsf{u}_k,\nu_{ik},\hat{\boldsymbol{\eta}}_{ik},\eta_{ik})^{n+1} \\ \qquad\qquad + \displaystyle\sum_{k\in\mathcal{K}_{i,\neq}^{[n-p,n+1)}} \boldsymbol{\Phi}(\mathsf{u}_i,\mathsf{u}_k,\nu_{ik},\hat{\boldsymbol{\eta}}_{ik},0)^{n+1}\Big]\Delta t, & i\in\mathcal{K} \\ \displaystyle\sum_{q=-1}^{p-1} \alpha_q \left(\Delta V_{ik}^{n-q} + \Delta A_{ik}^{n-q}\right) = \nu_{ik}^{n+1}\Delta t, & k\in\mathcal{K}_{i,\neq}^{[n-p,n+1]} \\ \displaystyle\sum_{q=-1}^{p-1} \alpha_q \Delta V_{\ell,\partial}^{n-q} = \nu_{\ell}^{n+1}\Delta t, & \ell\in\mathcal{K}_{\partial} \end{cases} \tag{11}$$

The pair of indices i-k, with $k \in \mathcal{K}_{i,\neq}^{[n-p,n+1)}$ indicates the edges removed during swapping operations performed during the time interval $[t^{n+1-p}, t^{n+1})$. This can be formalized as

$$\mathcal{K}_{i,\neq}^{[n-p,n+1)} = \{k\in\mathcal{K}, k\notin\mathcal{K}_{i,\neq}^{n+1} \quad \text{such that} \quad \nu_{ik}^{n+1}\neq 0\},$$

while $\mathcal{K}_{i,\neq}^{[n-p,n+1]}$ indicates the extension to the edges of the grid at the time level $n+1$, i.e.

$$\mathcal{K}_{i,\neq}^{[n-p,n+1]} = \mathcal{K}_{i,\neq}^{[n-p,n+1)} \cup \mathcal{K}_{i,\neq}^{n+1}.$$

To preserve the conservation properties of the scheme the computed swept areas have to be modified to account for the change in shape of the finite volumes during the swapping phase. This is indicated by the correction term ΔA appearing in Eq. (11). In particular during a swapping operation an edge disappears and a new one is created. To this purpose a continuous deformation of the cells involved in the topology change is used to compute the corrected swept areas ΔA_{ik}^{n+1} according to the procedure proposed by [27].

When an edge i-k is inserted into the grid a new interface between two finite volumes is created at the time t^{n+1}. The size of the integrated normal η_{ik}^m is identically zero for all $m \leq n$, while η_{ik}^{n+1} is computed from the given grid configuration. Therefore for $t^m \leq t^n$ the numerical fluxes are null and for

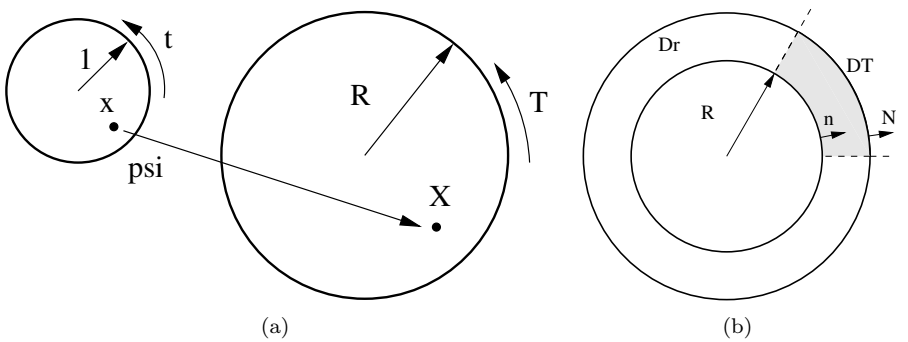


Figure 6: Mapping function between two circles of radius 1 and R. a) Mapping function. b) Area swept by the interface of arc θ from the time t to T.

t^{n+1} they are evaluated as indicated by the third term of the first equation of system (11).

When an edge j-ℓ is removed from the grid at the time t^{n+1}, the corresponding portion of the finite volume interface $\partial\mathcal{C}_{j\ell}$ is deleted. The size of the integrated normal vector is thus set as zero from this point on, i.e. $\eta_{j\ell}^m = 0$ for all $m \geq n+1$. On the contrary the interface velocity $\nu_{j\ell}^{n+1}$ is different than zero due to the area swept during the adaptation phase. Because of this, the numerical flux reduces to the the fourth term of the first equation of system (11), which is given by the ALE contribution only while the contribution of the Euler fluxes is identically zero. To correctly conserve the solution, the contribution to the fluxes associated to a removed edge can be dropped only when the corresponding interface velocity is identically null and this, in turns, depends on the adopted time integration scheme. For example the contribution of an edge j-ℓ removed during the adaptation step occurring in the time lapse between t^n and t^{n+1} can be dropped when $\nu_{j\ell}^m = 0$, that for a backward Euler scheme means $m > n+1$ while for a BDF scheme of order $p+1$ correspond to the condition $m > n+p+1$.

Such remarkable result is a direct consequence of the differential nature of the GCL constraint, i.e. Eq. (7) and (8) do not depend directly on the adopted integration scheme. Indeed the above is also valid in the time-continuous framework, as it is shown in the following example. Let us take as a reference configuration a circular control volume C of unit radius that can be transformed into a circular control volume c of radius variable in time $R(t)$, as shown in figure 6. The mapping between the points of the reference configuration and the transformed one is $\boldsymbol{x} = \boldsymbol{\psi}(\boldsymbol{X}, t) = R(t)\boldsymbol{X}$, i.e. c is a obtained by scaling C. The inverse transformation is $\boldsymbol{X} = \boldsymbol{\Psi}(\boldsymbol{x}, t) = \boldsymbol{x}/R(t)$ and the Jacobian of the mapping is $J(\boldsymbol{X}, t) = R^2(t)$. The velocity of deformation of the element is therefore given by

$$v(\boldsymbol{X}, t) = \frac{d\boldsymbol{\psi}}{dt} = \frac{dR}{dt}\boldsymbol{X} \qquad \text{or} \qquad v(\boldsymbol{x}, t) = v(\boldsymbol{\Psi}(\boldsymbol{x}, t), t) = \frac{dR}{dt}\frac{\boldsymbol{x}}{R}.$$

The integrated interface velocity for a boundary portion $\partial c(t)$, expressed in Eq. (6) becomes

$$\nu(t) = \int_{\partial c(t)} \frac{1}{R}\frac{dR}{dt}(\boldsymbol{x}\cdot\boldsymbol{n})\, ds. \tag{12}$$

For a circumference centered in the origin the product $\boldsymbol{x}\cdot\boldsymbol{n}$ is equal to R since $\boldsymbol{x}$ is always locally aligned with $\boldsymbol{n}$. Using a polar coordinate system one obtains $ds = R\, dR d\vartheta$, therefore Eq. (12) becomes

$$\nu(t) = \int_0^\theta R\frac{dR}{dt}\, d\vartheta = \theta R(t)\frac{dR(t)}{dt} \tag{13}$$

where θ is the angle span of the arc corresponding to ∂c and does not varies with the time.

From Eq. (13) the value of ν depends on both R and dR/dt. If the radius is progressively reduced until a null area is obtained at $t = T$, the value of $\nu(T)$ depends on how quickly R goes to zero with respect to dR/dt, and it is not necessarily zero. For example if the position of points of interface depends linearly on the time,

$$R(t) = R_0\frac{T-t}{T}, \qquad R_0 = R(0) \qquad \text{and} \qquad \nu(t) = -\theta R_0^2\frac{T-t}{T^2}$$

which gives $\nu(T) = 0$. If however the radius is made proportional to the square root of the time, then

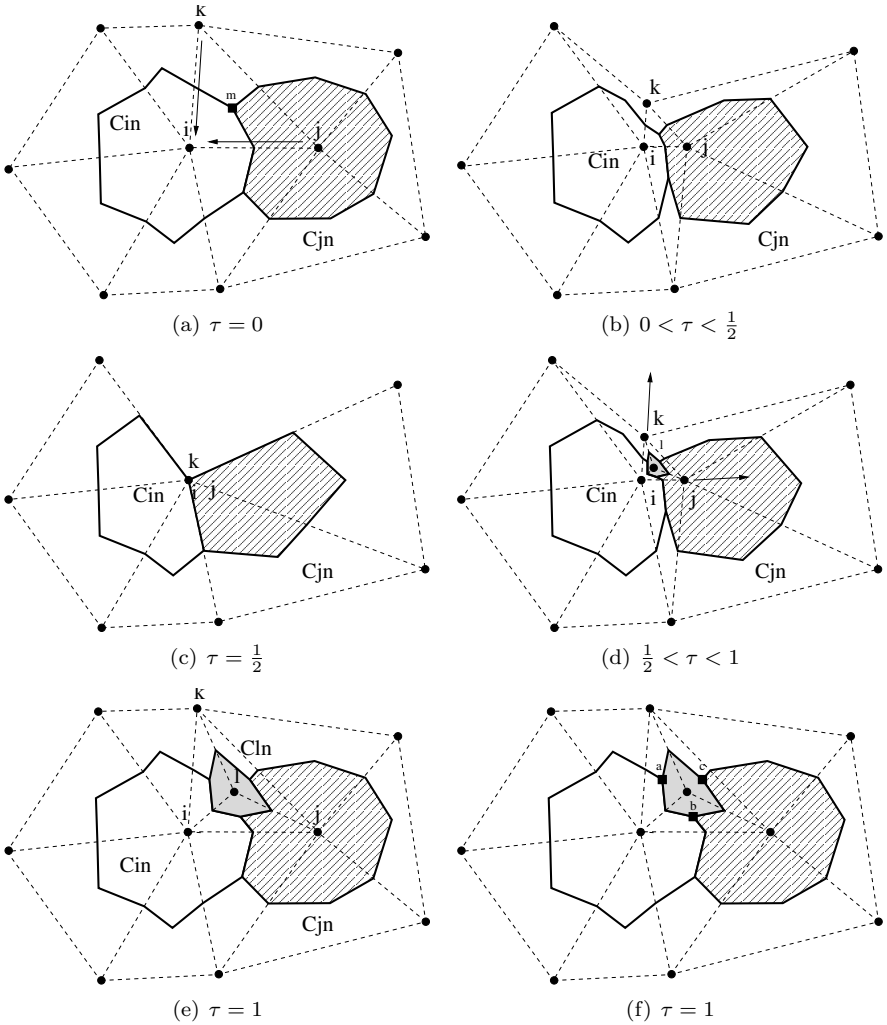
$$R(t) = R_0\sqrt{\frac{T-t}{T}} \qquad \text{gives} \qquad \nu(t) = -\theta\frac{R_0^2}{2T}.$$

As indicated by these two examples even in the continuous framework $\nu(t)$ does not necessarily approach zero as the interface shrinks. Additionally, it is noted that in the second example the area swept by the interface depends linearly on the time, which is consistent with the Backward Euler approximation of equation (8). Additional details can be found in [27].

4.2. ALE scheme with node insertion

The procedure described in section 4.1 to account for the edge-swapping in a conservative way, is now extended to the case of element refinement. Figure 7 illustrates a possible interpretation of the node insertion procedure as a continuous deformation of the finite volumes in the time interval $[t^n, t^{n+1})$. Figure 8 shows the same process for the insertion of a node along the boundary. A three step approach is followed, where: the selected triangle collapses at the location where the new node is to be inserted, the new node is added and connected to the other nodes of the initial triangle, thus generating three new triangles, and finally the triangles expands while the nodes returns to their original locations.

In the collapse phase, from (a) to (c), and in the expansion one, (d) to (f), the area swept by the interfaces is calculated as in system (9). Indeed the

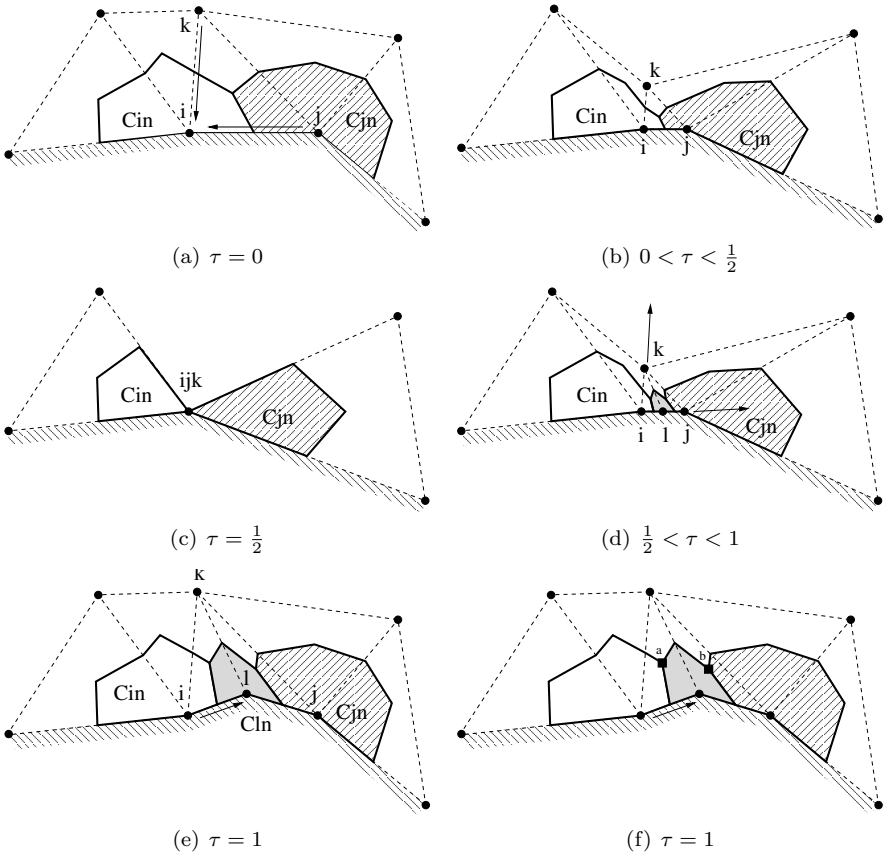


(a) $\tau = 0$ (b) $0 < \tau < \frac{1}{2}$

(c) $\tau = \frac{1}{2}$ (d) $\frac{1}{2} < \tau < 1$

(e) $\tau = 1$ (f) $\tau = 1$

Figure 7: Interpretation of the single node insertion procedure as continuous finite volumes deformation in the non-dimensional time $\tau = (t - t^n)/(t^{n+1} - t^n)$. (a) Initial state. (b) The nodes 2 and 3 collapse over the node 1, the interfaces of the cells $\mathcal{C}_2$ and $\mathcal{C}_3$ sweep a non null area. (c) Intermediate state, the element to be refined has entirely collapsed. Any change of connectivity, due to the refinement, produces no variation of area of the cells. (d) The nodes 2 and 3 return to their original locations. During the expansion step the interfaces of the cells $\mathcal{C}_4$, $\mathcal{C}_2$, and $\mathcal{C}_3$ sweep a non null area. (e) Final state.



(a) $\tau = 0$ (b) $0 < \tau < \frac{1}{2}$

(c) $\tau = \frac{1}{2}$ (d) $\frac{1}{2} < \tau < 1$

(e) $\tau = 1$ (f) $\tau = 1$

Figure 8: Interpretation of the node insertion procedure along the boundary as continuous finite volumes deformation in the non-dimensional time $\tau = (t - t^n)/(t^{n+1} - t^n)$. (a) Initial state. (b) The nodes 2 and 3 collapse at the location where node 1 is inserted, the interfaces of the cells $\mathcal{C}_2$ and $\mathcal{C}_3$ and the boundary interfaces $\partial\mathcal{C}_1 \cap \partial\Omega$ and $\partial\mathcal{C}_2 \cap \partial\Omega$ sweep a non null area. (c) Intermediate state, the element to be refined has entirely collapsed. Any change of connectivity, due to the refinement, produces no variation of area of the cells. (d) The nodes 2 and 3 return to their original locations. During the expansion step the interfaces of the cells $\mathcal{C}_4$, $\mathcal{C}_2$, and $\mathcal{C}_3$ and the boundary interfaces $\partial\mathcal{C}_4 \cap \partial\Omega$, $\partial\mathcal{C}_1 \cap \partial\Omega$ and $\partial\mathcal{C}_2 \cap \partial\Omega$ sweep a non null area. (e) Final state, the node $\boldsymbol{x}_4$ is positioned as prescribed by the continuous shape of the boundary.

grid movement is a simple grid deformation with constant number of nodes and connectivity.

The insertion step has no effects in terms of the satisfaction of the GCL since the involved cells have a zero interface length , see figure 7(c) and 8(c). The overall area swept by the interfaces outside the triangle $1, 2, 3$ is equal to zero and the net numerical flux is therefore null, indeed no changes of shape of those portion of cells occurs. Therefore, the governing equation of a new cell is

$$\begin{cases} a_{-1}\, V_i^{n+1}\, \mathsf{u}_i^{n+1} = \Delta t \sum\limits_{k \in \mathcal{K}_{i,\neq}^{n+1}} \Phi(\mathsf{u}_i, \mathsf{u}_k, \nu_{ik}, \hat{\boldsymbol{\eta}}_{ik}, \eta_{ik})^{n+1}, & i \text{ is new} \\ \\ \alpha_{-1}\left(\Delta V_{ik}^{n+1} + \Delta A_{ik}^{n+1}\right) = \Delta t\, \nu_{ik}^{n+1}, & k \in \mathcal{K}_{i,\neq}^{n+1} \end{cases} \tag{14}$$

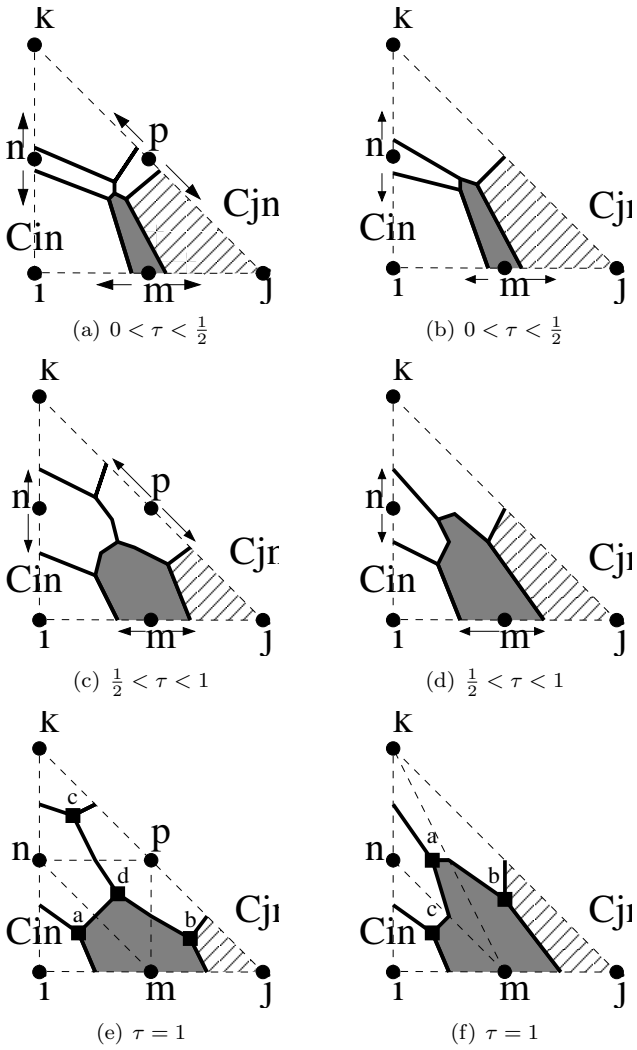
since $V^{n-q} = 0$ and $\Delta V^{n-q} = 0$ for any $q \geq 0$. In Eq. (14) it is apparent that the knowledge of the solution at time levels previous than $n+1$ is not necessary, since the cell area is null, and u^{n+1} is computed by simply integrating the conservation equation for the full ALE system.

A similar procedure is followed in the case of boundary nodes. Figure 8 illustrates how the three step procedure can be adopted to a boundary node. More complex refinement strategies can be also adopted if more than one node is inserted. Figure 9 (left) and 9(right) illustrate a possible way of interpreting in a time-continuous fashion the insertion of three and two vertices, respectively, along the edges of triangle defined by the vertices 1, 2 and 3. In this case the procedure to enforce the GCL does not require the sequence of collapse, insertion and expansion but instead it begins with the insertion of the vertices that lie on the edges of the element and the following expansion of the internal edges. This new procedure described, albeit simpler than the three step one, is in fact complicated by the need of directly assigning the position of the edges midpoint and of the centroid of the elements. Indeed in the three-step procedure only the cells located inside the refined element always respect the definition of dual mesh. This is greatly simplifies the calculation of the ΔA terms which are uniquely defined by the coordinates of the vertices of the triangulation.

4.3. ALE scheme with node deletion

In figure 10 is illustrated the ALE interpretation of the node deletion procedure for a domain cell. Again a three steps procedure is adopted where the elements connecting a node collapse to an arbitrary point, the connectivity is reconstructed and the new elements return to the correct position. As for the node insertion, a three-step procedure is followed. The swept areas of the collapse/expansion phases can be calculated as if the grid is simply deforming, while no ΔA contribution is required during the insertion phase, since the area of the elements is zero.

Therefore, the governing equation for a deleted domain cell, i.e. not consid-



(a) $0 < \tau < \frac{1}{2}$ (b) $0 < \tau < \frac{1}{2}$

(c) $\frac{1}{2} < \tau < 1$ (d) $\frac{1}{2} < \tau < 1$

(e) $\tau = 1$ (f) $\tau = 1$

Figure 9: Interpretation of the triple (left) and double (right) node insertion procedure as continuous finite volumes deformation in the non-dimensional time $\tau = (t - t^n)/(t^{n+1} - t^n)$. (a,c) and (b,d) The points that define the cells $\mathcal{C}_4$, $\mathcal{C}_5$ and $\mathcal{C}_6$ are collapsed over $\boldsymbol{x}_e$, $\boldsymbol{x}_{12}$, $\boldsymbol{x}_{23}$ or $\boldsymbol{x}_{31}$. Specifically, the cell points that at the end of the refinement step are located along the edges that are not inside the refinement pattern, i.e. $\boldsymbol{x}_{26}$, $\boldsymbol{x}_{36}$, $\boldsymbol{x}_{14}$, $\boldsymbol{x}_{42}$, $\boldsymbol{x}_{35}$ and $\boldsymbol{x}_{51}$, move along the edges to which they belong to. Their initial position correspond to the closest midpoint, while the final location is prescribed in the adopted refinement pattern. A linear function of the time is chosen, for example $\boldsymbol{x}_{14}(\tau) = \tau\boldsymbol{x}_{14} - (1-\tau)\boldsymbol{x}_{12}$ and $\boldsymbol{x}_{51}(\tau) = \tau\boldsymbol{x}_{51} - (1-\tau)\boldsymbol{x}_{31}$. Similarly at the initial time the edge midpoints or the centers of mass located inside the pattern are collapsed over $\boldsymbol{x}_e$ and their position is updated as a linear function of the time. For example the motion of the midpoint of the edge 4-5 is given by $\boldsymbol{x}_{45}(\tau) = \tau\boldsymbol{x}_{45} - (1-\tau)\boldsymbol{x}_e$, while the motion of the center of mass of the element of a is computed by $\boldsymbol{x}_a(\tau) = \tau\boldsymbol{x}_a - (1-\tau)\boldsymbol{x}_e$. (e) and (f) final states.

ering the boundary terms, is

$$\begin{cases} \displaystyle\sum_{q=0}^{p} a_q V_i^{n-q}\, \mathsf{u}_i^{n-q} = \Delta t \sum_{k \in \mathcal{K}_{i,\neq}^{[n-p,n+1)}} \Phi(\mathsf{u}_i, \mathsf{u}_k, \nu_{ik}, \hat{\boldsymbol{\eta}}_{ik}, 0)^{n+1}, & i \text{ is removed} \\ \displaystyle\sum_{q=-1}^{p-1} \alpha_q \big(\Delta V_{ik}^{n-q} + \Delta A_{ik}^{n-q}\big) = \Delta t\, \nu_{ik}^{n+1}, & k \in \mathcal{K}_{i,\neq}^{[n-p,n+1)} \end{cases} \tag{15}$$

where the left hand side of the first equation does not depend on u_i^{n+1}, since $V_i^{n+1} = 0$. Note that the summation starts at $q = 0$. The right hand side is given by the ALE contribution to the fluxes of the removed edges and it depends on the solution evaluated at t^{n+1}. As removed edges are associated to additional ALE contribution to the fluxes, removed cells are associated to additional ALE equations.

Integrating the governing equations with an implicit scheme at the time level $n+1$, as in system (9) or (15), requires one to calculate the value of the solution on each removed cell to determine in a conservative manner the solution on all remaining grid nodes. Indeed, given a removed cell $\mathcal{C}_i$ previously adjacent to the cell $\mathcal{C}_k$, the numerical fluxes across $\partial\mathcal{C}_{ik}$ are function of u_i^{n+1}, u_k^{n+1} and ν_{ik}^{n+1}, as given by Eq. (15). It is worth noting that, since the edges that are connected to a deleted node must be removed as well, the second equation of system (15) is defined for every interface that is part of the grid from t^{n-p} to t^{n+1} but it is no longer part of the grid at t^{n+1}.

Eq. (15) represent the balance of the ALE fluxes exchanged by a removed cell and the adjacent ones due to the movement of the interfaces with constant velocity ν_{ik}^{n+1} in the interval $[t^{n-p}, t^{n+1})$ and it is therefore not related to the Euler equations. Moreover Eq. (15) becomes a trivial identity only when $\nu_{ik}^{n+1} = 0,\ \forall k \in \mathcal{K}_{i,\neq}^{[n-p,n+1)}$. Thus $p+1$ time steps after the node removal the equation is erased from the system and the solution on the removed cell is no longer computed.

A similar procedure is used to handle boundary node deletion, as illustrated in figure 11, which includes a collapse and an expansion phase. The main difference is the presence of the boundary interfaces that span a non-negative area during the adaptation procedure, indicated as $\Delta A_{i,\partial}$. Similarly to domain edge, additional ALE fluxes associated to the removed boundary interfaces have to be accounted for.

4.4. Fully discrete ALE scheme for adaptive grids

Introducing the set $\mathcal{K}^{n+1} = \mathcal{K}(t^{n+1})$ of all nodes of the triangulation of Ω at time t^{n+1}, system (9) can be rewritten for grids with variable number of nodes

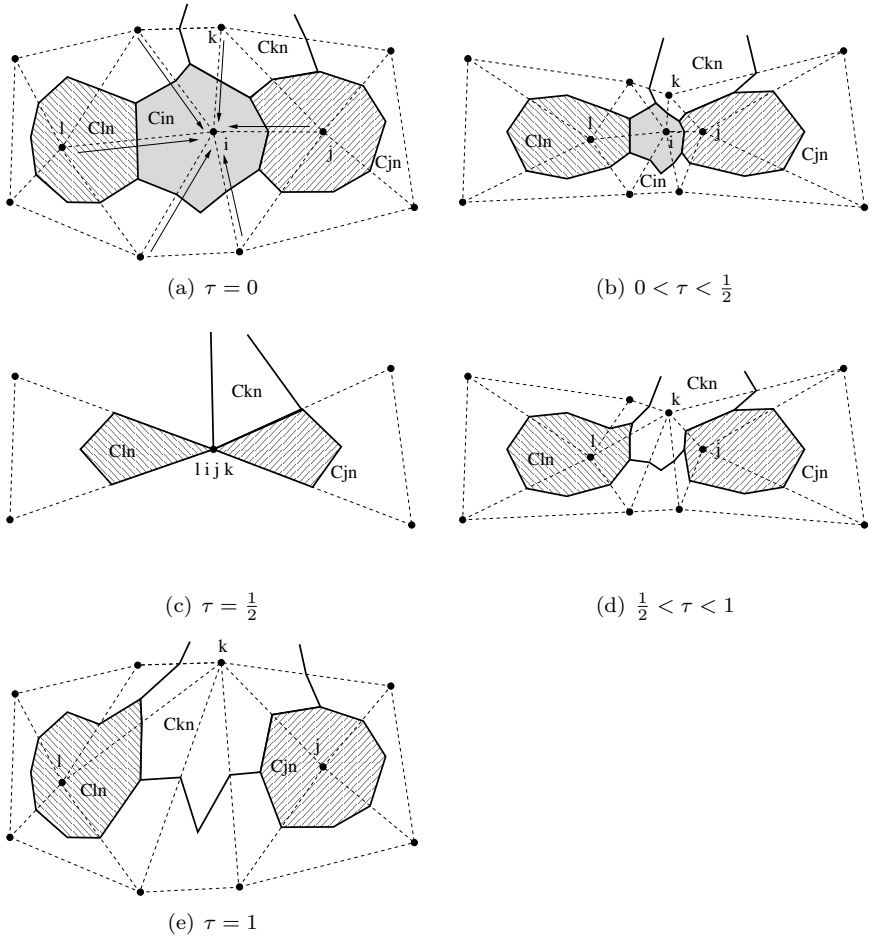


Figure 10: Interpretation of the node deletion procedure as continuous finite volumes deformation in the non-dimensional time $\tau = (t - t^n)/(t^{n+1} - t^n)$. (a) Initial state. (b) The nodes surrounding the node to be removed, i.e. 1, collapses on one location. The interfaces of the cells surrounding $\mathcal{C}_i$, e.g. $\mathcal{C}_2$, $\mathcal{C}_3$, and $\mathcal{C}_4$, sweep a non null area. (c) Intermediate state, bubble of nodes surrounding the node 1 has entirely collapsed. No variation in area of the cells is produced by the node deletion and the successive reconstruction of the local connectivity. (d) The nodes of the bubble return to their original locations. During the expansion step the interfaces of the cells, e.g. $\mathcal{C}_4$, $\mathcal{C}_2$, and $\mathcal{C}_3$ sweep a non null area. (e) Final state, the choice of the new connectivity is arbitrary.

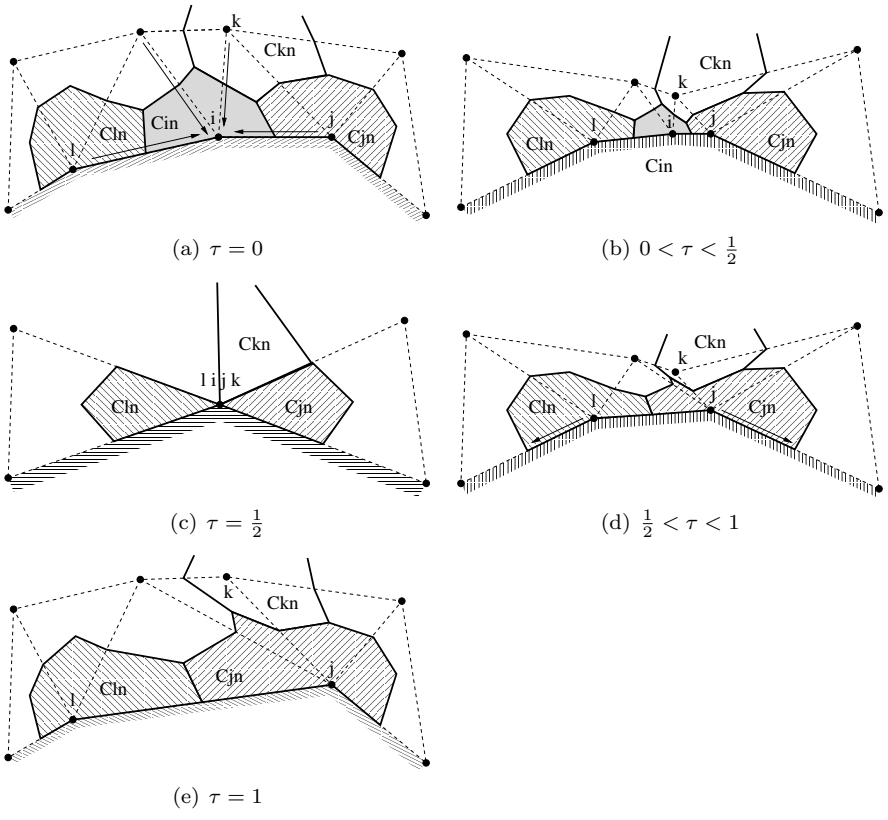


Figure 11: Interpretation of the boundary node deletion procedure as continuous finite volumes deformation in the non-dimensional time $\tau = (t - t^n)/(t^{n+1} - t^n)$. (a) Initial state. (b) The nodes surrounding the node to be removed, i.e. 1, collapses on one location. The interfaces of the cells surrounding $\mathcal{C}_i$, e.g. $\mathcal{C}_2$, $\mathcal{C}_3$, and $\mathcal{C}_4$ and the boundary interfaces $\partial\mathcal{C}_i \cap \partial\Omega$, $\partial\mathcal{C}_4 \cap \partial\Omega$ and $\partial\mathcal{C}_2 \cap \partial\Omega$ sweep a non null area. (c) Intermediate state, bubble of nodes surrounding the node 1 has entirely collapsed. No variation in area of the cells is produced by the node deletion and the successive reconstruction of the local connectivity. (d) The nodes of the bubble return to their original locations. During the expansion step the interfaces of the cells, e.g. $\mathcal{C}_4$, $\mathcal{C}_2$, and $\mathcal{C}_3$ sweep a non null area. Moreover the boundary interfaces $\partial\mathcal{C}_4 \cap \partial\Omega$ and $\partial\mathcal{C}_2 \cap \partial\Omega$ sweep a non null area. (e) Final state, the choice of the new connectivity is arbitrary.

and connectivity as follows

$$
\begin{cases}
\displaystyle\sum_{q=-1}^{p} a_q \frac{V_i^{n-q}\,\mathsf{u}_i^{n-q}}{\Delta t} = \Phi^{\partial}(\mathsf{u}_i, \nu_i, \hat{\boldsymbol{\xi}}_i, \xi_i)^{n+1} + \sum_{k \in \mathcal{K}_{i,\neq}^{n+1}} \Phi(\mathsf{u}_i, \mathsf{u}_k, \nu_{ik}, \hat{\boldsymbol{\eta}}_{ik},)^{n+1} \\
\qquad\qquad + \displaystyle\sum_{k \in \mathcal{K}_{i,\neq}^{[n-p,n+1)}} \Phi(\mathsf{u}_i, \mathsf{u}_k, \nu_{ik}, \hat{\boldsymbol{\eta}}_{ik}, 0)^{n+1}, \quad \left(i \in \mathcal{K}^{n+1}\right) \\
\displaystyle\sum_{q=-1}^{p} a_q \frac{V_j^{n-q}\,\mathsf{u}_j^{n-q}}{\Delta t} = \displaystyle\sum_{k \in \mathcal{K}_{j,\neq}^{[n-p,n+1)}} \Phi(\mathsf{u}_j, \mathsf{u}_k, \nu_{jk}, \hat{\boldsymbol{\eta}}_{jk}, 0)^{n+1} \\
\qquad\qquad + \Phi^{\partial}(\mathsf{u}_j, \nu_j, \hat{\boldsymbol{\xi}}_j, 0)^{n+1}, \quad \left(j \in \mathcal{K}^{[n-p,n+1)}\right) \\
\displaystyle\sum_{q=-1}^{p-1} \alpha_q \left(\Delta V_{ik}^{n-q} + \Delta A_{ik}^{n-q}\right) = \nu_{ik}^{n+1}\, \Delta t, \qquad \left(k \in \mathcal{K}_{i,\neq}^{[n-p,n+1]}\right) \\
\displaystyle\sum_{q=-1}^{p-1} \alpha_q \left(\Delta V_{\ell,\partial}^{n-q} + \Delta A_{\ell,\partial}^{n-q}\right) = \nu_{\ell}^{n+1}\, \Delta t, \qquad \left(\ell \in \mathcal{K}_{\partial}^{[n-p,n+1]}\right)
\end{cases}
\tag{16}
$$

where the set of removed nodes is expressed as

$$\mathcal{K}^{[n-p,n+1)} = \{i \notin \mathcal{K}^{n+1} : \exists k \text{ such that } \nu_{ik}^{n+1} \neq 0\},$$

the set of the removed interfaces as

$$\mathcal{K}_{\partial}^{[n-p,n+1)} = \{\ell \notin \mathcal{K}_{\partial}^{n+1} \quad \text{such that} \quad \nu_{\ell}^{n+1} \neq 0\}$$

$\mathcal{K}_{\partial}^{[n-p,n+1]}$ represent its extension to $\mathcal{K}_{\partial}^{n+1}$. Note that, adopting a BDF scheme of order $p+1$, $\mathcal{K}^{[n-p,n+1)}$ is the set of the nodes removed not before than t^{n-p}.

4.5. Implementation and data structures

Due to the peculiar nature of the proposed approach, the implementation and the associated data structure are not straightforward and are therefore briefly described here for the reader's convenience.

To deal efficiently with mesh adaptation, a *pointer-based* representation of the grid is preferred to the standard *index-based* one. Instead of referring to a grid entity, e.g. element, node or edge, by an index which indicates its position in an array, it is used the address to a chunk of memory which contains the required information, see figure 12. For example an `element` points to its own `nodes` and `edges` and a `node` data-structure contains its coordinates only. Every entity is linked to the prior and following entity in a two-way list, which admittedly does not provide the same efficiency of a random-access one but it requires minimal care when adding or removing an entity from the mesh. For example when a node is deleted it is sufficient to extract it from the list without having to deal with an array with a "hole" in it.

Figure 12: Pointer-based data structure of the mesh. Every entity (`element`,`node`,...) is part of a concatenated list, and the connectivity is expressed as a set of memory address that point to the required data structure. `Elements` are divided into two sections: the mesh, i.e. `nodes` and `edges`, and the discretization, i.e. `cells` and `fluxes`. Every `edge` points to two distinct `nodes`, while every `fluxe` points to two distinct `cells`. `Fluxes`/`cells` associated to edges/ndoes that have been removed from the mesh are flagged as `dead` and highlighted in red.

The second feature of interest is the distinction between mesh and space discretization. The former includes the nodes, the edges and the elements, whereas the latter includes the cells and the numerical fluxes needed by the finite-volume representation of the governing equation. The space discretization is indeed the set of data necessary to compute the residual vector. As in standard finite-volume schemes, for every node there exists a finite volume, or `cell`, with an associated unknown variable. Remarkably enough, the contrary is not always true in the present approach, namely, a given finite volume is not necessary associated to a grid node. Indeed, as described in section 4.4, the governing equation associated to a deleted node must be kept for N_p additional steps, where N_p is the order of the BDF. For this reason the `cell` type has two members: an index used to locate the position of the equation in the right hand side array and a status variable to flags the cell as active/removed. A very similar philosophy has been followed to create the `flux` type which represent the numerical flux exchanged by two cells connected with an edge. When an edge is removed the corresponding `flux` is extracted from the list only after N_p steps, when its contribution to the fluxes is no longer required to satisfy the GCL.

The presented approach allows maximum flexibility, which is required to treat adaptive meshes where the number of governing equations does not correspond to the number of nodes. However the separation between space discretization and mesh is above all a conceptual one, which could be implemented in different ways. The major drawback of the illustrated strategy is the inherent inefficiency of non random-access data structures, however more efficient implementations are possible and currently under investigation.

5. Numerical results

Two test cases are considered to assess the proposed approach in connection with two-dimensional flows of interest for rotary-wing aircraft. These are the simple translating airfoil in section 5.1 and the oscillating airfoil in section 5.2. In both cases, the solution is computed in the airfoil and in the laboratory reference to check the consistency of the dynamic grid algorithm. In particular, for the case of a translating airfoil, the solution is time-independent (steady problem) in the airfoil reference and time-dependent (unsteady problem) in the laboratory reference.

5.1. Translating Airfoil

In the present section the adaptation scheme described above is tested on a very simple problem, i.e. a NACA 0012 airfoil translating at constant speed for 200 chords inside the fixed domain depicted in figure 13. In figure 13, $a = 240\,c$ and $b = 40\,c$. The corresponding free stream Mach number is 0.755 and the angle of attack is 1.175°. The governing equations are integrated using a backward Euler scheme; a time step of 0.115 has been adopted and the motion is completed in 2 000 steps. Two different initial grids are adopted: a coarse

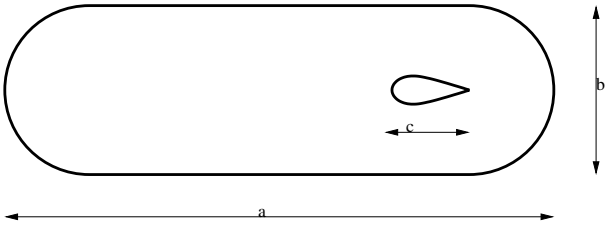


Figure 13: Computational domain for the translating NACA 0012 airfoil test case. The airfoil is displaced for 200 chords toward the left. The corresponding free stream Mach number is 0.755 with an angle of attack of 1.175°.

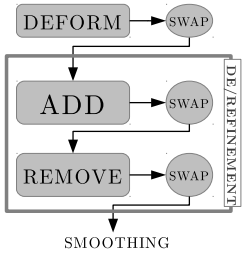
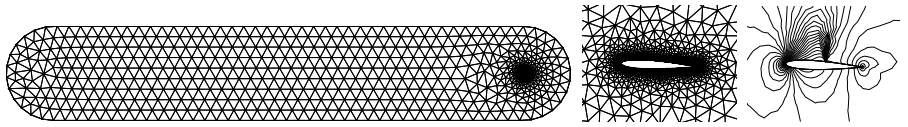
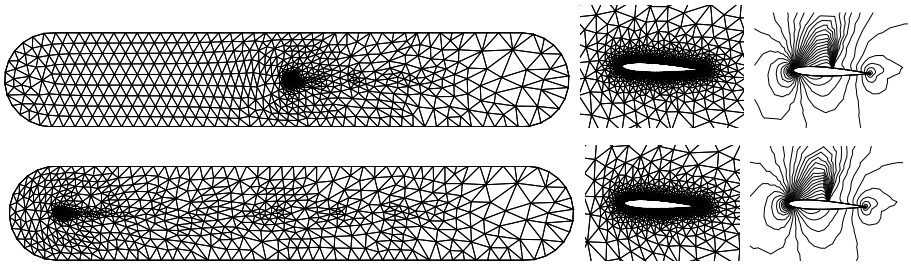


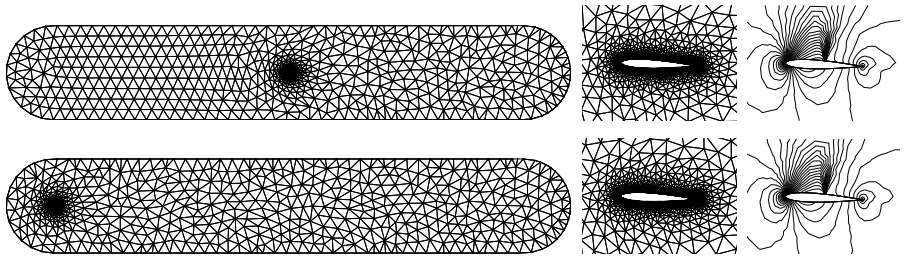
Figure 14: Fully adaptive scheme used in the translating NACA 0012 case.



(a) Initial configuration

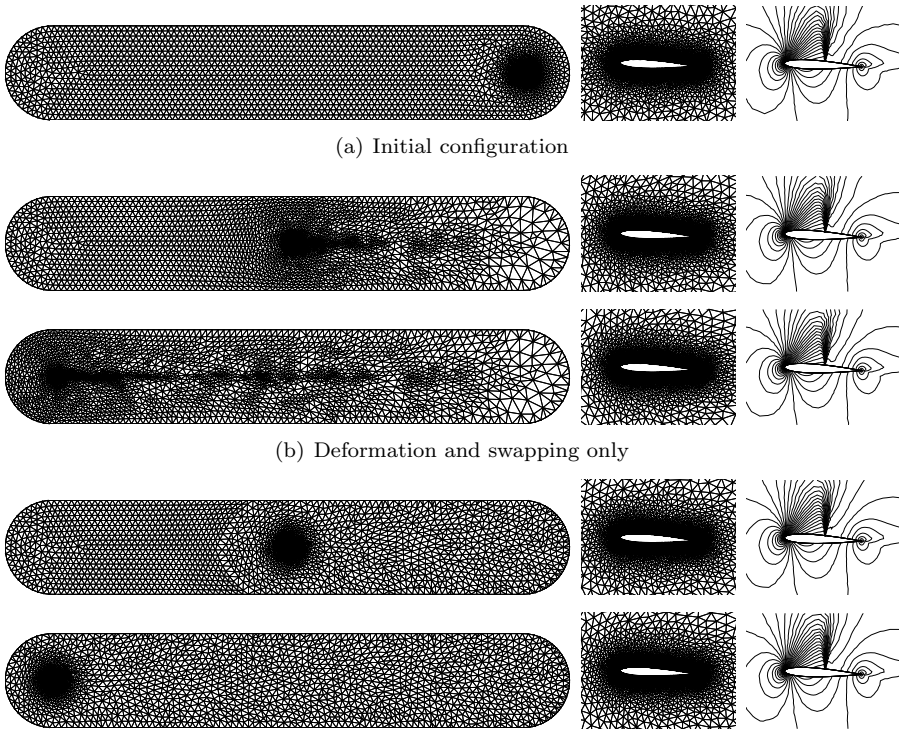


(b) Deformation and swapping only



(c) Fully adaptive

Figure 15: Computational grid with close up and Mach number contour lines for the translating NACA 0012 airfoil test case over a coarse grid. (a) Initial condition, 3 097 vertices and 5 675 elements. (b) Deformation and swapping scheme with intermediate configuration, with the airfoil at $x/c = 100$, and final condition, at $x/c = 200$. In the right column the Mach contour lines range from 0 to 1.5 with a spacing of 0.03. The swapping procedure has been carried out every step. (c) Fully adaptive scheme with intermediate (3 101 vertices and 5 663 elements) and final condition (3 108 vertices and 5 664 elements). The adaptation procedure has been carried out every step.



(a) Initial configuration

(b) Deformation and swapping only

(c) Fully adaptive

Figure 16: Computational grid with close up and Mach number contour lines for the translating NACA 0012 airfoil test case over a fine grid. (a) Initial condition, 10 870 vertices and 20 706 elements. (b) Deformation and swapping scheme with intermediate configuration, with the airfoil at $x/c = 100$, and final condition, at $x/c = 200$. In the right column the Mach contour lines range from 0 to 1.5 with a spacing of 0.03. The swapping procedure has been carried out every step. (c) Fully adaptive scheme with intermediate (10 685 vertices and 20 311 elements) and final condition (10 569 vertices and 20 071 elements). The adaptation procedure has been carried out every step.

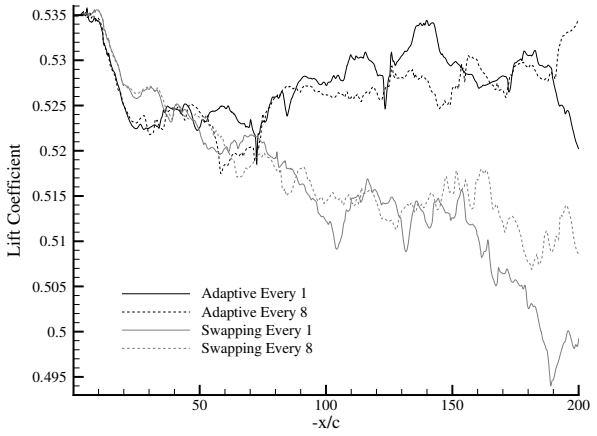


Figure 17: Lift coefficient versus position of the nose of the airfoil for the translating airfoil problem, coarse grid. Both the full adaptive and the deformation and swapping schemes are shown. The results obtained carrying out the adaptation procedure every one and eight iterations are also compared.

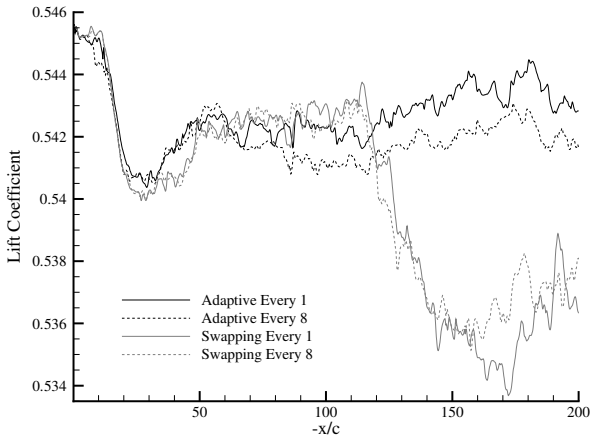


Figure 18: Lift coefficient versus position of the nose of the airfoil for the translating airfoil problem, fine grid. Both the full adaptive and the deformation and swapping schemes are shown. The results obtained carrying out the adaptation procedure every one and eight iterations are also compared.

boundary	$c_{1,b}$	$c_{2,b}$	$c_{3,b}$	ε
airfoil	$4.81 \cdot 10^{-6}$	$9.61 \cdot 10^{-4}$	$4.80 \cdot 10^{-2}$	0.3
far-field	10.82	0	0	0.3

Table 1: Coefficients used to describe the parabolic imposed area function for the case of the translating/pitching airfoil on the coarse mesh. In the last column the unique refinement/derefinement tolerance is indicated.

boundary	$c_{1,b}$	$c_{2,b}$	$c_{3,b}$	ε
airfoil	$9.10 \cdot 10^{-6}$	$6.60 \cdot 10^{-4}$	$1.98 \cdot 10^{-2}$	0.3
far-field	2.70	0	0	0.3

Table 2: Coefficients used to describe the parabolic imposed area function for the case of the translating/pitching airfoil on the fine mesh. In the last column the unique refinement/derefinement tolerance is indicated.

and a fine one. The coarse one has 3 101 vertices and a node spacing along the airfoil of $5 \cdot 10^{-3}c$, where c is the chord, and of $5c$ along the far field boundary. The fine one has 10 870 vertices and an halved grid spacing. The corresponding free stream Courant number is roughly 50 for the coarse mesh and 100 in the fine one.

First a steady state solution is computed by imposing a Mach 0.755 free-stream around the fixed airfoil. The initial grid and the corresponding Mach contour lines are shown in figure 15(a) and 16(a). The computed lift coefficient is 0.53487 in the first case and 0.54553 in the second one.

The distance-based function, described in section 2, is tuned to preserve the initial distribution shown in figure 15(a) and figure 16(a). The values of the coefficients describing $\mathcal{A}$ are shown in table 1 and 2 and the minimum edge size is thus $5 \cdot 10^{-3}$ for the coarse mesh and $2.5 \cdot 10^{-3}$ for the fine one. Two different adaptation procedures are used to perform the movement. The first resorts to grid deformation, swapping and smoothing only. The second procedure includes also the grid refinement and coarsening phases described in section 2 followed by a swapping cycle. The full adaptive scheme is shown in figure 14, while in the case of the grid deformation and swapping scheme the de/refinement phase is skipped. To better preserve the initial node spacing while increasing the quality, the Laplacian smoothing is applied only to the nodes that belong to the elements of a swapped edge.

In the coarse case, the computational grids and Mach number contour lines obtained with the airfoil at $x/c = 100$ and at $x/c = 200$ are shown in figure 15(b) and 15(c). After a translation of 200 chords the quality of the grid obtained with the adaptation scheme made of swapping and deformation only is very high close to the airfoil, but the same does not hold in the free stream region. Indeed, the elements in front of the airfoil are deformed due to the movement of the airfoil. The fully adaptive scheme, instead, allows to to maintain both the quality and the spacing of the grid during the entire motion.

Similar results can be obtained if the grid spacing is halved, as shown in

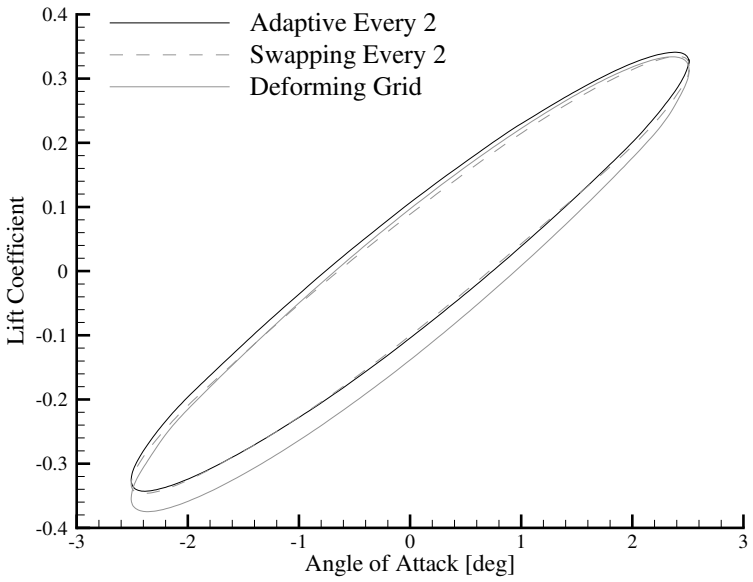


Figure 19: Lift coefficient versus angle of attack computed in the last period of oscillation for the pitching airfoil problem, coarse grid. The pitch reduced frequency is 0.1628. The solution obtained with swapping and deformation and the one obtained with the fully adaptive scheme are compared with a reference solution. The reference solution is computed over the initial fixed grid by imposing the corresponding free stream and deforming the grid due to the variation of the pitch angle only.

figure 16(b) and 16(c). Grid deformation and swapping allows one to obtain large displacement with a reasonable level of grid quality, while node insertion and removal maintain the spacing and improve the overall regularity.

The beneficial effect of the use of grid adaptation is apparent in the lift coefficient versus displacement curve shown in figure 17 and 18. During the first 25 chords of the movement the value of C_L drops quite abruptly with both the coarse and the fine grids and both adaptation procedures. Indeed in the first phase of the motion the elements are distorted but only few of them are swapped or refined/derefined due to the relatively high-level of quality. Once a reasonable decrease in quality is reached the adaptation procedure is triggered and the losses in terms of C_L are reduced. At the end of the motion, the lift coefficient computed with the fully adaptive scheme is closer to the initial value of C_L than the one obtained with the swapping only, for both the coarse and the fine grids. Applying the swapping or the adaptation procedure every 8 steps, instead of every step, does not effect significantly the solution in terms of lift coefficient.

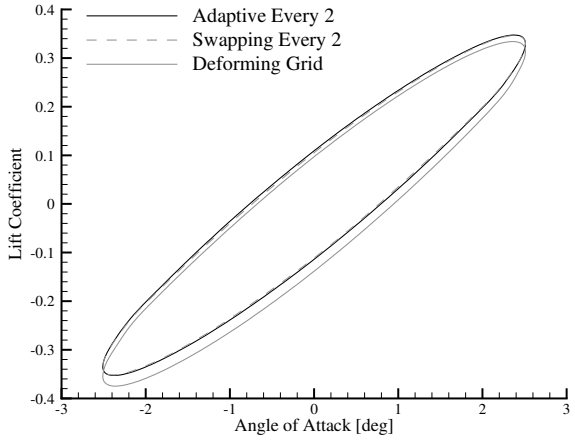


Figure 20: Lift coefficient versus angle of attack computed in the last period of oscillation for the pitching airfoil problem, fine grid. The pitch reduced frequency is 0.1628. The solution obtained with swapping and deformation and the one obtained with the fully adaptive scheme are compared with a reference solution. The reference solution is computed over the initial fixed grid by imposing the corresponding free stream and deforming the grid due to the variation of the pitch angle only.

5.2. Pitching Airfoil

In the present section compressible inviscid transonic flow computations around a pitching NACA 0012 airfoil are presented. The variation of the angle of attack in time is prescribed analytically as a sinusoidal function, namely

$$\alpha(t) = \alpha_0 + \bar{\alpha}\cos(\omega t) \tag{17}$$

where $\alpha_0 = 1.175°$, $\Delta\alpha = 2.51°$ and $\omega = 0.1628$ is the dimensionless reduced frequency, obtained scaling the dimensional quantity with the asymptotic velocity u_∞ and the chord c. The angle of attack of the airfoil with respect to the asymptotic flow is changed by rotating the airfoil around the nose. The airfoil is simultaneously translated forward as already described in section 5.1.

Similarly to what observed in the simple translation case the grid quality reduces with the displacement. The average of the element quality is shown in figure 21 and 20, for the fine and the coarse grid. In both cases the use of the node insertion and deletion procedures reduces the loss of quality. However, this beneficial effect is only mildly reflected in the solution, as shown by the C_L-α curve of figure 19 and 20. A reference solution is first computed over the initial, fixed grid by imposing the corresponding free stream. Only the pitch angle of the airfoil is varied and the grid is deformed accordingly. The adaptation procedure have been carried out every other step. In both the coarse and the fine case the curves obtained with either adaptation procedure are fairly overlapping, and a general increase in C_L with respect to the reference solution is featured.

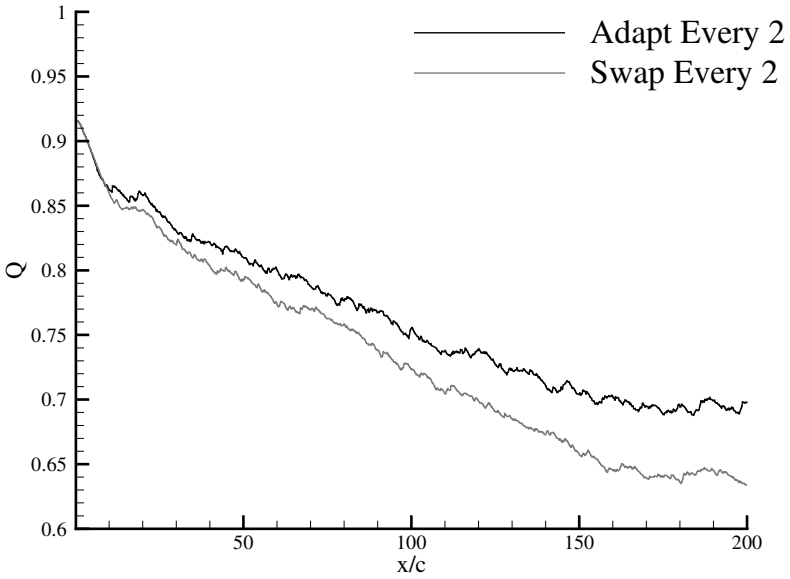


Figure 21: Variation of the average element quality with respect to the airfoil displacement in the pitching airfoil case, coarse grid. For both the coarse and the fine grid resorting to the fully adaptive scheme reduces the loss in overall quality with respect to the case of swapping and deformation only. The quality index is the one defined in ref [27].

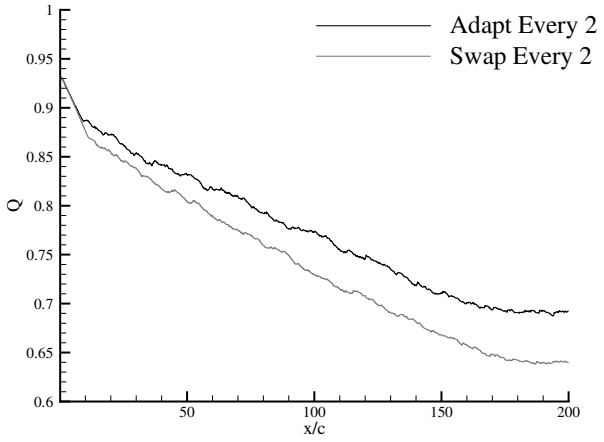


Figure 22: Variation of the average element quality with respect to the airfoil displacement in the pitching airfoil case, fine grid. For both the coarse and the fine grid resorting to the fully adaptive scheme reduces the loss in overall quality with respect to the case of swapping and deformation only. The quality index is the one defined in ref [27].

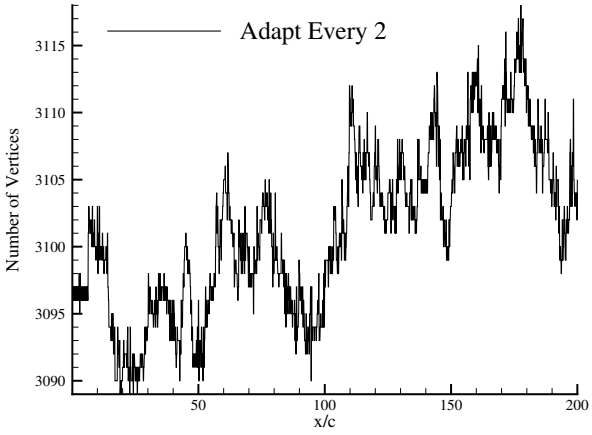


Figure 23: Variation of the total number of nodes with respect to the airfoil displacement, coarse g.

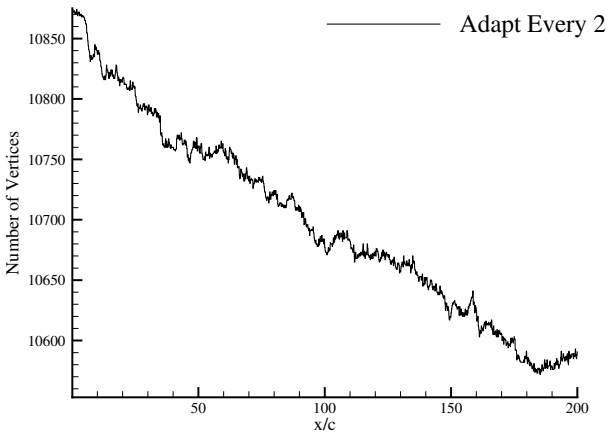


Figure 24: Variation of the total number of nodes with respect to the airfoil displacement, fine grid.

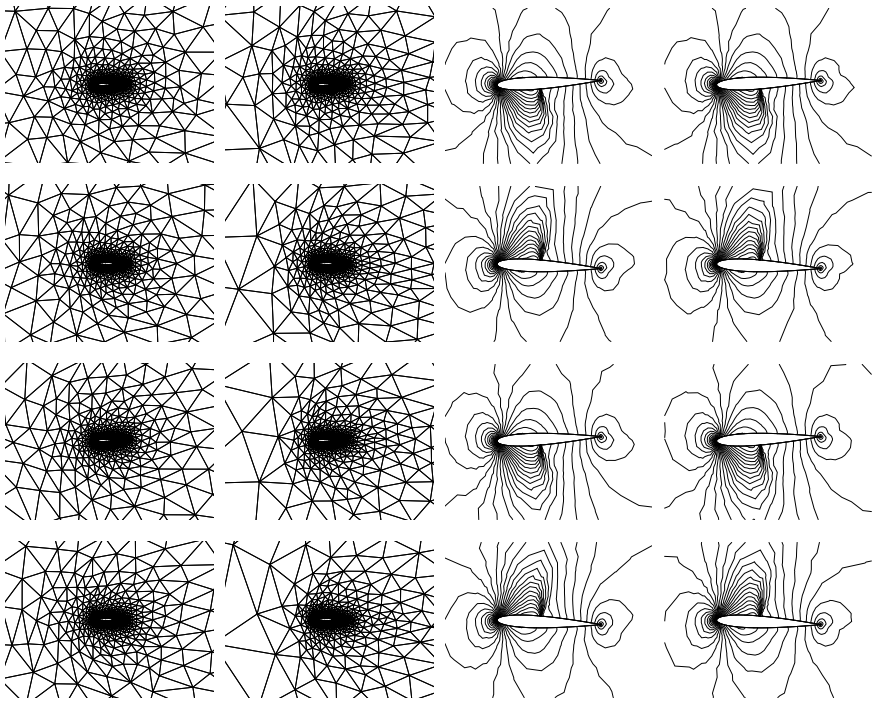


Figure 25: Comparison between the grid and the Mach number distribution obtained with the fully adaptive scheme (on the left) and the swapping plus deformation scheme (on the right) in the coarse grid case. The first row corresponds to $x/c = 50$, the second row to $x/c = 100$, the third to to $x/c = 150$ and the forth to $x/c = 200$.

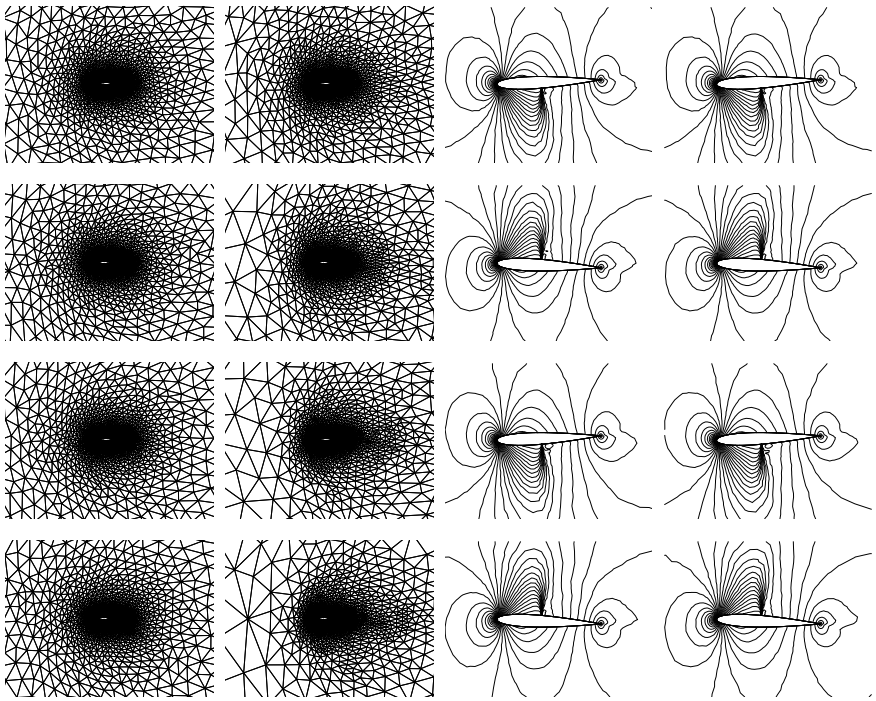


Figure 26: Comparison between the grid and the Mach number distribution obtained with the fully adaptive scheme (on the left) and the swapping plus deformation scheme (on the right) in the fine grid case. The first row corresponds to $x/c = 50$, the second row to $x/c = 100$, the third to to $x/c = 150$ and the forth to $x/c = 200$.

6. Conclusion

The paper presents a finite-volume approach for the ALE formulation of the Euler equations for two dimensional adaptive grids. The grid adaptation is performed by a combination of nodes displacement and local topology modifications, i.e. edge-swapping and node insertion/deletion. It is shown how the local modifications of the grid can be recast in terms of continuous deformations of the finite-volumes generated from the dual grid. As a result, the value of the local grid velocities is modified, while the area of the cells in the old and new configuration is described by the respective grids. The solution over the new adapted is calculated integrating the ALE equations.

Preliminary results for simple two-dimensional problems where large deformations occur were presented. The steady flow around an airfoil translating 200 chords inside a fixed domain was computed with both the fully adaptive scheme and the one that uses only edge swapping. The comparison between the two highlights how an overall higher grid quality can be obtained with the former scheme and this is also reflected on the computed lift coefficient. The unsteady flow around the same airfoil translating and pitching inside the fixed domain was also investigated with the same schemes. The overall increase in the grid quality that is obtained with the fully adaptive scheme is only mildly reflected in the C_L-α curve, if compared to the solution computed in the airfoil frame of reference.

References

[1] P. Chadwick, Continuum mechanics: concise theory and problems, Dover books on physics, Dover Publications, 1999.

[2] C. Hirt, A. Amsden, J. Cook, An Arbitrary Lagrangian-Eulerian computing method for all flow speeds, J. Comput. Phys. 14 (3) (1974) 227–253.

[3] J. Donea, Arbitrary Lagrangian–Eulerian finite element methods, in: T. Belytschko, T. J. Hughes (Eds.), Computational Methods for Transient Analysis, Elsevier Science Publisher, Amsterdam, The Netherlands, 1983, Ch. 10, pp. 474–516.

[4] M. Kucharik, Arbitrary Lagrangian-Eulerian (ALE) methods in plasma physics, Ph.D. thesis, Czech Technical University in Prague (2006).

[5] A. Winslow, Equipotential zoning of two-dimensional meshes, Tech. Rep. UCRL-7312, California Univ., Livermore (USA). Lawrence Livermore Lab. (1963).

[6] P. Váchal, R. V. Garimella, M. J. Shashkov, Untangling of 2D meshes in ALE simulations, J. Comput. Phys 196 (2004) 627–644.

[7] M. Kucharik, M. Shashkov, B. Wendroff, An efficient linearity-and-bound-preserving remapping method, J. Comput. Phys. 188 (2003) 462–471.

[8] R. Garimella, M. Kucharik, M. Shashkov, An efficient linearity and bound preserving conservative interpolation (remapping) on polyhedral meshes, Comp. Fluids 36 (2) (2007) 224 – 237.

[9] P. Váchal, R. Liska, Sequential flux-corrected remapping for ALE methods, in: Numerical Mathematics and Advanced Applications, Springer Berlin Heidelberg, 2006, pp. 671–679.

[10] J. Waltz, N. R. Morgan, T. R. Canfield, M. R. J. Charest, L. D. Risinger, J. G. Wohlbier, A three-dimensional finite element arbitrary LagrangianEulerian method for shock hydrodynamics on unstructured grids, Comput. Fluids, 92 (2014), 172–187.

[11] D. J. Mavriplis, Z. Yang, Construction of the discrete geometric conservation law for high-order time-accurate simulations on dynamic meshes, J. Comput. Phys. 213 (2006) 557–573.

[12] P. Geuzaine, C. Grandmont, C. Farhat, Design and analysis of ALE schemes with provable second-order time-accuracy for inviscid and viscous flow simulations, J. Comput. Phys. 191 (2003) 206–227.

[13] A.J. Barlow, ALE and AMR mesh refinement techniques for multi-material hydrodynamics problems, ICFD Workshop on Mesh Refinement Techniques 7th December (2005).

[14] A.J. Barlow, Challenges and recent progress in developing numerical methods for multi-material ALE hydrocodes, ICFD 25 Year Anniversary Conference, Oxford University, 15–16th September (2008).

[15] M. Berndt, J. Breil, S. Galera, M. Kucharik, P.-H. Maire, M. Shashkov, M., Two-step hybrid conservative remapping for multimaterial arbitrary LagrangianEulerian methods., J. Comput. Phys. 230, 66646687 (2011).

[16] C. Michler, H. D. Sterck, H. Deconinck, An Arbitrary Lagrangian Eulerian formulation for residual distribution schemes on moving grids, Comp. Fluids 32 (1) (2003) 59 – 71.

[17] F. Nobile, Numerical approximation of fluid–structure interaction problems with application to hemodynamics, Ph.D. thesis, École Polytechnique Fédérale de Lausanne, Switzerland (September 2001).

[18] C. Farhat, P. Geuzaine, Design and analysis of robust ALE time-integrators for the solution of unsteady flow problems on moving grids, Comp. Meth. Appl. Mech. Engng. 193 (2004) 4073–4095.

[19] D. Mavriplis, Z. Yang, Construction of the discrete geometric conservation law for high-order time-accurate simulations on dynamic meshes, J. Comput. Phys. 213 (3) (2006) 557–573, 2006.

[20] T. Fanion, M. A. Fernández, P. Le Tallec, Deriving adequate formulations for fluid-structure interaction problems: from ALE to transpiration, Rev. Européenne Élém. Finis 9 (6–7) (2000) 681–708.

[21] P. D. Thomas, C. K. Lombard, Geometric conservation law and its application to flow computations on moving grids, AIAA J. 17 (1979) 1030–1037.

[22] W. Shyy, R. W. Smith, H. S. Udaykumar, M. M. Rao, Computational Fluid Dynamics with Moving Boundaries, Taylor & Francis, Inc., Bristol, PA, USA, 1996.

[23] S. Piperno, C. Farhat, Partitioned procedure for the transient solution of coupled aeroelastic problems. Part II: Energy transfer analysis and three dimensional applications, Comp. Meth. Appl. Mech. Engng. 190 (2001) 3147–3170.

[24] S. Etienne, A. Garon, D. Pelletier, Perspective on the geometric conservation law and finite element methods for ALE simulations of incompressible flow, J. Comput. Phys. 228 (7) (2009) 2313 – 2333.

[25] L. Freitag, C. Ollivier-Gooch, Tetrahedral mesh improvement using swapping and smoothing, Int. J. Numer. Anal. Met. 40 (21) (1997) 3979–4002.

[26] P. Frey, P. George, Mesh Generation: Application to Finite Elements, Hermes Science, Paris, 2000.

[27] A. Guardone, D. Isola, G. Quaranta, Arbitrary Lagrangian Eulerian formulation for two-dimensional flows using dynamic meshes with edge swapping, J. Comput. Phys. 230 (20) (2011) 7706–7722.

[28] O. Hassan, K. Morgan, N. P. Weatherill, Unstructured mesh methods for the solution of the unsteady compressible flow equations with moving boundary components, Phil. Trans. R. Soc. Lond. A 365 (2007) 2531–2552.

[29] R. Wang, P. Keast, P. Muir, A comparison of adaptive software for 1D parabolic PDEs, J. Comput. Appl. Math. 169 (1) (2004) 127–150.

[30] R. Abgrall, How to prevent pressure oscillations in mulicomponent flow calculations: A quasi conservative approach, J. Comput. Phys. 125 (1996) 150160.

[31] R. Saurel, R. Abgrall, A simple method for compressible multifluid flows, SIAM J. Sci. Comput. 21 (3) (1999) 11151145.

[32] P. Jenny, B. Mu ller, H. Thomann, correction of conservative Euler solvers for gas mixtures, J. Comput. Phys. 132 (1997) 91107.

[33] C. Hirt, A. Amsden, J. Cook, An arbitrary Lagrangian-Eulerian computing method for all speed flows, J. Comput. Phys. 14 (3) (1974) 227253.

[34] R. Anderson, N. Elliott, R. Pember, A dynamically adaptive arbitrary Lagrangian-Eulerian method for solution of the Euler equations, Tech. Rep. Research Report UCRL-JC-151904, Lawrence Livermore Laboratory National Laboratory (2003).

[35] E. Godlewski, P. A. Raviart, Numerical approximation of hyperbolic systems of conservation laws, Springer-Verlag, New York, 1994.

[36] L. Formaggia, F. Nobile, A stability analysis for the Arbitrary Lagrangian Eulerian formulation with finite elements, East-West J. Num. Math. 7 (1999) 105–132.

[37] V. Selmin, The node-centred finite volume approach: bridge between finite differences and finite elements, Comp. Meth. Appl. Mech. Engng. 102 (1993) 107–138.

[38] R. J. LeVeque, Finite volume methods for conservation laws and hyperbolic systems, Cambridge University Press, 2002.

[39] B. van Leer, Towards the ultimate conservative difference scheme II. Monotonicity and conservation combined in a second order scheme, J. Comput. Phys. 14 (1974) 361–370.

[40] P. L. Roe, Approximate Riemann solvers, parameter vectors, and difference schemes, J. Comput. Phys. 43 (1981) 357–372.

[41] N. P. Weatherill, O. Hassan, M. Marchant, D. Marcum, Adaptive inviscid solutions for aerospace geometries on efficiently generated unstructured tetrahedral meshes, in: AIAA Paper 93-3390-CP, 11th AIAA Computational Fluid Dynamics Conference, Orlando, FL, 1993.

[42] A. Guardone, L. Quartapelle, Spatially factorized Galerkin and Taylor-Galerkin schemes for multidimensional conservation laws, Scientific Report DIA-SR 00-18, Politecnico di Milano, Italy (2000).

[43] V. Selmin, The node-centred finite volume approach: bridge between finite differences and finite elements, Comp. Meth. Appl. Mech. Engng. 102 (1) (1993) 107–138.

[44] D. Isola, An interpolation-free two-dimensional conservative ALE scheme over adaptive unstructured grids for rotorcraft aerodynamics, Ph.D. thesis, Politecnico di Milano (2012), `www.politesi.polimi.it`.

[45] J. D. Lambert, Numerical Methods for Ordinary Differential Systems: the Initial Value Problem, John Wiley & Sons, Chichester, UK, 1991.

[46] C. Farhat, P. Geuzaine, C. Crandmont, The discrete geometric conservation law and the nonlinear stability of ALE schemes for the solution of flow problems on moving grids, J. Comput. Phys. 174 (2001) 669–694.

[47] V. Venkatakrishnan, D. J. Mavriplis, Implicit method for the computation of unsteady flows on unstructured grids, J. Comput. Phys. 127 (1996) 380–397.

[48] L. Margolin, M. Shashkov, Second-order sign-preserving conservative interpolation (remapping) on general grids, J. Comput. Phys. 184 (2003) 266–298.

[49] R. Loubère, P. H. Maire, M. Shashkov, J. Breil, S. Galera, Reale: A reconnection-based Arbitrary-Lagrangian-Eulerian method, J. Comput. Phys. 229 (2010) 4724–4761.

[50] L. Cavagna, G. Quaranta, P. Mantegazza, Application of Navier-Stokes simulations for aeroelastic assessment in transonic regime, Comp. Struct. 85 (11-14) (2007) 818–832.

[51] J. F. Thompson, Numerical grid generation: foundations and applications, Elsevier Science Publishing, 1985.

[52] L. A. Freitag, C. Ollivier-Gooch, Tetrahedral mesh improvement using swapping and smoothing, Int. J. Numer. Meth. Eng. 40 (21) (1997) 3979–4002.

[53] J. Shewchuk, What is a good linear element? Interpolation, conditioning, and quality measures, in: 11^{th} International Meshing Roundtable, Ithaca, NY, 2002, pp. 115–126.

[54] M. Fossati, A. Guardone, L. Vigevano, A node-pair finite element / finite volume mesh adaptation technique for compressible flows, in: 40th AIAA Fluid Dynamics Conference and Exhibit, 2010.