

Predictive mesh adaptation for unsteady 2D Euler simulations

A. Vacca*, R. Guglielmi*, A. Rausa* and B. Re*

Corresponding author: alberto.vacca@mail.polimi.it, barbara.re@polimi.it

* Department of Aerospace Science and Technology, Politecnico di Milano, Italy.

Abstract: This work investigates a novel predictive strategy for unsteady mesh adaptation. Standard adaptation procedures typically require frequent remeshing to prevent the mesh from lagging behind the evolving flow solution. The proposed approach instead relies on computer-vision algorithms to reduce the remeshing frequency while preserving solution accuracy. The method is based on computing a metric function that represents the target grid spacing at the latest available solution time step and at a previous one. The displacement of the relevant flow features between these two time levels is then identified, and the metric function is advected to predict the future location of such features over subsequent time steps. The predicted metrics are finally combined into a single metric field, which is used to drive mesh adaptation. Three test cases are considered to assess the proposed methodology through *a posteriori* analyses performed outside the CFD simulation environment: a one-dimensional piston problem, a cylindrical shock without obstacles, and a cylindrical shock interacting with an obstacle. Although the complete prediction workflow still has to be integrated and tested within a CFD solver to assess its effectiveness in fully coupled simulations, the results presented in this work demonstrate the potential of the proposed approach. They also indicate promising extensions toward more complex scenarios, including anisotropic adaptation, three-dimensional simulations, and unsteady Navier–Stokes computations with turbulence modeling.

Keywords: Mesh adaptation criteria, Unstructured mesh, Error estimation, Computer Vision.

1. Introduction

In Computational Fluid Dynamics (CFD), increasingly detailed and accurate simulations are required to address flows of growing complexity, driven by both scientific interest and industrial demand. The computational mesh plays a central role in determining the accuracy and reliability of a simulation, since an inadequate mesh may lead to inaccurate or even physically incorrect numerical solutions. To accelerate mesh generation, improve solution quality, and reduce the need for manual intervention, mesh adaptation techniques have been extensively investigated. The objective of mesh adaptation is twofold: on the one hand, it provides control over the desired accuracy of the numerical solution; on the other hand, it reduces the overall computational cost. The underlying idea is to relate the local grid spacing to the relevant flow features through suitable indicators, so that the mesh is automatically refined in critical regions without requiring *a priori* knowledge of the solution. Intuitively, small cells are required near steep gradients or discontinuities to accurately resolve the flow, whereas larger cells are sufficient in smooth regions, where they improve computational efficiency. To drive the adaptation process, suitable indicators are therefore needed to define the target grid spacing, namely the spatial distribution of the desired element size, as a function of the flow features. Several techniques have been developed for steady simulations [1–4].

In unsteady simulations, flow structures evolve and move in time, making grid adaptation particularly important for computational efficiency [5]. Standard approaches aim to update the mesh at every time step [6–8], or at least very frequently [9], to accurately track the temporal evolution of the flow. If this is not done, the mesh may lag behind the solution and progressively lose its ability to resolve the moving features. Frequent remeshing is therefore the most direct and intuitive strategy. However, it is not only computationally expensive, but may also reduce solution accuracy because of interpolation errors: transferring the solution from one mesh to another can introduce numerical diffusion and degrade the

overall performance of the method [10]. An alternative strategy consists of avoiding explicit interpolation between different grids by using Arbitrary Lagrangian–Eulerian (ALE) techniques [6]. Nevertheless, this approach remains computationally demanding and still requires frequent local remeshing.

Since, unlike in the steady case, efficiency is often more critical than strict optimality in unsteady simulations, alternative strategies have been proposed to reduce the remeshing frequency. In RANS/LES simulations of high-Reynolds-number flows, the behaviour of turbulence models depends on the local grid size. As a consequence, time-dependent mesh modifications induced by adaptive refinement may affect the production and destruction mechanisms of turbulence. For this reason, one approach to unsteady grid adaptation defines refinement criteria based on time-averaged quantities, producing meshes that do not evolve rapidly in time while remaining suitable for capturing the unsteady flow at each instant [11]. In that work, two averaging strategies are considered: the first one averages the instantaneous flow field, whereas the second one averages the refinement criterion itself. Compared with adaptation based directly on the instantaneous solution, this strategy reduces the computational cost and improves solution accuracy. Such an approach is effective for turbulent simulations in which vortical structures often exhibit approximately periodic motion. However, it is not designed to predict the propagation of shock fronts in problems such as Riemann-type configurations, where discontinuities continuously enter regions that were not previously refined.

To overcome both the limitations associated with frequent remeshing and the lack of predictive capability in the temporal evolution of the flow features, a transient fixed-point formulation has been proposed for unsteady mesh adaptation [12, 13]. This method introduces an additional loop within the main solution algorithm. At each internal iteration of the adaptation loop, over the time interval from t to $t + \Delta t$, the solution is advanced starting from the solution at time t up to time $t + \Delta t$. A metric associated with the computed solution is then defined and used to generate a new adapted mesh. Instead of continuing the computation as in a standard time-marching procedure, the calculation is restarted from the same initial solution at time t , but on the newly adapted mesh. This process is repeated until convergence toward a fixed point at $t + \Delta t$ is achieved. The computation then proceeds to the following time interval, with the solution at $t + \Delta t$ becoming the new initial condition. At each iteration, the new mesh is constructed using an *a posteriori* error estimate based on the Hessian of the solution. A time intersection of the metrics is then introduced, allowing refinement over the whole region swept by the evolving phenomena during the considered time interval, rather than only at their initial or final locations. If the interval between t and $t + \Delta t$ is too large, the intersection between the metric associated with the initial solution at time t and the one associated with the computed solution at time $t + \Delta t$ may leave under-resolved regions between the two refined areas. To avoid this issue, intermediate solutions must be considered, and the corresponding intermediate metrics must be intersected in time to define the final metric used to generate the mesh for the simulation over the interval from t to $t + \Delta t$.

In this work, a predictive mesh adaptation strategy based on computer-vision algorithms is proposed. The solution at the current time level and at a previous one is analysed to predict a target grid spacing suitable for the following time steps. In this way, remeshing can be performed less frequently than in standard unsteady adaptation approaches. The objective is to avoid recomputing the CFD solution several times on different meshes within an iterative adaptation process. Instead, the target grid spacing is predicted in advance over a prescribed time interval, allowing the flow solution to be computed only once at each time step.

The proposed algorithm takes as input a computational mesh and the associated discrete solution field. In the present work, the flow solution is obtained by solving the compressible Euler equations using a node-centred finite volume formulation. However, the proposed framework is not restricted either to this specific spatial discretization or to inviscid flow models. The computed flow solution is used to construct a metric field that prescribes the target grid spacing. By comparing the metric field at the current time level with that at a previous time level, a template-matching algorithm is employed to estimate its motion and predict its evolution over a future time interval. This interval can be significantly larger than a single time step, extending up to several tens of time steps. The algorithm therefore estimates a target grid

spacing that can be used by the mesh adaptation procedure to generate a mesh suited to the solution to be computed at subsequent time steps.

The structure of the paper is as follows. Sec. 2 describes the workflow employed for metric prediction, from the computation of the metric field to the estimation of its motion and its extrapolation in time. The workflow is then assessed on selected unsteady test cases governed by the one- and two-dimensional Euler equations: the one-dimensional piston problem in Sec. 3, the cylindrical shock without obstacles in Sec. 4, and the cylindrical shock interacting with an obstacle in Sec. 5. In the test cases considered in this work, the metric-prediction workflow is assessed only through *a posteriori* analyses, using as input either analytical solutions or numerical solutions computed on a fixed mesh.

2. Metric prediction workflow

Rather than local adaptation, such as h-refinement, this work adopts a remeshing strategy. After a certain number of time steps, the mesh is regenerated from scratch based on the target grid spacing, that is, the metric function that drives the adaptation of the mesh to the flow solution. Various techniques are available in the literature [14] to define the target grid spacing. Among the available feature-based mesh adaptation techniques [5], the projected Hessian-based mesh adaptation (PHMA) method [15] is applied in this work to the solution of a 2D simulation of the Euler equations obtained with a node-centered finite-volume method.

2.1 Error estimation and metric computation

An error estimator is necessary to drive the mesh adaptation process. In the PHMA method, the estimator is represented by the discretization error related to the truncated Taylor series expansion of a flow variable at a specific point. In this work, the selected flow variable is the Mach number, M . Considering the gradient of the Mach number in two adjacent grid points x_1 and x_2 , the error estimate for each edge is computed as

$$S = |\vec{h}|^p \cdot \left| \hat{h} \cdot \frac{\partial M}{\partial x_2} - \hat{h} \cdot \frac{\partial M}{\partial x_1} \right|, \quad (1)$$

where $\vec{h}$ is the edge (vector that connects the two grid points), $\hat{h}$ is the normalized $\vec{h}$, and p is the multi-scale parameter, set equal to 3 to emphasize larger-than-average edges and protect against solution discontinuities. After taking for each grid point the maximum S among the values of mesh edges connected to that point, the target grid spacing of that point is computed as

$$h_{\text{target}} = h \cdot \left(\frac{S_{\text{mean}}}{S} \right)^{1/p}, \quad (2)$$

where h is the current grid spacing associated to that point, S_{mean} is the mean value of S considering all the nodes in the mesh.

For metric prediction computations, it was decided to deal with $\log_{10} h_{\text{target}}$ instead of h_{target} . This allows for smoothing the metric field, which would contain values with very different length scales.

2.2 Interpolation of metric over a structured grid

Template matching algorithms (Sec. 2.3) are originally developed for image processing, where data are stored on structured Cartesian grids. In contrast, the present framework addresses general CFD simulations on unstructured grids. So a pre-processing step is necessary, where $\log_{10} h_{\text{target}}$ is interpolated onto a structured rectangular grid. Hereafter, the original unstructured grid where the simulation is performed is indicated as $\mathcal{K}^{\text{un}}$, while the structured rectangular grid is called $\mathcal{K}^{\text{st}}$. Consequentially, $h_{\text{target}}^{\mathcal{K}^{\text{un}}}$ denotes h_{target} over $\mathcal{K}^{\text{un}}$ and similarly does $h_{\text{target}}^{\mathcal{K}^{\text{st}}}$. Defining $x_{\min}$ and $x_{\max}$ the minimum and maximum x -coordinates of points of $\mathcal{K}^{\text{un}}$, and $y_{\min}$ and $y_{\max}$ the minimum and maximum y -coordinates, domain

length and height are defined as $L = x_{\max} - x_{\min}$ and $H = y_{\max} - y_{\min}$.

The resolution of $\mathcal{K}^{\text{st}}$ is chosen to match approximately the smallest characteristic element size of $\mathcal{K}^{\text{un}}$. In particular, the parameter s regulates the structured grid scale, seen as the ratio between the size of the smallest possible cell of $\mathcal{K}^{\text{un}}$, $h_{\min}$, and the size of a cell of $\mathcal{K}^{\text{st}}$ (pixel), h_{st} . It represents the “pixels per cell”:

$$s = h_{\min}/h_{\text{st}}. \quad (3)$$

s should be set to a value near 1 to preserve all the information of the unstructured grid. However, in real-world applications, values near 0.5 are also possible and allow for saving computational resources since a flow feature, even if it is a discontinuity like a shock wave, in the h_{target} field spans more than one cell. Finally, the resolution of $\mathcal{K}^{\text{st}}$, r , expressed in pix/m, is computed as

$$r = 1/h_{\text{st}} = s/h_{\min}. \quad (4)$$

The structured lattice is denoted by G^{st} and corresponds to the matrix containing the point coordinates of $\mathcal{K}^{\text{st}}$. These coordinates are linearly distributed from $x_{\min}$ to $x_{\max}$ along the x -direction and from $y_{\min}$ to $y_{\max}$ along the y -direction. The dimensions of G^{st} are $N_y \times N_x$, where $N_x = \lceil L r \rceil + 1$ and $N_y = \lceil H r \rceil + 1$.

Once the structured grid has been defined, $\log_{10} h_{\text{target}}$ is interpolated onto the coordinates of G^{st} , yielding $\log_{10} h_{\text{target}}^{\mathcal{K}^{\text{st}}}$. Linear interpolation is used for the structured points lying inside the source unstructured grid. Since, by construction, the structured grid also covers regions that do not belong to the unstructured grid, and since a value must be assigned to every structured-grid point, the remaining points are assigned values using a nearest-neighbour interpolation.

2.3 Computation of metric motion

Let t_m denote the current time step at which the flow solution is available. The objective is to estimate, using template matching algorithms, the velocity field underlying the evolution of the metric function $\log_{10} h_{\text{target}}^{\mathcal{K}^{\text{st}}}$ from a previous time level $t_l < t_m$ to t_m . This velocity, that is an advection velocity, is hereafter referred to as *motion field*. The key assumption of the whole metric prediction procedure is that this estimated motion field also governs the evolution of the metric function beyond t_m , up to a successive time $t_n > t_m$. The employed notation adds the index of the time step as a subscript of the quantities: for example, to indicate h_{target} at t_m , the expression $h_{\text{target},m}$ is used. The motion field matrix, denoted by v_{lm} , represents the advection velocity of the metric between t_l and t_m .

Template matching algorithms have been chosen as object tracking algorithms since, using the right similarity metric, constant brightness in time is not required, in opposition to other methods like optical flow. In fact, during a CFD simulation with mesh adaptation, the average grid spacing for a given feature is not necessarily constant in time or across different meshes. Namely, a window in I_l in the frame at t_l , and the corresponding displaced window in I_m in the frame at t_m , could have different mean values of brightness. For example, if inside a window there is a shock wave that is refined in a remeshing phase between t_l and t_m , the values of $\log_{10} h_{\text{target}}^{\mathcal{K}^{\text{st}}}$ (that is, the image brightness) decrease in the shock region from I_l to I_m . If the brightness change is limited, optical flow algorithms should also work well, but template matching algorithms are more suitable.

Template matching algorithms consist of searching for the location of a template image in a larger image. I_l is divided into square windows, then each window T is slid over I_m , and a similarity metric is computed for each possible displacement. The displacement returned by the algorithm is the one that minimizes or maximizes the similarity metric, depending on the metric itself. Several choices can be made regarding the similarity metric [16]: the most common are the sum of absolute differences, the sum of squared differences (SSD), and the normalized cross-correlation. While the first two have to be minimized to find the best match of the template, the last one has to be maximized. The metric chosen for this work is the normalized sum of squared differences (NSSD): compared with the standard SSD, the normalization step reduces sensitivity to global lighting changes, making the method more robust when the brightness

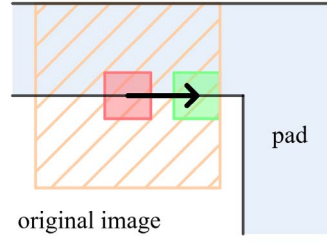


Figure 1: Pad added to the pair of images. ■ is T in I_l , ■ is T in I_m , ■ is the pad, while the diagonal stripe pattern colored with ■ is the search area. The thick black arrow is the displacement of T between I_l and I_m and in this example corresponds to the maximum possible displacement.

constancy constraint is not satisfied.

For each window T , the computed pixel displacement (d_i, d_j) is the one that minimizes the NSSD between the window in I_l and the corresponding shifted window in I_m . The NSSD can be expressed as

$$\text{NSSD}(d_i, d_j) = \frac{\sum_{(i,j) \in T} |I_l(i, j) - I_m(i + d_i, j + d_j)|^2}{\sqrt{\sum_{(i,j) \in T} I_l(i, j)^2 \cdot \sum_{(i,j) \in T} I_m(i + d_i, j + d_j)^2}}. \quad (5)$$

The size of T is determined by the parameter w . It can be set by imposing w_{rel} , the relative window size, which is defined as $w_{\text{rel}} = w / (N_x - 1)$. The window size has to be odd by construction to obtain a window that is centered on a pixel. The advantage is that the choice of w_{rel} is independent of the resolution r , unlike w . The number of windows along the x -direction is represented by the parameter N_{wx} . The number of windows along the y -direction, N_{wy} , is computed to get the same spacing between the centers of the windows as in the x -direction. Windows are equally spaced, so they are arranged according to a regular rectangular grid. The overlap between two adjacent windows is defined as

$$o = 1 - \frac{1/w_{\text{rel}}}{N_{wx} - 1}, \quad (6)$$

where $1/w_{\text{rel}}$ would be the number of windows along the x -direction minus one if there was no overlap. The overlap parameter has to be positive, otherwise there would be points of $\mathcal{K}^{\text{st}}$ that are not considered in any window. This could result in the loss of detection of the motion of some small features. To compute the maximum possible displacement (d_i, d_j) , and so the size of the search area, it is sufficient to estimate the maximum speed of a flow feature, $v_{\text{max}}^{\text{est}}$. Knowing $(t_m - t_l)$, the estimated maximum displacement is obtained (firstly in meters, then in pixels, multiplying by r).

A pad has been added to the edges of the images I_l and I_m , to be able to treat windows centered on image edges or corners like the other windows in the rest of the image. In Figure 1, there is a graphical example of a search area around a window T centered on the upper right corner. The chosen pad mode is the reflection of the matrix by mirroring its values along each edge, without repeating the values on the edge.

Without any further procedure, the obtainable displacement (d_i, d_j) would be discretized with an accuracy of 1 pixel. If further spatial resolution is needed, instead of increasing the resolution of the structured grid, which would increase the computational effort of the entire procedure, a subpixel interpolation can be exploited. Various subpixel interpolation methods are applied to Particle Image Velocimetry (PIV) in experimental fluid dynamics [17]. In this work, a parabolic peak fit has been employed. Defining with $\bar{d}_i$ and $\bar{d}_j$ the values for which $\text{NSSD}(\bar{d}_i, \bar{d}_j)$ is minimum, the displacement $\bar{d}_i$ refined by the subpixel interpolation is called $\bar{\bar{d}}_i$ and computed as

$$\bar{\bar{d}}_i = \bar{d}_i + \frac{1}{2} \frac{\text{NSSD}(\bar{d}_i - 1, \bar{d}_j) - \text{NSSD}(\bar{d}_i + 1, \bar{d}_j)}{\text{NSSD}(\bar{d}_i - 1, \bar{d}_j) - 2\text{NSSD}(\bar{d}_i, \bar{d}_j) + \text{NSSD}(\bar{d}_i + 1, \bar{d}_j)}. \quad (7)$$

In other terms, $\bar{\bar{d}}_i$ represents the abscissa of the vertex of the parabola passing through the point with minimum NSSD and its adjacent points: $(\bar{d}_i, \text{NSSD}(\bar{d}_i, \bar{d}_j))$, $(\bar{d}_i - 1, \text{NSSD}(\bar{d}_i - 1, \bar{d}_j))$ and $(\bar{d}_i + 1, \text{NSSD}(\bar{d}_i + 1, \bar{d}_j))$. Similarly, $\bar{\bar{d}}_j$ is computed. This procedure can only be applied in cases where $\bar{d}_i$ and $\bar{d}_j$ do not correspond to the maximum possible displacement, otherwise there is no adjacent point. Since $\bar{\bar{d}}_i$ and $\bar{\bar{d}}_j$ are computed only in $N_{wx} \times N_{wy}$ points (the centers of the windows), they have to be interpolated in all other grid points. A linear interpolation is performed to avoid discontinuities in the motion field that would appear with nearest-neighbor interpolation, and that could lead to a poorly extrapolated target grid spacing. From the two components of pixel displacement between t_l and t_m , $\bar{\bar{d}}_i$ and $\bar{\bar{d}}_j$, the motion field v_{lm} is obtained dividing by $(t_m - t_l)$ and by r .

2.4 Extrapolation of the target grid spacing

The metric prediction procedure is based on the assumption that the estimated motion field v_{lm} , that is the advection velocity of points (pixels) in the field $\log_{10} h_{\text{target}}^{\mathcal{K}^{\text{st}}}$ (the metric function) from t_l to t_m , governs the evolution of the metric function beyond t_m , up to successive time steps $t_n \in \{t_{m+1}, t_{m+2}, \dots\}$. The adopted advection scheme is a forward scheme that advects the points downstream while keeping their velocity constant [18]. This assumption is justified when the advection time is sufficiently short.

The motion field v_{lm} represents the velocity field associated with the evolution of the metric function from time t_l to time t_m . If this field is computed from I_l to I_m , the advection of the point positions would naturally start from time t_l and proceed toward time t_n , rather than from time t_m to time t_n . This is because the points to which v_{lm} is assigned are the pixels of I_l , not those of I_m . However, extrapolating the metric from t_l to t_m is unnecessary, since the metric function at time t_m is already available when the metric-prediction procedure is applied. Moreover, interpolating the motion field at the pixel centres of I_m would be inconsistent with the underlying Lagrangian interpretation of the prediction procedure: each vector of the motion field would be associated with a fixed spatial location rather than with a moving particle, here interpreted as an advected pixel. As a result, the motion field would no longer describe the displacement of the metric features according to the assumptions on which the proposed prediction strategy is based.

To avoid this unnecessary extrapolation from t_l to t_m , which would only contribute to the propagation of errors caused by possible under- or overestimation of the motion field, an *opposite motion field* approach is adopted. In this formulation, the opposite motion field v_{ml} , mapping I_m back to I_l , is computed. The points to which v_{ml} is assigned are therefore the pixels of I_m , so that the advection process can start directly from the available metric field at time t_m .

The pixel displacement returned by the algorithm described in the previous section does not necessarily correspond to an integer number of pixels, either because the template matching algorithm with subpixel interpolation itself provides decimal displacements or because the motion field in the majority of pixels is interpolated from the values at the centers of the windows. This means that, adding the computed displacement for a certain pixel to the original position of the pixel at t_m , this advected pixel does not necessarily land exactly on another pixel. While G^{st} , introduced in Sec. 2.2, contains the point coordinates of a regular structured grid ($\mathcal{K}^{\text{st}}$), the advected positions of the points of this structured grid do not represent a structured grid anymore. The matrix containing the advected positions is called G^{adv} (with a subscript that indicates the time step) and follows over time the positions of the points that at t_m lay on the structured grid:

$$G_n^{\text{adv}} = G^{\text{st}} - v_{ml}(t_n - t_m) . \quad (8)$$

The metric function associated with coordinates G_n^{adv} at t_n is the one associated with coordinates G^{st} at t_m , that is $\log_{10} h_{\text{target},m}^{\mathcal{K}^{\text{st}}}$. Figure 2 shows how the motion field vector $-v_{ml}(i, j)$ is associated, in every advection step starting from t_m , to the point that at t_m has coordinates $G^{\text{st}}(i, j)$; for every advection step, $-v_{ml}(i, j)$ is translated to follow the point, leading to the forward scheme employed in Eq. (8).

At the end of the advection step, G_n^{adv} is still a matrix but represents sparse coordinates. To obtain

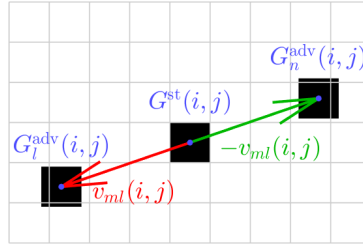
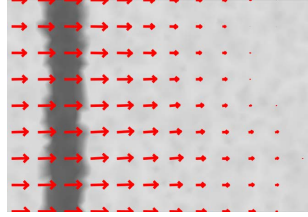


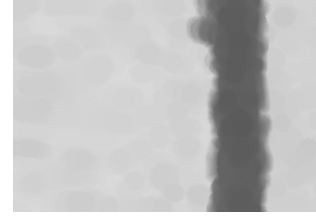
Figure 2: Pixel with indexes (i, j) has coordinates G^{st} at t_m . Its opposite motion field vector $v_{ml}(i, j)$ is the red arrow. The green arrow is $-v_{ml}(i, j)$, used to advect the pixel. $G_l^{\text{adv}}(i, j)$ comes from an advection back in time and represents the position of the pixel at t_l according to the opposite motion field.



(a) Motion field (red arrows) of the metric function from t_l to t_m with the background contour plot of $\log_{10} h_{\text{target},l}^{\mathcal{K}^{\text{st}}}$.



(b) $\log_{10} h_{\text{target},n}^{\mathcal{K}^{\text{st}}}$ computed with nearest-neighbor interpolation.



(c) $\log_{10} h_{\text{target},n}^{\mathcal{K}^{\text{st}}}$ computed with the *minimum search* approach.

Figure 3: Interpolation of the advected metric function over $\mathcal{K}^{\text{st}}$. The example is taken from Sec. 3.

$\log_{10} h_{\text{target},n}^{\mathcal{K}^{\text{st}}}$, that is the metric function at t_n on grid $\mathcal{K}^{\text{st}}$, $\log_{10} h_{\text{target},m}^{\mathcal{K}^{\text{st}}}$ has to be interpolated from coordinates G_n^{adv} to coordinates G^{st} .

Among the possible interpolation strategies, nearest-neighbor interpolation is generally fast and provides good results, but a particular aspect has been considered to refine this simple choice. For example, a shock wave moves from left to right through a uniform spacing region, like in Figure 3a (the example is taken from the test case described in Sec. 3). If the extrapolation is done for several time steps ahead of t_m , the dark points corresponding to the shock wave reach and mix with the light points in the top right corner of the image, which are characterized by a null (or really small) motion field. As a consequence, for certain points of $\mathcal{K}^{\text{st}}$ where the shock is located at t_n , the nearest advected point with coordinates G_n^{adv} is a light point that was already there at t_m and did not move, and not a point of the shock wave that has been advected from left to right, as shown in Figure 3b. To avoid this behavior, for each point of $\mathcal{K}^{\text{st}}$ the lowest value of $\log_{10} h_{\text{target},m}^{\mathcal{K}^{\text{st}}}$ is taken among advected pixels having coordinates G_n^{adv} within a certain radius r_{ms} from that point. If no advected pixel is present within that radius, then nearest-neighbor interpolation is done for that point. As Figure 3c shows, this *minimum search* approach preserves the shape of flow features, which are characterized by a smaller grid spacing than the surrounding space. This approach has the effect of enlarging the width of the features, because advected points with small grid spacing are spread around. This can be considered a benefit, like a safety factor that makes the predicted metric function more conservative. Moreover, r_{ms} can vary as a function of the time step difference between t_n and t_m : the bigger $t_n - t_m$ is, the bigger r_{ms} can be.

The predictive mesh adaptation approach proposed in this work is advantageous if the remeshing phase is done less frequently than the standard approaches. In the remeshing phase, the goal is to obtain a single metric function that will provide a good mesh to be used from t_{m+1} up to a certain $t_{\bar{n}}$. The extrapolation is then performed for every $t_n \in \{t_{m+\Delta n}, t_{m+2\Delta n}, \dots, t_{\bar{n}^*}\}$, where Δn indicates how many time steps are skipped between two consecutive extrapolations and $t_{\bar{n}^*}$ is the last time step of extrapolation of the current metric function. $t_{\bar{n}^*} > t_{\bar{n}}$ acts like a safety factor for an eventual underestimation of the motion field or an eventual positive acceleration of the position of some features. Then, for every point of $\mathcal{K}^{\text{st}}$,

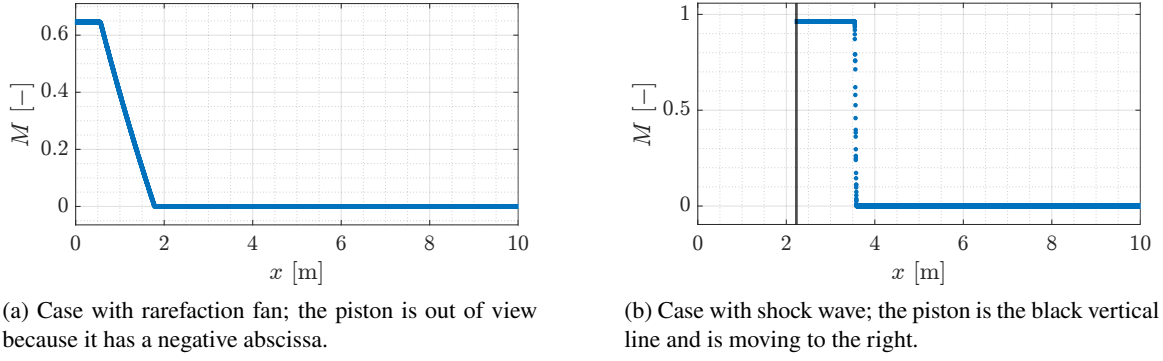


Figure 4: Mach number averaged on the dual cells starting from the analytical solution at $t_m = 5.1 \cdot 10^{-3}$ s.

the minimum among all the extrapolated metrics is taken:

$$\log_{10} h_{\text{target}, \bar{n} \text{ cum}}^{\mathcal{K}^{\text{st}}} = \min\{\log_{10} h_{\text{target}, m}^{\mathcal{K}^{\text{st}}}, \log_{10} h_{\text{target}, m+\Delta n}^{\mathcal{K}^{\text{st}}}, \log_{10} h_{\text{target}, m+2\Delta n}^{\mathcal{K}^{\text{st}}}, \dots, \log_{10} h_{\text{target}, \bar{n}^*}^{\mathcal{K}^{\text{st}}}\}. \quad (9)$$

This cumulative metric is given as input to the remeshing software. Also, the metric at t_m is included as a safety factor, even if the solution at t_m has already been computed.

2.5 Interpolation of predicted metric over original mesh

Comparisons between metrics can be done with $\log_{10} h_{\text{target}}$ expressed on both $\mathcal{K}^{\text{st}}$ and $\mathcal{K}^{\text{un}}$, but the software that adapts the mesh usually requires h_{target} in the points of the mesh of the CFD simulation, $\mathcal{K}^{\text{un}}$. So, $\log_{10} h_{\text{target}, \bar{n} \text{ cum}}^{\mathcal{K}^{\text{un}}}$ can be obtained from $\log_{10} h_{\text{target}, \bar{n} \text{ cum}}^{\mathcal{K}^{\text{st}}}$ by doing a simple linear interpolation on the points of $\mathcal{K}^{\text{un}}$. Taking the exponential of $\log_{10} h_{\text{target}, \bar{n} \text{ cum}}^{\mathcal{K}^{\text{un}}}$, $h_{\text{target}, \bar{n} \text{ cum}}^{\mathcal{K}^{\text{un}}}$ is obtained.

3. Piston problem

3.1 Problem description and analytical solution

The first test we address is the 1D piston problem governed by unsteady Euler equations, for which the analytical solution can be computed [19]. To test the metric prediction workflow in a more general way, the considered domain is 2d. The rectangular domain spans from $x_{\min} = 0$ m and $y_{\min} = 0$ m to $x_{\max} = 10$ m and $y_{\max} = 1$ m, obtaining $L = 10$ m and $H = 1$ m. Using the software GMSH [20], the domain has been discretized with a triangular mesh with a characteristic element size of $h = 0.04$ m.

The fluid inside the piston is air, with $R = 287$ J/(kg K) and $\gamma = 1.4$. The initial state, at rest, corresponds to $T_0 = 304.15$ K and $P_0 = 1.8 \cdot 10^5$ Pa. Then, at time $t = 0$ s, the piston located at $x = 0$ m is impulsively started with velocity V_p . Two cases are considered: $V_{p,f} = -200$ m/s, which leads to the formation of a rarefaction fan inside the domain, and $V_{p,s} = 436.955$ m/s, which leads to the formation of a shock wave. Both the fan and the shock wave move from left to right. This problem has an analytical solution where each physical quantity only depends on x and t . To replicate the results of a 2D simulation obtained with a node-centered finite volume formulation, the solution associated with each point of the mesh is computed as the cell average of the analytical solution over the dual cell centered on that point. The resulting Mach field is shown in Figure 4, for both the fan case and the shock case.

The gradient of the Mach number, necessary for the computation of h_{target} with the PHMA method (Sec. 2.1), is computed by resorting to the edge-based finite volume discretization, following [2]. The following limits for the target grid spacing have been set: $h_{\min} = 0.001$ m, $h_{\max} = 0.4$ m.

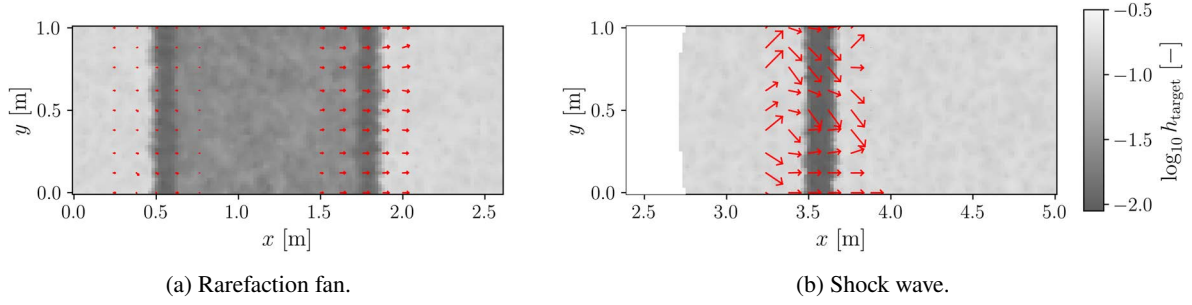


Figure 5: Motion field $-v_{ml}$ represented (red arrows), where the length of the vectors corresponds to that of the displacement vectors from t_m to t_l . The displayed vectors are those directly computed by minimization of the NSSD (the interpolated ones are not visible). The background contour plot is $\log_{10} h_{\text{target},m}^{\text{Kst}}$ being $t_m = 5.1 \cdot 10^{-3}$ s.

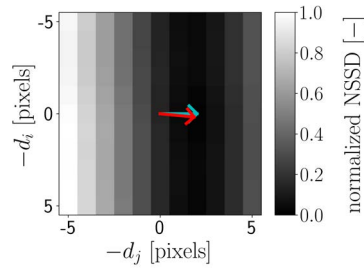


Figure 6: NSSD matrix relative to the point with coordinates $x = 1.76$ m, $y = 0.5$ m in the fan case. The NSSD matrix has been normalized by its maximum value. The cyan arrow is the displacement vector $-(\bar{d}_i, \bar{d}_j)$, characterized by a normalized NSSD of 0.015, while the red arrow (almost superimposed on the cyan one) is the displacement vector $-(\bar{\bar{d}}_i, \bar{\bar{d}}_j)$.

3.2 Results of the metric prediction workflow

After computing h_{target} , the metric prediction procedure has been applied. Reminding that the chosen t_m is $5.1 \cdot 10^{-3}$ s, t_l has been set equal to $5.0 \cdot 10^{-3}$ s. For convenience, it is assumed $t_l = t_{m-1}$, meaning that t_l is the previous time step of t_m .

3.2.1 Evaluation of motion field estimation

The chosen structured grid scale is $s = 2$, to have, on average, two pixels for each cell of $\mathcal{K}^{\text{un}}$, which leads to $N_x = 501$ and $N_y = 51$. The relative window size has been set to $w_{\text{rel}} = 0.04$, which corresponds to $w = 21$ pixels. $N_{w,x} = 81$ has been chosen, to obtain an overlap $o = 0.7$. $v_{\text{max}}^{\text{est}}$ has been set to 900 m/s, since in this specific case the exact speed of flow features is known and is lower than this value.

Figure 5 represents the motion field, in the cases with the rarefaction fan (Figure 5a) and with the shock wave (Figure 5b), where the left part of the domain is blank because the cells to the left of the piston are not considered part of the solution.

As an example, Figure 6 shows the NSSD matrix of the point at $x = 1.76$ m, $y = 0.5$ m in the fan case. The appearance as a vertical repetition of the same line is the consequence of the 1D nature of the problem, which leads to a not well-determined vertical component of displacement. In fact, the motion field vectors in some points also exhibit a high vertical component, parallel to the features, in addition to the normal component. Despite this quite big vertical component detected in some points, the normal (i.e., horizontal) component is well estimated (as shown in Table 1).

		fl	fr	s
v^{ex}	[m/s]	109.6	349.6	699.1
$\bar{\eta}$	[-]	0.827	1.027	1.043
σ	[-]	0.162	0.064	0.007

Table 1: Exact propagation speed of the flow features and indicators of the goodness of the motion field estimation.

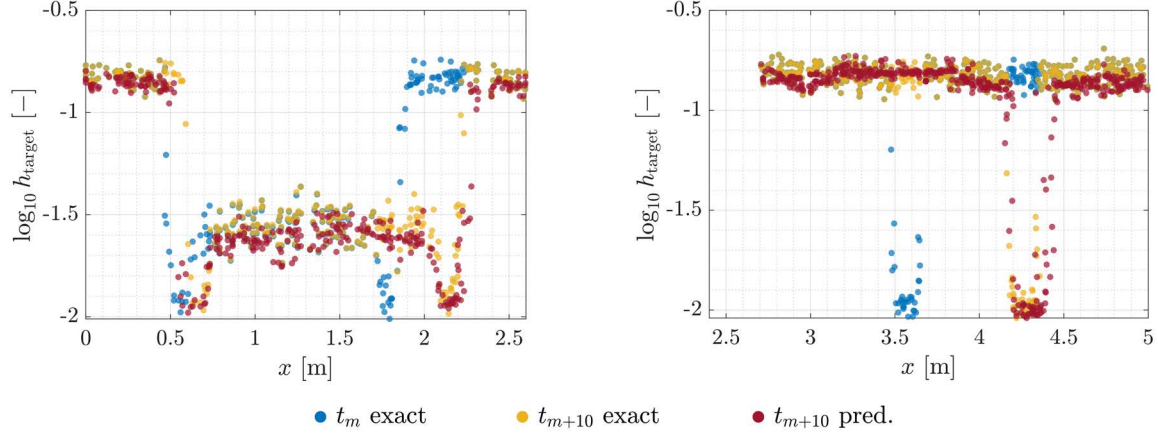


Figure 7: Evaluation of the goodness of the extrapolated metric. Case with the rarefaction fan (left) and case with the shock wave (right).

The accuracy of the computed motion field is assessed by comparing the exact propagation speed, analytically available, of the most important flow features. For each feature, the motion field is sampled at 11 points uniformly distributed along the vertical line, overlapping the feature at t_m . The exact propagation speed is denoted by v^{ex} , while the estimated speed v^{est} is defined as the horizontal component of the 11 motion field vectors. The following features are analyzed: left edge of the rarefaction fan, right edge of the fan, shock wave, identified by fl , fr , s , respectively. The following normalization is useful to easily analyze the results:

$$\eta = \frac{v^{est}}{v^{ex}}, \quad (10)$$

where η is a vector because v^{est} is a vector. Two useful quantities that can be extracted are $\bar{\eta}$, which is the mean value of η , and σ , the standard deviation of η . Ideally, if $\bar{\eta} = 1$ and $\sigma = 0$, the motion field would perfectly match the analytical speed. Usually, $\bar{\eta}$ close to 1 means that the motion field is correctly estimated, but it is not always true. For example, having a certain number of elements of η equal to 0 and an equal number of elements of η equal to 2 would provide $\bar{\eta} = 1$ but the absolute relative error regarding the estimation of the motion field would be 100% for every point, leading to $\sigma = 1$; so, also watching at σ is useful to evaluate the goodness of the estimation of the motion field. Table 1 provides values of v^{ex} , $\bar{\eta}$ and σ .

3.2.2 Comparison between exact and predicted metric

Using the computed motion field $-v_{ml}$, the extrapolated metric $\log_{10} h_{target,n}^{\mathcal{K}^{st}}$ is computed for $t_n = t_{m+10}$, which means 10 time steps ahead. The *minimum search* approach has been applied with radius $r_{ms} = 0.04$ m. Figure 7 shows the exact metrics at t_m and t_n , and the predicted metrics at t_n interpolated over $\mathcal{K}^{un}$. The predicted metric overlaps well with the exact one, marking a first successful example of the metric prediction workflow.

4. Cylindrical shock (no obstacles)

The aim now is to test the metric prediction workflow in a truly 2D problem that is still sufficiently simple, such as the cylindrical shock problem. The domain is a circular sector, and the problem is axisymmetric. The motion field is compared to the true speed of the flow features. Finally, the extrapolated metric is compared to the exact one.

4.1 Problem description and simulation setup

The problem is the radial version of the Riemann problem. Initially, there is a diaphragm that separates an inner region from an outer one. The simulation deals with air, modelled an ideal gas, with gas constant $R = 287.058 \text{ J/(kg K)}$.

The Euler equations are solved using SU2 using a first-order implicit dual-time stepping strategy for the time integration. The convective scheme is the upwind ROE scheme [21], extended to second-order through the MUSCL reconstruction [22]. Venkatakrishnan's slope limiter [23] has been used to avoid oscillations near shocks and other regions with sharp gradients. The convergence criterion for the simulation relies on the root-mean square residual of the density, with a threshold equal to 10^{-9} . The chosen time step of the simulation is $\Delta t = 1 \cdot 10^{-6} \text{ s}$, assessed after a time step convergence analysis that is not reported here for brevity.

4.1.1 Mesh

A circular sector is used for the simulation. The radius of the domain is $R_D = 1 \text{ m}$, while its angle of amplitude is $\alpha = 30^\circ$. The unstructured mesh, created with Gmsh, is made of triangles with a uniform cell size of $h = 0.001 \text{ m}$, leading to 347169 points.

4.1.2 Initial and boundary conditions

The diaphragm is placed at radius $R_0 = 0.65 \text{ m}$ and is removed at $t = 0 \text{ s}$. Velocity is initially zero in the entire domain. The internal state, $r \leq R_0$, is characterized by dimensionless temperature and pressure $\hat{T}_i = 1$, $\hat{P}_i = 1$. The external state, $r > R_0$, has the same temperature $\hat{T}_e = \hat{T}_i$, but a 10 times larger pressure, i.e., $\hat{P}_e = \beta_P \cdot \hat{P}_i$. Reference quantities are $T_{\text{ref}} = 304.15 \text{ K}$, $P_{\text{ref}} = 1.8 \cdot 10^5 \text{ Pa}$.

The boundary condition for the upper and lower boundaries is a slip-wall, while for the outer arc a pressure equal to the one in the external region is imposed.

Once the diaphragm is removed, converging shock wave and contact discontinuity are generated, as well as a diverging rarefaction fan. Figure 8 shows the solution at time instants $t = 3.2 \cdot 10^{-4} \text{ s}$ and $t = 6.4 \cdot 10^{-4} \text{ s}$. Due to the curvature, these features are by nature unsteady: shock intensity progressively grows up to the focus point (the center of the domain), so post-shock states are time dependent.

4.2 Features position in time

Converging shock waves have been widely studied through both experimental and numerical approaches. The numerical results obtained in this work are compared with the self-similar solution proposed by Guderley [24]. According to Guderley's law, the radius of the shock wave R_s evolves in time as

$$R_s = R_0 \left(1 - \frac{t}{\tilde{\tau}}\right)^\alpha, \quad (11)$$

where R_0 is the initial radius of the shock, $\tilde{\tau}$ is the total focusing time (the time needed for the shock to reach the focus point), and α is the self-similar exponent, which for a cylindrical shock in a polytropic ideal gas is $\alpha = 0.834$.

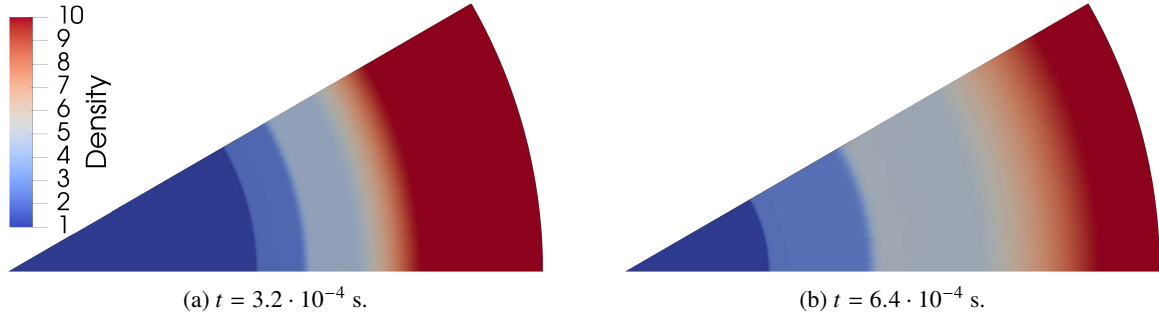


Figure 8: Dimensionless density $\hat{\rho} = \rho / \rho_{\text{ref}}$, where reference density is computed as $\rho_{\text{ref}} = P_{\text{ref}} / (R T_{\text{ref}})$.

To identify the shock position in time along the bisector of the domain, a shock tracking algorithm has been implemented, following [25]. The shock radius is identified as the minimum radial coordinate over the bisector for which: the pressure derivative is positive (compression shock), the second derivative of pressure is negative (inflection point or peak), and the pressure differs by a certain tolerance from P_i (numerical oscillations). The shock radius obtained in this way is used to fit the exponent α in Guderley's law. Being the focusing time $\tilde{\tau} = 1.006 \cdot 10^{-3}$ s, the fitted value is $\alpha = 0.869$, close to the theoretical value. This discrepancy may be due to the fact that, to fit α , all the time steps from 0 to $\tilde{\tau}$ were considered, whereas the power law only applies where the phenomenon is truly self-similar, namely after the initial, transient shock formation.

The fact that the velocity of the shock wave changes with time is in contrast with the assumption of constant advection velocity made in Sec. 2.3. To estimate the error introduced by this assumption, ξ is defined as the percentual overestimation of the shock wave radius at t_n :

$$\xi = \frac{R_{s,n}^{\text{est}} - R_{s,n}^{\text{ex}}}{R_{s,m}^{\text{ex}} - R_{s,n}^{\text{ex}}} \cdot 100\% , \quad (12)$$

where the second letter of the pedices refers to time step indexes. $R_{s,n}^{\text{est}}$ is computed as

$$R_{s,n}^{\text{est}} = R_{s,m}^{\text{ex}} + (t_n - t_m) \cdot (R_{s,m}^{\text{ex}} - R_{s,l}^{\text{ex}}) / (t_m - t_l) , \quad (13)$$

while R_s^{ex} is the radius given by the fitted Guderley's law. $t_l = t_{m-10}$ and $t_n = t_{m+10}$ are chosen as they are the values used in the results section. At the beginning of the simulation, ξ is below 0.2%; it is below 1% up to $t = 8.76 \cdot 10^{-4}$ s and just before the focusing time is still just below 20%, that is quite a small value: imposing $(t_{\tilde{n}} - t_m) = 1.25 \cdot (t_{\tilde{n}} - t_m)$ when performing metric extrapolation, this safety margin is sufficient to remedy the invalidity of the hypothesis of constancy of advection velocity.

For the other flow features, which are the contact discontinuity and the rarefaction fan, analytical laws that express their position as a function of time have not been found, so a procedure to detect their position from the numerical solution has been developed, like the one just described to identify the shock wave. For the contact discontinuity, firstly the point over the bisector of the domain that satisfies the following conditions is identified, starting from the origin: its dimensionless temperature is ≥ 1 , the dimensionless temperature of its next point (with greater radial coordinate) is ≤ 1 , the pressure differs by a certain tolerance from P_i (numerical oscillations). Then, starting from this point, the contact discontinuity position is identified as that of the nearest inflection point of the density.

The inner border of the rarefaction fan moves a bit outwards, but with a really low speed: for this reason it is not considered in this analysis. Regarding the outer border of the rarefaction fan, instead, its position is found as the outermost point of the bisector for which: the third derivative of the Mach number has a local extreme (the second derivative has an inflection point), the pressure differs by a certain tolerance from P_e (numerical oscillations). The choice of using this criterion is linked to the particular error estimator

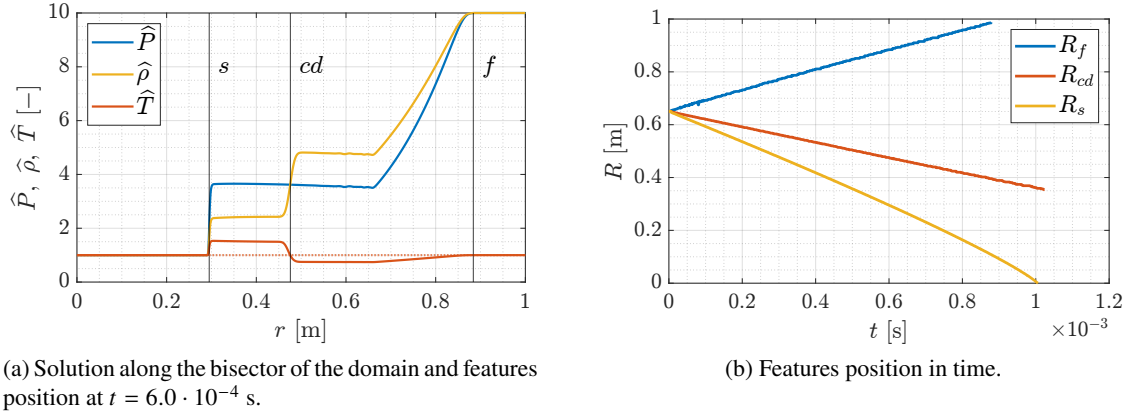


Figure 9: Position of the main flow features.

used in the PHMA method: the point where the third derivative of the Mach number has a local extreme is the point where the contrast in the $\log_{10} h_{\text{target}}$ field (related to the Hessian) is greater; this is the most natural position where the motion field has to be identified correctly.

The position of the features identified with the procedure described above is shown in Figure 9; the shock wave, the contact discontinuity and the fan outer border are identified by s , cd , f respectively.

4.3 Results of the metric prediction workflow

The limits for the target grid spacing have been set as follows: $f_{\text{max}}^{\text{ref}} = 0.25$, $f_{\text{max}}^{\text{coa}} = 4$, $h_{\text{min}} = 2.5 \cdot 10^{-4}$ m, $h_{\text{max}} = 0.01$ m.

4.3.1 Evaluation of motion field estimation

The structured grid scale has been set $s = 1$, leading to $N_x = 1001$, $N_y = 501$, that means 501501 points in $\mathcal{K}^{\text{st}}$. The relative winsize has been kept equal to the previous test case, $w_{\text{rel}} = 0.04$, since there are no small features close to each other; the value in pixels corresponds to $w = 41$. $N_{wx} = 81$ has been chosen, to obtain an overlap $o = 0.7$. $v_{\text{max}}^{\text{est}}$ has been set to 1000 m/s, a reasonable value, confirmed by the fact that the speed of the shock wave is always lower than this value, apart from just a few time steps before the shock reaches the focus point. If $t_l = t_{m-1}$ is chosen, the features move too little to guarantee a correctly estimated motion field, especially for the contact discontinuity and the rarefaction fan, which are not sharp discontinuities like the shock and move more slowly. After some trials, $t_l = t_{m-10}$ has been chosen, meaning that the two metric functions used to compute the motion field are separated by 10 time steps. This leads to a maximum possible displacement of 10 pixels between a window in I_l and the corresponding window in I_m . Also in this case the subpixel interpolation is useful: the contact discontinuity, for example, moves with a speed that is more or less constant and equal to 300 m/s, which corresponds to a displacement of about only 3 pixels between I_l and I_m .

Figure 10 shows the motion field at time instant $t_m = 6.0 \cdot 10^{-4}$ s. As expected, the motion field is non-zero only in correspondence with the flow features. Then, especially for the fan outer border, some arrows also have a component parallel to the feature, while in the other features only the normal component is detected (this could be an advantage during the advection phase).

Like in Sec. 3.2.1, the accuracy of the computed motion field is verified in the proximity of the flow features; the analytical values of the speed of the features are not available, but for the shock wave, the value can be extracted by differentiating the Guderley's law with α fitted from the simulation results. For the contact discontinuity and the fan outer border, a modified version of the Guderley's law has been used as interpolant of the radii computed from the solution (Sec. 4.2); in particular, in the expression for the

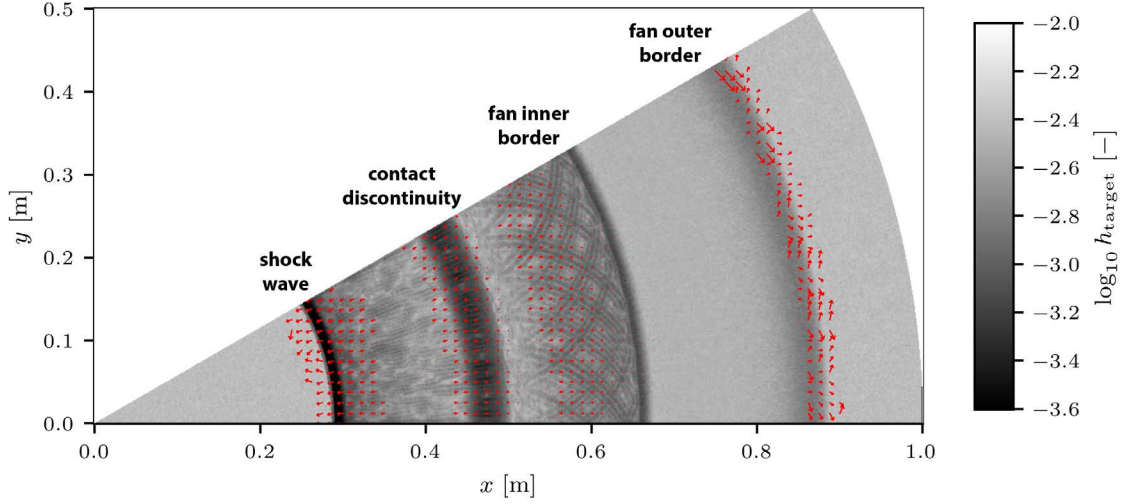


Figure 10: Motion field $-v_{ml}$ represented (red arrows), where the length of the vectors corresponds to that of the displacement vectors from t_m to t_l . The displayed vectors are the ones directly computed by minimization of the NSSD (the interpolated ones are not visible). The background contour plot is $\log_{10} h_{\text{target},m}^{\mathcal{K}^{\text{st}}}$ being $t_m = 6.0 \cdot 10^{-4}$ s.

	$\bar{\eta}_s$ [-]	σ_s [-]	$\bar{\eta}_{cd}$ [-]	σ_{cd} [-]	$\bar{\eta}_f$ [-]	σ_f [-]
$t_m = 3 \cdot 10^{-5}$ s	0.979	0.031	1.400	0.106	0.846	0.073
$t_m = 1 \cdot 10^{-4}$ s	0.975	0.014	1.161	0.055	0.955	0.020
$t_m = 3 \cdot 10^{-4}$ s	0.993	0.013	0.975	0.037	0.948	0.021
$t_m = 6 \cdot 10^{-4}$ s	0.991	0.016	0.918	0.055	0.984	0.039

Table 2: Indicators of the goodness of the motion field estimation for various time steps.

fan outer border, a positive sign has been put inside the brackets of Eq. (11) instead of a negative one, since this feature moves outwards. These interpolants have no physical meaning; they have been chosen because, fitting the values of $\tilde{\tau}$ and α for each feature, they faithfully reproduce the actual positions of the features. Furthermore, they are more suitable for differentiation due to the absence of numerical noise.

Following the definitions introduced in Sec. 3.2.1, v^{ex} represents the exact propagation speed of the flow features, computed by differentiating the interpolants just described; v^{est} is defined as the radial component of 101 motion field vectors, computed in equally spaced points along the features at t_m . η , $\bar{\eta}$ and σ follow the definitions given in Sec. 3.2.1. Table 2 provides the values of $\bar{\eta}$ and σ for the considered flow features in various time steps of the simulation. In general, the motion field of the features is well estimated, with no significant outliers. In particular, the shock wave speed is really well estimated at each considered time step. The contact discontinuity is the feature for which the motion field is most poorly estimated; however, the only value of $\bar{\eta}_{cd}$ which is considerably different from 1 is 1.400, at $t_m = 3 \cdot 10^{-5}$ s. At this time, the features are still not well separated, so some windows that include the contact discontinuity also include the shock wave in the left part: the slight overestimation of the motion field is a natural consequence.

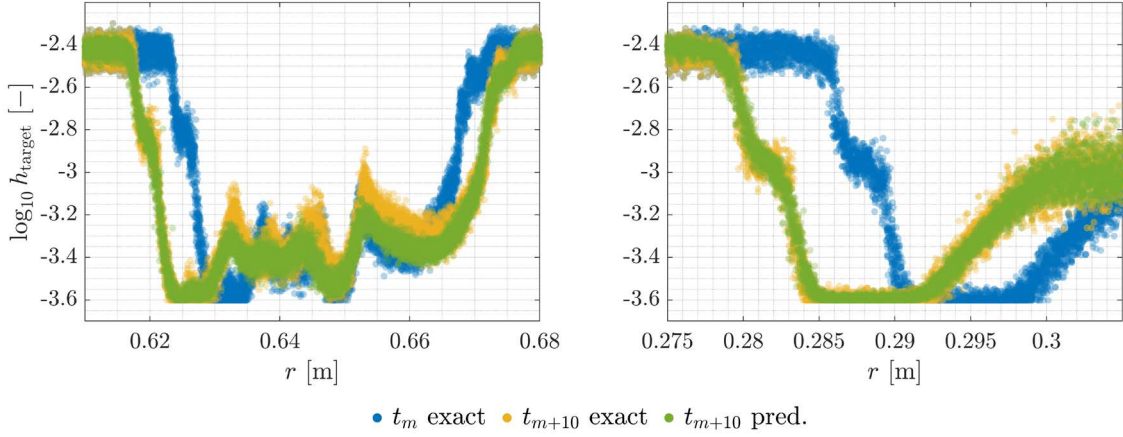


Figure 11: Evaluation of the goodness of the extrapolated metric. On the left, all the features are visible when they are still not well separated, at $t_m = 3 \cdot 10^{-5}$ s; on the right, the region around the shock at $t_m = 6 \cdot 10^{-4}$ s.

4.3.2 Comparison between exact and predicted metric

The extrapolated metric function has been evaluated for $t_n = t_{m+10}$, like in the previous test case. The fact that $t_n - t_m = t_m - t_l$ did not make it necessary to use the *minimum search* approach; its application would produce nearly the same results with an increase in computational time, so a simple nearest-neighbor interpolation has been done for all the points of $\mathcal{K}^{\text{st}}$. In Figure 11, the predicted metric field at $t_n = t_{m+10}$ is compared against the exact metric field computed from the CFD solution at the same time level. The comparison of the predicted metric field, interpolated over $\mathcal{K}^{\text{un}}$, is done for two time instants of those considered in Table 2: $t_m = 3 \cdot 10^{-5}$ s on the left, $t_m = 6 \cdot 10^{-4}$ s on the right. For both time instants, the predicted metric field closely matches the reference one around all the flow features.

5. Cylindrical shock with an obstacle

This test case is a modified version of the problem discussed in the previous section. The difference consists of the addition of an obstacle, which is an arc-shaped profile located in proximity to the focus point. The interaction of the shock front with the obstacle is an intrinsically 2D process, which represents the ideal test case for this work, without adding too much complexity.

5.1 Problem description and simulation setup

The simulation deals with air, considered an ideal gas with constant $R = 287.058 \text{ J/(kg K)}$. The geometry and input data for this test case are mostly taken from Vignati's study about cylindrical converging shock waves [26]. All the settings of the flow solver SU2 are left equal to those of the previous test case. The chosen time step of the simulation is $\Delta t = 3.125 \cdot 10^{-8}$ s, assessed after a time step convergence analysis that is not reported in this paper.

5.1.1 Mesh

Like in the previous test case, a circular sector is used for the simulation. The radius of the domain is $R_D = 0.8$ m, while its angle of amplitude is $\alpha = 45^\circ$. The arc-shaped obstacle, which lies along the lower symmetry wall boundary, is defined through the following geometrical quantities, which are reported dimensionless, being $l_{\text{ref}} = 0.01$ m:

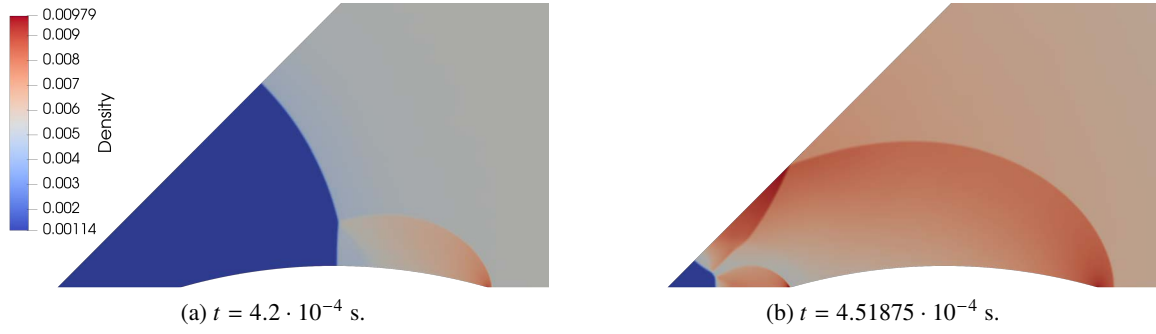


Figure 12: Dimensionless density $\hat{\rho} = \rho / \rho_{\text{ref}}$.

- obstacle chord, 5
- maximum thickness to chord ratio, 0.07
- focus point to obstacle leading edge distance, 7

The mesh is of an unstructured type, composed of triangular elements and created by the software Gmsh. The cell size is not uniform: the reference value is $h = 0.001$ m, characteristic of the central part of the domain, while the region around the obstacle and the focus point has a spacing of $0.15 \cdot h$, to better refine the shock pattern that arises from the shock-obstacle interaction. The outer region instead (where the fan is located) has a spacing of $2.5 \cdot h$, to save computational resources in a zone that is not crossed by features of interest. The resulting number of points is 278221. The chosen reference value of the cell size comes from a mesh convergence analysis, not reported in this paper.

5.1.2 Initial and boundary conditions

Rather than exactly reproducing Vignati's simulations, the diaphragm radius was increased from $R_0 = 0.25$ m to 0.4 m. This adjustment was necessary because using the original radius causes the contact discontinuity to encounter the reflected shock pattern before the latter can return to the obstacle's leading edge. Increasing the radius ensures that the shock pattern remains isolated during this phase of the simulation, which represents the most challenging test for the metric prediction workflow. The diaphragm is removed at $t = 0$ s, allowing the fluid to move, since the initial velocity is zero in the entire domain. Internal state, corresponding to the points with radius $r \leq R_0$, is given by dimensionless temperature and pressure $\hat{T}_i = 2.2476$, $\hat{P}_i = 2.5826 \cdot 10^{-3}$. External state, for points with radius $r > R_0$, is characterized by $\hat{P}_e = \beta_P \cdot \hat{P}_i$ with $\beta_P = 16$ and $\hat{\rho}_e = 5.0662 \cdot 10^{-3}$, computed from Hugoniot adiabat for a polytropic ideal gas, following [26]. Reference quantities, used for nondimensionalization, are $T_{\text{ref}} = 132.65$ K, $P_{\text{ref}} = 3.872 \cdot 10^6$ Pa (the critical temperature and pressure of air), $\rho_{\text{ref}} = P_{\text{ref}} / (R T_{\text{ref}})$.

The boundary conditions are equal to the previous test case, described in Sec. 4.1.2.

As in the case without obstacle, once the diaphragm is removed, there is the generation of a converging shock wave and contact discontinuity, and a diverging rarefaction fan. Figure 12 shows the solution at two instants of time, in the region that includes the obstacle and the focus point: Fig. 12a at $t = 4.2 \cdot 10^{-4}$ s, where the Mach reflection after the interaction of the shock with the obstacle is clearly visible, and Fig. 12b at $t = 4.51875 \cdot 10^{-4}$ s, just before the shocks pattern reaches the focus point.

5.2 Results of the metric prediction workflow

Regarding the parameters for the computation of the target grid spacing, the following values have been set: $f_{\text{max}}^{\text{ref}} = 0.25$, $f_{\text{max}}^{\text{coa}} = 4$, $h_{\text{min}} = 5 \cdot 10^{-5}$ m, $h_{\text{max}} = 0.01$ m.

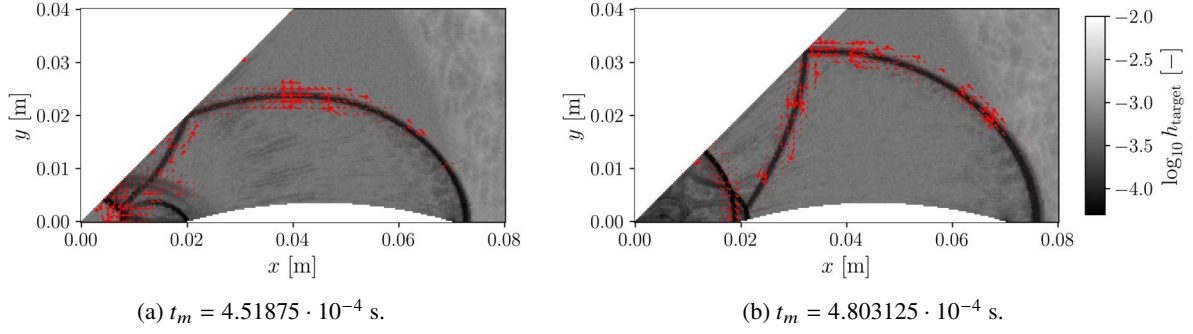


Figure 13: Motion field $-v_{ml}$ represented (red arrows), where the length of the vectors corresponds to that of the displacement vectors from t_m to t_l . The displayed vectors are the ones directly computed by minimization of the NSSD (the interpolated ones are not visible). The background contour plot is $\log_{10} h_{\mathcal{K}^{\text{st}}_{\text{target},m}}$.

5.2.1 Evaluation of motion field estimation

Since the minimum cell size is smaller than the previous test case, the structured grid scale has been set $s = 0.5$, a value smaller than 1 but, as stated in Sec. 2.2, still a good value to preserve the information of the solution on the original grid; this allows to reduce the resolution of $\mathcal{K}^{\text{st}}$, saving computational resources. This value of s gives $N_x = 2668$ and $N_y = 1887$, that correspond to 5034516 points. Since the interaction between the shock and the obstacle leads to small flow structures, the relative winsize has been reduced with respect to the previous test cases, in order to capture the different directions of motion of the small features. $w_{\text{rel}} = 0.005$ has been chosen, that corresponds to $w = 13$ pixels. The number of windows along the x -direction has been set to $N_{wx} = 641$, to obtain an overlap $o = 0.7$, the same as the previous test cases. Since the maximum speed of the shock wave near the focus point is ~ 2000 m/s, $v_{\text{max}}^{\text{est}}$ has been set equal to 3000 m/s to keep a margin, and considering that usually the exact value is not available a priori before performing the simulation. A good value for the time step difference between t_l and t_m is $t_l = t_m - 20$: t_l closer to t_m would provide less accuracy in the estimation of the motion field, while with t_l further from t_m some features may form, reflect, or change shape in the time steps between t_l and t_m . The resulting maximum possible displacement of windows between t_l and t_m is 7 pixels, making the subpixel interpolation useful also in this case.

Figure 13 shows the motion field at time instants $t_m = 4.51875 \cdot 10^{-4}$ s (Figure 13a) and $t_m = 4.803125 \cdot 10^{-4}$ s (Figure 13b). Right behind the main shock wave in Figure 13a, close to the focus point, two shock waves are running towards each other; the red arrows show that this motion is correctly identified. In Figure 13b, at the obstacle trailing edge ($x = 0.02$ m), the main shock wave, after the reflection at the focus point, is running back over the obstacle and is reaching the slower minor shock wave that originated after the interaction of the main shock with the trailing edge. The different speeds of these shocks are well captured, as it is visible from the different lengths of the arrows. In this test case, analytical values of the speed of the features are not available, since the pattern of reflected shocks depends a lot on the specific geometry of the obstacle and becomes intricate after a few reflections. So, the accuracy of the entire metric prediction workflow is directly evaluated by comparing the exact and the predicted metrics (in the next section).

5.2.2 Comparison between exact and predicted metric

The comparison between the exact and predicted metric has been performed using two images: the first one represents the absolute value of the difference between the logarithms of the exact metric at t_m and of the exact metric at t_n , reminding that the exact metric is computed directly from the CFD solution.

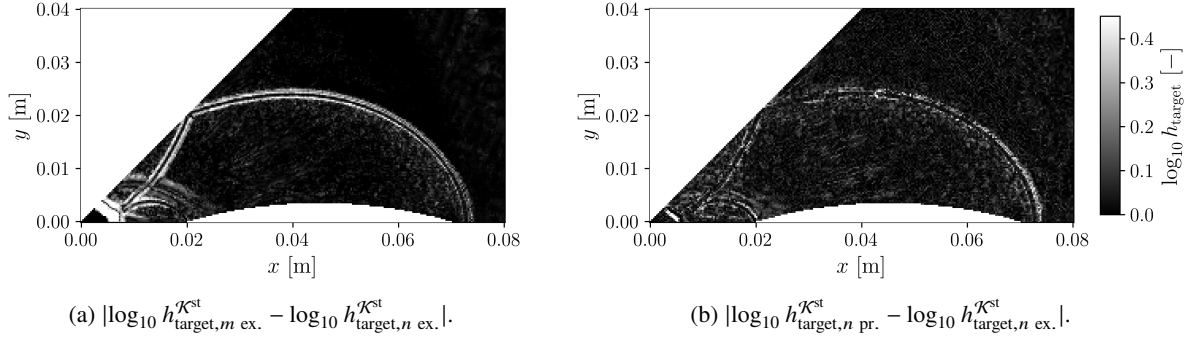


Figure 14: Comparison between exact and predicted metric, $t_m = 4.51875 \cdot 10^{-4}$ s.

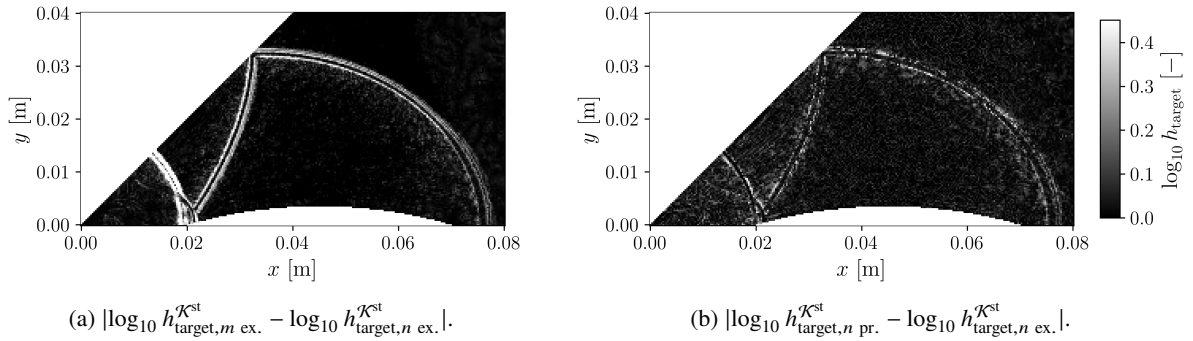


Figure 15: Comparison between exact and predicted metric, $t_m = 4.803125 \cdot 10^{-4}$ s.

This difference represents the error that would be present in the metric if the mesh created by the metric computed at t_m is used to obtain the solution at t_n (like most standard unsteady mesh adaptation strategies, as stated in Sec. 1). The second image, instead, represents the absolute value of the difference between the logarithms of the predicted metric at t_n and of the exact metric at t_n , where “predicted” refers to the extrapolated metric using the prediction workflow. For this test case, since in the proximity of the focus point the shock waves change topology quite often, too long extrapolations are avoided; in particular, $t_n = t_{m+40}$ has been chosen. The *minimum search* approach has been applied, with radius $r_{\text{ms}} = 0.0003$ m.

Figures 14 and 15 show the comparison respectively for time instants $t_m = 4.51875 \cdot 10^{-4}$ s (motion field represented in Figure 13a) and $t_m = 4.803125 \cdot 10^{-4}$ s (motion field represented in Figure 13b). For both time instants, the predicted metric at t_n matches the exact metric at t_n much better than the exact metric at t_m , meaning that all the flow features of the predicted metric have been correctly advected to overlap with their exact location. From this comparison, it is clear that, also in cases like this, where the shape and interactions of the features are complex, the metric prediction algorithm is able to get close to the exact metric in advance with good accuracy.

6. Conclusions and future work

This work introduced a predictive mesh adaptation strategy based on computer-vision techniques. The method estimates the motion of the metric field between two available time levels and extrapolates its evolution to define a target grid spacing suitable for subsequent time steps.

The approach was assessed through *a posteriori* analyses of three unsteady test cases: a one-dimensional piston problem, a cylindrical shock propagating in an unobstructed domain, and a cylindrical shock interacting with an obstacle. In all cases, the predicted metric followed the evolution of the main flow features over multiple time steps, demonstrating the potential of the method to reduce the remeshing

frequency while maintaining adequate spatial resolution.

The main limitation of the current formulation is its inability to predict changes in the motion of flow structures caused by reflections, interactions with solid boundaries, or interactions with other flow features. Excessively long extrapolation intervals should therefore be avoided. Moreover, the current implementation requires manual selection of parameters such as the temporal separations among t_l , t_m , and t_n , as well as the estimated maximum propagation speed of the flow features.

Future developments will focus on reducing the computational cost by evaluating the motion field only in regions containing relevant flow features, extending the method to three-dimensional and multiscale problems, and generalizing it to anisotropic mesh adaptation. Applications involving unsteady Navier–Stokes equations, turbulence models, and moving geometries will also be considered.

The next step is to assess the method within a fully coupled CFD and mesh adaptation workflow. This evaluation will determine whether the reduction in remeshing frequency produces measurable improvements in computational cost and solution accuracy compared with standard unsteady adaptation strategies.

References

- [1] A. Loseille, A. Dervieux, and F. Alauzet. Fully anisotropic goal-oriented mesh adaptation for 3D steady Euler equations. *J. Comput. Phys.*, v.229:2866–2897, 2010.
- [2] M. Fossati, A. Guardone, and L. Vigevano. A Node-Pair Finite Element/Volume Mesh Adaptation Technique for Compressible Flows. *Int. J. Numer. Meth. Fluids*, v.70(8):1004–1026, 2012.
- [3] N. Wyman, P. Galpin, and T. Hansen. Robust, Efficient and Accurate Mesh Adaptation for Turbomachinery CFD Simulations. In *AIAA Propulsion and Energy Forum*, 2020.
- [4] W. G. Habashi, J. Dompierre, Y. Bourgault, D. Ait-Ali-Yahia, M. Fortin, and M.-G. Vallet. Anisotropic mesh adaptation: towards user-independent, mesh-independent and solver-independent CFD. Part I: general principles. *Int. J. Numer. Meth. Fluids*, v.32:725–744, 2000.
- [5] F. Alauzet and A. Loseille. A decade of progress on anisotropic mesh adaptation for computational fluid dynamics. *Computer-Aided Design*, v.72:13–39, 2016.
- [6] B. Re, C. Dobrzynski, and A. Guardone. An interpolation-free ALE scheme for unsteady inviscid flows computations with large boundary displacements over three-dimensional adaptive grids. *J. Comput. Phys.*, v.340:26–54, 2017.
- [7] C. Dobrzynski and P. Frey. Anisotropic Delaunay Mesh Adaptation for Unsteady Simulations. In *Proceedings of the 17th International Meshing Roundtable*, pages 177–194, 2008.
- [8] J.-F. Remacle, X. Li, M. S. Shephard, and J. E. Flaherty. Anisotropic adaptive simulation of transient flows using discontinuous Galerkin methods. *Int. J. Numer. Meth. Engng*, v.62:899–923, 2005.
- [9] M. J. Castro-Díaz, F. Hecht, B. Mohammadi, and O. Pironneau. Anisotropic Unstructured Mesh Adaption for Flow Simulations. *Int. J. Numer. Meth. Fluids*, v.25:475–491, 1997.
- [10] O. Hassan, K. Morgan, and N. Weatherill. Unstructured mesh methods for the solution of the unsteady compressible flow equations with moving boundary components. *Phil. Trans. R. Soc. A*, v.365:2531–2552, 2007.
- [11] S. Mozaffari, E. Guilmineau, M. Visonneau, and J. Wackers. Average-based mesh adaptation for hybrid RANS/LES simulation of complex flows. *Computers and Fluids*, v.232:105202, 2022.

- [12] F. Alauzet, P. L. George, B. Mohammadi, P. Frey, and H. Borouchaki. Transient fixed point-based unstructured mesh adaptation. *Int. J. Numer. Meth. Fluids*, v.43:729–745, 2003.
- [13] F. Alauzet, P. J. Frey, P. L. George, and B. Mohammadi. 3D transient fixed point mesh adaptation for time-dependent problems: Application to CFD simulations. *J. Comput. Phys.*, v.222:592–623, 2007.
- [14] R. Löhner. Mesh adaptation in fluid mechanics. *Eng. Fract. Mech.*, v.50(5/6):819–847, 1995.
- [15] T. Carrigan and N. Wyman. Geometry-Sensitive, Solver-Independent Mesh Adaptation Using Hybrid Viscous and Hex-Core Meshing Techniques. *Advanced Modeling & Simulation (AMS) Seminar Series*, NASA Ames Research Center, 2021.
- [16] S. Nayar. Object Tracking using Template Matching. *First Principles of Computer Vision (Columbia University)*, <https://www.youtube.com/watch?v=2wwVogo88aY>, 2021.
- [17] M. Raffel, C. E. Willert, S. T. Wereley, and J. Kompenhans. *Particle Image Velocimetry, a practical guide*. Springer, 2007.
- [18] U. Germann and I. Zawadzki. Scale-Dependence of the Predictability of Precipitation from Continental Radar Images. Part I: Description of the Methodology. *Monthly Weather Review*, 130(12), 2002.
- [19] P. A. Thompson. *Compressible-Fluid Dynamics*. McGraw-Hill, 1988.
- [20] C. Geuzaine and J.-F. Remacle. Gmsh: A 3-D finite element mesh generator with built-in pre- and post-processing facilities. *Int. J. Numer. Meth. Engng*, v.79(11):1309–1331, 2009.
- [21] P. Roe. Approximate Riemann Solvers, Parameter Vectors, and Difference Schemes. *J. Comput. Phys.*, v.43:357–372, 1981.
- [22] B. van Leer. Towards the Ultimate Conservative Difference Scheme. V. A Second-Order Sequel to Godunov’s Method. *J. Comput. Phys.*, v.32(1):101–136, 1979.
- [23] V. Venkatakrishnan. Convergence to Steady State Solutions of the Euler Equations on Unstructured Grids with Limiters. *J. Comput. Phys.*, v.118:120–130, 1995.
- [24] G. Guderley. Starke Kugelige Zylindrische Verdichtungsstöße in Der Nähe Des Kugelmittelpunktes Bzw. Der Zylinderachse. *Luftfahrtforschung*, v.19, 1942.
- [25] F. Vignati and A. Guardone. Dynamics of Cylindrical Converging Shock Waves Interacting with Aerodynamic Obstacle Arrays. *Phys. Fluids*, v.27(6), 2015.
- [26] F. Vignati. *Dynamics of cylindrical converging shock waves interacting with circular-arc obstacles*. PhD thesis, Politecnico di Milano, DAER, 2015.