MOX-Report No. 57/2017

A POD-Selective Inverse Distance Weighting method for fast parametrized shape morphing

Ballarin, F.; D'Amario, A.; Perotto, S.; Rozza, G.

MOX, Dipartimento di Matematica
Politecnico di Milano, Via Bonardi 9 - 20133 Milano (Italy)

mox-dmat@polimi.it http://mox.polimi.it

A POD-Selective Inverse Distance Weighting method for fast parametrized shape morphing

Francesco Ballarin[♯], Alessandro D'Amario[♯,‡], Simona Perotto[♭], Gianluigi Rozza[♯]

October 27, 2017

[♯] mathLab, Mathematics Area, SISSA, via Bonomea 265, I-34136 Trieste, Italy
francesco.ballarin@sissa.it, gianluigi.rozza@sissa.it
[♭] MOX, Dipartimento di Matematica, Politecnico di Milano,
Piazza Leonardo da Vinci 32, I-20133 Milano, Italy
simona.perotto@polimi.it
[‡] Politecnico di Milano, via Giuseppe La Masa, 34, I-20156 Milano, Italy
alessandro.damario1989@gmail.com

Keywords: Shape Morphing, Inverse Distance Weighting, Proper Orthogonal Decomposition

Abstract

Efficient shape morphing techniques play a crucial role in the approximation of partial differential equations defined in parametrized domains, such as for fluid-structure interaction or shape optimization problems. In this paper, we focus on Inverse Distance Weighting (IDW) interpolation techniques, where a reference domain is morphed into a deformed one via the displacement of a set of control points. We aim at reducing the computational burden characterizing a standard IDW approach without compromising the accuracy. To this aim, first we propose an improvement of IDW based on a geometric criterion which automatically selects a subset of the original set of control points. Then, we combine this new approach with a model reduction technique based on a Proper Orthogonal Decomposition of the set of admissible displacements. This choice further reduces computational costs. We verify the performances of the new IDW techniques on several tests by investigating the trade-off reached in terms of accuracy and efficiency.

1 Introduction

Shape morphing plays a meaningful role in several engineering and life science fields, such as, for instance, aero-elasticity [40], high performance boat design [31, 30], modelling of the cardiovascular system [4, 35]. On one hand, the same

physical problem may involve the deformation of the domain, for instance, when dealing with fluid-structure interaction problems [8]. On the other hand, several applications imply iterative procedures where the same problem is solved in different geometric configurations. This is the case, e.g., of a multi-query context, such as shape optimization problems, where the shape of the domain is varied until a prescribed cost functional is minimized (or maximized) [43].

Our interest is focused on partial differential equations solved in a domain which changes in time. In such a case, the goal becomes twofold, since we aim at accurately approximate the domain as well as the differential problem. In general, it is not computationally affordable to generate a new discretization (mesh) of the domain at each deformed configuration. Indeed, mesh generation may be a time consuming procedure (both in terms of CPU time and assembling) and, sometimes, it is not directly integrated in the solver at hand. Therefore, it is often more convenient to generate a mesh for a reference configuration and then to morph it into the deformed grid. This work can be framed in such a context.

Several shape morphing techniques are available in the computer graphics community. A reference mesh is deformed by displacing some points (the so-called control points); then the shape morphing map associated with these points is applied to the whole discretized domain, thus avoiding any remeshing. These techniques have been recently applied to parametrized partial differential equations (PPDEs). For instance, shape morphing techniques based on *Free Form Deformation* (FFD) [41] and *Radial Basis Functions* (RBF) interpolation [12] have been successfully applied to PPDEs [29, 33], shape optimization problems [6, 44, 45], reconstruction of scattered geometrical data [18], mesh motion and interface coupling for fluid-structure interaction problems [10, 21, 23], interpolation between non-conforming meshes [19], sensitivity analysis studies in complex geometrical configurations [5].

In this paper we focus on a different approach, the *Inverse Distance Weighting* (IDW) interpolation that relies on the inverse of a weighted sum of distances between mesh nodes, some of which will be chosen as control points [42, 50, 51, 22].

A proper choice of the control points is a crucial step independently of the adopted morphing technique. A first requirement is to keep the number of control points as small as possible, since the complexity of the morphing map evaluation increases as the number of control points becomes larger. Such selection is usually driven by a prior knowledge of the phenomenon at hand. In some cases an automatic selection can be carried out. For instance, a sensitivity analysis to the control points is proposed for shape optimization problems in [6], where the control points selected are the ones providing the largest variation of a cost functional. Local minima and maxima of structure eigenmodes are employed in [23] as control points to perform mesh motion in fluid-structure interaction problems.

Here, we propose a selective version of the IDW shape morphing procedure

based on geometrical considerations and, consequently, independent of the underlying partial differential equations. We further reduce the computational cost of the morphing by coupling, in a straightforward way, the Selective IDW (SIDW) interpolation with a suitable model reduction technique. In particular, thanks to the employment of a reference domain, we identify any shape deformation with a certain parametric configuration of the original structure, the parameter being strictly related to the control point displacement. This suggests us to resort to a Proper Orthogonal Decomposition (POD) as model reduction approach. We will refer to this combined technique as to POD-SIDW. POD, also known as Principal Component Analysis (PCA), is used in the literature, for instance, in statistical shape analysis [20], as well as for mesh deformation and optimization [3, 48]. Manifold learning techniques are combined with PCA for efficient structural shape optimization in [36]. An equivalent shape representation by means of currents is sought in [46] in order to introduce an Hilbert space over shapes and apply PCA. Application to life sciences of PCA has been proposed in [26, 34].

The paper is organized as follows. In Section 2, after formalizing the standard approach, we set the selective variant of IDW interpolation. Both the approaches are customized in a shape morphing setting and numerically compared on structural configurations of interest in aeronautic and naval engineering. Section 3 deals with model reduction. POD is directly applied to the selective IDW interpolation and the numerical benefits due to such a merging of techniques are investigated when dealing with shape morphing. Some conclusions are finally drawn in Section 4 together with perspectives for a future development of the present work.

2 Inverse Distance Weighting techniques

In this section, we consider two techniques to drive a shape morphing process, with particular interest in fluid-structure interaction (FSI) problems. In particular, first we introduce the original Inverse Distance Weighting (IDW) interpolation, and then we propose a new variant to overcome some of the intrinsic computational limits of such an approach.

2.1 The Inverse Distance Weighting (IDW) approach

The IDW method has been originally proposed in [42] to deal with two-dimensional interpolation problems, and successively extended to higher-dimensional and applicative settings (see, e.g., [15, 32, 51, 50]).

Let $\Omega_r \subset \mathbb{R}^d$, with $d = 2, 3$, be the reference domain, and let $u : \Omega_r \to \mathbb{R}$ be a continuous function. We select a prescribed set $\mathcal{C} = \{\mathbf{c}_k\}_{k=1}^{\mathcal{N}_c} \subset \Omega_r$ of points in Ω_r, known as *control points*. Then, the *IDW interpolant* of u, $\Pi_{\mathrm{IDW}}(u)$, coincides

with the continuously differentiable function

$$\Pi_{\mathrm{IDW}}(u)(\mathbf{x}) = \sum_{k=1}^{\mathcal{N}_c} w_k(\mathbf{x})\, u(\mathbf{c}_k) \quad \mathbf{x} \in \Omega_r, \tag{1}$$

where the local weight functions $w_k : \Omega_r \to \mathbb{R}$, for $k = 1, \ldots, \mathcal{N}_c$, are defined by

$$w_k(\mathbf{x}) = \begin{cases} \dfrac{\|\mathbf{x} - \mathbf{c}_k\|^{-p}}{\displaystyle\sum_{j=1}^{\mathcal{N}_c} \|\mathbf{x} - \mathbf{c}_j\|^{-p}} & \text{if } \mathbf{x} \neq \mathbf{c}_k, \\ 1 & \text{if } \mathbf{x} \equiv \mathbf{c}_k, \\ 0 & \text{otherwise,} \end{cases} \tag{2}$$

with $\|\cdot\|$ the standard Euclidean norm, and for some integer p. Notice that weights are automatically selected such that the influence of the k-th control point $\mathbf{c}_k$ on $\mathbf{x}$ diminishes as the distance between $\mathbf{x}$ and $\mathbf{c}_k$ increases. Power p tunes such an inverse dependence, so that the weight assigned to a point $\mathbf{x}$ far from the control points becomes smaller and smaller as p increases. With reference to FSI problems, the choice of parameter p is crucial to avoid compenetration effects between fluid and structure [50].

2.1.1 IDW for shape morphing

To set shape morphing into the IDW interpolation framework, we identify the reference domain Ω_r with the initial configuration of the physical domain. Then, we consider a discretization $\Omega_h = \{\mathbf{x}_i\}_{i=1}^{\mathcal{N}_h}$ of Ω_r, and we identify function u in (1) with the displacement d of points $\mathbf{x}_i$ and the set $\mathcal{C}$ of the control points (also denoted as parameters) with a subset of Ω_h. In such a context, the IDW interpolant, $\Pi_{\mathrm{IDW}}(d)$, represents the so-called *shape parametrization map*.
The choice of the control points clearly represents a crucial issue. In general, $\mathcal{C}$ coincides with the set of the boundary points of Ω_h. This is a common practice in several engineering applications, in particular, in FSI problems, where the displacement is prescribed at the interface between fluid and structure and, successively, extended to the interior of the domain to identify the deformed domain, Ω_d [9, 13, 37].

To formalize shape morphing in a more computationally practical way, we collect the displacements assigned at the control points in the vector $\mathbf{d}_c \in \mathbb{R}^{\mathcal{N}_c}$, with $[\mathbf{d}_c]_i = d(\mathbf{c}_i)$ and $i = 1, \ldots, \mathcal{N}_c$. In practice, values $d(\mathbf{c}_i)$ are often constrained to suitable ranges in order to satisfy admissible shape configurations (see, e.g., [29]). Then, the displacement $\mathbf{d} \in \mathbb{R}^{\mathcal{N}_h}$ of points $\mathbf{x}_i$ in Ω_h, with $[\mathbf{d}]_i = d(\mathbf{x}_i)$ and $i = 1, \ldots, \mathcal{N}_h$, is computed via the IDW interpolant (1)-(2) as

$$\mathbf{d} = W \mathbf{d}_c, \tag{3}$$

with $W \in \mathbb{R}^{\mathcal{N}_h \times \mathcal{N}_c}$ the IDW matrix of generic component

$$W_{ik} = w_k(\mathbf{x}_i) \quad \text{for } i = 1, \dots \mathcal{N}_h \text{ and } k = 1, \dots, \mathcal{N}_c. \tag{4}$$

Matrix W keeps trace of the internal structure of Ω_h, recording the reciprocal distance between control and internal points. For this reason, the IDW matrix is computed before morphing takes place, once and for all. The actual motion is impressed by vector $\mathbf{d}_c$, which, vice versa, varies during the morphing process in order to follow the shape deformation. Finally, vector $\mathbf{d}$ identifies the deformed domain Ω_d.

Relation (3) highlights the easiness of implementation of a morphing strategy driven by IDW interpolation. In particular, when applied to FSI configurations, this approach provides a sharp description of the interface displacement, by properly tackling also portions of the domain characterized by a null displacement and by avoiding compenetrations effects. Additionally, a good mesh quality is usually guaranteed (we refer to Section 2.3 for more details), even in the presence of large deformations.
Nevertheless, the standard IDW approach does not prescribe, a priori, any smart criterion to select the control points. The value $\mathcal{N}_c$ may consequently become very large, especially in the presence of practical configurations, so that the computational effort required by the assembly and by the storage of matrix W may be very massive. This justifies the proposal of the new IDW formulation in the next section, where a new set $\widehat{\mathcal{C}}$ of control points is properly selected via an automatic procedure, so that $\text{card}(\widehat{\mathcal{C}}) \ll \mathcal{N}_c$, $\text{card}(\mathcal{S})$ denoting the cardinality of the generic set $\mathcal{S}$. Sparsification of matrix W provides an alternative way to reduce the computational burden of IDW [38, 39], though beyond the goal of this work.

2.2 The Selective Inverse Distance Weighting (SIDW) approach

The new procedure proposed in this section aims at reducing the computational effort demanded by the standard IDW interpolation, without loosing the good properties of the original approach. In particular, since the most computationally demanding part of the IDW algorithm is the memory storage, we aim at reducing the number of control points by automatically selecting the most relevant points in $\mathcal{C}$ to sharply describe the initial configuration Ω_h as well as the deformed domain Ω_d. For this reason, we refer to the new approach as to the Selective IDW (SIDW) formulation.

Starting point is the approach proposed in [42] which is essentially suited to deal with regimes of small deformations. Our goal is to improve this procedure to tackle also large displacement configurations, without violating the no-compenetration constraint. Additionally, we aim at guaranteeing a uniform distribution in the reduced set of control points.

The complete SIDW procedure is provided by **Algorithm 1**. To set notation, we denote by: $\widehat{\mathcal{C}} = \{\widehat{\mathbf{c}}_j\} \subset \mathcal{C}$ the subset of the selected control points; $\omega \subseteq \Omega_h$ the region where the selection is applied; $R > 0$ the selection radius; $B(\mathbf{c}; r)$ the

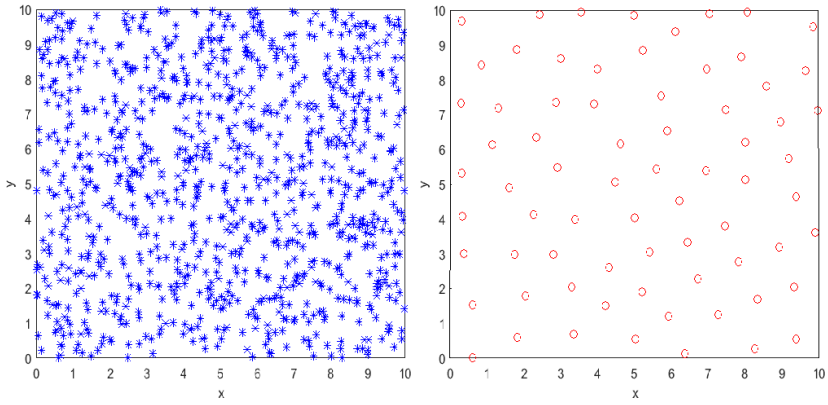


Figure 1: SIDW algorithm: initial (left) and final (right) distribution of control points.

ball of center $\mathbf{c}$ and radius $r > 0$; $B(\mathbf{c}; r_i, r_e)$ the circular annulus of center $\mathbf{c}$, inner radius r_i and outer radius r_e, with $r_e > r_i > 0$.
We first exemplify the action of the SIDW procedure starting from the configuration in Figure 1, left, where control points are distributed on the whole domain $\Omega_h = (0, 10)^2$, and where $\omega \equiv \Omega_h$. Then, we will particularize such a procedure to a shape morphing setting.

SIDW algorithm consists essentially of two phases, i.e., an *initialization* phase and the actual *selection* phase. Now, let $\widehat{\mathbf{c}}_1 \in \mathcal{C}$ be a randomly selected control point.

Goal of the initialization phase is to perform a tessellation of the region $\omega \setminus B(\widehat{\mathbf{c}}_1; R)$ via n concentric annuli, $\{\alpha_m\}_{m=1}^n$, of thickness aR (`lines 2-7`), being $a \in \mathbb{R}$ a positive constant picked by the user, with $a < 1$. Concerning the specific example in Figure 1, we refer to Figure 2(a), where the tessellation corresponding to $R = 2.1$ and $a = 0.8$ is shown, the control point $\widehat{\mathbf{c}}_1$ being marked in the plot. To simplify the graphical representation, we highlight only the control points constituting $\widehat{\mathcal{C}}$. Four annular regions are identified by the choice done for the input parameters, thus inducing a partitioning of the original control points in $\mathcal{C}$. Then, all the control points in $\mathcal{C}$ belonging to the *area of influence*, $B(\widehat{\mathbf{c}}_1; R)$, around $\widehat{\mathbf{c}}_1$ are removed from $\mathcal{C}$. This operation concludes the first phase of SIDW algorithm.

The selection phase begins with a first initialization of the *area of selection*, β, around the first control point $\widehat{\mathbf{c}}_1$, defined as the intersection between α_1 and the circular annulus $B(\widehat{\mathbf{c}}_1; R, bR)$, with $b > 1$ a user defined positive constant (`line 8`). Then, a loop on the annular regions $\{\alpha_m\}_{m=1}^n$ of the tessellation is started (`line 9`). Inside this loop, until β is empty, a new control point is selected in β via the `SelectControlPoint` function (`line 11`). Then, all the control points in $\mathcal{C}$ belonging to each area of influence α_l around the current control point $\widehat{\mathbf{c}}_{k+1}$ are removed (`lines 12-14`), and a new area of selection β

Algorithm 1 : SIDW interpolation

function SIDW(ω, $\mathcal{C}$, $\widehat{\mathbf{c}}_1$, R, a, b, u)
  $n \leftarrow 1$, $r \leftarrow R$, $r_\omega \leftarrow \min\{r > 0 : \omega \supseteq B(\widehat{\mathbf{c}}_1; r)\}$
  while $r \leq r_\omega$ **do**
    $\alpha_n \leftarrow B(\widehat{\mathbf{c}}_1; r, r + aR) \cap \mathcal{C}$
    $r \leftarrow r + aR$
    $n \leftarrow n + 1$
  end while
  $k \leftarrow 1$, $m \leftarrow 1$, $\beta \leftarrow \alpha_1 \cap B(\widehat{\mathbf{c}}_1; R, bR)$
  while $m \leq n$ **do**
    while $\beta \neq \emptyset$ **do**
      $\widehat{\mathbf{c}}_{k+1} \leftarrow$ SelectControlPoint(β)
      for $l = m, \ldots, n$ **do**
        $\alpha_l \leftarrow \alpha_l \setminus B(\widehat{\mathbf{c}}_{k+1}; R)$
      end for
      $\beta \leftarrow \alpha_m \cap B(\widehat{\mathbf{c}}_{k+1}; R, bR)$
      $k \leftarrow k + 1$
    end while
    if $\alpha_m = \emptyset$ **then**
      $\beta \leftarrow \alpha_{m+1} \cap B(\widehat{\mathbf{c}}_k; R, bR)$
      $m \leftarrow m + 1$
    else
      $\beta \leftarrow \alpha_m$
    end if
  end while
  $\widehat{\mathcal{C}} \leftarrow \{\widehat{\mathbf{c}}_j\}_{j=1}^{k}$
  computation of $\Pi_{\mathrm{SIDW}}(u)$
end function

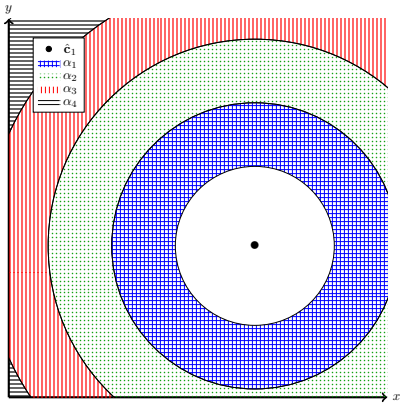
is computed as the intersection between the current area of influence α_m and a circular annulus $B(\widehat{\mathbf{c}}_{k+1}; R, bR)$ (`line 15`).
Concerning function `SelectControlPoint`, different criteria can be implemented to select the new control point $\widehat{\mathbf{c}}_{k+1}$. For instance, a geometric selection can be applied by picking the control point closest to the centroid of β or the farthest control point with respect to the already selected points in $\widehat{\mathcal{C}}$. To simplify the procedure, in the numerical assessment below, we resort to a random choice of $\widehat{\mathbf{c}}_{k+1}$ in β.
Figure 2(b) displays the firstly initialized area of selection β, together with the second control point $\widehat{\mathbf{c}}_2$, being $b = 1.4$. The remaining panels in Figure 2 show the effects of the successive removals operated by the `while` loop. Six iterations are associated with the first area of influence α_1. In particular, the new area of selection β and the corresponding control point $\widehat{\mathbf{c}}_{k+1}$ associated with the first two (Figure 2(c)-(d)) and the last (Figure 2(e)) removals are highlighted. We observe that, as expected, the initial tessellation gradually becomes empty (the white area becomes larger and larger), while the new control points uniformly distribute.
When β is empty, a re-initialization of the area of selection is performed (`lines 18-23`). If the current annulus α_m is exhausted, then the area of selection is sought in the next annulus, α_{m+1}, (`lines 19-20`). Figure 2(f) shows the transition from α_1 to α_2. Additionally, `line 22` handles the very peculiar case when α_m is not actually exhausted, yet it is disconnected from the area of selection around the current control point $\widehat{\mathbf{c}}_{k+1}$. In such a case, β is simply reset to α_m.

Finally, once all the circular annuli in the tessellation $\{\alpha_m\}_{m=1}^n$ are exhausted, SIDW algorithm ends, and the new set $\widehat{\mathcal{C}}$ of selected control points is returned (`line 25`). The final distribution of control points for the considered specific configuration is provided in Figure 1, right. The selected points are uniformly distributed as desired. At this point, the *SIDW interpolant* of u, $\Pi_{\text{SIDW}}(u)$, can be computed (`line 26`) as

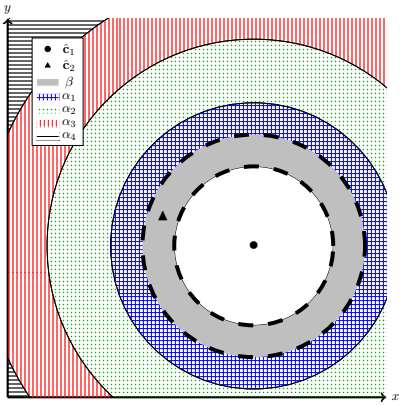
$$\Pi_{\text{SIDW}}(u)(\mathbf{x}) = \sum_{k=1}^{\mathcal{N}_{\widehat{c}}} w_k(\mathbf{x})\, u(\widehat{\mathbf{c}}_k) \quad \mathbf{x} \in \Omega_r, \tag{5}$$

with $\text{card}(\widehat{\mathcal{C}}) = \mathcal{N}_{\widehat{c}} \ll \mathcal{N}_c$, and where the weight functions w_k are defined according to (2).

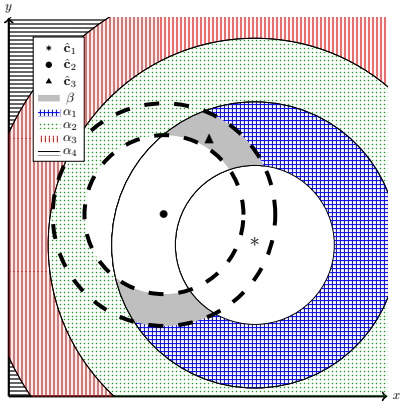
Few remarks are in order. The three input parameters, R, a and b, tune the SIDW algorithm and, as a consequence, the final selection $\widehat{\mathcal{C}}$ of control points. In particular, they vary the area of the annular regions $\{\alpha_m\}_{m=1}^n$, as well as of the areas of influence and selection. In more details, concerning the radius R, it turns out that the smaller R, the larger the number of selected control points. As far as a and b are concerned, smaller values of a lead to a larger number n of annular regions α_m, which generally results in a finer control point distribution; larger values of b leads to larger areas of selection β, resulting in a smaller number of selected control points. The choice of R, a and b as well



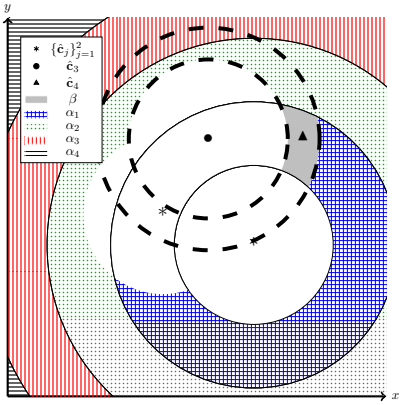
(a) Initialization phase.



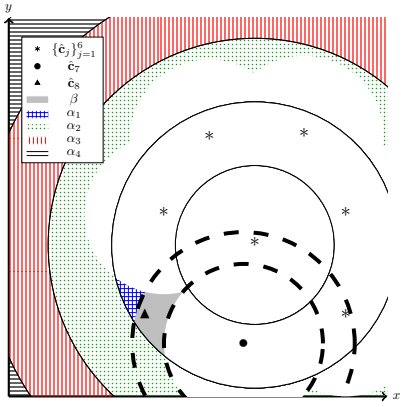
(b) Selection phase: identification of β and of $\widehat{\mathbf{c}}_2$.



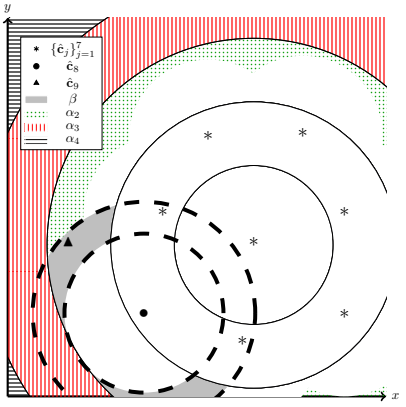
(c) Selection phase: first removal associated with α_1.



(d) Selection phase: second removal associated with α_1.



(e) Selection phase: sixth removal associated with α_1.



(f) Selection phase: first removal associated with α_2.

Figure 2: SIDW algorithm: initialization (a) and first iterations (b)-(f) of the selection phase applied to the configuration in Figure 1, left.

as the corresponding interplay is difficult to be established *a priori* and, clearly, it is problem-dependent. A sensitivity analysis of SIDW with respect to these parameters will be carried out in Section 2.3.

Finally, we remark that, especially in practical applications, it might be advisable to split ω into smaller subregions, ω_p, and to use various values for the parameters R, a and b to tackle possible local different discretizations.

2.2.1 SIDW for shape morphing

Now we customize SIDW algorithm to manage shape morphing. We consider the discretization $\Omega_h = \{\mathbf{x}_i\}_{i=1}^{\mathcal{N}_h}$ of the initial configuration of the physical domain and we assume to know the displacement d at each control point $\mathbf{c}_k \in \mathcal{C}$, for $k = 1, \ldots, \mathcal{N}_c$. We remind that, in such a case, $\mathcal{C} \subseteq \partial\Omega_h$.

Our idea consists in deforming the boundary $\partial\Omega_h$ of Ω_h, according to the deformation prescribed at $\{\mathbf{c}_k\}_{k=1}^{\mathcal{N}_c}$, while selecting a subset of control points to update the displacement of the internal nodes of Ω_h.
For this purpose, we resort to the procedure itemized in **Algorithm 2**, following a selection-deformation paradigm.
During the selection phase, which is performed once and for all, we first filter set $\mathcal{C}$ via the selection procedure in **Algorithm 1**, to obtain the subset $\widehat{\mathcal{C}}$ of control points $\widehat{\mathbf{c}}_k$, with $k = 1, \ldots, \mathcal{N}_{\widehat{c}}$ (`item (i)`). Then, we build the SIDW matrix $\widehat{W}$ according (4) associated with $\widehat{\mathcal{C}}$ and with the set of the internal nodes of Ω_h (`item (iii)`). As an option, we can add to the filtered set $\widehat{\mathcal{C}}$ extra control points to include possible specific constraints to the problem (`item (ii)`). This occurs, for instance, when a null displacement is assigned to a portion of the domain. In such a case, the fixed nodes have to be necessarily included in the set $\widehat{\mathcal{C}}$. In the sequel, we refer to this variant of the SIDW algorithm as to ESIDW (*Enriched* SIDW) interpolation.
The online phase performs the actual shape morphing. The boundary $\partial\Omega_h$ is deformed via the displacement assigned at the control points in $\mathcal{C}$ (`item (iv)`). Successively, we deform the internal nodes of Ω_h as

$$\mathbf{d}^i = \widehat{W}\mathbf{d}_{\widehat{c}}, \tag{6}$$

where $\mathbf{d}^i \in \mathbb{R}^{\mathcal{N}_h^i}$ collects the displacements of the internal nodes, $\mathbf{x}_j$ for $j = 1, \ldots, \mathcal{N}_h^i$, of Ω_h, $\widehat{W} \in \mathbb{R}^{\mathcal{N}_h^i \times \mathcal{N}_{\widehat{c}}}$ is the (E)SIDW matrix computed in the offline phase, and $\mathbf{d}_{\widehat{c}} \in \mathbb{R}^{\mathcal{N}_{\widehat{c}}}$ is the vector of the displacements of the selected control points in $\widehat{\mathcal{C}}$ (`item (v)`).

Algorithm 2 can be advantageously exploited to deal with FSI problems. In such a case, we identify domain Ω_h with the fluid domain. The displacement of the interface is generally provided by a structure solver, so that it suffices to compute the displacement of the fluid nodes via the procedure detailed in the algorithm.

Algorithm 2 SIDW interpolation for shape morphing

SELECTION PHASE:

(i) apply the selection procedure in **Algorithm 1** to $\mathcal{C}$ to extract the subset $\widehat{\mathcal{C}}$;

(ii) *Optional* : add to the set $\widehat{\mathcal{C}}$ extra control points to account for problem constraints;

(iii) assemble the SIDW matrix $\widehat{W}$ associated with $\widehat{\mathcal{C}}$ and with the internal nodes of Ω_h.

DEFORMATION PHASE:

(iv) deform the boundary $\partial\Omega$ via the displacement assigned at the control points in $\mathcal{C}$;

(v) deform the internal nodes of Ω_h according to (6).

2.3 IDW versus SIDW interpolation

In this section we investigate the performances of IDW and SIDW interpolation algorithm. In particular, we consider configurations of interest in aeronautic and naval engineering. We focus on 3D test cases and on tetrahedral meshes, although the procedure can be generalized to any dimension and to arbitrary meshes. Concerning the choice of the parameter p in (2), we set $p = 4$, essentially driven by numerical considerations. Such a choice allows us also to properly tackle the no-compenetration issue when dealing with FSI settings. To check the improvements led by the new approach, we compare IDW and SIDW interpolations in terms of computational effort. Both the procedures are implemented in the `C++` open source library `libMesh` [28], while the visualization software `ParaView` is employed for the post-processing of the solutions [2]. Concerning mesh generation, we resort to `SALOME` [1]. Finally, all the simulations are performed on a laptop with Intel® Core™ i7 CPU and 4GB RAM.

2.3.1 Structural deformation of a wing

We consider a wing characterized by a NACA0012 profile, clamped on the left side (see Figure 3, top). Table 1 gathers the main properties of the reference domain Ω_r and of the corresponding mesh Ω_h. We impose a vertical displacement to the wing. In particular, denoting by z the distance from the clamped side and by y the vertical direction, we assign the displacement

$$\delta y = \delta y(z) = 0.01 z^2 \tag{7}$$

in the y direction (see Figure 3, bottom).

First, we apply the standard IDW approach, after identifying $\mathcal{C}$ with the set of all the boundary nodes of $\partial\Omega_h$ (see Figure 4, top), so that the IDW matrix W in (3) belongs to $\mathbb{R}^{797\times 1666}$. The resulting deformed wing, shown in Figure 3, bottom, is obtained after 0.25 [s].

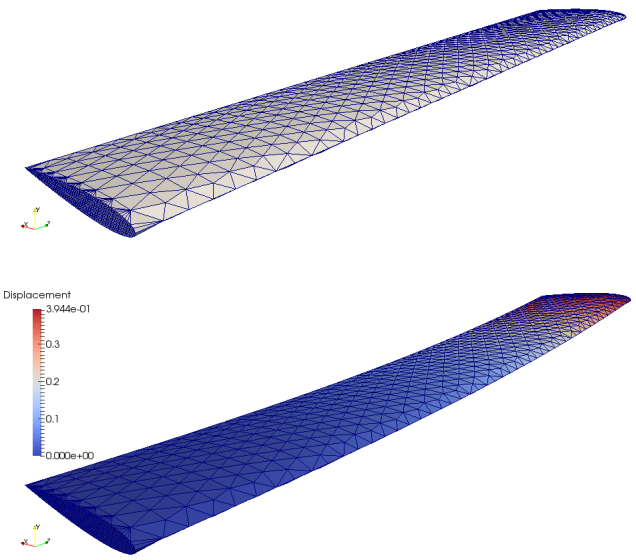


Figure 3: Structural deformation of a wing: reference (top) and IDW deformed (bottom) configuration.

longitudinal dimension	2π [m]
profile chord length	1.01 [m]
card(elements)	8850
card(nodes)	2463
card(internal nodes)	797
card(boundary nodes)	1666

Table 1: Structural deformation of a wing: main properties of Ω_r and of Ω_h.

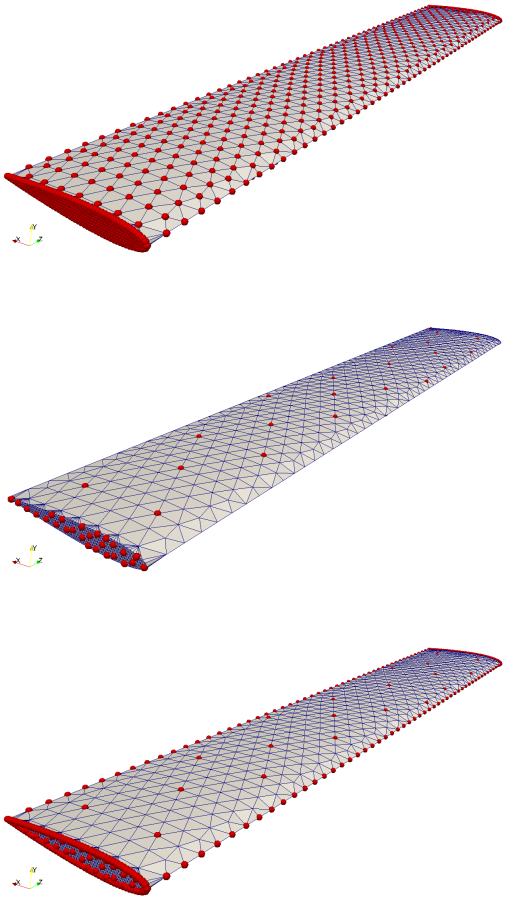


Figure 4: Structural deformation of a wing: control points associated with the IDW (top), SIDW (center) and ESIDW (bottom) interpolation.

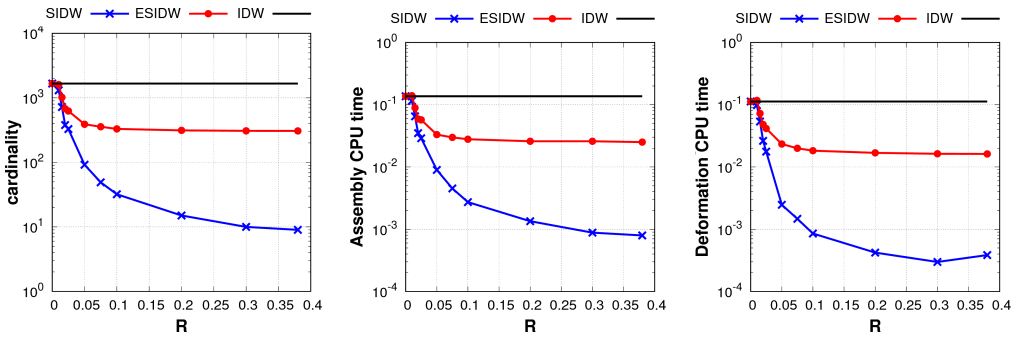


Figure 5: Structural deformation of a wing: cardinality of the control point set (left), assembly (center) and deformation (right) CPU time as a function of R.

We now resort to the SIDW interpolation algorithm. Set $\mathcal{C}$ still contains all the boundary nodes. We adopt two different values for the selection radius, i.e., $R = R_{lr} = 0.05$ [m] for the left and the right lateral surfaces of the wing, and $R = R_{tb} = 10 \cdot R_{lr} = 0.5$ [m] for the top and the bottom surfaces. Factor 10 is approximately the square root of the ratio, A_{tb}/A_{lr}, between the sum, A_{tb}, of the areas of the top and of the bottom surfaces of the wing, and the sum, A_{lr}, of the areas of the left and of the right sides. Finally, parameters a and b are set to 0.8 and 1.3, respectively.
We first employ the basic SIDW interpolation procedure, by skipping the enrichment step (ii). Figure 4, center highlights the new set $\widehat{\mathcal{C}} \subset \mathcal{C}$, consisting of 92 control points. Notice the very coarse and uniform distribution of control points $\widehat{\mathbf{c}}_k$. The deformed configuration yielded by the SIDW approach essentially coincides with the one in Figure 3, bottom, despite a considerable reduction of the interpolation matrix, being $\widehat{W} \in \mathbb{R}^{797\times 92}$, and of the corresponding computational time (the deformation phase takes only 0.015 [s]).
We successively resort to the ESIDW variant. We preserve the same values for a, b, R_{lr} and R_{tb} as in the previous numerical check. Additionally, we constrain the selection procedure to include in $\widehat{\mathcal{C}}$ all the nodes along the left and the right profiles of the wing as well as the nodes along the horizontal edges of the NACA profile. This requirement increases the cardinality of $\widehat{\mathcal{C}}$, now consisting of 390 control points. As shown in Figure 4, bottom, the constraints are guaranteed, while the distribution of the control points on the top (and on the bottom) of the wing is essentially the same as in Figure 4, center. Also in this case the deformed configuration is essentially the same as the one in Figure 3, bottom. Concerning the computational time, the increased cardinality of $\widehat{\mathcal{C}}$ results in an deformation stage that takes 0.034 [s]. This time, despite larger than the one demanded by SIDW, is still one order of magnitude less with respect to the time required by the standard IDW interpolation.

Sensitivity to R. We investigate the sensitivity of SIDW and ESIDW interpolation to the selection radius. In particular, in Figure 5, we show the trend, as a function of R, of the number of the control points (left), and of the CPU time (in seconds) required to assemble the interpolation matrix (center) and to compute the deformation of the wing (right), respectively.
The relation between the number of control points and R is nonlinear for both the SIDW and ESIDW approaches. Nevertheless, while for the SIDW interpolation, the number of control points (and, consequently, the associated CPU time) increases as R becomes smaller, a low sensitivity to R is shown by the ESIDW variant. In particular, the cardinality of $\widehat{\mathcal{C}}$ remains the same for R greater than 0.1.
Concerning the CPU times, no significant difference distinguishes the trend associated with the matrix assembly and the deformation step, also quantitatively. As expected, ESIDW interpolation is more computationally demanding than the

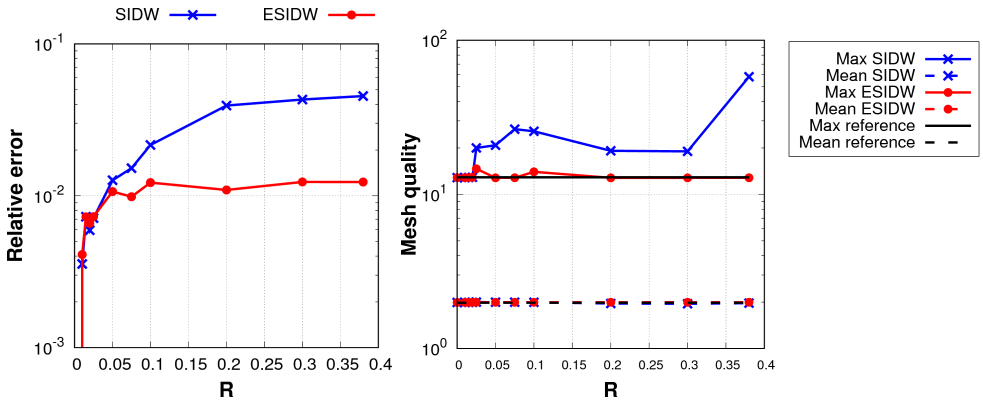


Figure 6: Structural deformation of a wing: relative error (left) and maximum and mean mesh quality (right) as a function of R.

basic SIDW approach, whereas the standard IDW approach coincides with the most expensive procedure.

In Figure 6, left we show the error trend as a function of the selection radius. We compute the $L^2(\Omega_h)$-norm of the relative error between the SIDW (ESIDW) and the IDW deformation. The larger number of control points employed by the ESIDW interpolation yields more accurate deformations compared with the ones provided by the SIDW approach. In particular, for large enough values of R, the error due to SIDW is approximately 4%, while, for the ESIDW procedure, it is about 1%. Convergence to zero is guaranteed by both the methods as R decreases.

Finally, in Figure 6, right we investigate the influence of the selected radius R on the quality of the elements of the deformed meshes. Different criteria can be employed to quantify the mesh quality $\mathcal{Q}$. Here, we adopt the ratio between the longest and the shortest edge [24]. In the figure, for different values of R, we compare the maximum and the mean value of $\mathcal{Q}$ on the meshes generated by SIDW and ESIDW algorithms, with the corresponding values associated with the initial configuration Ω_h. While, on average, the mean value of $\mathcal{Q}$ is essentially independent of the deformation procedure and of the selected R, more sensitivity is appreciable on the maximum value of $\mathcal{Q}$. The sufficiently large number of control points allows ESIDW to preserve about the same value of $\mathcal{Q}$ as for the initial mesh, also when R increases. On the contrary, a deterioration on the maximum mesh quality is evident when dealing with the SIDW interpolation algorithm, especially for large values of R.

Sensitivity to a and b. We conclude this first test case, by studying the sensitivity of the control point cardinality and of the accuracy of SIDW interpolation on the parameters a and b. Similar conclusions can be drawn for the ESIDW approach. To simplify this analysis, we relate a and b so that $b = 1/a$,

we pick $a < 1$ (i.e., $b > 1$), and we consider four different values of the selection radius, namely, $R = 0.02$, 0.05, 0.2 and 0.3. Figure 7 collects the results of this check.

For $a \ll 1$ (i.e., $b \gg 1$), the number of selected control points becomes larger and larger, as expected, with a consequent increment of the CPU time. Nevertheless, this does not necessarily entail an improvement in terms of accuracy, especially for large values of R. Indeed, for $R = 0.2, 0.3$, choosing $a \approx 1$ (i.e., $b \approx 1$) reduces the number of control points of about one order of magnitude with respect to the choice $a \ll 1$ (i.e., $b \gg 1$), while only a slightly lower relative error is guaranteed (see Figure 7, right). On the contrary, when R is small (see Figure 7, left), decreasing a (i.e., increasing b) improves the accuracy. This is due the fact that, for these choices of a (b) and R, the SIDW procedure selects almost all the available control points.

Thus, since the actual goal of SIDW procedures, is to reduce the computational burden, we are essentially interested in sufficiently large values of R. This suggests us that the choice $a \approx 1$ (i.e., $b \approx 1$) ensures a reasonable trade-off between efficiency and accuracy to the morphing procedure.

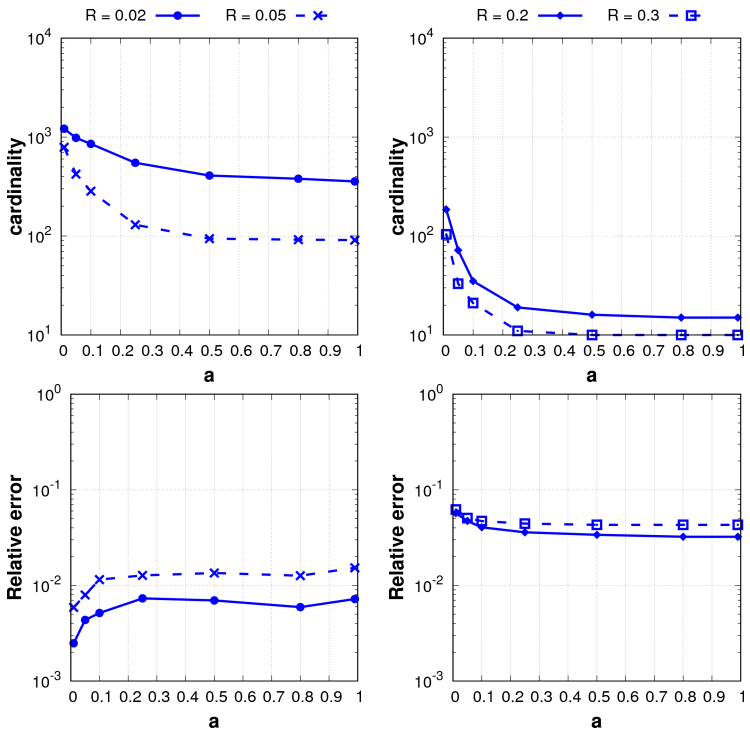


Figure 7: Structural deformation of a wing: cardinality of the control point set (top) and $L^2(\Omega_h)$-norm of the relative error (bottom) as a function of a, for $R = 0.02, 0.05$ (left) and for $R = 0.2, 0.3$ (right).

wind tunnel dimensions	$10 \times 5 \times 4\pi$ [m^3]
wing longitudinal dimension	2π [m]
wing chord length	1.01 [m]
card(elements)	169598
card(nodes)	36036
card(internal nodes)	21910
card(boundary nodes)	14126

Table 2: Fluid mesh motion around a wing: main properties of Ω_r and of Ω_h.

2.3.2 Fluid mesh motion around a wing

In this section we move to a FSI setting, by considering the fluid mesh motion around the NACA0012 profile in Figure 8, top. This test case mimics a typical study performed in a wind tunnel, where the wing is clamped on one side while, on the other side, it is deformed by a vertical displacement, such as the one in (7). Table 2 gathers the main properties of the physical domain and of the corresponding discretization.

To investigate the deformation in the fluid mesh, we start by applying the standard IDW approach, thus identifying the set $\mathcal{C}$ with all boundary nodes (see Figure 9, top for a detail in correspondence with the clamped side). The corresponding interpolation matrix in (3) belongs to $\mathbb{R}^{14126\times 21910}$. The resulting deformed configuration, shown in Figure 8, bottom, is obtained after 83.09 [s].

In order to reduce the computational costs, we resort to the ESIDW interpolation, by enriching the selection of control points in $\widehat{\mathcal{C}}$ with all the nodes on the left and right sides as well as along the horizontal edges of the NACA profile. Different values of radius R are selected for the faces of the (outer) box and for the sides of the (inner) wing. In particular, with similar considerations as in the previous section, we choose: $R_{top,b} = R_{bottom,b} = R_{front,b} = R_{rear,b} = 0.5$ [m] and $R_{right,b} = R_{left,b} = 0.25$ [m] for the box; $R_{top,w} = R_{bottom,w} = R_{right,w} = 0.05$ [m] for the wing. Parameters a and b are set to 0.8 and 1.3, respectively. **Algorithm 2** provides a subset $\widehat{\mathcal{C}}$ consisting of 9339 control points (see the enlarged view in Figure 9, bottom), so that matrix $\widehat{W}$ in (6) is now in $\mathbb{R}^{9339\times 21910}$. As shown in the figure, control points are essentially located on the structure profile only.
The selection process reduces the computational time more than one third with respect to the IDW approach, the time required by the deformation phase being equal to 23.9 [s].

Finally, the $L^2(\Omega_h)$-norm of the relative error between the ESIDW and the IDW deformation is approximately 5.86%.

Sensitivity to R. We numerically check the sensitivity of IDW and ESIDW techniques to the selection radius, by mimicking the investigation in the previous section. For this purpose, we choose $R_{top,b} = R_{bottom,b} = R_{front,b} = R_{rear,b} = R$,

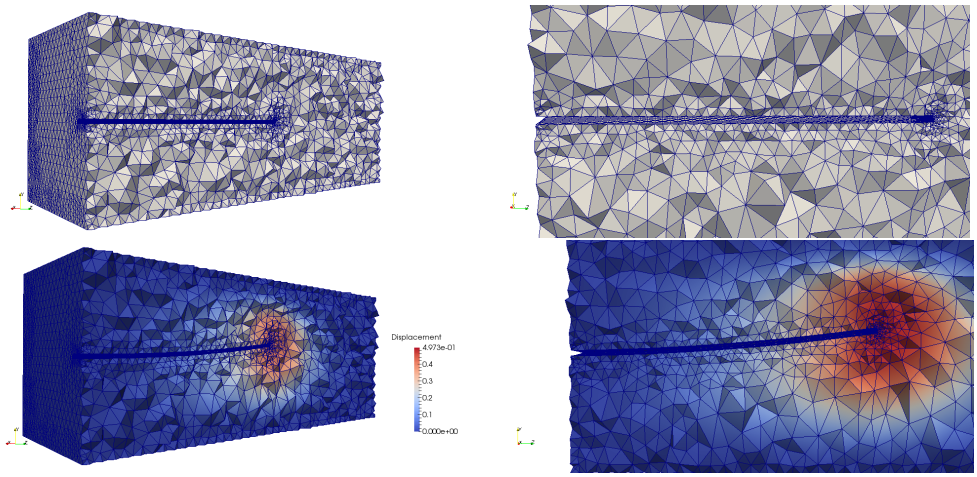


Figure 8: Fluid mesh motion around a wing: rest (top) and deformed (bottom) configuration; 3D (left) and section (right) view.

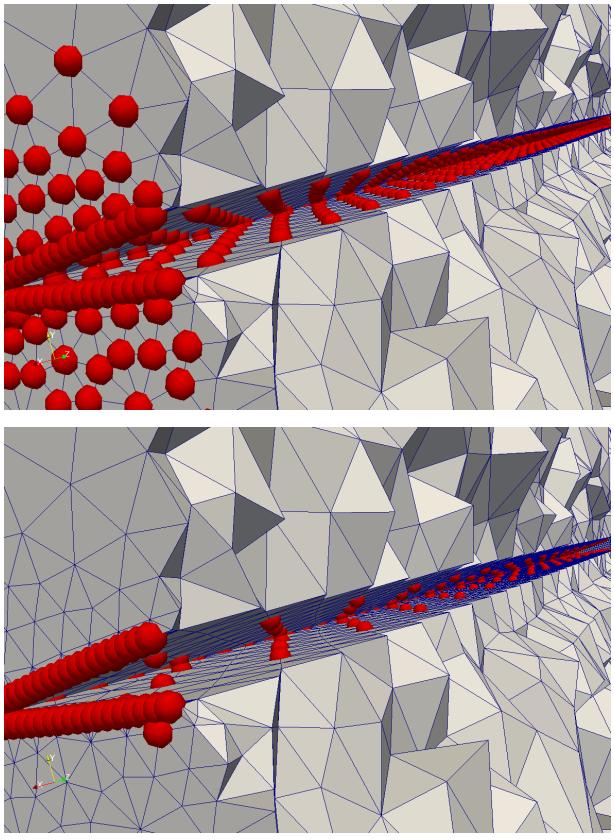


Figure 9: Fluid mesh motion around a wing: control point distribution associated with the IDW (top) and ESIDW (bottom) interpolation in correspondence with the clamped side.

$R_{right,b} = R_{left,b} = 0.5R$ for the box, and $R_{top,w} = R_{bottom,w} = R_{right,w} = 0.1R$ for the wing, while preserving the values previously adopted for a and b.

The trend exhibited by the number of control points in Figure 10, left is completely different with respect to the one in Figure 5, left. The cardinality of $\widehat{\mathcal{C}}$ is essentially the same as for the IDW approach until R sufficiently increases. Then, for $R > 0.1$, the cardinality reduces more and more ensuring, for instance, a gain of one order of magnitude for $R > 2$.
The central panel in Figure 10 computes the associated CPU time (in seconds), where now we have quantified the assembly and the deformation time, altogether. As expected, the trend of the computational time follows the one of the control point cardinality, as well as the $L^2(\Omega_h)$-norm of the relative error between ESIDW and IDW deformation, as shown in Figure 10, right. In contrast to Figure 6, left no stagnation of the error is detected when varying R.

Finally, concerning the quality characterizing the meshes yielded by IDW and ESIDW procedures, we have that the maximum and the mean value of $\mathcal{Q}$, independently of the adopted interpolation, is equal to 4.31 and 1.48, respectively with negligible variations (on the second decimal digit only) for ESIDW, as long as $R \leq 1$.

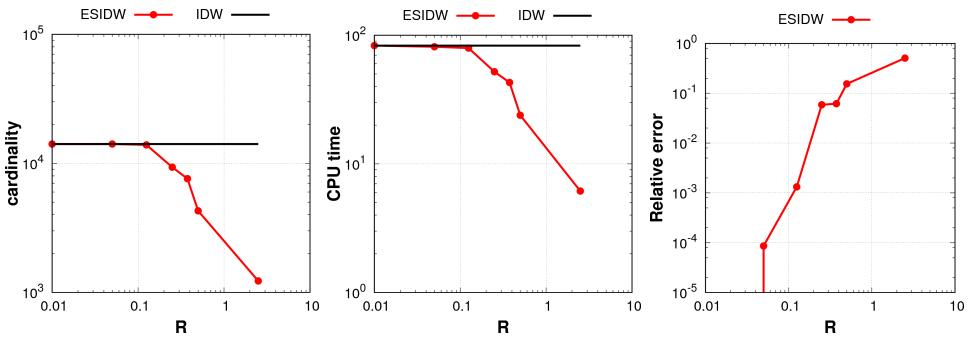


Figure 10: Fluid mesh motion around a wing: cardinality of the control point set (left), total CPU time (center), and the relative error (right) as a function of R.

2.3.3 Fluid mesh motion around a rotating hull

We conclude this section by testing the performances of the ESIDW interpolation when dealing with a FSI setting where a large deformation occurs. In more detail, we model a fluid mesh motion which results from the rotation of a structural domain in a naval engineering context. The initial configuration coincides with an outer box containing an inner Wigley hull [49] (see Figure 11, top). A rotation of $-5°$ with respect to the z-axis is successively applied. We refer to Table 3 for a summary of the main properties of the physical domain and of the corresponding discretization.

mesh dimension	$50 \times 25 \times 10$ [m^3]
hull length	2.5 [m]
card(elements)	30265
card(nodes)	7186
card(internal nodes)	3322
card(boundary nodes)	3864

Table 3: Fluid mesh motion around a rotating hull: main properties of Ω_r and of Ω_h.

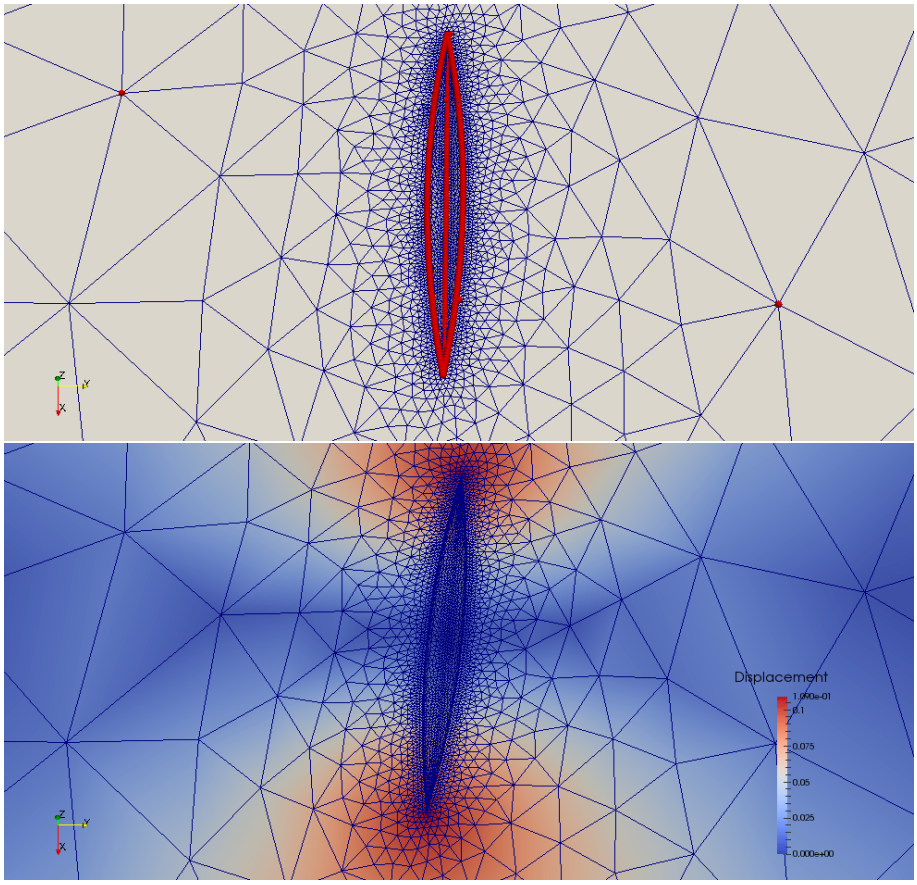


Figure 11: Fluid mesh motion around a rotating hull: control points (top) and deformed configuration (bottom) provided by the ESIDW interpolation.

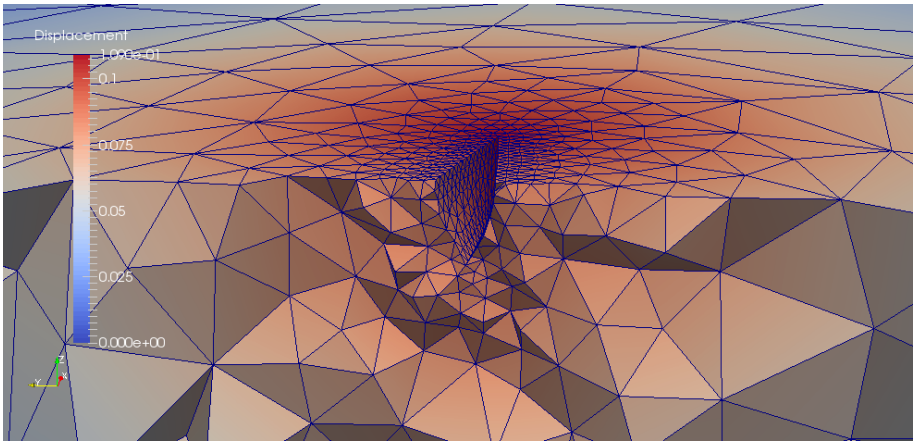


Figure 12: Fluid mesh motion around a rotating hull: detail of the deformed configuration provided by ESIDW interpolation.

The standard IDW approach results in solving the linear system (3), matrix W being in $\mathbb{R}^{3322\times3864}$. This leads to a CPU time of 4.51 [s].

To reduce the computational burden, we apply **Algorithm 2**. The selection is performed by considering a radius $R = 1.5$ [m], $a = 0.8$ and $b = 1.3$. Notice that the value of R is comparable with the hull length. This allows a considerable reduction in the number of control points along the boundaries of the fluid domain. Moreover, in order to accurately capture the rotation of the hull, we switch on the enriching step `(ii)` of the selection procedure, by adding to $\widehat{\mathcal{C}}$ all the nodes belonging to the edge of the hull. The resulting distribution of control points is shown in Figure 11, top. Most of the points is identified by the constraint step, while very few nodes (only two in the specific case) are retained by the selection at step `(i)`. As a result, the hull turns out to be sharply described by the control points. ESIDW procedure essentially halves the computational time, the CPU time of the deformation phase being now equal to 2.56 [s]. Figure 11, bottom displays the deformed fluid mesh, obtained by solving system (6), with $\widehat{W} \in \mathbb{R}^{3322\times1057}$. We also refer to Figure 12 for a detail of the deformed grid.

The computational cost reduction provided by ESIDW does not compromise the accuracy of the approximation. Indeed, the $L^2(\Omega_h)$-norm of the relative error between the ESIDW and the IDW deformation is 2.42%.

We have investigated the sensitivity to R also for this configuration, by mimicking the analysis in the previous section. Figure 13 provides the same results as in Figure 10. In contrast to this last case, we remark a less significant reduction in the number of the control points and of the total CPU time. Moreover, while for the wing configuration the ESIDW interpolation allows to gain six orders of magnitude in terms of accuracy for a sufficiently small R, only 2 orders less are obtained for the rotating hull configuration.

Finally, the rotation causes a slight increment in the maximum mesh quality (which changes from 2.47 in the reference configuration to 2.56 for the deformed one), while essentially preserving the mean value of $\mathcal{Q}$ (varying from 1.62 to 1.63 before and after the deformation). These values are essentially independent of the adopted interpolation and of R.

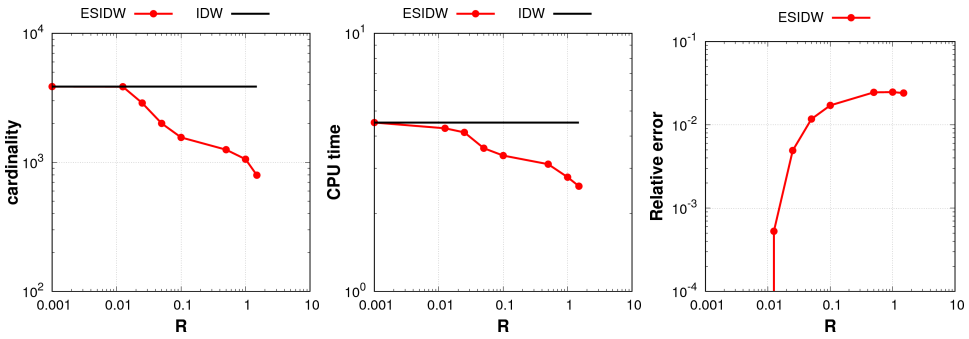


Figure 13: Fluid mesh motion around a rotating hull: cardinality of the control point set (left), total CPU time (center), and the relative error (right) as a function of R.

3 POD for SIDW shape morphing

We merge now the SIDW techniques presented in the previous section with a model reduction technique, following an offline-online paradigm [27]. The general idea is to exploit, during the offline phase, the deformations of the reference structure associated with certain loads to identify a reduced basis which allows us to predict, during the online phase, the structure deformation due to a new weight. In more detail, each deformation will be identified by a certain displacement of the control points. *A priori*, we can employ all the points in $\mathcal{C}$ for this purpose. Nevertheless, to contain the computational effort, we propose to properly filter points $\{\mathbf{c}_k\}_{k=1}^{\mathcal{N}_c}$ via SIDW algorithms and to resort to the control points in $\widehat{\mathcal{C}}$ only.
The reduced basis will be determined via a *Proper Orthogonal Decomposition* (POD) technique (see, e.g., [14, 47, 11, 27]). POD reduces the dimensionality of a system by transforming the original variables into a new set of uncorrelated variables (called POD modes, or principal components), so that, ideally, the first few modes retain most of the 'energy' of the original system.

Generalizing notation, let $\boldsymbol{\mu} \in \mathbb{D}$ denote the generic parameter identifying the displacement $\mathbf{d}_{\widehat{c}}(\boldsymbol{\mu})$ of the control points $\widehat{\mathbf{c}}_k$ in $\widehat{\mathcal{C}}$. According to (6), the deformation of the internal nodes $\mathbf{x}_j$, for $j = 1, \ldots, \mathcal{N}_h^i$, of the discretized reference structure Ω_h can be computed as

$$\mathbf{d}^i(\boldsymbol{\mu}) = \widehat{W}\mathbf{d}_{\widehat{c}}(\boldsymbol{\mu}), \tag{8}$$

the dependence on the parameter being highlighted. Notice that, while both the input displacement $\mathbf{d}_{\widehat{\mathcal{C}}}(\boldsymbol{\mu})$ and the output deformation $\mathbf{d}^i(\boldsymbol{\mu})$ depend on $\boldsymbol{\mu}$, the SIDW matrix $\widehat{W}$ is parameter independent.

During the POD offline stage, we consider a training set, $\Xi_{\text{train}} = \{\boldsymbol{\mu}_i\}_{i=1}^{n_{\text{train}}}$, of n_{train} values for the parameter $\boldsymbol{\mu}$. With each parameter $\boldsymbol{\mu}_i$, we associate a certain displacement $\mathbf{d}_{\widehat{\mathcal{C}}}(\boldsymbol{\mu}_i)$ of $\widehat{\mathcal{C}}$, and we compute the corresponding deformation $\mathbf{d}^i(\boldsymbol{\mu}_i)$ in (8). This allows us to assemble the so-called *snapshot* matrix

$$\mathcal{U} = [\mathbf{d}^i(\boldsymbol{\mu}_1), \cdots, \mathbf{d}^i(\boldsymbol{\mu}_{n_{\text{train}}})] \in \mathbb{R}^{\mathcal{N}_h^i \times n_{\text{train}}}, \tag{9}$$

whose columns store the different scenarios induced by the training set of deformations. To extract the desired reduced basis, we perform now the Singular Value Decomposition (SVD) of matrix $\mathcal{U}$, so that

$$\mathcal{U} = \mathcal{W} \begin{pmatrix} \Sigma & 0 \\ 0 & 0 \end{pmatrix} \mathcal{V}^T, \tag{10}$$

where

$$\mathcal{W} = [\boldsymbol{\zeta}_1, \boldsymbol{\zeta}_2, \cdots, \boldsymbol{\zeta}_{\mathcal{N}_h^i}] \in \mathbb{R}^{\mathcal{N}_h^i \times \mathcal{N}_h^i}$$

and

$$\mathcal{V} = [\boldsymbol{\Psi}_1, \boldsymbol{\Psi}_2, \cdots, \boldsymbol{\Psi}_{n_{\text{train}}}] \in \mathbb{R}^{n_{\text{train}} \times n_{\text{train}}}$$

are the orthogonal matrices of the *left* and of the *right singular vectors* of $\mathcal{U}$, while $\Sigma = \text{diag}(\sigma_1, \cdots, \sigma_r) \in \mathbb{R}^{\mathcal{N}_h^i \times n_{\text{train}}}$ is a diagonal rectangular matrix of the same size as $\mathcal{U}$, whose non-negative real diagonal entries, $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r$, represent the *singular* (or *principal*) values of $\mathcal{U}$, $r \leq n_{\text{train}}$ being the rank of $\mathcal{U}$ [25].

The reduced *POD basis* is thus identified by a subset of the left singular vectors of $\mathcal{U}$ collected in the *POD basis matrix*

$$\mathcal{Z} = [\boldsymbol{\zeta}_1, \cdots, \boldsymbol{\zeta}_N] \in \mathbb{R}^{\mathcal{N}_h^i \times N}. \tag{11}$$

with $N \ll r$. The integer N can be selected via different criteria. In particular, since the energy contained in the discarded (i.e., the last $r - N$) POD modes is provided by

$$E_{\mathcal{U}}(N) = \sum_{i=N+1}^{r} \sigma_i^2, \tag{12}$$

we set a tolerance ε and choose N as the first integer such that $E_{\mathcal{U}}(N) \leq \varepsilon$. The identification of the POD basis concludes the offline phase.

The online phase starts from the choice of a new value $\boldsymbol{\mu}^* \in \mathbb{D}$ for the parameter, i.e., with the assignment of a new displacement $\mathbf{d}_{\widehat{\mathcal{C}}}(\boldsymbol{\mu}^*)$ to the control points in $\widehat{\mathcal{C}}$. The POD basis is then exploited to approximate the corresponding deformation $\mathbf{d}^i(\boldsymbol{\mu}^*)$ of the internal nodes in Ω_h. In particular, we look for a suitable linear combination of the POD basis functions $\{\boldsymbol{\zeta}_l\}_{l=1}^N$, such that

$$\mathcal{Z}\, \boldsymbol{\beta}(\boldsymbol{\mu}^*) \approx \mathbf{d}^i(\boldsymbol{\mu}^*), \tag{13}$$

where

$$\mathbf{d}^i(\boldsymbol{\mu}^*) = \widehat{W}\mathbf{d}_{\widehat{\mathcal{C}}}(\boldsymbol{\mu}^*) \tag{14}$$

and where $\boldsymbol{\beta}(\boldsymbol{\mu}^*) \in \mathbb{R}^N$, with $[\boldsymbol{\beta}(\boldsymbol{\mu}^*)]_l = \beta_l$ and $l = 1, \ldots, N$, is a vector of unknown coefficients depending on parameter $\boldsymbol{\mu}^*$. Now, in order to contain the computational effort, we aim at computing $\boldsymbol{\beta}(\boldsymbol{\mu}^*)$ in (13) by skipping the direct computation of $\mathbf{d}^i(\boldsymbol{\mu}^*)$ via (14) and by exploiting the displacement vector $\mathbf{d}_{\widehat{\mathcal{C}}}(\boldsymbol{\mu}^*)$ only.
For this purpose, we first solve system (14) with respect to $\mathbf{d}_{\widehat{\mathcal{C}}}(\boldsymbol{\mu}^*)$ in a least squares sense, so that

$$\mathbf{d}_{\widehat{\mathcal{C}}}(\boldsymbol{\mu}^*) = \widehat{W}^+\mathbf{d}^i(\boldsymbol{\mu}^*)$$

where $\widehat{W}^+ \in \mathbb{R}^{\mathcal{N}_{\widehat{\mathcal{C}}} \times \mathcal{N}_h^i}$ denotes the pseudo-inverse of W [25]. Thanks to (13), we obtain

$$\mathbf{d}_{\widehat{\mathcal{C}}}(\boldsymbol{\mu}^*) \approx \widehat{W}^+ \mathcal{Z}\, \boldsymbol{\beta}(\boldsymbol{\mu}^*). \tag{15}$$

Vector $\boldsymbol{\beta}(\boldsymbol{\mu}^*)$ can thus be computed by resorting to the normal equations associated with (15) as

$$\mathcal{Z}^T(\widehat{W}^+)^T\, \widehat{W}^+\mathcal{Z}\, \boldsymbol{\beta}(\boldsymbol{\mu}^*) = \mathcal{Z}^T(\widehat{W}^+)^T\mathbf{d}_{\widehat{\mathcal{C}}}(\boldsymbol{\mu}^*). \tag{16}$$

The system we are lead to solve has size N, being $\mathcal{Z}^T(\widehat{W}^+)^T\, \widehat{W}^+\mathcal{Z} \in \mathbb{R}^{N\times N}$ and $\mathcal{Z}^T(\widehat{W}^+)^T \in \mathbb{R}^{N\times \mathcal{N}_{\widehat{\mathcal{C}}}}$. The deformation of the structure is finally computed via the product in (13).
We observe that both the matrices in (16) can be computed once and for all at the end of the offline stage. Thus, the online phase reduces to evaluate the matrix-vector product on the right-hand side of (16), to solve a linear system of size N and then to compute the linear combination $\mathcal{Z}\ \boldsymbol{\beta}(\boldsymbol{\mu}^*)$. In terms of computational burden, the most effort involves the internal nodes. In more detail, we are comparing the full problem (14) demanding $O(\mathcal{N}_h^i \cdot \mathcal{N}_{\widehat{\mathcal{C}}})$ floating point operations with the POD approach characterized by $O(N^3 + N\cdot\mathcal{N}_{\widehat{\mathcal{C}}} + \mathcal{N}_h^i \cdot N)$ operations. This suggests that, if $N \ll \mathcal{N}_{\widehat{\mathcal{C}}}$, we expect a computational saving via the reduced approach. This will be numerically verified in the next section.

The complete POD-SIDW interpolation procedure is itemized in **Algorithm 3** when applied to a generic shape morphing context. The POD-ESIDW variant can be set in a straightforward way simply by switching on the optional step `(ii)` of **Algorithm 2** at the first item of the offline phase. With a view to a FSI problem, the same setting as for **Algorithm 2** is adopted.

3.1 POD-SIDW algorithms in action

We come back to the test cases in Section 2.3 to investigate possible benefits on SIDW interpolation techniques due to POD. In particular, we quantify the computational improvements in terms of CPU time and accuracy of the approximation.

Algorithm 3 POD-SIDW interpolation for shape morphing

OFFLINE PHASE:

(a) apply the SELECTION PHASE of **Algorithm 2** to extract $\widehat{\mathcal{C}}$ and to assemble the SIDW matrix $\widehat{W}$;

(b) **for each** $\boldsymbol{\mu}_i \in \Xi_{\mathbf{train}}$

apply the DEFORMATION PHASE of **Algorithm 2** to compute $\mathbf{d}^i(\boldsymbol{\mu}_i)$ via (8);

end for

(c) assemble the snapshot matrix $\mathcal{U}$ in (9);

(d) extract the POD basis matrix $\mathcal{Z}$ in (11);

(e) compute matrices $\mathcal{Z}^T(\widehat{W}^+)^T\,\widehat{W}^+\mathcal{Z}$ and $\mathcal{Z}^T(\widehat{W}^+)^T$ in (16).

ONLINE PHASE: chosen $\boldsymbol{\mu}^* \in \mathbb{D}$:

(f) solve system (16) to derive the weights in $\boldsymbol{\beta}(\boldsymbol{\mu}^*)$;

(g) compute the linear combination $\mathcal{Z}\,\boldsymbol{\beta}(\boldsymbol{\mu}^*)$.

method	card($\widehat{\mathcal{C}}$)	**CPU time**(OFFLINE)	card($\mathcal{Z}$)	**CPU time**(ONLINE)	**Error**
IDW	1666	-	-	0.25 [s]	-
ESIDW	390	-	-	0.034 [s]	1.06 %
POD-IDW	1666	90.93 [s]	1	0.013 [s]	negligible
POD-ESIDW	390	76.84 [s]	1	0.013 [s]	1.09 %

Table 4: Structural deformation of a wing: comparison between the basic IDW and ESIDW techniques and the corresponding POD variants.

3.1.1 Structural deformation of a wing

We apply the POD reduction procedure to a parameter dependent variant of the configuration in Section 2.3.1. To this aim, we impose the parametrized vertical displacement

$$\delta y = \delta y(z;\mu) = \mu z^2 \tag{17}$$

to the wing, where the parameter μ is a scalar varying in the interval $\mathbb{D} = [0, 1.3]$.

We focus on the standard IDW and on the ESIDW interpolation techniques. With reference to the ESIDW approach, we preserve the two different choices done for the selection radius in Section 2.3.1, by picking $R = R_{lr} = 0.05$ [m] and $R = R_{tb} = 0.5$ [m] for the left and right and for the top and bottom surfaces of the wing, respectively. Then, accordingly to Section 2.3.1, the enrichment is performed by adding in $\widehat{\mathcal{C}}$ all the nodes along the left, the right and the horizontal edges of the NACA profile.

To build the snapshot matrix $\mathcal{U}$, we randomly select $n_{\text{train}} = 100$ values in $\mathbb{D}$ which identify the set Ξ_{train}. Then, to extract the POD basis we fix $\varepsilon = 10^{-5}$ as tolerance on the energy $E_{\mathcal{U}}(N)$ in (12). Finally, we choose $\mu^* = 0.01$ as parameter value during the online phase.

Table 4 compares the performances of the plain IDW and ESIDW interpo-

lations with the corresponding POD variants. The second column provides the number of control points employed for morphing the original structure, being understood that $\widehat{\mathcal{C}} \equiv \mathcal{C}$ when dealing with the IDW approach. The third column gathers the CPU time required by the offline phase of **Algorithm 3** to construct the response matrix, to extract the POD basis, whose cardinality is furnished in the fourth column, and to assemble matrices $\mathcal{Z}^T(\widehat{W}^+)^T\,\widehat{W}^+\mathcal{Z}$ and $\mathcal{Z}^T(\widehat{W}^+)^T$ in (16). The fifth column summarizes the CPU time required to perform the shape morphing via (3) and (6) in the case of the basic IDW and SIDW interpolations, respectively; for the POD variants, this coincides with the CPU time demanded by the online phase, i.e., by the resolution of system (16) and by the computation of the linear combination $\mathcal{Z}\boldsymbol{\beta}(\boldsymbol{\mu}^*)$. Finally, the last column investigates the accuracy of the provided deformation by collecting the value of the $L^2(\Omega_h)$-norm of the relative error between the computationally cheaper deformations and the reference IDW shape.

Concerning the specific values in Table 4, we observe that both the POD variants identify a minimal reduced basis, a single POD mode being sufficient to ensure the desired tolerance. Figure 14, left shows the decay of the spectrum, normalized to the maximum singular value, for both the POD-IDW and the POD-ESIDW reduction. The first approach exhibits a more evident drop so that a single mode ensures actually an accuracy of 10^{-7} (considerably higher than the one demanded). The POD-ESIDW procedure in this case requires a larger number of modes to guarantee the same accuracy, for instance, two modes are demanded for $\varepsilon = 10^{-6}$ whereas four modes are required for $\varepsilon = 10^{-7}$. The

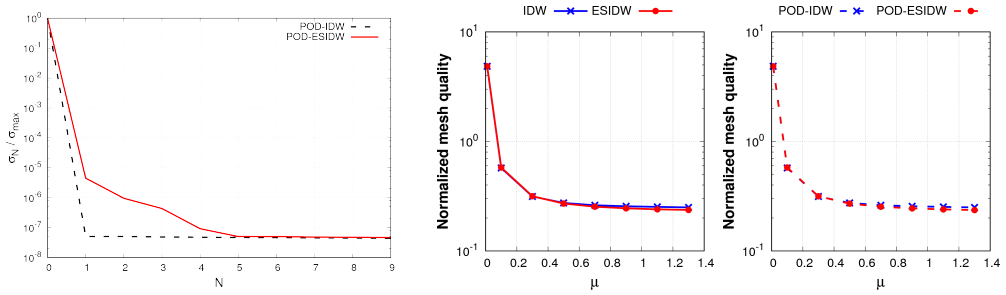


Figure 14: Structural deformation of a wing: spectrum decay (of the first ten singular values) for POD-IDW and POD-ESIDW procedures (left); comparison in terms of mesh quality between IDW and ESIDW interpolation (center) and POD-IDW and POD-ESIDW reduction (right) in the presence of large deformations.

computational time demanded by the offline phase of both the POD-IDW and POD-ESIDW procedures is not negligible, being equal to 90.93 [s] and to 76.84 [s], respectively. Nevertheless, this phase takes place just once and, as expected, allows a considerable saving in the actual computation of the shape morphing (0.013 [s] to be compared with 0.25 [s] and with 0.034 [s], respectively), thus becoming the ideal tool, for instance, for a multi-query context. Moreover, the

method	card($\widehat{\mathcal{C}}$)	**CPU time**(OFFLINE)	card($\mathcal{Z}$)	**CPU time**(ONLINE)	**Error**
IDW	14126	-	-	83.09 [s]	-
ESIDW	9339	-	-	23.90 [s]	5.86%
POD-IDW	14126	2479.51 [s]	1	0.55 [s]	negligible
POD-ESIDW	9339	1613.65 [s]	1	0.55 [s]	5.94 %

Table 5: Fluid mesh motion around a wing: comparison between the basic IDW and ESIDW techniques and the corresponding POD variants.

POD procedure does not compromise the accuracy of the approximation, the error (with respect to standard IDW interpolation) being negligible (less than 10^{-8}) in the POD-IDW case, while increasing from 1.06% to 1.09% when dealing with the enriched procedure.

Finally, we check how the POD procedure does influence the quality of the deformed mesh, with emphasis on large deformations. In particular, since mesh quality is expected to deteriorate for larger and larger displacements, we consider a normalized mesh quality index, defined as the ratio between the mean mesh quality and the maximum displacement measured at the side of the wing which is not clamped. A cross comparison of the plots in Figure 14, center and right, shows that the POD approach essentially has no influence on the mesh quality as well as the ESIDW interpolation, the selected index preserving a constant value of about 0.28 for $\mu > 0.3$.

3.1.2 Fluid mesh motion around a wing

We move to the parametric version of the FSI test case in Section 2.3.2, dealing with the fluid mesh motion around a NACA0012 profile. The wing is now deformed by the parametrized vertical displacement (17), with $\mu \in \mathbb{R}$ varying in $\mathbb{D} = [0, 0.05]$.

We compare POD-IDW with POD-ESIDW and with the corresponding procedures without any model reduction. The parameters R, a and b characterizing the selection procedure are the same as in Section 2.3.2, as well as the constraints driving the enrichment step of **Algorithm 2**. The training set Ξ_{train} is now constituted of 20 samples randomly distributed in $\mathbb{D}$, and the tolerance ε employed in the POD offline phase is set to 10^{-5} as in the previous test case. The actual deformation is then identified by $\mu^* = 0.01$.

Table 5 offers a summary of the performances of the proposed methods. The successive columns collect the same quantities as in Table 4. The advantages due to the enriched selective procedure have been already highlighted in Section 2.3.2, both in terms of reduction of the control points and of the CPU time.
A further saving in the computational time demanded for the actual structure deformation is obtained via POD, the online CPU time reducing to 0.55 [s] for both the POD-IDW and the POD-ESIDW procedures. The global CPU time reduction due to a selection of the control points combined with a POD procedure

is of two orders of magnitude with respect to the standard IDW approach, for each new morphing. This considerable saving is justified by the small dimension of the POD basis, consisting of a unique mode (notice that, for this test case, the trend of the first ten singular values associated with the two POD procedures is exactly the same, as Figure 15, left shows). The POD offline phase remains the most time consuming part of the whole procedure, requiring 2479.51 [s] and 1613.65 [s] in the POD-IDW and POD-ESIDW case, respectively.

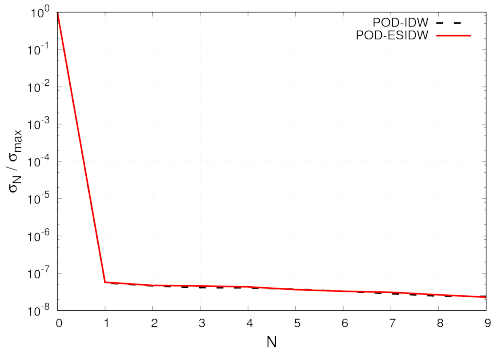


Figure 15: Fluid mesh motion around a wing: spectrum decay (for the first ten singular values) for POD-IDW and POD-ESIDW procedures.

Finally, we remark that the accuracy of the POD-ESIDW interpolation is essentially limited by the selection applied to control points. Indeed, the POD-ESIDW approach entails a relative error of 5.94% to be compared with an error of 5.86% for the basic ESIDW method.

3.1.3 Fluid mesh motion around a rotating hull

We conclude the numerical assessment by studying the benefits provided by the POD reduction onto the large deformation setting in Section 2.3.3. The rotation of the hull with respect to the z axis is now parametrized, μ coinciding with the rotation angle assuming values in the range $\mathbb{D} = [-36°,\ 0°]$.

As in the two previous sections, POD is combined with IDW and ESIDW interpolations. For the control point selection procedure, we adopt the same values for R, a and b as in Section 2.3.3. To build the snapshot matrix and to extract the POD basis, we exploits $n_{\text{train}} = 50$ values for parameter μ, randomly distributed in $\mathbb{D}$. The tolerance ε for the energy check is set to 10^{-5}. Finally, the target configuration is identified by the parameter $\mu^* = -5°$. From Figure 16, left, we realize that few POD modes suffice to describe the new rotation with a good accuracy. In particular, tolerance ε is reached by resorting only to two modes.

The performances of POD-IDW and POD-ESIDW methods are summarized in Table 6, whose columns are organized as in Tables 4 and 5. The computational gain yielded by ESIDW with respect to the standard IDW interpolation

method	card($\widehat{\mathcal{C}}$)	**CPU time**(OFFLINE)	card($\mathcal{Z}$)	**CPU time**(ONLINE)	**Error**
IDW	3864	-	-	4.51 [s]	-
ESIDW	1057	-	-	2.56 [s]	2.42%
POD-IDW	3864	203.07 [s]	1	0.38 [s]	0.23% $\sim$ 2.85%
			2	0.38 [s]	negligible
POD-ESIDW	1057	150.26 [s]	1	0.38 [s]	2.62% $\sim$ 3.84%
			2	0.38 [s]	2.43%

Table 6: Fluid mesh motion around a rotating hull: comparison between the basic IDW and ESIDW techniques and the corresponding POD variants.

is evident, following the analysis in Section 2.3.3.
Concerning the combined effect of selecting the control points and resorting to a POD reduction, Table 6 quantitatively confirms what already remarked for the two previous test cases. In particular, while ESIDW manages almost to halve the CPU time required for a shape morphing compared to the standard IDW approach, a successive reduction of about one seventh is reached via POD, two basis functions being enough to reach the prescribed tolerance ε. Similar conclusions hold when comparing IDW with POD-IDW, the CPU time being reduced of about eleven times when resorting to a POD space constituted of two basis functions only. The price to pay for this computationally cheap shape morphing is represented, according to an offline/online paradigm, by the offline phase in **Algorithm 3**, which demands 203.07 [s] and 150.26 [s] for the POD-IDW and the POD-ESIDW approach, respectively.

Figure 16, center compares the trend of the $L^2(\Omega_h)$-norm of the relative error between the POD-IDW (POD-ESIDW) and the standard IDW deformation as a function of the rotation angle, when either one or two POD modes are adopted to predict the new deformation. The plot of the POD-IDW procedure associated with two POD modes is omitted, the corresponding relative error being essentially negligible (less than 10^{-8}). The accuracy guaranteed by the POD-IDW approach strongly depends on the selected angle when a single POD mode is employed, with a significant improvement in the presence of large rotations. On the contrary, a low sensitivity to the rotation angle is exhibited by the POD-ESIDW reduction, the error always being of the order of 10^{-2}. Moreover, from the last column in Table 6, we deduce that the control point selection still represents the principal responsible for an accuracy deterioration regardless of the selected angle, the relative error remaining essentially the same when combining the ESIDW approach with a POD reduction.

Finally, as shown in Figure 16, right, the POD reduction does not perturb essentially the quality of the deformed mesh with respect to the standard ESIDW approach, and exhibits a contained dependence on the applied rotation.

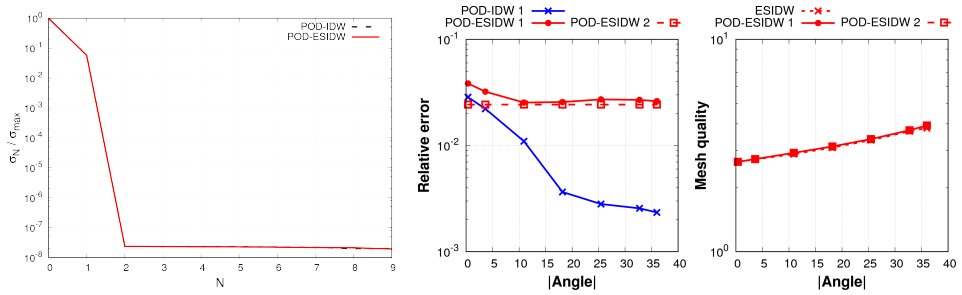


Figure 16: Fluid mesh motion around a rotating hull: spectrum decay (for the first ten singular values) for POD-IDW and POD-ESIDW procedures; relative error as a function of the rotation angle for POD-IDW and POD-ESIDW reductions (center); comparison in terms of mesh quality between ESIDW and POD-ESIDW procedures (right).

4 Conclusions and perspectives

Shape morphing techniques based on IDW are actually powerful computational tools, even though, often, very time consuming. We have proposed two strategies (and a possible combination of these) to limit this drawback. The first approach (the SIDW/ESIDW interpolation), based on a geometric selection of the control points, is very general and can be applied to any dimension and to arbitrary meshes. SIDW/ESIDW variants, tested on configurations of interest in engineering applications, such as airfoils and hulls, proved to be very effective since they considerably reduce the set of control points (i.e., the CPU time) without excessively compromising the accuracy of the approximation.
A further reduction of the computational burden is then carried out by means of a POD (offline phase) and a least squares regression (online phase) techniques. This allows us to convert any shape morphing into the resolution of a linear system of size N, with N the number of selected POD basis functions. In the considered test cases, the number N turns out to be very small. Indeed, one or, at most, two POD basis functions were enough to guarantee a tolerance of 10^{-5} to the energy contained in the discarded POD modes. We have combined POD model reduction with IDW and ESIDW interpolations, event though any shape morphing algorithm can be alternatively considered.
The numerical verification in Section 3.1 shows that the combined POD-IDW and POD-ESIDW techniques lead to a large computational saving, up to two orders of magnitude on the CPU time. The offline phase remains the most time-consuming part of the procedure, according to an offline/online paradigm. Finally, an accuracy analysis highlights that SIDW/ESIDW entail very small relative errors (few percentage points) with respect to the standard IDW procedure, while POD reduction does not significantly contribute further.

A possible future development of this work might concern the integration of

the proposed methods into a FSI solver, or the application to several optimization contexts. An adaptive selection of the control points driven by some quantity of interest rather than the employment of other model reduction procedures for parametrized problems (e.g., [7, 16, 27]), as well as the use of active subspaces method [17, 45] as pre-processing, represent further topics of interest for the following of the current work.

Acknowledgments

We gratefully acknowledge Dr. Andrea Mola (SISSA mathLab) for helpful discussions and his support with mesh generation. We acknowledge the support by European Union Funding for Research and Innovation – Horizon 2020 Program – in the framework of European Research Council Executive Agency (H2020 ERC CoG 2015 AROMA-CFD project 681447 "Advanced Reduced Order Methods with Applications in Computational Fluid Dynamics"). We also acknowledge the INDAM-GNCS project "Metodi Numerici Avanzati Combinati con Tecniche di Riduzione Computazionale per PDEs Parametrizzate e Applicazioni". Alessandro D'Amario has been supported by a SISSA scholarship (Mathematics Area).

References

[1] SALOME Platform. http://www.salome-platform.org/.

[2] J Ahrens, B Gavenci, and C Law. *ParaView: An End-User Tool for Large Data Visualization, Visualization Handbook*. Elsevier, 2005.

[3] JSR Anttonen, PI King, and PS Beran. POD-based reduced-order models with deforming grids. *Mathematical and Computer Modelling*, 38(1):41 – 62, 2003.

[4] F Ballarin. *Reduced-Order Models for Patient-Specific Haemodynamics of Coronary Artery Bypass Grafts*. PhD thesis, Department of Mathematics, Politecnico di Milano, 2015. http://hdl.handle.net/10589/102804.

[5] F Ballarin, E Faggiano, S Ippolito, A Manzoni, A Quarteroni, G Rozza, and R Scrofani. Fast simulations of patient-specific haemodynamics of coronary artery bypass grafts based on a POD–Galerkin method and a vascular shape parametrization. *Journal of Computational Physics*, 315:609–628, 2016.

[6] F Ballarin, A Manzoni, G Rozza, and S Salsa. Shape optimization by Free-Form Deformation: existence results and numerical solution for Stokes flows. *Journal of Scientific Computing*, 60(3):537–563, 2014.

[7] D Baroli, CM Cova, S Perotto, L Sala, and A Veneziani. Hi-POD solution of parametrized fluid dynamics problems: preliminary results. In P Benner,

M Ohlberger, A Patera, G Rozza, and K Urban, editors, *Model Reduction of Parametrized Systems*, volume 17 of *MS&A Series*, chapter 15, pages 235–254. Springer International Publishing, 2017.

[8] Y Bazilevs, K Takizawa, and TE Tezduyar. *Computational Fluid-Structure Interaction: Methods and Applications*. John Wiley & Sons, 2013.

[9] Y Bazilevs, K Takizawa, and TE Tezduyar. *Computational Fluid-Structure Interaction: Methods and Applications*. John Wiley & Sons, 2013.

[10] A Beckert and H Wendland. Multivariate interpolation for fluid-structure-interaction problems using Radial Basis Functions. *Aerospace Science and Technology*, 5:125–134, 2001.

[11] P Benner, S Gugercin, and K Willcox. A survey of projection-based model reduction methods for parametric dynamical systems. *SIAM Review*, 57(4):483–531, 2015.

[12] MD Buhmann. *Radial Basis Functions*, volume 12 of *Cambridge Monographs on Applied and Computational Mathematics*. Cambridge University Press, UK, 2003.

[13] H-J Bungartz and M Schäfer. *Fluid-Structure Interaction: Modelling, Simulation, Optimisation*, volume 53. Springer Science & Business Media, 2006.

[14] J Burkardt, M Gunzburger, and H-C Lee. POD and CVT-based reduced-order modeling of Navier–Stokes flows. *Computer Methods in Applied Mechanics and Engineering*, 196:337–355, 2006.

[15] JR Cebral and R Lohner. Conservative load projection and tracking for fluid-structure problems. *AIAA Journal*, 35:687–692, 1997.

[16] F Chinesta, R Keunings, and A Leygue. *The Proper Generalized Decomposition for Advanced Numerical Simulations: A Primer*. Springer, 2014. Springer Briefs in Applied Sciences and Technologies.

[17] PG Constantine. *Active Subspaces: Emerging Ideas for Dimension Reduction in Parameter Studies*. SIAM, 2015.

[18] A Crivellaro, S Perotto, and S Zonca. Reconstruction of 3D scattered data via radial basis functions by efficient and robust techniques. *Applied Numerical Mathematics*, 113:93–108, 2017.

[19] S Deparis, D Forti, and A Quarteroni. A rescaled localized radial basis function interpolation on non-Cartesian and non-conforming grids. *SIAM Journal on Scientific Computing*, 36(6), 2014.

[20] IL Dryden and KV Mardia. *Statistical Shape Analysis*. John Wiley and Sons, Chichester, 1998.

[21] C Farhat, M Lesoinne, and P Le Tallec. Load and motion transfer algorithms for fluid/structure interaction problems with non-matching discrete interfaces: momentum and energy conservation, optimal discretization and application to aeroelasticity. *Computer Methods in Applied Mechanics and Engineering*, 157(1):95–114, 1998.

[22] D Forti. Comparison of Shape Parametrization Techniques for Fluid-Structure Interaction Problems. Master's thesis, Politecnico di Milano, Italy, 2012.

[23] D Forti and G Rozza. Efficient geometrical parametrization techniques of interfaces for reduced-order modelling: application to fluid–structure interaction coupling problems. *International Journal of Computational Fluid Dynamics*, 28(3–4):158–169, 2014.

[24] PJ Frey and P-L George. *Mesh Generation: Application to Finite Elements*. ISTE Ltd and John Wiley & Sons, Inc, 2nd edition, 2010.

[25] GH Golub and CF Van Loan. *Matrix Computation*. The Johns Hopkins University Press, Baltimore, Fourth edition, 2013.

[26] R Guibert, K McLeod, A Caiazzo, T Mansi, MA Fernández, M Sermesant, X Pennec, IE Vignon-Clementel, Y Boudjemline, and J-F Gerbeau. Group-wise construction of reduced models for understanding and characterization of pulmonary blood flows from medical images. *Medical Image Analysis*, 18(1):63–82, 2014.

[27] JS Hesthaven, G Rozza, and B Stamm. *Certified Reduced Basis Methods for Parametrized Partial Differential Equations*. MS&A Series. Springer International Publishing, 2016.

[28] BS Kirk, JW Peterson, RH Stogner, and GF Carey. libMesh: a C++ library for parallel adaptive mesh refinement/coarsening simulations. *Engineering with Computers*, 22(3-4):237–254, 2006.

[29] T Lassila and G Rozza. Parametric free-form shape design with PDE models and reduced basis method. *Computer Methods in Applied Mechanics and Engineering*, 199(23–24):1583–1592, 2010.

[30] M Lombardi. *Numerical Simulation of a Sailing Boat: Free Surface, Fluid-Structure Interaction and Shape Optimization*. PhD thesis, Ecole Polyechnique Fédérale de Lausanne, 2012.

[31] M Lombardi, N Parolini, A Quarteroni, and G Rozza. Numerical simulation of sailing boats: dynamics, FSI, and shape optimization. In Buttazzo G Frediani, A, editor, *Variational Analysis and Aerospace Engineering: Mathematical Challenges for Aerospace Design*, volume 66, pages 339–377. Springer US, 2012.

[32] GY Lu and DW Wong. An adaptive inverse-distance weighting spatial interpolation technique. *Computers & Geosciences*, 34:1044–1055, 2008.

[33] A Manzoni, A Quarteroni, and G Rozza. Model reduction techniques for fast blood flow simulation in parametrized geometries. *International Journal for Numerical Methods in Biomedical Engineering*, 28(6–7):604–625, 2012.

[34] K McLeod, A Caiazzo, M Fernández, T Mansi, IE Vignon-Clementel, M Sermesant, X Pennec, Y Boudjemline, and J-F Gerbeau. Atlas-based reduced models of blood flows for fast patient-specific simulations. In O Camara, M Pop, K Rhode, M Sermesant, N Smith, and A Young, editors, *Statistical Atlases and Computational Models of the Heart*, volume 6364 of *Lecture Notes in Computer Science*, pages 95–104. Springer Berlin, 2010.

[35] J Park, SM Shontz, and CS Drapaca. A combined level set/mesh warping algorithm for tracking brain and cerebrospinal fluid evolution in hydrocephalic patients. In Y Zhang, editor, *Image-Based Geometric Modeling and Mesh Generation*, pages 107–141. Springer Netherlands, 2013.

[36] B Raghavan, P Breitkopf, Y Tourbier, and P Villon. Towards a space reduction approach for efficient structural shape optimization. *Structural and Multidisciplinary Optimization*, 48:987–1000, 2013.

[37] T Richter. *Fluid-structure Interactions: Models, Analysis and Finite Elements*, volume 118. Springer, 2017.

[38] M Ripepi. *Model Order Reduction for Computational Aeroelasticity*. PhD thesis, Politecnico di Milano, Italy, 2015.

[39] G Romanelli. *Computational Aeroservoelasticity of Free-Flying Deformable Aircraft*. PhD thesis, Politecnico di Milano, Italy, 2012.

[40] JA Samareh. Aerodynamic shape optimization based on Free-Form Deformation. In *Proceedings of the 10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*. AIAA, 2004.

[41] TW Sederberg and SR Parry. Free-Form Deformation of solid geometric models. In *Proceedings of SIGGRAPH - Special Interest Group on GRAPHics and Interactive Techniques*, pages 151–159. SIGGRAPH, 1986.

[42] D Shepard. A two-dimensional interpolation function for irregularly-spaced data. In *Proceedings-1968 ACM National Conference*, pages 517–524. ACM, 1968.

[43] J Sokolowski and J-P Zolésio. *Introduction to Shape Optimization: Shape Sensitivity Analysis*. Springer, 1992.

[44] ML Staten, SJ Owen, SM Shontz, AG Salinger, and TS Coffey. A comparison of mesh morphing methods for 3d shape optimization. In *Proceedings of the 20th international meshing roundtable*, pages 293–311. Springer, 2011.

[45] M Tezzele, F Salmoiraghi, A Mola, and G Rozza. Dimension reduction in heterogeneous parametric spaces applied to a naval engineering shape design problem. submitted, 2017.

[46] M Vaillant and J Glaunès. Surface matching via currents. In GE Christensen and M Sonka, editors, *Information Processing in Medical Imaging: 19th International Conference, IPMI 2005, Glenwood Springs, CO, USA, July 10-15, 2005. Proceedings*, pages 381–392. Springer Berlin Heidelberg, 2005.

[47] S Volkwein. Proper Orthogonal Decomposition: Theory and Reduced-Order Modeling. *Lecture Notes, University of Konstanz*, 2012.

[48] S Walton, O Hassan, and K Morgan. Reduced order mesh optimisation using proper orthogonal decomposition and a modified cuckoo search. *International Journal for Numerical Methods in Engineering*, 93(5):527–550, 2013.

[49] WCS Wigley. A comparison of experiment and calculated wave-profiles and wave-resistances for a form having parabolic waterlines. *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 144(851):144–159, 1934.

[50] JAS Witteveen. Explicit and robust Inverse Distance Weighting mesh deformation for CFD. In *48th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition*. AIAA, 2010.

[51] JAS Witteveen and H Bijl. Explicit mesh deformation using Inverse Distance Weighting interpolation. In *19th AIAA Computational Fluid Dynamics*. AIAA, 2009.

MOX Technical Reports, last issues

Dipartimento di Matematica
Politecnico di Milano, Via Bonardi 9 - 20133 Milano (Italy)

56/2017 Alberti, G. S.; Santacesaria, M.
Infinite dimensional compressed sensing from anisotropic measurements

55/2017 Agosti, A.; Cattaneo, C.; Giverso, C.; Ambrosi, D.; Ciarletta, P.
A computational platform for the personalized clinical treatment of glioblastoma multiforme

54/2017 Dede', L; Quarteroni, A.
Isogeometric Analysis of a Phase Field Model for Darcy Flows with Discontinuous Data

53/2017 Bertagna, L.; Deparis, S.;Formaggia, L.; Forti, D.; Veneziani A.
The LifeV library: engineering mathematics beyond the proof of concept

52/2017 Beretta, E.; Ratti, L.; Verani, M.
A phase-field approach for the interface reconstruction in a nonlinear elliptic problem arising from cardiac electrophysiology

51/2017 Gerbi, A.; Dede', L.; Quarteroni, A.
A monolithic algorithm for the simulation of cardiac electromechanics in the human left ventricle

50/2017 Formaggia, F.; Vergara, C.
Defective boundary conditions for PDEs with applications in haemodynamics

49/2017 Antonietti, P. F.,; Pennesi, G.
V-cycle multigrid algorithms for discontinuous Galerkin methods on non-nested polytopic meshes

48/2017 Regazzoni, F.; Dedè, L.; Quarteroni, A.
Active contraction of cardiac cells: a model for sarcomere dynamics with cooperative interactions

47/2017 Menghini, F.; Dede, L.; Forti, D.; Quarteroni, A.
Hemodynamics in a left atrium based on a Variational Multiscale-LES numerical model