Available online at www.sciencedirect.com
ScienceDirect
Journal of Computational Physics 224 (2007) 414–430
JOURNAL OF COMPUTATIONAL PHYSICS
www.elsevier.com/locate/jcp

Higher-order time integration through smooth mesh deformation for 3D fluid–structure interaction simulations

A.H. van Zuijlen *, A. de Boer, H. Bijl

Faculty of Aerospace Engineering, Delft University of Technology, P.O. Box 5058, 2600 GB Delft, The Netherlands

Received 13 October 2006; received in revised form 20 March 2007; accepted 28 March 2007
Available online 5 April 2007

Abstract

In this paper, we present a higher-order accurate in time, partitioned integration scheme (IMEX) for fluid–structure interaction. The scheme is based on a combination of an implicit, *L*-stable, multi-stage Runge–Kutta scheme and an explicit Runge–Kutta scheme. Fluid and structure dynamics are integrated using the implicit scheme and only the pressure loads acting on the structure are integrated explicitly.

For an academic problem we show that mesh optimization functions, which are often necessary in standard mesh deformation algorithms, can have a detrimental effect on the temporal order and accuracy. We use a radial basis function (RBF) interpolation with a thin plate spline to create a smooth displacement field for the whole fluid domain, which does not affect the order of the IMEX time integration scheme. For reasonable accuracies, the IMEX schemes outperform a second-order staggered scheme by a factor of 2–3.

As an example for a three-dimensional, real-world problem, a simulation of a transonic wing flutter case, the AGARD 445.6 wing, is performed. For this test case, a clear third-order time accuracy is observed for IMEX3.

Keywords: Fluid–structure interaction; Partitioned coupling; Higher-order time integration; Mesh deformation

1. Introduction

For a wide range of engineering problems, dynamic multi-physics interactions like fluid–structure interaction play a key role in the safety of the design. For example, the construction of a wing can fail when flutter occurs. The computation of fluid–structure interaction phenomena has therefore received a lot of interest over the past decades. Nowadays, complex real-world fluid–structure interaction problems can be simulated such as complete jet-fighter aircraft [1], although simulation times are very high.

In the fluid–structure interaction literature, two distinct ways are identified for solving the coupled problem: the monolithic approach and the partitioned approach [2]. In the monolithic approach, the equations governing the multi-physics are discretized and solved within the same solver. A monolithic approach requires the

* Corresponding author. Tel.: +31 15 278 2046; fax: +31 15 278 7077.
E-mail address: a.h.vanzuijlen@tudelft.nl (A.H. van Zuijlen).

0021-9991/$ - see front matter
doi:10.1016/j.jcp.2007.03.024

implementation of a whole new solver in which the structure and fluid dynamics are merged. This generally results in ill-conditioned systems for which existing solution algorithms are far from optimal [3].

In the partitioned approach, the flow and structure domains are treated separately. This has the advantage that different, already existing solvers can be used, which can be maintained separately. Different solvers can use different meshes and discretizations that do not have to match at the fluid–structure interface. The coupling at the interface, therefore, requires an interpolation scheme to transfer displacements from the structure to the flow and pressure forces from the flow to the structure. In [4], a review of several interpolation schemes revealed that radial basis function interpolation with compact support or thin plate splines [5,6] performed best. Drawback of the partitioned approach is that in numerical simulations the coupling is not solved implicitly, but explicitly as boundary conditions imposed by one system onto the other. This explicit coupling can cause numerical instability, so that either the time step needs to be decreased or sub-iterations have to be performed. By sub-iterating the partitioned scheme, the fully coupled solution can be obtained. The efficiency and stability of the sub-iterations can be increased by, e.g. using under relaxation [7], reduced order modeling [8], Newton–Krylov acceleration techniques at the interface [9] or on the whole domain [10]. In any case, still a number of sub-iterations is required for every time step, increasing the computational effort. In order to obtain a stable and accurate solution with a partitioned scheme it depends on the problem whether it is more efficient to reduce the time step or to use sub-iterations.

Due to the interaction with the structure a mesh deformation algorithm is required that deforms the flow mesh based on the boundary displacements. Some mesh deformation techniques exploit the connectivity of the internal grid points, e.g. by representing the connectivity by springs [11–13] or as solid body elasticity [14]. Special instances of this continuous approach include moving grids based on Laplacian and Biharmonic operators [15]. All methods based on grid connectivity require solving a sparse system of equations involving all the flow points, which can be solved approximately using a small number of iterations.

Other mesh deformation strategies move each grid point individually based on its position in space and this results in the so-called point-by-point schemes. In de Boer et al. [16], Radial Basis Functions – originally only used at the fluid–structure interface – were extended to interpolate the displacement to all the nodes of the flow mesh. This method has to solve a system involving only the flow points on the boundary. The displacements of the nodes interior to the fluid domain can also be easily performed in parallel.

The fluid dynamics equations that are solved on dynamic meshes are often written in the Arbitrary Lagrangian–Eulerian (ALE) formulation [17]. This introduces an additional term to the convective fluxes that depends on the velocity of the deforming mesh. A geometric conservation law (GCL) provides the relation between the deformation of the mesh and the velocity of the mesh. When time is discretized a discrete geometric conservation law (D-GCL) provides the relation between the discrete mesh displacement and the discrete mesh velocity. Violating the D-GCL generally results in a numerically less stable scheme and a loss of accuracy and efficiency [18,19].

Fluid–structure interaction covers a wide range of physical applications. For the weak interactions partitioned methods can be used. For instance, wing flutter has been simulated successfully using partitioned methods by Farhat et al. [20]. For applications with a strong interaction, loosely coupled partitioned schemes require an overly restrictive time step for numerical stability [21]. For this type of applications, like flags or sails, a monolithic or strongly coupled scheme is more appropriate. In this paper, we only consider weak interaction, hence we choose a partitioned integration scheme without sub-iterations.

Promising for increasing the efficiency of fluid–structure interaction simulations are higher-order time integration schemes. Compared to second-order methods that are current practice in engineering and fluid–structure interaction simulations [1,22,23], multi-stage, L-stable, Runge–Kutta schemes (ESDIRK) have shown their potential for computational fluid dynamics simulations [24,25]. More recently, these schemes were also applied to fluid dynamics with moving boundaries [19], but without the interaction with a structure dynamics model. Higher-order time integration has been used for the simulation of academic structure dynamics problems in [26] and for the partitioned computation of relatively simple fluid–structure interaction problems in [27]. In the partitioned fluid–structure interaction simulations a combination of implicit/explicit Runge–Kutta time integration schemes is used, referred to as IMEX schemes. The implicit scheme integrates the fluid and structure dynamics and the explicit scheme only integrates the coupling term from the fluid to the structure. High order accurate convergence in time could be obtained without a need for sub-iterating the partitioned

algorithm and in terms of computational efficiency the high order IMEX schemes were also outperforming second-order time accurate methods. However, only a one-dimensional problem was considered.

In this paper, we extend the IMEX partitioned algorithm to a general three-dimensional approach and investigate its efficiency for real-world applications. The displacement of the fluid–structure interface is not prescribed anymore, as in [19], but actually depends on the interaction with the structure dynamics. Apart from the partitioned time integration we also have to deal with the mesh deformation for arbitrary moving boundaries. For obtaining higher-order accuracy in time, the smoothness of the mesh deformation algorithm plays a key role. In this paper, we therefore show the potentially adverse effect of the mesh deformation on the order of the time accuracy. Secondly, we show the potential efficiency gains of high order methods over second-order schemes when the higher-order time accuracy is obtained.

First the general layout of a fluid–structure interaction solver is given, where after all the components are discussed. Some numerical tests are performed on a one-dimensional piston problem to demonstrate the influence of mesh deformation on the order of convergence in time. As an illustration for a real-world three-dimensional test case, the scheme is applied to the simulation of a transonic wing flutter case, the AGARD 445.6 wing [28].

2. Fluid–structure interaction model

The aim is to develop a partitioned fluid–structure interaction solver that has higher-order accuracy in time. Because of the partitioned approach, separate solvers are used for the flow and structure and a coupling algorithm is needed both in space and time. In space an interpolation scheme is required at the fluid structure interface that connects the spatial discretization (meshes). In time a partitioning algorithm is required that connects the time integration schemes. Summarizing, the fluid–structure interaction solver consists of the following six ingredients:

- *A flow solver*: the solver can be, e.g. an Euler or Navier–Stokes solver. Any kind of spatial discretization can be used by the solver, e.g. a finite element or finite volume method.
- *A structure solver*: the structure solver can use any kind of spatial discretization for solving linear or nonlinear problems.
- *A spatial coupling algorithm*: in general the flow and structure meshes do not match at the interface and an interpolation scheme is necessary to transfer the displacements from the structure to the flow and the pressure loads from the fluid to the structure.
- *A mesh deformation algorithm*: since the fluid boundaries are dynamic, the interior mesh has to be deformed to match the fluid boundaries.
- *A time integration scheme*: this can be any time integration scheme, e.g. ESDIRK or BDF with a mesh velocity computation algorithm $\mathscr{F}$ that satisfies the Discrete Geometric Conservation Law (D-GCL)
- *A temporal coupling algorithm*: this scheme is required when separate solvers or solution techniques are used for the flow and the structure. A sub-iteration method can also be included to obtain a strongly-coupled scheme.

In the following sections, each of these ingredients is described in more depth.

2.1. Flow model

The flow used in this paper is compressible and inviscid. The fluid is an ideal gas. The governing equations for the flow are the Euler equations and the ideal gas law: $p = \rho R_g T$, with p the pressure, ρ the density, T the temperature and R_g the gas constant. Since the fluid domain is deforming, the Euler equations are written in the Arbitrary Lagrangian–Eulerian (ALE) formulation [17]. A general purpose, compressible Finite Volume flow solver is used to solve the equations on an unstructured, hexahedral mesh [29]. Since the flow is inviscid, only the mesh velocity normal to the fluid faces ($\boldsymbol{\kappa} \cdot \mathbf{n}$) is required. The semi-discrete system for a cell l can be written as

$$\frac{\mathrm{d}(\Omega \mathbf{U})_l}{\mathrm{d}t} + \sum_{i=1}^{N_{l,\mathrm{face}}} (\mathbf{\Phi}_i - \mathbf{U}_i(\boldsymbol{\kappa} \cdot \mathbf{n}S)_{l,i}) = 0, \tag{1}$$

where $\mathbf{U}$ is the fluid state in conservative variables, Ω is the cell volume, $N_{l,\mathrm{face}}$ denotes the number of faces that define cell l, $\mathbf{\Phi}_i$ the numerical flux for the fluid dynamics equations as computed on static meshes for face i, $\mathbf{U}_i$ the fluid state at face i, $S_{l,i}$ is the face surface, $\boldsymbol{\kappa}_{l,i}$ its velocity in the moving mesh and $\mathbf{n}_{l,i}$ the face normal pointing outward of cell l. In our case, the numerical flux $\mathbf{\Phi}$ is computed using the standard second-order central scheme with Jameson type artificial dissipation [30]. The semi-discrete system can be written more compactly as

$$\frac{\mathrm{d}\mathbf{U}}{\mathrm{d}t} + \mathbf{R}_{\mathrm{f}}(\mathbf{U}, \boldsymbol{\kappa}) = 0, \tag{2}$$

wherein $\mathbf{R}_{\mathrm{f}}$ the semi-discrete fluid dynamics model. Although we only consider the Euler equations for the fluid in this paper, we expect similar results for the full Navier–Stokes equations.

2.2. Structure model

The structure model is a linear structure dynamics model without damping:

$$M\frac{\mathrm{d}^2\mathbf{Q}}{\mathrm{d}t^2} + K\mathbf{Q} = \mathbf{F}_{\mathrm{sf}}, \tag{3}$$

with M the mass matrix, K the stiffness matrix, $\mathbf{F}_{\mathrm{sf}}$ the pressure load from the flow that acts on the structure and $\mathbf{Q}$ the structural displacement vector. In this paper, we do not use a separate structure solver to simulate the structural dynamics, but we use *OpenFEM* [31], a Finite Element toolbox for *Matlab*, to construct the matrices M and K. For modeling a three-dimensional solid, we use the standard 20 node, 60 degrees-of-freedom (DOF), brick-element and the standard 15 node, 45 DOF, pentahedral element [32]. Both element types use quadratic shape functions. The resulting mass matrix M is not lumped. The structure state vector $\mathbf{W} = \begin{pmatrix} M\dot{\mathbf{Q}} \\ \mathbf{Q} \end{pmatrix}$, which contains the structural momentum vector $M\dot{\mathbf{Q}}$ and the structural displacement vector $\mathbf{Q}$, is used to write the structure dynamics as a system of ordinary differential equations

$$\frac{\mathrm{d}\mathbf{W}}{\mathrm{d}t} + A_{\mathrm{s}}\mathbf{W} = \begin{pmatrix} \mathbf{F}_{\mathrm{sf}} \\ \mathbf{0} \end{pmatrix}, \tag{4}$$

with

$$A_{\mathrm{s}} = \begin{bmatrix} 0 & K \\ -M^{-1} & 0 \end{bmatrix}, \tag{5}$$

which is the semi-discrete formulation of the structure dynamics. A more general formulation of (4) is

$$\frac{\mathrm{d}\mathbf{W}}{\mathrm{d}t} + \mathbf{R}_{\mathrm{s}}(\mathbf{W}, \mathbf{F}_{\mathrm{sf}}) = 0, \tag{6}$$

with $\mathbf{R}_{\mathrm{s}}$ the semi-discrete structure dynamics model.

2.3. Spatial coupling

In the partitioned fluid–structure interaction solver two interpolation schemes are required to transfer data at the fluid–structure interface from the flow mesh to the structure mesh and vice versa. The first scheme transfers displacements $\mathbf{d}_{\mathrm{s}}$ of the interface of the structure mesh to displacements $\mathbf{d}_{\mathrm{fs}}$ of the interface of the flow mesh

$$\mathbf{d}_{\mathrm{fs}} = H_{\mathrm{fs}}\mathbf{d}_{\mathrm{s}}, \tag{7}$$

with H_{fs} the interpolation matrix for transferring data from the structure interface mesh to the fluid interface mesh. The subscript fs denotes that a quantity defined on the discrete fluid mesh at the interface is obtained

from the interpolation of that quantity from the discrete structure mesh at the interface. The second interpolation scheme transfers the pressure forces $\mathbf{F}_{\mathrm{f}}$ acting on the fluid interface mesh to a pressure load $\mathbf{F}_{\mathrm{sf}}$ acting on the structure interface mesh

$$\mathbf{F}_{\mathrm{sf}} = H_{\mathrm{sf}}\mathbf{F}_{\mathrm{f}}, \tag{8}$$

with H_{sf} the interpolation matrix for transferring data from the fluid interface mesh to the structure interface mesh. The subscript sf denotes that a quantity defined on the discrete structure mesh at the interface is obtained from the interpolation of that quantity from the discrete fluid mesh at the interface. Many different interpolation methods are available for the construction of the interpolation matrices. In an earlier study on these interpolation methods [4,33], the radial basis function (RBF) interpolation [5,34], using thin plate spline or compact support, turned out to be the most accurate method for interpolating the structural displacements to the flow mesh when a second-order spatial discretization is used. In [33], it was shown that a very simple (and inexpensive) nearest neighbor interpolation scheme can still have the same high accuracy as a radial basis function interpolation as long as the interpolation is performed from a fine mesh to a coarse mesh. In general, fluid–structure interaction applications the flow mesh is much denser at the interface than the structure mesh. Therefore, we use the nearest neighbor interpolation to transfer the pressure forces from the flow mesh to the structure mesh and the RBF interpolation to transfer the displacements to the flow mesh.

2.3.1. Radial basis function interpolation

Radial basis functions (RBF's) have become a well-established tool to interpolate scattered data. A smooth interpolation function is used to transfer the displacements known at the boundary of the structural mesh to the boundary of the aerodynamic mesh.

The interpolation function, s, describing the displacement of the fluid interface mesh, can be approximated by a sum of basis functions

$$s(\mathbf{x}) = \sum_{j=1}^{n_{\mathrm{s}}} \alpha_j \phi(\|\mathbf{x} - \mathbf{x}_{\mathrm{s}_j}\|) + p(\mathbf{x}), \tag{9}$$

where $\mathbf{x}_{\mathrm{s}_j} = [x_{\mathrm{s}_j}, y_{\mathrm{s}_j}, z_{\mathrm{s}_j}]$ are the centers in which the values are known, in this case the structure boundary nodes, p a linear polynomial, n_{s} the number of boundary nodes in the structure mesh and ϕ a given basis function with respect to the Euclidean distance $\|\mathbf{x}\|$. Adding a linear polynomial ensures that rigid body translations are exactly recovered. The coefficients α_j and the polynomial p are determined by the interpolation conditions

$$s(\mathbf{x}_{\mathrm{s}_j}) = \mathbf{d}_{\mathrm{s}_j} \quad \text{and} \quad \sum_{j=1}^{n_{\mathrm{s}}} \alpha_j q(\mathbf{x}_{\mathrm{s}_j}) = 0, \tag{10}$$

for all linear and constant polynomials q, and $\mathbf{d}_{\mathrm{s}_j}$ the vector containing the discrete known values of the displacement at the structure boundary mesh $\mathbf{x}_{\mathrm{s}_j}$.

The values of the coefficients α_j and the linear polynomial can be obtained by solving the system

$$\begin{bmatrix} \mathbf{d}_{\mathrm{s}} \\ 0 \end{bmatrix} = \begin{bmatrix} M_{\mathrm{s,s}} & P_{\mathrm{s}} \\ P_{\mathrm{s}}^T & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha} \\ \boldsymbol{\beta} \end{bmatrix}, \tag{11}$$

with $\boldsymbol{\alpha}$ the vector containing the coefficients α_j, $\boldsymbol{\beta}$ the coefficients of the linear polynomial p, $M_{\mathrm{s,s}}$ an $n_{\mathrm{s}} \times n_{\mathrm{s}}$ matrix containing the evaluation of the basisfunction $\phi_{\mathrm{s}_i\mathrm{s}_j} = \phi(\|\mathbf{x}_{\mathrm{s}_i} - \mathbf{x}_{\mathrm{s}_j}\|)$ and P_{s} an $n_{\mathrm{s}} \times 4$ matrix with row j given by $[\,1 \quad x_{\mathrm{s}_j} \quad y_{\mathrm{s}_j} \quad z_{\mathrm{s}_j}\,]$. For details, see [5].

The values for the displacement of the boundary vertices of the flow mesh $\mathbf{d}_{\mathrm{fs}}$, can then be obtained by evaluating the interpolation function in the fluid boundary vertices:

$$\mathbf{d}_{\mathrm{fs}_j} = s(\mathbf{x}_{\mathrm{f}_j}). \tag{12}$$

When it is assumed that the displacements are not extremely large compared to the initial position, the displacements $\mathbf{d}_{\mathrm{fs}_j}$ can be computed with respect to the initial position of the mesh $\mathbf{x}_{\mathrm{f}_j}$. This results in the interpolation matrix H_{fs} to be constant during the simulation. Note that after computing the coefficients $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$,

the method can be easily partitioned, since the displacement in each fluid boundary vertex can be obtained independently.

2.3.2. Nearest neighbor interpolation

In the nearest neighbor interpolation, the pressure in the fluid is integrated over each of the fluid cell faces on the interface. Each fluid cell face i contributes a pressure force $\mathbf{F}_{\mathrm{f},i}$ that is added to the structural load $\mathbf{F}_{\mathrm{sf},j}$ of the structure node j closest to the fluid cell face i. This method results in a Boolean matrix H_{sf}, which can be generally kept constant during the simulation.

2.4. Mesh deformation

The flow mesh deformation matrix $H_{\Omega_{\mathrm{f}}\Gamma_{\mathrm{f}}}$ interpolates the displacement of the fluid boundary vertices $\mathbf{d}_{\mathrm{f}}$ to displacements for all interior flow mesh vertices $\mathbf{d}_{\mathrm{f_{in}}}$. Note that $\mathbf{d}_{\mathrm{f}}$ not only includes the vertex displacements at the fluid structure interface $\mathbf{d}_{\mathrm{fs}}$, but also the flow vertices at the static boundaries. Mesh deformation techniques that exploit the connectivity of the internal grid points, e.g. by representing the connectivity by springs [11–13] or as solid body elasticity [14] require solving a system of equations involving all the flow points. Other mesh deformation strategies move each grid point individually based on its position in space and this results in the so-called point-by-point schemes. However, until now point-by-point schemes are only applied to the boundary nodes of multi-grid blocks [35]. In [16], radial basis functions were used to interpolate the displacement to all the nodes of the flow mesh. In the same study, it was concluded that the thin plate spline generated meshes with a high quality after deformation and is therefore used in this paper.

The mesh deformation uses the same radial basis function interpolation algorithm as in Section 2.3. The size of the system that has to be solved is equal to $(n_{\mathrm{b}}+4)\times(n_{\mathrm{b}}+4)$, with n_{b} the total number of boundary vertices for the flow mesh. This system is usually very small compared to the systems that have to be solved in mesh-connectivity schemes. The systems encountered there are approximately as large as $n_{\mathrm{in}}\times n_{\mathrm{in}}$, with n_{in} the total number of mesh points. Another advantage of the RBF interpolation is that it results in a linear system as opposed to, e.g. mesh deformation by torsional springs [13]. The total number of mesh points is a dimension higher than the number of points on the boundary of the mesh. The moving mesh technique based on RBF's is very easy to implement, even for 3D applications, because no mesh-connectivity information is needed.

The efficiency for the mesh deformation can be increased even more when the mesh deformation $H_{\Omega_{\mathrm{f}}\Gamma_{\mathrm{f}}}$ is combined with the displacement interpolation H_{fs}. Hereto a radial basis function interpolation is made that uses the fluid boundary vertices at the boundaries with a pre-imposed displacement (here we assume zero displacements, hence static boundaries) and the structure nodes at the interface. For these imposed displacements an interpolation field is generated for all internal flow vertices, including those on the fluid–structure interface. In this case, the size of the system that has to be solved is equal to $(n_{\mathrm{b}}-n_{\mathrm{fs}}+n_{\mathrm{s}}+4)\times(n_{\mathrm{b}}-n_{\mathrm{fs}}+n_{\mathrm{s}}+4)$, with n_{fs} the number of vertices on the *dynamic* boundary of the flow mesh. Therefore when $n_{\mathrm{fs}}\gg n_{\mathrm{s}}$, as it generally is, the size of the system is reduced even further. The interpolation matrix $H_{\Omega_{\mathrm{f}}\mathrm{s}}$ transfers the structure displacements directly to all the flow vertices

$$\mathbf{d}_{\mathrm{f_{in}}} = H_{\Omega_{\mathrm{f}}\mathrm{s}}\mathbf{d}_{\mathrm{s}}. \tag{13}$$

Although (13) denotes the resulting interpolation, the actual evaluation follows the procedure mentioned in Section 2.3.1. When the displacements are not extremely large, $H_{\Omega_{\mathrm{f}}\mathrm{s}}$ is kept constant throughout the computation. Therefore, the flow mesh does not degenerate when it is moved in a periodic fashion (as for flutter cases).

2.5. Time integration scheme

Both the fluid and the structure dynamics, (2) and (6), can be expressed as a semi-discrete system of the form

$$\dot{\mathbf{w}} + \mathbf{R}(\mathbf{w},\mathbf{v}) = 0, \tag{14}$$

wherein $\mathbf{w}$ is the fluid or structure state vector, $\mathbf{R}$ is the residual function (spatial discretization) of the fluid or structure dynamics and $\mathbf{v}$ is any other variable, e.g. mesh velocity or external forcing. In this paper, we use an

implicit, L-stable, multi-stage Runge–Kutta scheme for the time integration of the fluid and structure dynamics. In particular, we use the so-called ESDIRK (explicit first stage, single diagonal coefficient, implicit Runge–Kutta) schemes obtained from [36]. These schemes consist of a number of second-order accurate, implicit stages within each time step. For each stage k we solve

$$\frac{\mathbf{w}^{(k)} - \mathbf{w}^n}{\Delta t} + \sum_{i=1}^{k} a_{ki}\mathbf{R}^{(i)} = 0, \tag{15}$$

where the superscript (k) denotes the solution at stage level k, Δt is the time step and a_{ki} are the coefficients defining the ESDIRK scheme and

$$\mathbf{R}^{(i)} \equiv \mathbf{R}\left(\mathbf{w}^{(i)}, \mathbf{v}^{(i)}\right). \tag{16}$$

The coefficients a_{ki} are generally represented in a Butcher-tableau, e.g. for a five-stage method the implicit ESDIRK scheme and explicit ERK schemes have the following Butcher-tableau.

ESDIRK

c_1	0	0	0	0	0
c_2	a_{21}	γ	0	0	0
c_3	a_{31}	a_{32}	γ	0	0
c_4	a_{41}	a_{42}	a_{43}	γ	0
c_5	a_{51}	a_{52}	a_{53}	a_{54}	γ
	b_1	b_2	b_3	b_4	b_5

ERK

c_1	0	0	0	0	0
c_2	$\hat{a}_{21}$	0	0	0	0
c_3	$\hat{a}_{31}$	$\hat{a}_{32}$	0	0	0
c_4	$\hat{a}_{41}$	$\hat{a}_{42}$	$\hat{a}_{43}$	0	0
c_5	$\hat{a}_{51}$	$\hat{a}_{52}$	$\hat{a}_{53}$	$\hat{a}_{54}$	0
	b_1	b_2	b_3	b_4	b_5

The values for the coefficients of the third-order (four stages) and fourth-order (six stages) schemes can be found in [36]. The stage level is defined by $c_k = \sum_{i=1}^{k} a_{ki}$ for which the time level is $t_k = t_n + c_k\Delta t$. Note the explicit first stage, $a_{11} = 0$, and the single diagonal coefficient $a_{ii} = \gamma$ for the ESDIRK scheme. Note also that the stage levels $\mathbf{c}$ for the ESDIRK and ERK schemes are identical, which means that these methods have colocated stage solutions. After solving the intermediate stages, the high order solution at the new time level t_{n+1} is obtained using the weights $\boldsymbol{b}$

$$\mathbf{w}^{n+1} = \mathbf{w}^n - \Delta t \sum_{i=1}^{s} b_i\mathbf{R}^{(i)}, \tag{17}$$

with s the total number of stages of the scheme. Since both the ESDIRK and ERK scheme use the same weights $\mathbf{b}$ to obtain a high order solution, they can be used in a combination and still obtain design order. For instance in [36] the combination was used to explicitly integrate the fluid dynamics and implicitly integrate the reaction equations in a convection–diffusion-reaction problem. We intend to use the explicit scheme only for the integration of the pressure load at the fluid–structure interface and use the implicit scheme for the integration of the fluid and structure dynamics.

2.6. Mesh velocity computation satisfying the D-GCL

For the chosen time integration method a discrete-geometric conservation law (D-GCL) exists that should be satisfied for accuracy and stability reasons [37,38]. The D-GCL prescribes the mesh face normal velocities $(\boldsymbol{\kappa}\cdot\mathbf{n})$ that are consistent with the mesh face displacement and time integration scheme. When we apply the ESDIRK scheme to integrate the flow equation (1) and assume a uniform flow condition $\mathbf{U}_0$, we obtain for every stage k

$$\frac{\Omega_l^{(k)} - \Omega_l^n}{\Delta t} - \sum_{m=1}^{k} a_{km}\left(\sum_{i=1}^{N_{l,\text{face}}} \boldsymbol{\kappa}_i^{(m)} \cdot (\mathbf{n}S)_i^{(m)}\right) = 0, \tag{18}$$

where $\Omega_l^{(k)}$ and Ω_l^n are the volumes of cell l at t_k and t_n, respectively. The D-GCL basically states that the volume change of a cell in a time interval should be equal to the volume swept by its boundary in the same interval. Or more discrete: the change in volume of a cell should be equal to the summation of the swept volumes for each face individually, hence

$$\Omega_l^{(k)} - \Omega_l^n = \sum_{i=1}^{N_{l,\text{face}}} \Delta\Omega_{l,i}^{(k)}, \tag{19}$$

where $\Delta\Omega_{l,i}$ is the swept volume of face i of cell l in the time interval $t_n \rightarrow t_k$. Substitution of (19) into (18) gives

$$\sum_{i=1}^{N_{l,\text{face}}} \left[\frac{\Delta\Omega_{l,i}^{(k)}}{\Delta t} - \sum_{m=1}^{k} a_{km} \boldsymbol{\kappa}_i^{(m)} \cdot (\mathbf{n}S)_i^{(m)} \right] = 0. \tag{20}$$

As a restriction (20) should hold for each face individually, hence

$$\frac{\Delta\Omega_{l,i}^{(k)}}{\Delta t} - \sum_{m=1}^{k} a_{km} \boldsymbol{\kappa}_i^{(m)} \cdot (\mathbf{n}S)_i^{(m)} = 0. \tag{21}$$

In the fluid–structure interaction computations, the fluid mesh is deformed for each stage k prior to the computation of the flow at stage k. Therefore, the geometric properties of the mesh $(\mathbf{n}S)_i^{(k)}$ are known. When the swept volume $\Delta\Omega_{l,i}^{(k)}$ is also known the only unknown is $\boldsymbol{\kappa}_i^{(k)}$ since all other values have already been computed in previous stages. So one obtains a constraint for $(\boldsymbol{\kappa}_i \cdot \mathbf{n})^{(k)}$

$$(\boldsymbol{\kappa}_i \cdot \mathbf{n})_i^{(k)} = \frac{1}{a_{kk} S_i^{(k)}} \left[\frac{\Delta\Omega_{l,i}^{(k)}}{\Delta t} - \sum_{m=1}^{k-1} a_{km} \boldsymbol{\kappa}_i^{(m)} \cdot (\mathbf{n}S)_i^{(m)} \right]. \tag{22}$$

Therefore, when (19) and (22) are satisfied, so is the D-GCL. In order to satisfy (19) the swept volumes are computed by defining each swept volume as a collection of surfaces with an outward pointing normal that form a closed surface. The swept volume for face i consists of the face at t_n, $(\mathbf{n}S)_i^n$, and at t_k, $(\mathbf{n}S)_i^{(k)}$, which we call *external* faces and a number of connecting *internal* surfaces that connect the edge of face i at t_n, $\mathbf{e}_i^n$, to the edge of face i at t_k, $\mathbf{e}_i^{(k)}$. As long as the internal surfaces are uniquely defined by the shape of the edges $\mathbf{e}_i^n$ and $\mathbf{e}_i^{(k)}$, (19) is satisfied.

Our mesh consists of hexahedral cells, Fig. 1, and the swept volumes are computed by simply making a triangulation of the internal faces and computing the volume contained herein.

2.7. Partitioning scheme

In this paper, we only consider partitioned coupling methods without sub-iterations. For many aeroelastic cases – such as flutter – partitioned methods should provide sufficient accuracy without loosing stability. The method used is the IMEX [27] partitioning scheme. This scheme uses the implicit ESDIRK schemes for the integration of the fluid and structure dynamics and explicit ERK scheme to integrate the coupling term of

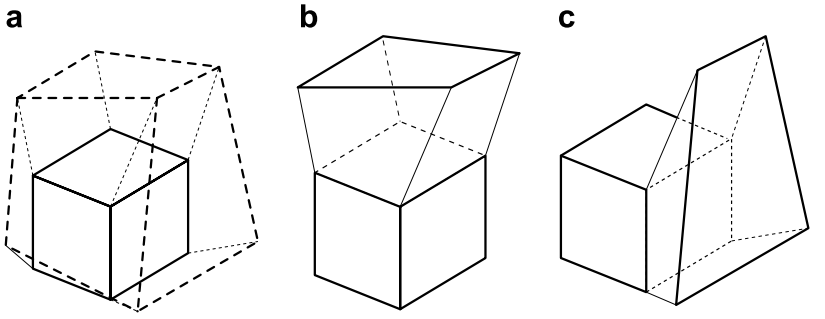

Fig. 1. Example of the swept volume for a hexahedral cell: (a) hexahedral cell at t_n (solid) and at t_k (dashed); (b) swept volume for the top face; and (c) swept volume for a side face.

the flow to the structure. Since the coupling term $\mathbf{F}_{sf}$ is a linear term in the structure dynamics (4), the IMEX scheme can be used as a state predictor for $\mathbf{F}_{sf}$ at the implicit stage level, without the risk of loosing the order or accuracy of the time integration scheme [39]. Therefore, we write the IMEX state predictor for stage k as

$$\mathbf{F}_{sf}^{(k)^*} = \sum_{i=1}^{k-1} \frac{\hat{a}_{ki} - a_{ki}}{a_{kk}} \mathbf{F}_{sf}^{(i)}, \tag{23}$$

where $\hat{a}_{ki}$ and a_{ki} are the Butcher coefficients of the ERK and ESDIRK schemes, respectively (Section 2.5). A step-by-step summary of the resulting partitioned algorithm is presented in Appendix A.

3. Test cases

In this paper, we consider two test cases: the first is a one-dimensional piston problem. This academic test case is used to show the importance of a good mesh deformation technique when high order time integration schemes are used. It also shows the potential efficiency gains over second-order methods. The second test case is the AGARD 445.6 wing flutter case. For this problem the accuracy and efficiency of the high order partitioned IMEX schemes with a smooth grid deformation module are investigated. The AGARD 445.6 wing test case demonstrates the practical applicability of high order time integration for real-world three-dimensional fluid–structure interaction.

3.1. One-dimensional piston problem

In order to investigate the effect of the mesh deformation algorithm on the temporal accuracy, the one-dimensional piston problem (Fig. 2) is used. This model has already been used in [27] as the test case for the IMEX schemes. The fluid is modelled by inviscid flow and the structure by a linear mass-spring system, which is governed by the scalar formulation of (3). The non-dimensional parameter settings used for the spring stiffness and structure mass are $\bar{k} = 1.429$ and $\bar{m} = 2$, respectively. The non-dimensional parameters are obtained by scaling with the equilibrium conditions for fluid domain length L, the fluid density ρ_0 and the speed-of-sound c_0: $\bar{k} = (kL)/(\rho_0 c_0^2)$ and $\bar{m} = m/(\rho_0 L)$. The Euler equations are solved in a two-dimensional domain. The undeformed mesh spans $(0,0) \rightarrow (1,1)$. The mesh consists of 64×4 cells in x,y-direction. Two different meshes are used: a mesh which is "ideal", e.g. all cells are perfectly orthogonal (Fig. 3a). The second mesh is "imperfect" and has cells that are not perfectly orthogonal (Fig. 3b). The second mesh mimics mesh imperfections that are generally present for real-world cases, e.g. meshes do not have to be perfectly aligned with important flow features like shocks. The fluid is an ideal gas with $\gamma = 1.4$ and $C_p = 3.5$. The initial displacement for the piston is chosen as $q = 0.5$ and the flow is assumed to be uniform and at rest: $u = 0$, $v = 0$, $\rho = 2/3$, $p = (2/3)^{\gamma}/\gamma$. Approximately, one coupled period is simulated till $t = 7$ at which instance the errors with respect to a temporally exact solution are computed. The comparisons are performed with two different mesh deformation techniques.

The first mesh deformation technique uses a dual mesh on which the Laplace equation is solved to create a displacement field [40]. After displacing the flow vertices, the mesh is optimized [41]. Optimization is necessary to avoid degenerate cells, e.g. when long simulations are run the accumulation of interpolation errors can cause the mesh to deteriorate. The optimization operation is generally not iterated until full convergence. The second mesh deformation technique is based on radial basis function interpolation as explained in Section 2.4 and does not require an additional optimization function.

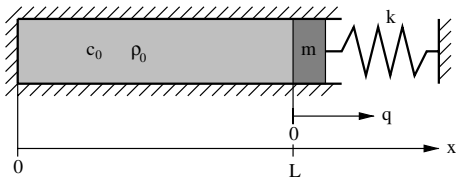

Fig. 2. Sketch of the one-dimensional piston problem.

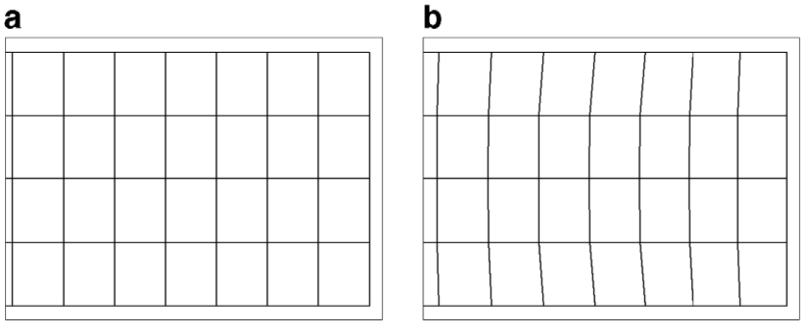

Fig. 3. A partial, stretched view of the initial mesh for the two-dimensional ideal and imperfect case: (a) ideal mesh and (b) imperfect mesh.

3.1.1. Influence of mesh deformation algorithm

In Fig. 4, the L_2-norms of the fluid density ρ, pressure p, u-velocity and v-velocity for the different meshes and mesh deformation schemes are shown. We use the fourth-order IMEX scheme for the partitioned time integration. The results obtained for the dual mesh with smoothing functions are not satisfactory. The fourth-order of the scheme is not observed and although the test problem is essentially one-dimensional, the v-velocity is not zero due to the imperfect mesh. For the large time steps this perturbation is only small compared to the errors in the density, pressure and u-velocity. However, since the convergence for the v-velocity is clearly not fourth-order, its influence becomes more apparent for the smaller time steps. Radial basis function interpolation for the imperfect mesh has the same non-zero v-velocity for the large time steps. This time, however, the perturbation does converge with fourth-order accuracy and therefore its influence on the solution remains negligible. Therefore, we can conclude that the radial basis function interpolation does not aggravate the imperfections in the flow mesh. For the ideal mesh, no perturbation in the v-velocity is generated and the convergence of the solution is perfectly fourth-order accurate.

In order to explain the bad convergence for the dual mesh we study the mesh face velocities computed by imposing the D-GCL. The mesh faces are identified by $\eta = x/L$, the location of the mesh face at the equilibrium position relative to the length of the fluid domain at equilibrium. The velocities for the cell faces $\eta = 0, 1/16, 2/16, \ldots, 1$ obtained with the dual mesh deformation method are displayed in Fig. 5. For $\eta = 0$ which is the location of the solid wall, the mesh velocity is identical to zero. For $\eta = 1$ the mesh velocity is approximately equal to the piston velocity (or equal when a fully coupled solution is obtained). Fig. 5 shows that the dual mesh with optimization introduces irregularities (wiggles) in the mesh face velocities which are worse for small Δt. The mesh face velocity does not converge to a consistent solution so the design order of the IMEX scheme cannot be expected. Since the optimization is not necessarily converged for every time step, the

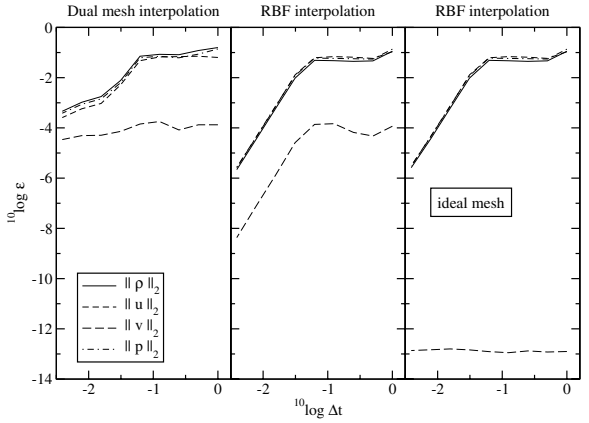

Fig. 4. Convergence for the L_2-norm of the density, pressure and u- and v-velocity components for the piston problem for the dual mesh interpolation with optimization and radial basis function interpolation.

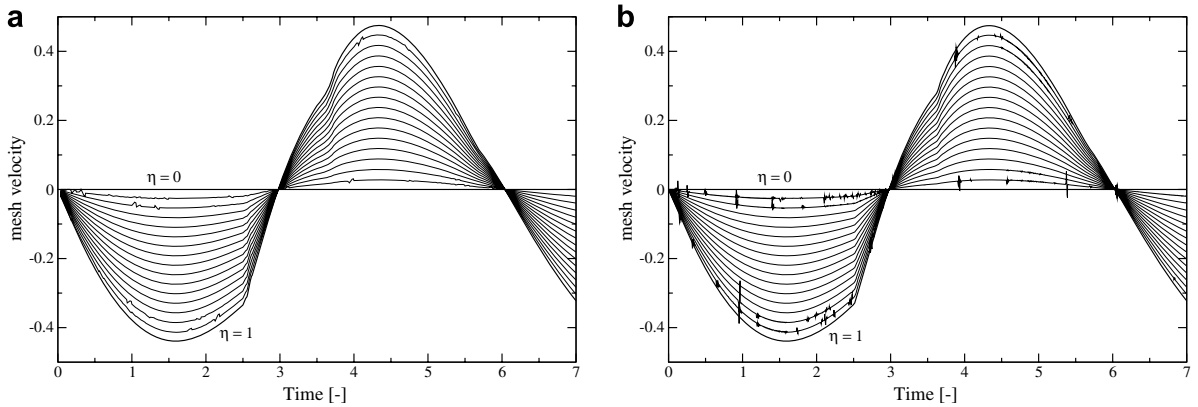

Fig. 5. Mesh face velocities for the dual mesh deformation with optimization: (a) $\Delta t = 1/16$ and (b) $\Delta t = 1/64$.

mesh vertex displacements do not have to relate to the prescribed displacements at the mesh boundary. Hence, the internal vertices can displace even when the boundary is not moving. This causes the highly irregular and inconsistent behavior for the computed mesh face velocities. Due to the high accuracy and regularity of the displacement field obtained with radial basis function interpolation the RBF mesh deformation algorithm does not exhibit these convergence problems.

3.1.2. Efficiency of higher-order time accuracy

The reason for using high order time integration schemes is that they allow the use of larger time steps compared to second-order schemes and still obtain the same accuracy in the solution. However, the amount of computational work for a single time step with the IMEX scheme is much higher than for a single time step with a second-order staggered [20] scheme. Solving an implicit time step for the staggered scheme or an implicit stage for the IMEX scheme is approximately the same amount of work. Therefore, the work is estimated as the total number of implicit stages (or time steps). The question is whether the increase in time step weighs up to the additional work per time step. In Fig. 6a, the convergence for the second-order staggered and third and fourth-order IMEX schemes is shown. The results are obtained for the ideal mesh with the RBF mesh deformation. All methods converge with their design orders. In Fig. 6b, the efficiency of the schemes is presented by the amount of work necessary to obtain a certain accuracy. At an error level of approximately $10^{-1.5}$ all the methods are still outside the asymptotic range. Below an error level of $10^{-1.5}$ the solutions should have a reasonable accuracy. At an error level of 10^{-2}, the third-order IMEX scheme performs best and is two times fas-

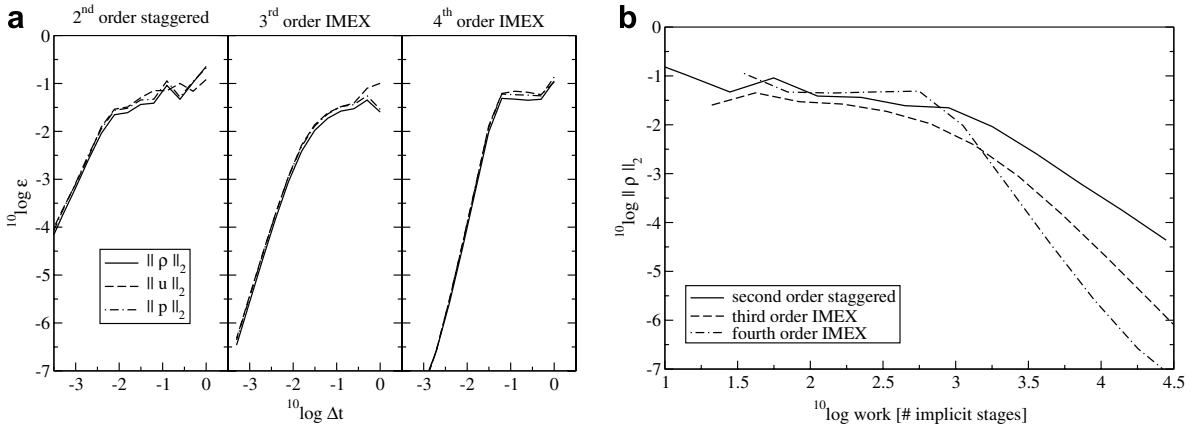

Fig. 6. Time accuracy and efficiency for the second-order staggered and third and fourth-order IMEX schemes: (a) convergence and (b) efficiency.

ter than the second-order scheme. For error levels below $10^{-2.5}$ the fourth-order IMEX scheme is more efficient, e.g. at an error level of 10^{-3} IMEX4 is already three times faster than the second-order scheme.

3.2. AGARD 445.6 wing

As a three-dimensional test case for the IMEX scheme and the grid deformation module, the AGARD 445.6 test case [28] is used. The model used is the 2.5-foot semi-span model, which consists of a symmetric NACA 65A004 airfoil, a sweep angle of 45° and a taper ratio of 0.66. The nose of the wing at the root is located in the origin. The flow domain is a box defined by the corner points $(-15, 0, -15)$ [m] and $(45, 15, 15)$ [m]. The boundary conditions for the domain consist of one symmetry plane at $y = 0$ to which the wing is attached and external boundary conditions all around prescribed at the static boundaries. The external conditions are set to the values obtained from [28] for the $M_\infty = 0.901$ case. The medium used is air for which the gas constant is $R_g = 287.05$ [K s^2/m^2] and the specific heat ratio $\gamma = 1.4$. From the measured density $\rho_\infty = 0.099468$ [kg/m^3] and velocity $V_\infty = 296.69$ [m/s] and the ideal gas assumption, one obtains $T_\infty = 269.82$ [K] and $p_\infty = 7704.05$ [Pa]. Several meshes are used, containing $223k$, $79k$, $28k$ and $15k$ hexahedral cells. The surface mesh on the wing for these meshes is shown in Fig. 7. The fine, medium, coarse and very coarse mesh have 26583, 7492, 2765 and 912 vertices on the wing's surface, respectively. The structure is modeled using the finite element (FE) method. Two FE models have been used, a fine model, consisting of 13×10 standard 20 node, 60 degrees-of-freedom (DOF), brick-elements (hexa20) in the chordal and spanwise direction and 2×10 standard 15 node, 45 DOF, pentahedral elements (penta15) at the leading and trailing edge of the wing. The coarse model uses 6×6 hexa20 and 2×6 penta15 elements, see Fig. 8. The wing simulated is the weakened model number 3 wing made out of Mahogany wood as described in [28]. The material properties are assumed to be orthotropic and their settings are obtained from [42] and given in Table 1. The fiber orientation (E_{11}) of the wood is taken along the quarter-chord line. The Finite Element toolbox *OpenFEM*, freely available for *Matlab*, is used to resolve the eigenmodes of the beam and to export the mass matrix M and the stiffness matrix K. The eigenmode frequencies for the fine and coarse models are compared to the experimental data of [28] and the numerical results of [42] in Table 2. The table shows that both the fine and the coarse

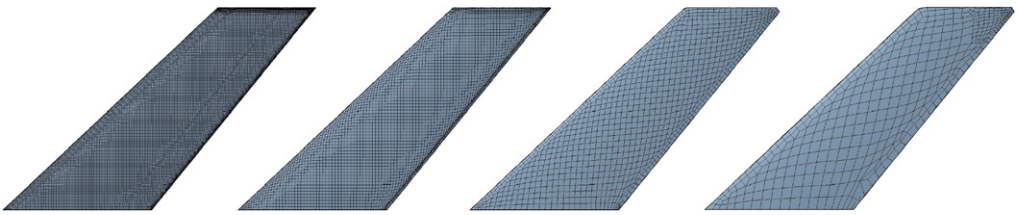

Fig. 7. Surface meshes for fine to very coarse CFD meshes.

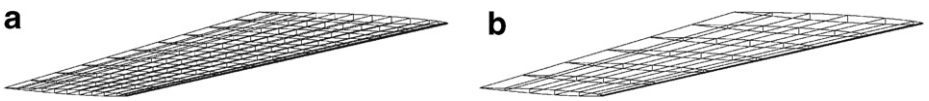

Fig. 8. Fine and coarse FE model of the AGARD 445.6 wing: (a) fine FE model and (b) coarse FE model.

Table 1
Material properties for the FE model

Property	Value	Dim
E_{11}	3.1511	[GPa]
E_{22}, E_{33}	0.4162	[GPa]
G_{12}, G_{13}, G_{23}	0.4392	[GPa]
ν_{12}, ν_{13}, ν_{23}	0.3100	[–]
ρ	381.98	[kg/m^3]

Table 2
Results for the fine and coarse FE model of the AGARD 445.6 wing

	Fine		Coarse		Yates [28] (baseline)	Beaubien [42]
Mode 1 [Hz]	9.59	(−0.1%)	9.71	(+1.1%)	9.60	9.46
Mode 2 [Hz]	40.23	(+8.2%)	41.86	(+9.7%)	38.10	39.44
Mode 3 [Hz]	50.69	(−0.0%)	52.92	(+4.4%)	50.70	49.71
Mode 4 [Hz]	97.52	(−1.0%)	104.24	(+5.8%)	98.50	94.39
Mass [kg]	1.68		1.64		1.86	1.69
DOFs	3120		990		–	–

model compare well with the experimental data [28] (within 10%). Therefore, we choose the coarsest model to represent the structural dynamics. The coarse structure model has 295 nodes on the interface. The very coarse flow mesh has already three times as many vertices on the fluid–structure interface. Therefore, the nearest neighbor interpolation of the fluid pressure force to the structure nodes should be accurate. The initial deformation of the wing is taken as the first bending mode and the amplitude is chosen such that the largest displacement for a structural node is 0.1 m, see Fig. 9. At first a steady-state solution is computed around the deformed wing shape. At $t = 0$ the wing is allowed to move freely according to its structural dynamics. For the time step refinement analysis the third-order IMEX scheme is used as it requires less computing time for a given time step compared to the fourth-order method. Therefore, the third-order IMEX scheme is more practical when the time step is reduced substantially. The fourth-order IMEX scheme is also used to investigate the computational efficiency of the scheme at large time steps.

3.3. Results

First the influence of the flow mesh is investigated by performing numerical simulations on the very coarse, coarse, medium and fine mesh with the third-order IMEX scheme and a time step $\Delta t = 0.001$ s, which is sufficiently small for the meshes used to have a larger spatial discretization error than time integration error. In Fig. 10, the lift history is shown for these simulations. The very coarse mesh is clearly too inaccurate and the coarse mesh still has an error of $\approx 15\%$. The medium mesh has an error of $\approx 5\%$ compared to the fine mesh which is considered sufficiently accurate for the main purpose of this study: to verify that high order IMEX

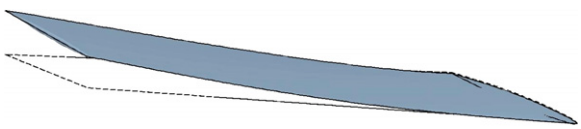

Fig. 9. The initial displacement of the AGARD 445.6 wing.

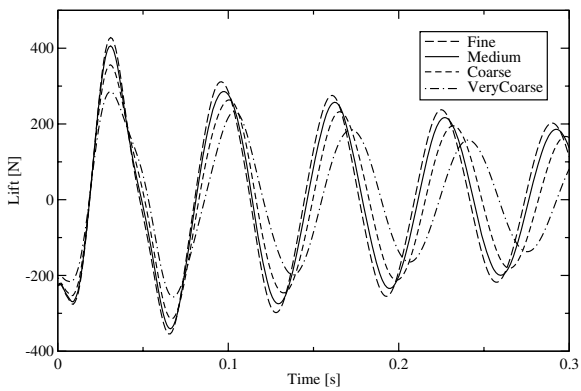

Fig. 10. Time history of the lift for the AGARD 445.6 wing, IMEX3, $\Delta t = 0.001$ s.

partitioned schemes in combination with the RBF mesh deformation are efficiently applicable to real world problems.

For the investigation into the performance of the IMEX scheme and RBF mesh deformation a time step convergence study is performed. As initial condition we use the solution at $t = 0.2$ obtained using the third-order IMEX scheme with $\Delta t = 0.001$. This eliminates the transient at the beginning of the computation. At $t = 0.2$ approximately one period is computed till $t = 0.264$ using the time steps $\Delta t = \{0.016, 0.008, 0.004, 0.002, 0.001\}$ s. The solution obtained with $\Delta t = 0.001$ is taken as the temporally exact solution. In Fig. 11, the results for the pressure field, w-velocity, structural displacement and lift at the end of the simulation are shown. The figures show a clear third-order convergence for all the properties in all the norms, which shows that the combination of the partitioned IMEX scheme with a radial basis function mesh deformation retains the order of the time integration scheme without the necessity to sub-iterate.

Even more important than the order of the scheme is the efficiency of the scheme, e.g. the amount of computational time required to obtain a solution with a certain accuracy. Therefore, we estimate the maximum allowable time step that still provides a physically correct solution, e.g. that still shows a damped oscillation. In Fig. 12a, the time history for the lift is shown for the third-order IMEX scheme used in the time step convergence study. The results show that a time step of $\Delta t = 0.016$ is too large, since the solution starts to diverge. For a time step of $\Delta t = 0.008$ the solution still shows a damped oscillation, but less physical damping is predicted. The error in the lift at the end of the simulation is $\approx 10\%$. In Fig. 12b, the fourth-order IMEX scheme is used with physical time steps of $\Delta t = 0.016$ and $\Delta t = 0.008$. The results show that the largest time step also

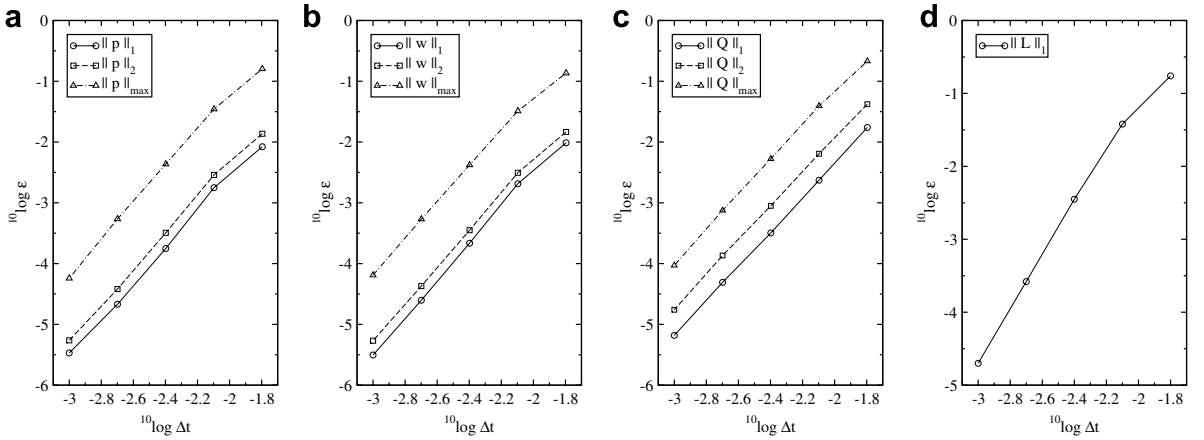

Fig. 11. Time step convergence for the third-order IMEX scheme: (a) pressure; (b) w-velocity; (c) displacement and (d) lift.

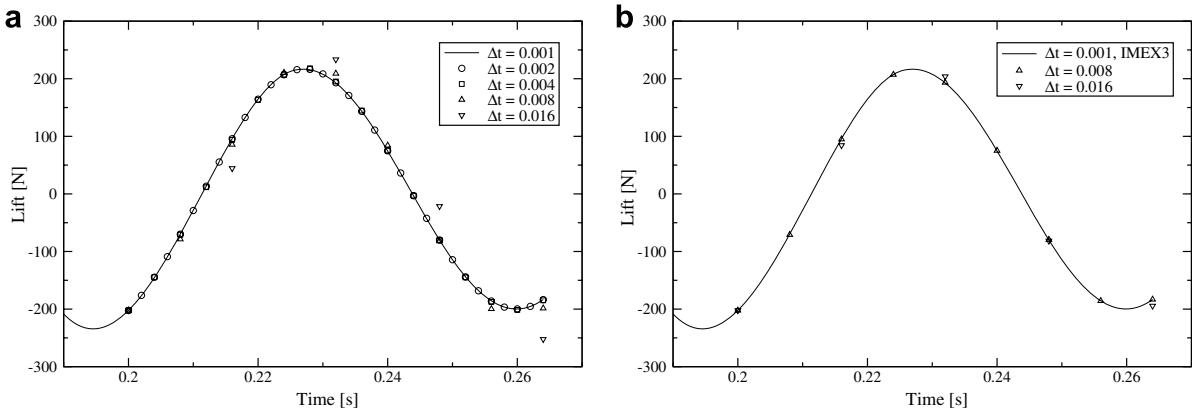

Fig. 12. Time step convergence for the third-order IMEX scheme: (a) IMEX3 and (b) IMEX4.

predicts a smaller physical damping and that the error in the lift is $\approx 10\%$. For $\Delta t = 0.008$ the results are basically equal to the temporally exact solution.

To give an idea of other partitioned methods we mention the second-order partitioned methods used in [43] and [20], which use a physical time step for the integration of the flow of $\Delta t = 0.001$ and $\Delta t = 0.0025$, respectively. The time step of $\Delta t = 0.0025$ is approximately the limit that can be allowed for the second-order partitioned scheme. At this time step the second-order scheme predicts a lower physical damping, comparable to IMEX3 with $\Delta t = 0.008$ and IMEX4 with $\Delta t = 0.016$. However, it is argued in [20] that it is difficult to identify whether a (slowly) diverging solution (like IMEX3, $\Delta t = 0.016$ in Fig. 12a) is caused by a numerical or a physical instability. The only way of identifying the cause of the instability would be to use a smaller time step. Comparing to a time step of $\Delta t = 0.001$ which is necessary for a highly accurate solution with the second-order partitioned schemes, the IMEX schemes can use time steps that are at least a factor of 4 and 8 larger for the third and fourth-order methods, respectively. Since the IMEX schemes use multiple implicit stages per time step (three stages for IMEX3 and five stages for IMEX4), a single time step is computationally more expensive than a time step with the second-order partitioned schemes. Solving an implicit stage is approximately as expensive as solving a single time step with the second-order methods. Taking this into consideration we expect a reduction in computing time compared to second-order partitioned schemes.

4. Conclusions

In this paper, we use a radial basis function interpolation for the deformation of the flow mesh in fluid–structure interaction simulations and high order IMEX schemes for the partitioned time integration. For an academic one-dimensional problem it is shown that the regularity of the flow mesh deformation has a profound influence on the accuracy and order of the time integration. It is also shown that through the accurate and smooth interpolation field obtained with radial basis function interpolation the higher order of the IMEX schemes is preserved.

In this paper, we have extended and applied the partitioned IMEX schemes to three-dimensional applications. By using a radial basis function interpolation the displacements of the structure nodes can be immediately interpolated to a displacement field for the whole fluid domain. For the AGARD 445.6 wing test case, a clear third-order convergence is observed for IMEX3. For simulations with a high temporal accuracy, necessary to discriminate between physical and numerical divergence, the high order schemes are expected to reduce computing times compared to second-order partitioned schemes.

Acknowledgements

The third author, Hester Bijl, would like to thank Piet Wesseling for introducing her into Computational Fluid Dynamics. Piet, you were, and still are, my role model.

Appendix A. Algorithm

Given the initial conditions at $t = 0$: $\mathbf{U}^0$, $\boldsymbol{\kappa}^0$, $\mathbf{x}_\mathrm{f}^0$ and $\mathbf{W}^0$ perform the time integration:

(1) *Initialize first stage*:

$$\mathbf{U}^{(1)} = \mathbf{U}^n, \quad \boldsymbol{\kappa}^{(1)} = \boldsymbol{\kappa}^n, \quad \mathbf{x}_\mathrm{f}^{(1)} = \mathbf{x}_\mathrm{f}^n, \quad \mathbf{R}_\mathrm{f}^{(1)} = \mathbf{R}(\mathbf{U}^{(1)}, \boldsymbol{\kappa}^{(1)}),$$
$$\mathbf{W}^{(1)} = \mathbf{W}^n, \quad \mathbf{F}_\mathrm{sf}^{(1)} = H_\mathrm{sf}\mathbf{p}^{(1)} \quad \text{and} \quad \mathbf{R}_\mathrm{s}^{(1)} = \mathbf{R}_\mathrm{s}(\mathbf{W}^{(1)}, \mathbf{F}_\mathrm{sf}^n).$$

(2) *For $k = 2 \ldots s$ stages do*: Estimate $\mathbf{F}_\mathrm{sf}^{(k)}$ by

$$\mathbf{F}_\mathrm{sf}^{(k)^*} = \sum_{i=1}^{k-1} \frac{\hat{a}_{ki} - a_{ki}}{a_{kk}} \mathbf{F}_\mathrm{sf}^{(i)}.$$

Compute the structure state vector $\mathbf{W}^{(k)}$ which satisfies

$$\mathbf{W}^{(k)} + a_{kk}\Delta t\mathbf{R}_{\mathrm{s}}(\mathbf{W}^{(k)}, \mathbf{F}_{\mathrm{sf}}^{(k)^*}) = \mathbf{W}^n - \Delta t\sum_{i=1}^{k-1} a_{ki}\mathbf{R}_{\mathrm{s}}^{(i)}.$$

Deform the fluid mesh according to the structure displacements at the interface $\mathbf{d}_{\mathrm{s}}^{(k)}$ and compute the mesh velocities satisfying the D-GCL

$$\mathbf{x}_{\mathrm{f}}^{(k)} = \mathbf{x}_{\mathrm{f}}^0 + H_{\Omega_{\mathrm{f}}\mathrm{s}}\mathbf{d}_{\mathrm{s}}^{(k)},$$
$$\boldsymbol{\kappa}^{(k)} = \mathscr{F}(\mathbf{x}_{\mathrm{f}}^n, \mathbf{x}_{\mathrm{f}}^{(i|i=1,\ldots,k)}).$$

Compute the fluid state vector $\mathbf{U}^{(k)}$ which satisfies

$$\mathbf{U}^{(k)} + a_{kk}\Delta t\mathbf{R}_{\mathrm{f}}(\mathbf{U}^{(k)}, \boldsymbol{\kappa}^{(k)}) = \mathbf{U}^n - \Delta t\sum_{i=1}^{k-1} a_{ki}\mathbf{R}_{\mathrm{f}}^{(i)}.$$

Store the flow residual

$$\mathbf{R}_{\mathrm{f}}^{(k)} = \mathbf{R}_{\mathrm{f}}(\mathbf{U}^{(k)}, \boldsymbol{\kappa}^{(k)}),$$

and transfer the boundary pressure $\mathbf{p}^{(k)}$ obtained from the new fluid state to the structure interface to compute and store the corrected structure residual

$$\mathbf{F}_{\mathrm{sf}}^{(k)} = H_{\mathrm{sf}}\mathbf{p}^{(k)},$$
$$\mathbf{R}_{\mathrm{s}}^{(k)} = \mathbf{R}_{\mathrm{s}}(\mathbf{W}_{\mathrm{s}}^{(k)}, \mathbf{F}_{\mathrm{sf}}^{(k)}). \tag{A.1}$$

(3) Update the flow and its mesh properties to the new time level t_{n+1}

$$\mathbf{U}^{n+1} = \mathbf{U}^{(s)}, \boldsymbol{\kappa}^{n+1} = \boldsymbol{\kappa}^{(s)}, \mathbf{x}_{\mathrm{f}}^{n+1} = \mathbf{x}_{\mathrm{f}}^{(s)},$$

and update the structure state

$$\mathbf{W}^{n+1} = \mathbf{W}^n - \Delta t\sum_{i=1}^{s} b_i\mathbf{R}_{\mathrm{s}}^{(i)}.$$

References

[1] P. Geuzaine, G. Brown, C. Harris, C. Farhat, Aeroelastic dynamic analysis of a full F-16 configuration for various flight conditions, AIAA Journal 41 (3) (2003) 363–371.

[2] C. Felippa, K. Park, C. Farhat, Partitioned analysis of coupled mechanical systems, Computer Methods in Applied Mechanics and Engineering 190 (2001) 3247–3270.

[3] S. Piperno, C. Farhat, Partitioned procedures for the transient solution of coupled aeroelastic problems – Part II: energy transfer analysis and three-dimensional applications, Computer Methods in Applied Mechanics and Engineering 190 (2001) 3147–3170.

[4] A.d. Boer, A.v. Zuijlen, H. Bijl, Review of coupling methods for non-matching meshes, Computer Methods in Applied Mechanics and Engineering 196 (2007) 1515–1525.

[5] A. Beckert, H. Wendland, Multivariate interpolation for fluid–structure-interaction problems using radial basis functions, Aerospace Science and Technology 5 (2) (2001) 125–134.

[6] M.J. Smith, C.E.S. Cesnik, D.H. Hodges, Evaluation of some data transfer algorithms for noncontiguous meshes, Journal of Aerospace Engineering 13 (2) (2000) 52–58.

[7] P. Causin, J. Gerbeau, F. Nobile, Added-mass effect in the design of partitioned algorithms for fluid–structure problems, Computer Methods in Applied Mechanics and Engineering 194 (2005) 4506–4527.

[8] J. Vierendeels, K. Dumont, E. Dick, P. Verdonck, Analysis and stabilization of fluid–structure interaction algorithm for rigid body motion, AIAA Journal 43 (12) (2005) 2549–2557.

[9] C. Michler, E. Brummelen, R.d. Borst, An interface Newton–Krylov solver for fluid–structure interaction, International Journal for Numerical Methods in Fluids 47 (10-11) (2005) 1189–1195.

[10] H. Matthies, J. Steindorf, Partitioned strong coupling algorithms for fluid–structure interaction, Computers & Structures 81 (2003) 805–812.

[11] J.T. Batina, Unsteady euler algorithm with unstructured dynamic mesh for complex-aircraft aeroelastic analysis, Technical Report AIAA-89-1189, 1989.

[12] C. Farhat, C. Degand, B. Koobus, M. Lesoinne, Torsional springs for two-dimensional dynamic unstructured fluid meshes, Computer Methods in Applied Mechanics and Engineering 163 (1998) 231–245.

[13] C. Degand, C. Farhat, A three-dimensional torsional spring analogy method for unstructured dynamic meshes, Computers and Structures 80 (2002) 305–316.
[14] D. Lynch, K. ONeill, Elastic grid deformation for moving boundary problems in two space dimensions, in: S. Wang (Ed.), Finite Elements in Water Resources, 1980.
[15] B.T. Helenbrook, Mesh deformation using the biharmonic operator, International Journal for Numerical Methods in Engineering 56 (2003) 1007–1021.
[16] A.d. Boer, M.v.d. Schoot, H. Bijl, Mesh deformation based on radial basis function interpolation, Computers and Structures, in press.
[17] J. Donea, An arbitrary Lagrangian–Eulerian finite element method for transient fluid–structure interactions, Computer Methods in Applied Mechanics and Engineering 33 (1982) 689–723.
[18] C. Farhat, P. Geuzaine, C. Grandmont, The discrete geometric conservation law and the nonlinear stability of ALE schemes for the solution of flow problems on moving grids, Journal of Computational Physics 174 (2001) 669–694.
[19] D. Mavriplis, Z. Yang, Construction of the discrete geometric conservation law for high-order time-accurate simulations on dynamic meshes, Journal of Computational Physics 213 (2006) 557–573.
[20] C. Farhat, M. Lesoinne, Two efficient staggered algorithms for the serial and parallel solution of three-dimensional nonlinear transient aeroelastic problems, Computer Methods in Applied Mechanics and Engineering 182 (2000) 499–515.
[21] M. Giles, Stability and accuracy of numerical boundary conditions in aeroelastic analysis, International Journal for Numerical Methods in Fluids 24 (1997) 739–757.
[22] C. Farhat, CFD on moving grids: from theory to realistic flutter, maneuvering, and multidisciplinary optimization, International Journal of Computational Fluid Dynamics 19 (8) (2005) 595–603.
[23] R. Gordnier, M. Visbal, Computation of the aeroelastic response of a flexible delta wing at high angles of attack, Journal of Fluids and Structures 19 (6) (2004) 785–800.
[24] M. Carpenter, C. Kennedy, H. Bijl, S. Viken, V. Vatsa, Fourth-order Runge–Kutta schemes for fluid mechanics applications, Journal of Scientific Computing 25 (1) (2005) 157–194.
[25] H. Bijl, H. Carpenter, V. Vatsa, C. Kennedy, Implicit time integration schemes for the unsteady compressible Navier–Stokes equations: laminar flow, Journal of Computational Physics 179 (2002) 1–17.
[26] A.v. Zuijlen, H. Bijl, Implicit and explicit high order time integration schemes for structural dynamics and fluid–structure interaction computations, Computers & Structures 83 (2-3) (2005) 93–105.
[27] A.v. Zuijlen, H. Bijl, Implicit and explicit higher-order time integration schemes for fluid–structure interaction computations, International Journal for Multiscale Computational Engineering 4 (2) (2006) 255–263.
[28] E. Yates Jr., AGARD standard aeroelastic configurations for dynamic response. Candidate configuration I.-Wing 445.6, Technical Memorandum 100492, NASA, 1987.
[29] H. Bijl, A.v. Zuijlen, A.v. Mameren, Validation of adaptive unstructured hexahedral mesh computations of flow around a wind turbine airfoil, International Journal for Numerical Methods in Fluids 48 (2005) 929–945.
[30] A. Jameson, W. Schmidt, E. Turkel, Numerical solutions of the Euler equations by finite volume methods using Runge–Kutta time-stepping schemes, AIAA-81-1259, 1981.
[31] Openfem – A Finite Element Toolbox for Matlab and Scilab. Available on: <http://www-rocq.inria.fr/OpenFEM/>, release 2006a, 2006.
[32] T. Hughes, The Finite Element Method, Linear Static and Dynamic Finite Element Analysis, Prentice-Hall Int., Englewood Cliffs, NJ, 1987.
[33] A. de Boer, A.H. van Zuijlen, H. Bijl, Comparison of the conservative and a consistent approach for the coupling of non-matching meshes, in: Proceedings of the European Conference on Computational Fluid Dynamics, ECCOMAS CFD 2006, 2006.
[34] M. Smith, C. Cesnik, D. Hodges, Evaluation of some data transfer algorithms for noncontiguous meshes, Journal of Aerospace Engineering 13 (2) (2000) 52–58.
[35] M. Potsdam, G. Guruswamy, A Parallel Multiblock Mesh Movement Scheme for Complex Aeroelastic Applications, Technical Report AIAA-2001-0716, 2001.
[36] C. Kennedy, M. Carpenter, Additive Runge–Kutta schemes for convection–diffusion-reaction equations, Applied Numerical Mathematics 44 (2003) 139–181.
[37] M. Lesoinne, C. Farhat, Geometric conservation laws for flow problems with moving boundaries and deformable meshes, and their impact on aeroelastic computations, Computer Methods in Applied Mechanics and Engineering 134 (1996) 71–90.
[38] H. Guillard, C. Farhat, On the significance of the geometric conservation law for flow computations on moving meshes, Computer Methods in Applied Mechanics and Engineering 190 (2000) 1467–1482.
[39] A.v. Zuijlen, Fluid–structure Interaction Simulations: Efficient Higher Order Time Integration of Partitioned Systems, Ph.D. thesis, Delft University of Technology, 2006.
[40] D. Martineau, J. Georgala, A Mesh Movement Algorithm for High Quality Generalized Meshes, in: Proceedings of the 42nd AIAA Fluid Dynamics Conference and Exhibit, 2004.
[41] K. Kovalev, M. Delanaye, C. Hirch, Untangling and optimization of unstructured hexahedral meshes, in: S. Ivanenko, V. Garanzha (Eds.), Grid Generation: Theory and Applications, Moscow, Russia, 2002.
[42] R. Beaubien, F. Nitzsche, D. Feszty, Time and frequency domain solutions for the AGARD 445 wing, in: Proceedings of the International Forum on Aeroelasticity and Structural Dynamics (IFASD), Munich, Germany, 2005.
[43] B. Koobus, C. Farhat, Second-order time-accurate and geometrically conservative implicit schemes for flow computations on unstructured dynamic meshes, Computer Methods in Applied Mechanics and Engineering 170 (1999) 103–129.