# Computational engineering analysis with the new-generation space-time methods

Kenji Takizawa

# Computational engineering analysis with the new-generation space–time methods

**Kenji Takizawa**

Department of Modern Mechanical Engineering and Waseda Institute for Advanced Study, Waseda University, 1-6-1 Nishi-Waseda, Shinjuku-ku, Tokyo 169-8050, Japan
e-mail: Kenji.Takizawa@tafsm.org

**Abstract** This is an overview of the new directions we have taken the space–time (ST) methods in bringing solution and analysis to different classes of computationally challenging engineering problems. The classes of problems we have focused on include bio-inspired flapping-wing aerodynamics, wind-turbine aerodynamics, and cardiovascular fluid mechanics. The new directions for the ST methods include the variational multiscale version of the Deforming-Spatial-Domain/Stabilized ST method, using NURBS basis functions in temporal representation of the unknown variables and motion of the solid surfaces and fluid meshes, ST techniques with continuous representation in time, and ST interface-tracking with topology change. We describe the new directions and present examples of the challenging problems solved.

**Keywords** Bio-inspired flapping-wing aerodynamics · MAV · Wind-turbine aerodynamics · Cardiovascular fluid mechanics · Space–time methods · DSD/SST method · ST-SUPS method · ST-VMS method · NURBS in time · STNMUM · ST with continuous temporal representation · ST-C · ST with topology change · ST-TC

## 1 Introduction

In computational engineering analysis, one frequently faces the challenges involved in solving flow problems with moving boundaries and interfaces (MBI). This wide class of problems include fluid–structure interaction (FSI), fluid–object interaction (FOI), fluid–particle interaction (FPI), free-surface and multi-fluid flows, and flows with solid surfaces in fast, linear or rotational relative motion. The computational challenges still being addressed include accurately representing the small-scale flow patterns, which requires a reliable multiscale method. They also include contact or near contact between moving solid surfaces and other cases of topology change (TC) or near TC, such as those in flapping-wing aerodynamics, wind-turbine aerodynamics, and cardiovascular fluid mechanics. These three specific classes of problems played a key role in motivating the development of the computational-analysis methods discussed in this article.

A method for flows with MBI can be viewed as an interface-tracking (moving-mesh) technique or an interface-capturing (nonmoving-mesh) technique, or possibly a combination of the two. In interface-tracking methods, as the interface moves, the mesh moves to follow (i.e. "track") the interface. The Arbitrary Lagrangian–Eulerian (ALE) finite element formulation [1] is the most widely used moving-mesh technique. It has been used for many flow problems with MBI, including FSI (see, for example, [2–35]). The Deforming-Spatial-Domain/Stabilized Space–Time (DSD/SST) method [30,36–42], introduced in 1992, is also a moving-mesh method. The costs and benefits of moving the fluid mechanics mesh to track a fluid–solid interface were articulated in many papers (see, for example, [39,43]).

Moving-mesh methods require mesh update methods. Mesh update typically consists of moving the mesh for as long as possible and remeshing as needed. With the key objectives being to maintain the element quality near solid surfaces and to minimize frequency of remeshing, a number of advanced mesh update methods [40,44–48] were developed in conjunction with the DSD/SST method, including those that minimize the deformation of the layers of small elements placed near solid surfaces.

Perceived challenges in mesh update are quite often given as reasons for avoiding interface-tracking methods in classes of problem that can be, and actually have been, solved with such methods. A robust moving-mesh method with effective mesh update can handle FSI or other MBI problems even when the solid surfaces undergo large displacements (see, for example, FPI [46,49] with the number of particles reaching 1,000 [46], parachute FSI [40,43,50–56], flapping-wing aerodynamics [57–59], and wind-turbine rotor and tower aerodynamics [60]. It can handle FSI or other MBI problems also even when the solid surfaces are in near contact or create near TC, if the "nearness" is sufficiently "near" for the purpose of solving the problem. Examples of such problems are FPI with collision between the particles [46,49], parachute-cluster FSI with contact between the parachutes of the cluster [52–54,56], flapping-wing aerodynamics with the forewings and hindwings crossing each other very close [57–59], and wind-turbine rotor and tower aerodynamics with the blades passing the tower close [60].

No method is a panacea for all classes of MBI problems. As mentioned in [30], certain classes of interfaces, such as those in splashing, might be too complex to deal with an interface-tracking technique, requiring an interface-capturing technique. The Mixed Interface-Tracking/Interface-Capturing Technique (MITICT) [46] was introduced for computations that involve both fluid–solid interfaces that can be accurately tracked with a moving-mesh method and fluid–fluid interfaces that are too complex or unsteady to be tracked. Those fluid–fluid interfaces are captured over the mesh tracking the fluid–solid interfaces. The MITICT was successfully tested in 2D computations with solid circles and free surfaces [61,62] and in 3D computation of ship hydrodynamics [21].

In some MBI problems with contact between the solid surfaces, the "nearness" that can be modeled with a moving-mesh method without actually bringing the surfaces into contact might not be "near" enough for the purpose of solving the problem. Cardiovascular FSI with heart valves, where the flow has to be completely blocked at contact, is an example. The Fluid–Solid Interface-Tracking/Interface-Capturing Technique (FSITICT) [51] was motivated by such FSI problems. In the FSITICT, we track the interface we can with a moving mesh, and capture over that moving mesh the interfaces we cannot track, specifically the interfaces where we need to have an actual contact between the solid surfaces. A specific application of the FSITICT was presented in [63], where the ALE method is used for interface tracking, and a fully Eulerian approach for interface capturing, with some 2D benchmark problems as test computations. This specific application was extended in [63] to 2D FSI models with flapping and contact, where the fully Eulerian interface-capturing is complemented with mesh adaptivity.

Since its inception, the DSD/SST method has been applied to some of the most challenging flow problems with MBI. The classes of problems solved include the free-surface and multi-fluid flows [36,38,44–46,49,61,64–69], FOI [36–38,45,49,61,65,67,70–77], aerodynamics of flapping wings [57–59,78–80], flows with solid surfaces in fast, linear or rotational relative motion [20,46,49,60,81–83], compressible flows [49,65,77,84–86], shallow-water flows [46,87,88], FPI [46,49,67,75,89–93], and FSI [40–43,50–56,59,64–66,69,70,78,86,94–138]. Much of the success with the DSD/SST method in recent years was due to the new directions we have taken the ST methods in bringing solution and analysis to different classes of computationally challenging engineering problems.

The original DSD/SST method is based on the SUPG/PSPG stabilization, where "SUPG" and "PSPG" stand for the Streamline-Upwind/Petrov–Galerkin [139] and Pressure-Stabilizing/Petrov–Galerkin [36,140] methods. Starting in its very early years, the DSD/SST method also included the "LSIC" (least-squares on incompressibility constraint) stabilization. The ST-VMS method [30,41,42] is the variational multiscale version of the DSD/SST method. It was called "DSD/SST-VMST" (i.e. the version with the VMS turbulence model) when it was first introduced in [41]. The VMS components are from the residual-based VMS method given in [141–144]. We demonstrated the increased accuracy brought by the ST-VMS method the first time with the computations reported in [41,83,145]. We have been using the ST-VMS method in most of our computations since then. The original DSD/SST method was named "DSD/SST-SUPS" in [41] (i.e. the version with the SUPG/PSPG stabilization), which was also called "ST-SUPS" in [30].

The ST methods give us the the option of using higher-order basis functions in time, including the NURBS, which have been used very effectively as spatial basis functions (see [4,8,146,147]). We started using that option with the methods and concepts introduced in [41]. This of course increases the order of accuracy in the computations [41,42,129], and the desired accuracy can be attained with larger time steps, but there are positive consequences beyond that. The ST context provides us better accuracy and efficiency in temporal representation of the motion and deformation of the moving interfaces and volume meshes, and better efficiency in remeshing. This has been demonstrated in a number of 3D computations, specifically, flapping-wing aerodynamics [57–59,79,80], separation aerodynamics of spacecraft [55], and wind-turbine aerodynamics [60]. The mesh update method based on using NURBS basis functions in mesh motion and remeshing was named "ST/NURBS Mesh Update Method (STNMUM)" in [60].

There are some advantages in using a discontinuous temporal representation in ST computations. For a given order of

temporal representation, we can reach a higher order accuracy than one would reach with a continuous representation of the same order. When we need to change the spatial discretization (i.e. remesh) between two ST slabs, the temporal discontinuity between the slabs provides a natural framework for that change. There are advantages also in continuous temporal representation. We obtain a smooth solution, NURBS-based when needed. We also can deal with the computed data in a more efficient way, because we can represent the data with fewer temporal control points, and that reduces the computer storage cost. These advantages motivated the development of the ST computation techniques with continuous temporal representation (ST-C) [148].

There are different types of nonmoving-mesh methods that can compute MBI problems involving an actual contact between solid surfaces or other cases of TC. Some of those methods give up on the accurate representation of the interface, and most give up on the consistent representation of the interface motion. The DSD/SST formulation does not need to give up on either, even where we have an actual contact or some other TC, provided that we can update the mesh even there. Using an ST mesh that is unstructured both in space and time, as proposed for contact problems in [46], would give us such a mesh update option. However, that would require a fully unstructured 4D mesh generation, and that is not easy in computing real-world problems. An ST interface-tracking method that can deal with TC was introduced in [138], and it is called ST-TC. It is a practical alternative to using unstructured ST meshes, but without giving up on the accurate representation of the interface or the consistent representation of the interface motion, even where there is an actual contact between solid surfaces or other TC. The ST-TC method is based on special mesh generation and update, and a master–slave system that maintains the connectivity of the "parent" mesh when there is a TC.

In this article we provide an overview of these four new directions we have taken the ST methods and show how that brought solution to the three specific classes of problems mentioned in the first paragraph. In Sect. 2, we briefly review the Navier–Stokes equations of incompressible flows. The ST-SUPS and ST-VMS methods are described in Sects. 3 and 4. Methods based on temporal representation with NURBS basis functions, including the STNMUM, are given in Sect. 5. The ST-C and ST-TC are described in Sects. 6 and 7. In the three sections following that, we present examples of the challenging problems solved. In Sect. 8, we present aerodynamic analysis of flapping wings of an actual locust and an MAV, in Sect. 9 aerodynamic analysis of wind turbines, and in Sect. 10 a proof-of-concept computation with two pairs of symmetrically-flapping surfaces with coordinated opening/closing actions. The concluding remarks are given in Sect. 11.

## 2 Governing equations

Let $\Omega_t \subset \mathbb{R}^{n_{\rm sd}}$ be the spatial domain with boundary $\Gamma_t$ at time $t \in (0, T)$. The subscript $t$ indicates the time-dependence of the domain. The Navier–Stokes equations of incompressible flows are written on $\Omega_t$ and $\forall t \in (0, T)$ as

$$\rho \left( \frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \boldsymbol{\nabla} \mathbf{u} - \mathbf{f} \right) - \boldsymbol{\nabla} \cdot \boldsymbol{\sigma} = \mathbf{0}, \tag{1}$$

$$\boldsymbol{\nabla} \cdot \mathbf{u} = 0, \tag{2}$$

where $\rho$, $\mathbf{u}$ and $\mathbf{f}$ are the density, velocity and the external force, respectively. The stress tensor $\boldsymbol{\sigma}$ is defined as $\boldsymbol{\sigma}(p, \mathbf{u}) = -p\mathbf{I} + 2\mu\boldsymbol{\varepsilon}(\mathbf{u})$, with $\boldsymbol{\varepsilon}(\mathbf{u}) = \left( (\boldsymbol{\nabla}\mathbf{u}) + (\boldsymbol{\nabla}\mathbf{u})^T \right)/2$. Here $p$ is the pressure, $\mathbf{I}$ is the identity tensor, $\mu = \rho\nu$ is the viscosity, $\nu$ is the kinematic viscosity, and $\boldsymbol{\varepsilon}(\mathbf{u})$ is the strain-rate tensor. The essential and natural boundary conditions for Eq. (1) are represented as $\mathbf{u} = \mathbf{g}$ on $(\Gamma_t)_{\rm g}$ and $\mathbf{n} \cdot \boldsymbol{\sigma} = \mathbf{h}$ on $(\Gamma_t)_{\rm h}$, where $(\Gamma_t)_{\rm g}$ and $(\Gamma_t)_{\rm h}$ are complementary subsets of the boundary $\Gamma_t$, $\mathbf{n}$ is the unit normal vector, and $\mathbf{g}$ and $\mathbf{h}$ are given functions. A divergence-free velocity field $\mathbf{u}_0(\mathbf{x})$ is specified as the initial condition.

## 3 ST-SUPS (DSD/SST-SUPS) method

In the DSD/SST method (see, e.g., [36–42]), the finite element formulation is written over a sequence of $N$ ST slabs $Q_n$, where $Q_n$ is the slice of the ST domain between the time levels $t_n$ and $t_{n+1}$ (see Fig. 1). At each time step, the integrations are performed over $Q_n$. The ST finite element interpolation functions are continuous within a ST slab, but discontinuous from one ST slab to another. The notation $(\cdot)_n^-$ and $(\cdot)_n^+$ will denote the function values at $t_n$ as approached from below and above. Each $Q_n$ is decomposed into elements $Q_n^e$, where $e = 1, 2, \ldots, (n_{\rm el})_n$. The subscript $n$ used with $n_{\rm el}$ is for the general case where the number of ST elements may change from one ST slab to another. The essential and natural boundary conditions are enforced over $(P_n)_{\rm g}$ and $(P_n)_{\rm h}$, the complementary subsets of the lateral boundary of

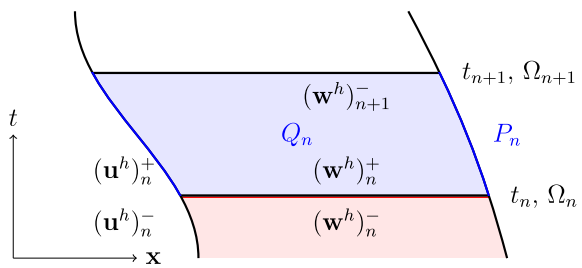


**Fig. 1** ST slab in an abstract representation

the ST slab. The finite element trial function spaces $\mathcal{S}_u^h$ for velocity and $\mathcal{S}_p^h$ for pressure, and the test function spaces $\mathcal{V}_u^h$ and $\mathcal{V}_p^h = \mathcal{S}_p^h$ are defined by using, over $Q_n$, typically first-order polynomials in space and time, but can also be of higher-order functions.

The ST-SUPS (DSD/SST-SUPS) method (from [39]) is written as follows: given $(\mathbf{u}^h)_n^-$, find $\mathbf{u}^h \in \mathcal{S}_u^h$ and $p^h \in \mathcal{S}_p^h$, such that $\forall \mathbf{w}^h \in \mathcal{V}_u^h$ and $\forall q^h \in \mathcal{V}_p^h$:

$$\begin{aligned}
&\int_{Q_n} \mathbf{w}^h \cdot \rho \left( \frac{\partial \mathbf{u}^h}{\partial t} + \mathbf{u}^h \cdot \boldsymbol{\nabla} \mathbf{u}^h - \mathbf{f}^h \right) \mathrm{d}Q \\
&\quad + \int_{Q_n} \boldsymbol{\varepsilon}(\mathbf{w}^h) : \boldsymbol{\sigma}(\mathbf{u}^h, p^h) \mathrm{d}Q - \int_{(P_n)_\mathrm{h}} \mathbf{w}^h \cdot \mathbf{h}^h \mathrm{d}P \\
&\quad + \int_{Q_n} q^h \boldsymbol{\nabla} \cdot \mathbf{u}^h \mathrm{d}Q + \int_{\Omega_n} (\mathbf{w}^h)_n^+ \cdot \rho \left( (\mathbf{u}^h)_n^+ - (\mathbf{u}^h)_n^- \right) \mathrm{d}\Omega \\
&\quad + \sum_{e=1}^{(n_\mathrm{el})_n} \int_{Q_n^e} \frac{1}{\rho} \left[ \tau_\mathrm{SUPG} \rho \left( \frac{\partial \mathbf{w}^h}{\partial t} + \mathbf{u}^h \cdot \boldsymbol{\nabla} \mathbf{w}^h \right) + \tau_\mathrm{PSPG} \boldsymbol{\nabla} q^h \right] \\
&\qquad \cdot \mathbf{r}_\mathrm{M} \mathrm{d}Q + \sum_{e=1}^{(n_\mathrm{el})_n} \int_{Q_n^e} \rho \nu_\mathrm{LSIC} \boldsymbol{\nabla} \cdot \mathbf{w}^h r_\mathrm{C} \mathrm{d}Q = 0, \qquad (3)
\end{aligned}$$

where $\mathbf{r}_\mathrm{M}$ and $r_\mathrm{C}$ are the the residuals of the momentum equation and incompressibility constraint (continuity equation). The ST-SUPS method has all the ingredients of the semi-discrete SUPG/PSPG finite element formulation. That includes the test functions, domain integrations, stress terms that have been integrated by parts, boundary integrations, and the SUPG, PSPG and LSIC stabilization terms with stabilization parameters $\tau_\mathrm{SUPG}$, $\tau_\mathrm{PSPG}$ and $\nu_\mathrm{LSIC}$. The stabilization is residual based because $\mathbf{r}_\mathrm{M}$ and $r_\mathrm{C}$ appear as factors in the stabilization terms. The stabilization parameters $\tau_\mathrm{SUPG}$, $\tau_\mathrm{PSPG}$ and $\nu_\mathrm{LSIC}$ originate from stabilized finite element methods for fluid dynamics (see, e.g., [36,39,139,149–152]). There are various ways of defining these parameters. In the computations included in this article, we mostly use the definitions given in [39], with some new options introduced in [40,60,79,145]. For more ways of calculating the stabilization parameters in finite element computation of flow problems, see [30,106,153–177].

## 4 ST-VMS (DSD/SST-VMST) method

### 4.1 Conservative form

The conservative form of the ST-VMS method is written as follows: given $(\mathbf{u}^h)_n^-$, find $\mathbf{u}^h \in \mathcal{S}_u^h$ and $p^h \in \mathcal{S}_p^h$, such that $\forall \mathbf{w}^h \in \mathcal{V}_u^h$ and $\forall q^h \in \mathcal{V}_p^h$:

$$\begin{aligned}
&\int_{Q_n} \mathbf{w}^h \cdot \rho \left( \frac{\partial \mathbf{u}^h}{\partial t} + \boldsymbol{\nabla} \cdot (\mathbf{u}^h \mathbf{u}^h) - \mathbf{f}^h \right) \mathrm{d}Q \\
&\quad + \int_{Q_n} \boldsymbol{\varepsilon}(\mathbf{w}^h) : \boldsymbol{\sigma}(\mathbf{u}^h, p^h) \mathrm{d}Q - \int_{(P_n)_\mathrm{h}} \mathbf{w}^h \cdot \mathbf{h}^h \mathrm{d}P \\
&\quad + \int_{Q_n} q^h \boldsymbol{\nabla} \cdot \mathbf{u}^h \mathrm{d}Q + \int_{\Omega_n} (\mathbf{w}^h)_n^+ \cdot \rho \left( (\mathbf{u}^h)_n^+ - (\mathbf{u}^h)_n^- \right) \mathrm{d}\Omega \\
&\quad + \sum_{e=1}^{(n_\mathrm{el})_n} \int_{Q_n^e} \left[ \frac{\tau_\mathrm{SUPS}}{\rho} \rho \left( \frac{\partial \mathbf{w}^h}{\partial t} + \mathbf{u}^h \cdot \boldsymbol{\nabla} \mathbf{w}^h \right) + \boldsymbol{\nabla} q^h \right] \cdot \mathbf{r}_\mathrm{M} \mathrm{d}Q \\
&\quad + \sum_{e=1}^{(n_\mathrm{el})_n} \int_{Q_n^e} \rho \nu_\mathrm{LSIC} \boldsymbol{\nabla} \cdot \mathbf{w}^h r_\mathrm{C} \mathrm{d}Q \\
&\quad + \sum_{e=1}^{(n_\mathrm{el})_n} \int_{Q_n^e} \tau_\mathrm{SUPS} \mathbf{r}_\mathrm{M} \cdot \left( \boldsymbol{\nabla} \mathbf{w}^h \right) \mathbf{u}^h \mathrm{d}Q \\
&\quad - \sum_{e=1}^{(n_\mathrm{el})_n} \int_{Q_n^e} \frac{\tau_\mathrm{SUPS}^2}{\rho} \mathbf{r}_\mathrm{M} \cdot \left( \boldsymbol{\nabla} \mathbf{w}^h \right) \mathbf{r}_\mathrm{M} \mathrm{d}Q = 0. \qquad (4)
\end{aligned}$$

The stabilization parameter $\tau_\mathrm{SUPS}$ is calculated in essentially the same way as $\tau_\mathrm{SUPG}$ is calculated. The notation "SUPS," introduced in [41], indicates that there is a single stabilization parameter for the SUPG and PSPG stabilizations, instead of two separate parameters.

*Remark 1* One of the main differences between the ALE and ST forms of the VMS method is that the ST form retains the fine-scale time derivative term $\frac{\partial \mathbf{u}'}{\partial t}|_{\boldsymbol{\xi}}$ ($\boldsymbol{\xi}$ is the vector of element coordinates). Dropping this term is called the "quasi-static" assumption (see [15] for the terminology). This is the same as the "WTSE" option in the DSD/SST formulation (see Remark 2 of [40]). We believe that this makes a significant difference, especially when the polynomial orders in space or time are higher (see [41]).

### 4.2 Convective form

$$\begin{aligned}
&\int_{Q_n} \mathbf{w}^h \cdot \rho \left( \frac{\partial \mathbf{u}^h}{\partial t} + \mathbf{u}^h \cdot \boldsymbol{\nabla} \mathbf{u}^h - \mathbf{f}^h \right) \mathrm{d}Q \\
&\quad + \int_{Q_n} \boldsymbol{\varepsilon}(\mathbf{w}^h) : \boldsymbol{\sigma}(\mathbf{u}^h, p^h) \mathrm{d}Q - \int_{(P_n)_\mathrm{h}} \mathbf{w}^h \cdot \mathbf{h}^h \mathrm{d}P \\
&\quad + \int_{Q_n} q^h \boldsymbol{\nabla} \cdot \mathbf{u}^h \mathrm{d}Q + \int_{\Omega_n} (\mathbf{w}^h)_n^+ \cdot \rho \left( (\mathbf{u}^h)_n^+ - (\mathbf{u}^h)_n^- \right) \mathrm{d}\Omega \\
&\quad + \sum_{e=1}^{(n_\mathrm{el})_n} \int_{Q_n^e} \left[ \frac{\tau_\mathrm{SUPS}}{\rho} \rho \left( \frac{\partial \mathbf{w}^h}{\partial t} + \mathbf{u}^h \cdot \boldsymbol{\nabla} \mathbf{w}^h \right) + \boldsymbol{\nabla} q^h \right] \cdot \mathbf{r}_\mathrm{M} \mathrm{d}Q
\end{aligned}$$

$$
\begin{aligned}
&+ \sum_{e=1}^{(n_{\mathrm{el}})_n} \int_{Q_n^e} \nu_{\mathrm{LSIC}} \boldsymbol{\nabla} \cdot \mathbf{w}^h \rho r_{\mathrm{C}} \mathrm{d}Q \\
&- \sum_{e=1}^{(n_{\mathrm{el}})_n} \int_{Q_n^e} \tau_{\mathrm{SUPS}} \mathbf{w}^h \cdot \left( \mathbf{r}_{\mathrm{M}} \cdot \boldsymbol{\nabla} \mathbf{u}^h \right) \mathrm{d}Q \\
&- \sum_{e=1}^{(n_{\mathrm{el}})_n} \int_{Q_n^e} \frac{\tau_{\mathrm{SUPS}}^2}{\rho} \mathbf{r}_{\mathrm{M}} \cdot \left( \boldsymbol{\nabla} \mathbf{w}^h \cdot \mathbf{r}_{\mathrm{M}} \right) \mathrm{d}Q = 0. \qquad (5)
\end{aligned}
$$

*Remark 2* The 6th and 7th terms of the ST-VMS method in either form are the SUPG/PSPG and LSIC stabilization terms, respectively. If we exclude the last two terms of the convective form, the method reduces to the ST-SUPS (DSD/SST-SUPS) method under the condition $\tau_{\mathrm{PSPG}} = \tau_{\mathrm{SUPG}}$.

## 5 Temporal representation with NURBS basis functions

The concept of using NURBS basis functions, in conjunction with the ST methods, in temporal representation of the unknown variables and motion of the solid surfaces and fluid meshes was first introduced in [41].

### 5.1 ST basis functions

An ST basis function can be written as a product of its spatial and temporal parts:

$$
\begin{aligned}
N_a^\alpha = T^\alpha(\theta)\, N_a(\boldsymbol{\xi}), \quad & a = 1, 2, \ldots, n_{\mathrm{en}}, \\
& \alpha = 1, 2, \ldots, n_{\mathrm{ent}}, \qquad (6)
\end{aligned}
$$

where $\theta \in [-1,\ 1]$ is the temporal element coordinate, and $n_{\mathrm{en}}$ and $n_{\mathrm{ent}}$ are the number of spatial and temporal element nodes. Figure 2 shows an example of temporal NURBS basis functions. Temporal NURBS basis functions can be used in an ST slab for the representation of the unknown variables and test functions as well as the spatial coordinates.

As pointed out in [30,41,42,60,79], different components (i.e. unknowns), and the corresponding test functions, can be discretized with different sets of temporal basis functions. This was shown in [30,41,42,60,79] by introducing a secondary mapping $\Theta_\zeta(\theta) \in [-1,\ 1]$, where $\Theta_\zeta(\theta)$ is a strictly increasing function, and rewriting the generalized ST basis function for the element indices $(a,\ \alpha)$ as

$$
\left(N_a^\alpha\right)_\zeta = T^\alpha\left(\Theta_\zeta(\theta)\right) N_a(\boldsymbol{\xi}). \qquad (7)
$$

For example, we can discretize time and position as

$$
t = \sum_{\alpha=1}^{n_{\mathrm{ent}}} T^\alpha(\Theta_t(\theta)) t^\alpha, \qquad (8)
$$

$$
\mathbf{x} = \sum_{\alpha=1}^{n_{\mathrm{ent}}} T^\alpha(\Theta_x(\theta)) \mathbf{x}^\alpha. \qquad (9)
$$

Here $\Theta_t(\theta)$ and $\Theta_x(\theta)$ are the secondary mappings for time and position, and $t^\alpha$ and $\mathbf{x}^\alpha$ are the time and position values corresponding to the basis function $T^\alpha$.

### 5.2 Motion of solid surfaces

As an example, Fig. 3 shows, from [79], how the motion of the forewing (FW) of a locust is represented temporally. In the preliminary computations reported in [79], quadratic NURBS basis functions were used in the temporal representation of the wing motion. Based on those computations, using even higher-order temporal basis functions was proposed in [79], so that the acceleration is continuous. Fig. 4 shows, from [57], how the path of a point on the hindwing (HW) of the locust is represented with cubic NURBS basis functions over four temporal patches (see [57] for details).

### 5.3 Rotation representation with constant angular velocity

With temporal NURBS basis functions, as described in [30, 41,42,60,79], we can represent a path accurately, such as representing a circular arc exactly with quadratic NURBS.

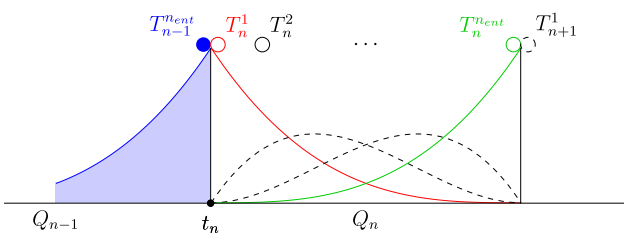


**Fig. 2** Temporal NURBS basis functions

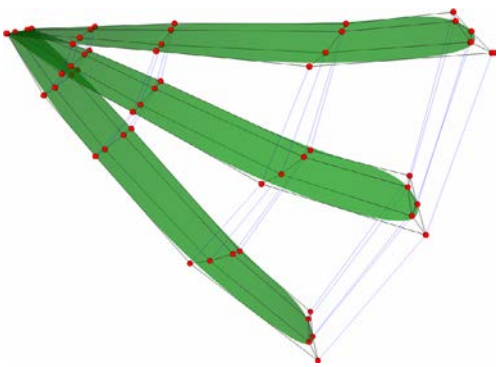

**Fig. 3** FW control mesh and corresponding surface at three temporal-control points

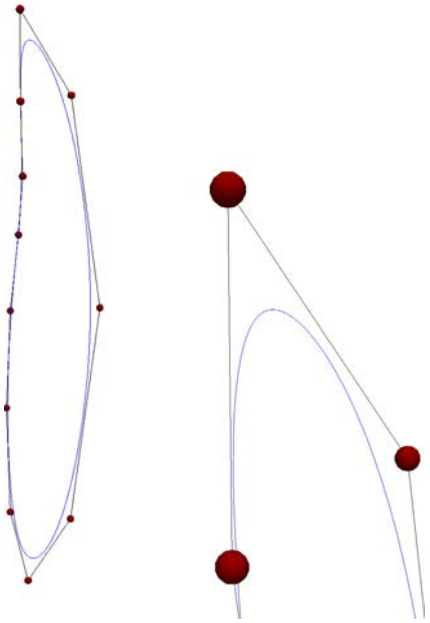

**Fig. 4** Path of a point on the HW as represented with cubic NURBS basis functions over four temporal patches

We can also specify a speed along that path, such as a constant angular velocity for a circular arc [30,60,79]. In this this subsection, from [60], we describe how to do that.

For the circular arc, with quadratic NURBS, $n_{\text{ent}} = 3$. With the secondary-mapping concept described in Sect. 5.1, the velocity can be expressed as follows:

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \sum_{\alpha=1}^{n_{\text{ent}}} \frac{\mathrm{d}T^{\alpha}}{\mathrm{d}\Theta_x}\frac{\mathrm{d}\Theta_x}{\mathrm{d}\theta}\mathbf{x}^{\alpha}\Bigg)\Bigg(\sum_{\alpha=1}^{n_{\text{ent}}} \frac{\mathrm{d}T^{\alpha}}{\mathrm{d}\Theta_t}\frac{\mathrm{d}\Theta_t}{\mathrm{d}\theta}t^{\alpha}\Bigg)^{-1}, \qquad (10)$$

leading to

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \sum_{\alpha=1}^{n_{\text{ent}}} \frac{\mathrm{d}T^{\alpha}}{\mathrm{d}\Theta_x}\mathbf{x}^{\alpha}\Bigg)\Bigg(\sum_{\alpha=1}^{n_{\text{ent}}} \frac{\mathrm{d}T^{\alpha}}{\mathrm{d}\Theta_t}t^{\alpha}\Bigg)^{-1}\left(\frac{\mathrm{d}\Theta_x}{\mathrm{d}\theta}\frac{\mathrm{d}\theta}{\mathrm{d}\Theta_t}\right). \qquad (11)$$

Thus, the speed along the path can be specified only by modifying the secondary mapping. For a circular arc, two methods were introduced in [79] and also described in [30]; one is modifying the secondary mapping for position and the other one is modifying both such that $\frac{\mathrm{d}t}{\mathrm{d}\theta}$ is constant. We note that, in theory, the secondary-mapping selections do not make any difference as long as the relationship $\frac{\mathrm{d}\Theta_x}{\mathrm{d}\Theta_t}$ is the same. In our implementation, to keep the process general, we search for the parametric coordinate $\theta$ by using an iterative solution method [30,79]. We use the latter set of the secondary mappings, having constant $\frac{\mathrm{d}t}{\mathrm{d}\theta}$.

## 5.4 STNMUM

### *5.4.1 Mesh computation and representation*

Given the fluid mechanics mesh on a moving solid surface, we compute the fluid mechanics volume mesh using the mesh moving techniques [40,44–46] developed in conjunction with the DSD/SST method. As proposed in [79] and also described in [30], these mesh moving techniques are applied to computing the meshes that will serve as temporal-control points. This allows us to do mesh computations with longer time in between, but get the mesh-related information, such as the coordinates and their time derivatives, from the temporal representation whenever we need. Obviously this also reduces the storage amount and access associated with the meshes. However, because of the longer time between the control meshes, linear interpolation of the surfaces between control points in time might be needed in computing those meshes with the mesh moving technique mentioned.

*Remark 3* Getting the meshes used in the computations from the temporal representation can be done independent of which time direction was used in computing the control meshes. For example, in flapping-wing aerodynamics, it does not matter if the control meshes were computed while the wings were flapping forward or backward in time.

### *5.4.2 Remeshing*

In many computations remeshing becomes unavoidable. Two choices were proposed in [79] and also described in [30]. To explain those choices, let us assume that when we try to move from control mesh $M_c^{\beta}$ to $M_c^{\beta+1}$, we find the quality of $M_c^{\beta+1}$ to be less than desirable. In the first choice, called "trimming," we remesh going back to $M_c^{\beta-p+1}$, where $p$ is the order of the NURBS basis functions. Then whenever our solution process needs a mesh, depending on the time, we use the control meshes belonging to either only the un-remeshed set or only the remeshed set (Fig. 5). In the second choice, we perform knot insertion $p$ times in the temporal representation of the surface at the right-most knot before the maximum value of the basis function corresponding to $t_c^{\beta+1}$, making that knot a new patch boundary. Then we do the mesh moving computation for the control meshes associated with the newly-defined basis functions, not only the one at the new patch boundary, but also going back $(p-1)$ basis functions (Fig. 6). We favor the second choice, because we believe that in many cases the need for remeshing is generated by a topological change, which we can avoid going over with a large step if we use the knot insertion process.

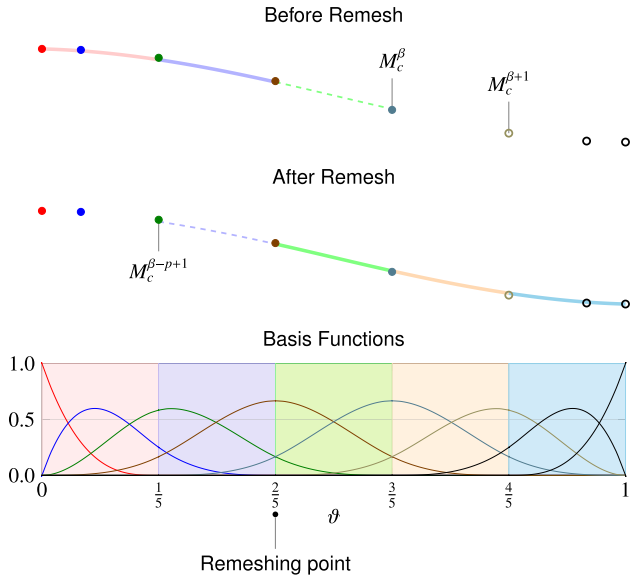


**Fig. 5** Remeshing and trimming NURBS. A set of un-remeshed meshes (*top*). A set of remeshed meshes (*middle*). Common basis functions (*bottom*)

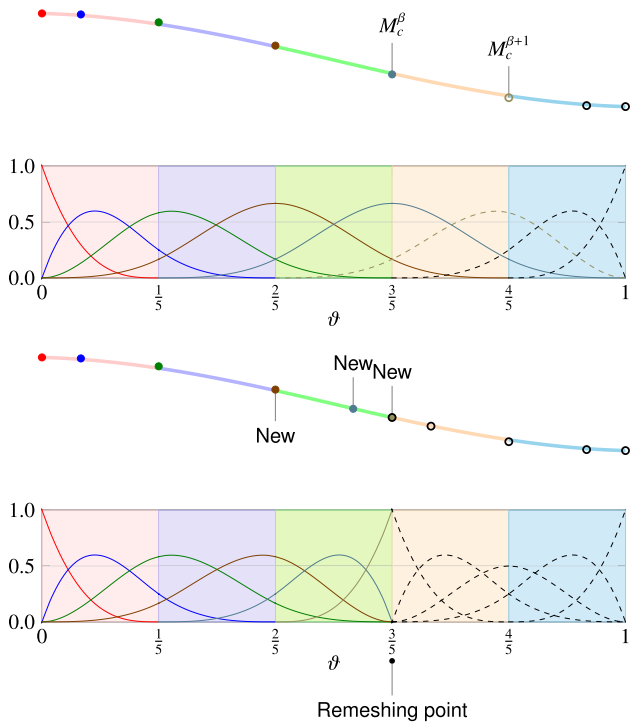


**Fig. 6** Remeshing with knot insertion. For the set of un-remeshed meshes, there are $p$ newly-defined basis functions and the corresponding control points are marked "New." We carry out the mesh moving computations for those meshes

## 6 ST-C

In this section, from [148], we describe the version of ST-C used in extracting continuous temporal representation from computed data. This is essentially a post-processing method, and can also be seen as a data compression method. For the version used in direct computation of the solution with continuous temporal representation, see [148].

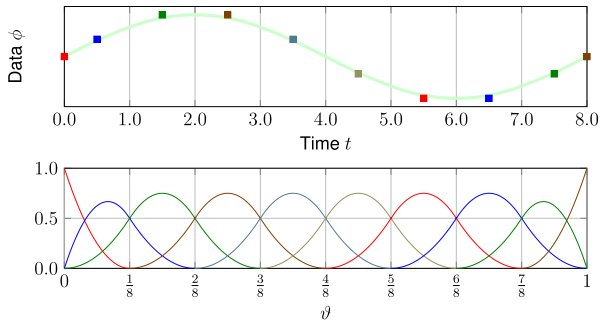


**Fig. 7** Continuous solution (*top*) and its basis functions (*bottom*), where $\vartheta$ is the parametric coordinate

### 6.1 Least-squares projection for full temporal domain

When we have the complete sequence of computed data, we can project that to a smooth representation, with basis functions that provide us that smooth representation, such as NURBS basis functions. As an example, Fig. 7 shows the goal continuous data $\phi_{\mathrm{C}}$ and its basis functions, where $\vartheta$ denotes the parametric temporal coordinate. The projection for each spatial node can be done independently from the other nodes. Consider the time-dependent, typically discontinuous computed data $\phi_{\mathrm{D}}$ for a node. We define the basis functions as $T_{\mathrm{C}}^{\alpha}$, where $\alpha = 0, 1, \ldots$, and the coefficients to be determined in the projection as $\phi^{\alpha}$. We use a standard least-squares projection: given $\phi_{\mathrm{D}}$, find the solution $\phi_{\mathrm{C}} \in \mathcal{S}_{\mathrm{C}}$, such that for all test functions $w_{\mathrm{C}} \in \mathcal{V}_{\mathrm{C}}$:

$$\int_0^T w_{\mathrm{C}} \left(\phi_{\mathrm{C}} - \phi_{\mathrm{D}}\right) \mathrm{d}t = 0, \tag{12}$$

where $T$ represents time period of the computation, and $\mathcal{S}_{\mathrm{C}}$ and $\mathcal{V}_{\mathrm{C}}$ are the solution and test function spaces constructed from the basis functions. This approach requires that we store all the computed data before the projection, and that would be a significant computer storage cost when the number of time steps is large.

### 6.2 Successive-projection technique

In ST-C with the successive-projection technique (ST-C-SPT), we extract the continuous solution shown in Fig. 7 without storing all the computed data. We describe the technique here for the special case with quadratic B-splines.

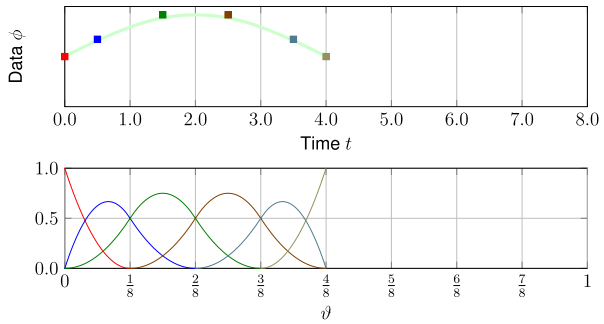


**Fig. 8** Continuous solution up to $t_n = 4.0$ (*top*) and its basis functions (*bottom*)

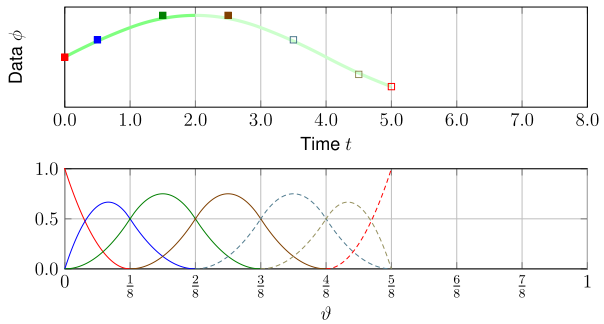


**Fig. 9** Continuous solution up to $t_{n+1} = 5.0$ (*top*) and its basis functions (*bottom*). The bold part of the top curve indicates the part of the solution that does not change. The empty squares denote the temporal control values to be determined. The dashed lines denote the modified and new basis functions, which correspond to those control values

To explain the successive nature of the SPT, let us suppose that we have the continuous solution extracted up to $t_n = 4.0$, as shown in Fig. 8. We assume that this continuous solution, which we will call $\overline{\phi}_{\mathrm{C}}$, has already replaced $\phi_{\mathrm{D}}$ up to $t_n = 4.0$. With that, we describe how we extract the continuous solution up to $t_{n+1} = 5.0$, as shown in Fig. 9. With the newly computed data $\phi_{\mathrm{D}}$ between $t_n = 4.0$ and $t_{n+1} = 5.0$, we solve the following projection equation: given $\phi_{\mathrm{D}}$ on $t \in (4.0, 5.0)$, $\overline{\phi}_{\mathrm{C}}$ on $t \in [2.0, 4.0]$, and $\phi_{\mathrm{C}}^{\alpha}$, $\alpha = 2, 3$, find $\phi_{\mathrm{C}} \in \mathcal{S}_{\mathrm{C}}$, such that $\forall w_{\mathrm{C}} \in \mathcal{V}_{\mathrm{C}}$:

$$\int_{2.0}^{4.0} w_{\mathrm{C}} \left(\phi_{\mathrm{C}} - \overline{\phi}_{\mathrm{C}}\right) \mathrm{d}t + \int_{4.0}^{5.0} w_{\mathrm{C}} \left(\phi_{\mathrm{C}} - \phi_{\mathrm{D}}\right) \mathrm{d}t = 0. \tag{13}$$

We note that Eq. (13) is essentially used for defining the coefficients $\phi_{\mathrm{C}}^{\alpha}$, $\alpha = 4, 5, 6$, which correspond to the basis functions $T_{\mathrm{C}}^{\alpha}$. How to deal with the initial part of the extraction, description of the ST-C-SPT for the general case (i.e. beyond quadratic B-splines), and comments on efficient implementation of the SPT can be found in [148].

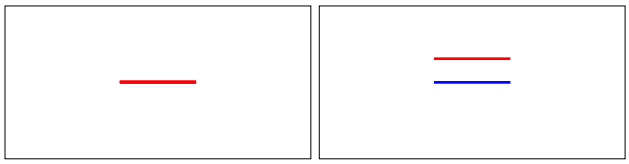

**Fig. 10** Hypothetical case of two bars that are initially coinciding, with one hole in the fluid mechanics domain (*left*). Then the red bar starts moving upward, creating a second hole in the domain (*right*)

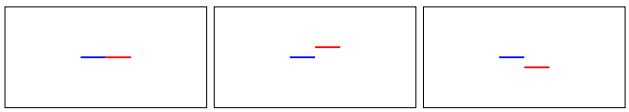

**Fig. 11** Hypothetical case of two bars that are are initially aligned with connected ends, with one hole in the domain (*left*). Then the red bar starts a flapping motion, up (*middle*) and down (*right*), creating a second hole in the domain, except when their ends become connected periodically during the flapping motion

## 7 ST-TC

### 7.1 TC

We consider two hypothetical cases of two bars to provide a context for TC. In the first case, shown in Fig. 10, the bars are initially coinciding, with just one hole in the fluid mechanics domain. Then the red bar starts moving upward, creating a second hole. In the second case, shown in Fig. 11, the bars are initially aligned with connected ends, again with a single hole in the domain. Then the red bar starts a flapping motion, up and down, creating a second hole in the domain, except when their ends become connected periodically during the flapping motion. When the red bar is in the upper position, the part of the domain below it is connected to the part of the domain above the blue bar. When the red bar is in the lower position, the part of the domain above it is connected to the part of the domain below the blue bar. These two cases are representatives of the typical TC challenges we expect to see in the classes of MBI problems we are targeting. Especially the first case is really not possible to treat in a consistent way without using an ST method.

#### *7.1.1 Master–slave system*

We propose a very simple technique in the ST context. Having a constraint between nodes in a finite element formulation is quite common. These constraints reduce the number of unknowns, but in our implementation we delay that unknown elimination until the iterative solution of the linear systems encountered at nonlinear iterations of a time step. The iterative solution of the linear systems is performed with reduced number of unknowns. The technique is easy to manage in a parallel-computing environment, especially if the precondi-

tioner is simple enough. Typically we assign a master node to each slave node, and we use only the unknowns of the master nodes in iterative solution of the linear systems. We can use different master–slave relationships at different time levels. This is a practical alternative to, but less general than, using ST meshes that are unstructured in time. Still, we can use this concept to deal with the TC cases considered above, and the important point is that the connectivity of the "parent" mesh does not change. Consequently, the distribution model in the parallel-computing environment does not change during the computations with moving meshes.

With this technique, we need to implement one more functionality. We exclude certain elements from the integration of the finite element formulation. The exclusion principles are given below.

- Exclude all spatial elements with zero volume from the spatial integration.
- Exclude all ST elements with zero ST volume from the ST integration.
- We assume that checking if an ST element has zero ST volume is equivalent to checking if all the spatial elements associated with that ST element have zero volume. Therefore, for this purpose, we check the spatial-element volumes.
- To identify the spatial elements with zero volume, which should have zero Jacobian at all the integration points, instead of evaluating the Jacobians, we make the determination for a given spatial element from the master–slave relationship of its nodes. The method is explained more in [138].

#### *7.1.2 Design of the master–slave system*

The data we need to provide to the solver is simple. It is just the master–slave relationship at each time level. However there are some restrictions, and here we explain the three that we want to emphasize. The first restriction is that we cannot have a node which is not part of any active (nonzero volume) spatial element. This is because the values at such nodes would no longer be in our equation system, and therefore would become undefined. If because of another TC such a node comes back to the equation system later as part of an active element, it would add an undefined component to the equation system. The second restriction is that when we construct the ST elements, we have to have matching lateral element-boundary faces between the active adjacent ST elements. This condition cannot be checked on the spatial mesh. Therefore we need to check it on the ST mesh. The third one is related to implementation. The master–slave relationship also extends to cases when we have boundary conditions on the master and slave nodes. In other words, the conditions at the master node also apply to the slave nodes.

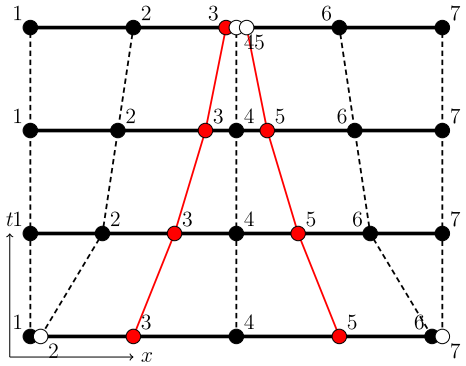


**Fig. 12** Contraction. The *red* nodes, 3 and 5, are on the contraction interface and are contacting. The *white* nodes are the slaves. They are in the same position as their masters, but for visualization purposes we slightly shift their positions in the figure. The numbers indicate the node numbers on the parent mesh. (Color figure online)

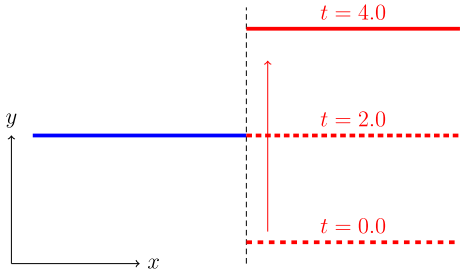


**Fig. 13** Flapping. *Red* and *blue bars* at different instants in time as the *red bar* crosses the *blue bar*. (Color figure online)

### 7.2 Conceptual examples

#### *7.2.1 Contraction and expansion*

This is related to the first one of the two cases of TC described in Sect. 7.1. Contraction and expansion are basically the same, except having different directions in time progression. Figure 12 shows a contraction example. The spatial element with nodes 1 and 2, for example, has zero volume at the first time level. However, it has nonzero volume at the second time level, and therefore the corresponding ST element has nonzero volume.

#### *7.2.2 Flapping*

This is related to the second one of the two cases of TC described in Sect. 7.1. Figure 13 shows the red and blue bars at three instants in time as the red bar crosses the blue bar. Figure 14 shows, for the flapping motion, the ST trajectories of the neighboring ends of the blue and red bars. Figure 15 shows the ST element edges for the line separating the two sides of the domain containing the blue and red bars (shown as the vertical dashed line in Fig. 13). For each side of the domain, the spatial node motions along the ST element edges

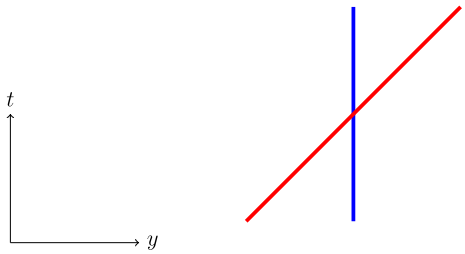


**Fig. 14** Flapping. The ST trajectories of the neighboring ends of the *blue* and *red bars*. (Color figure online)

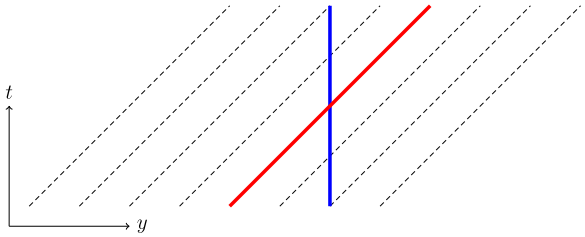


**Fig. 15** Flapping. The ST element edges for the *vertical dashed line* in Fig. 13

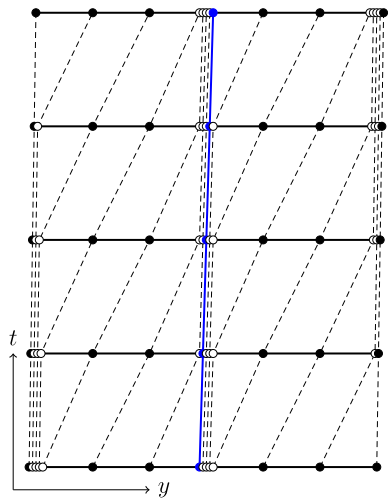


**Fig. 16** Flapping. *Blue-bar* side of the ST boundary between the two sides. (Color figure online)

have to be designed in a fashion that does not lead to mesh entanglement. Figure 16 shows the master–slave relationship for the blue-bar side of the domain, and Fig. 17 the red-bar side. In addition, those two sides are in a master–slave relationship along the vertical dashed line in Fig. 13.

## 8 Aerodynamic analysis of flapping wings of an actual locust and an MAV

This section is from [57,58]. The fluid mechanics computations are carried out with the conservative form of the ST-VMS method. More information on the computational para-

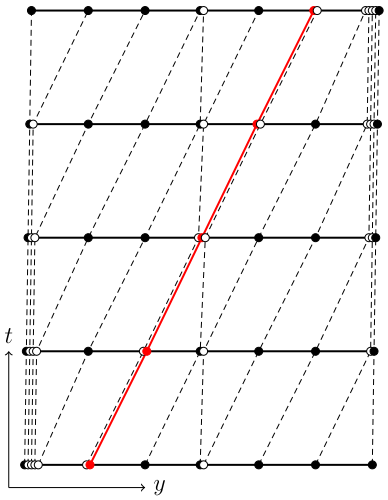


**Fig. 17** Flapping. *Red-bar* side of the ST boundary between the two sides. (Color figure online)

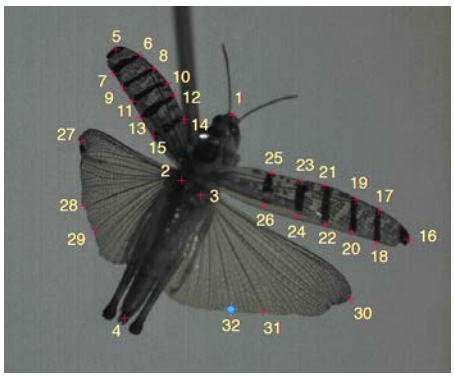


**Fig. 18** Tracking points in the data set from the BCM wind tunnel

meters and conditions can be found in [57,58]. The motion and deformation data for the wings is extracted from the high-speed, multi-camera video recordings of a locust in a wind tunnel at Baylor College of Medicine (BCM), Houston, Texas. The video recording is accomplished by using a set of tracking points marked on the FWs and HWs of the locust. The tracking points can be seen in Fig. 18. The interested reader can find in [57] the details of how the wing motion and deformation data is extracted from the video data and represented in space and time with the methods described in Sect. 5, including the STNMUM, and some additional techniques. Figure 19 illustrates, in the context of the HW wingtip trajectory, how the STNMUM is used in the computation. Figure 20 show how the body and wings compare for the locust and MAV models. Figure 21 shows for the locust the vorticity magnitude during the second flapping cycle. Figure 22 shows for the MAV the vorticity magnitude during the third flapping cycle. In Figs. 21 and 22, the color range from blue to red corresponds to a vorticity range from low to high, and lighter and darker shades of a color correspond to

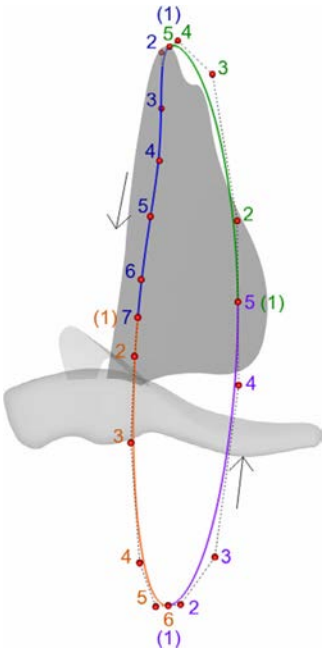


**Fig. 19** Mesh update with the STNMUM in the context of the HW wingtip trajectory, represented with cubic NURBS over four temporal patches. Control point numbering is local to each patch. A control point at the start of a patch and collocated with a control point at the end of the previous patch is in parentheses. A fluid mechanics volume mesh is generated for each patch at the middle control point, and the meshes at the other control points are computed with the mesh moving techniques developed in conjunction with the DSD/SST method. That gives us a temporal representation of the mesh

**Fig. 20** Locust body and wings (*left*) and MAV body and wings (*right*)

lower and higher values. Figure 23 shows the lift and thrust generated by the locust and MAV.

## 9 Aerodynamic analysis of wind turbines

This section is from [83] and [60]. Computer modeling of wind-turbine aerodynamics is challenging because correct aerodynamic torque calculation requires correct separation-point calculation, which requires an accurate flow field, which in turn requires good mesh resolution and turbulence model. We use an actual wind-turbine model, which is NREL 5MW offshore baseline wind turbine, and the geometric complexity is also a computational challenge. Including the tower in the model increases the computational challenge because of the mesh update requirements of the fast, rotational relative motion between the rotor and tower.

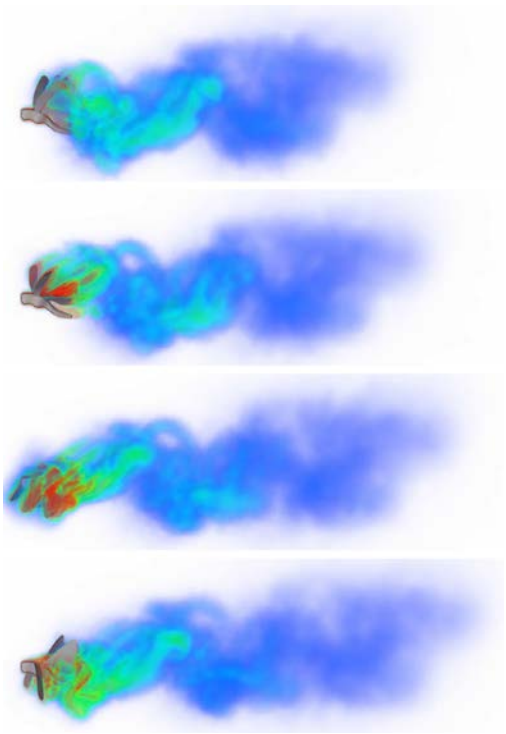

**Fig. 21** Locust. Vorticity for four equally-spaced points during the second flapping cycle

First we describe how we computed the aerodynamics of a rotor without the tower by using the ST-SUPS and ST-VMS methods. More information on the computational parameters and conditions can be found in [83]. Figure 24 shows, from [15], the airfoil cross-sections of the wind-turbine blade superposed on the blade. Figure 25 shows time history of the aerodynamic torque generated by a single blade, as computed with the ST-SUPS, ST-VMS (conservative form) and ALE methods. The ALE results are from [15], which we take as the reference solution. The figure demonstrates how the ST-VMS method increases the accuracy in this particular computational analysis.

When we include the tower, we deal with the mesh update requirements with the methods described in Sect. 5, including the STNMUM. Figure 26 illustrates, in the context of the blade tip trajectory, how the STNMUM is used in the computation. More information can be found in [60]. Figure 27 shows the vorticity magnitude, computed with the conservative form of the ST-VMS method and the STNMUM. In that figure, the color range from blue to red corresponds to a vorticity range from low to high, and lighter and darker shades of a color correspond to lower and higher values. Figure 28 shows the torque for the individual blades.

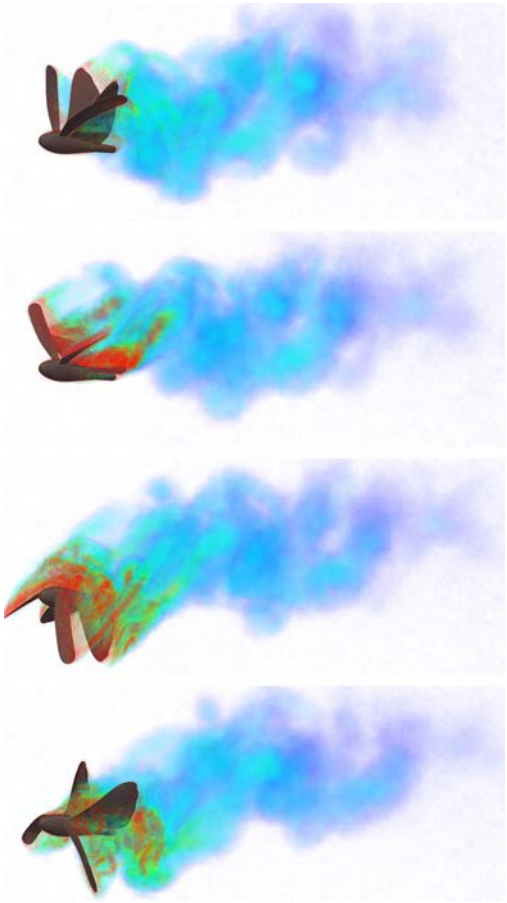

**Fig. 22** MAV. Vorticity for four equally-spaced points during the third flapping cycle

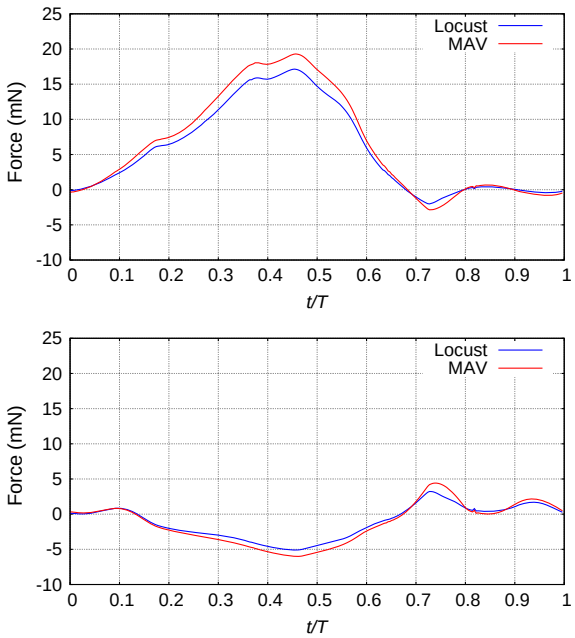


**Fig. 23** Total lift (*top*) and thrust (*bottom*) generated over one cycle

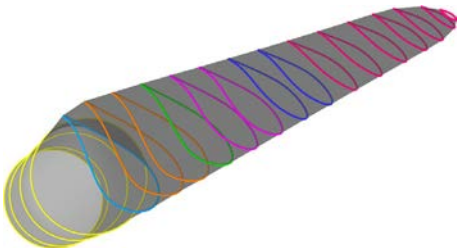

**Fig. 24** Airfoil cross-sections of the wind-turbine blade superposed on the blade (from [15])

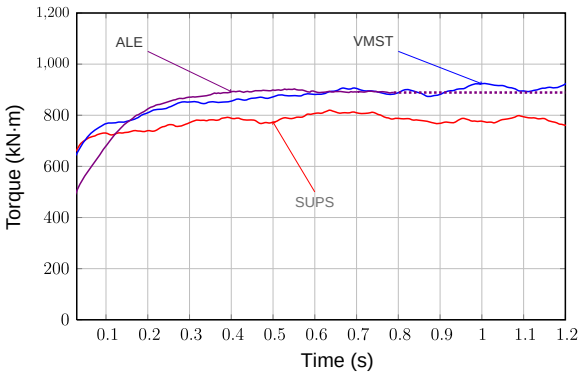


**Fig. 25** Time history of the aerodynamic torque generated by a single blade. Computed with the ST-SUPS ("SUPS"), ST-VMS ("VMST"), and ALE methods

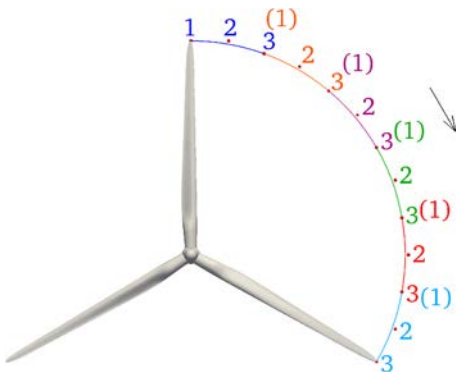

**Fig. 26** Mesh update with the STNMUM in the context of the blade tip trajectory, represented with quadratic NURBS over six temporal patches in 1/3 rotation. Control point numbering is local to each patch. A control point at the start of a patch and collocated with a control point at the end of the previous patch is in parentheses. A fluid mechanics volume mesh is generated for each patch at the middle control point, and the meshes at the other control points are computed with the mesh moving techniques developed in conjunction with the DSD/SST method. That gives us a temporal representation of the mesh

*Remark 4* The drop in the aerodynamic torque as the blade passes the tower is a well-known phenomenon, observed in experiments and in other computations of wind-turbine aerodynamics (see, for example, [35]). In [35], a sliding-interface technique [147] was used in conjunction with the ALE-VMS formulation [20] to compute wind-turbine aerodynamics, including the rotor–tower interaction.

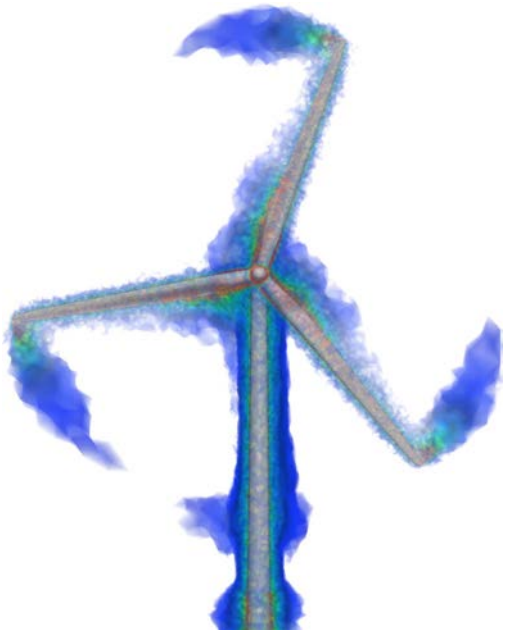

**Fig. 27** Vorticity, computed with the ST-VMS method and the STNMUM

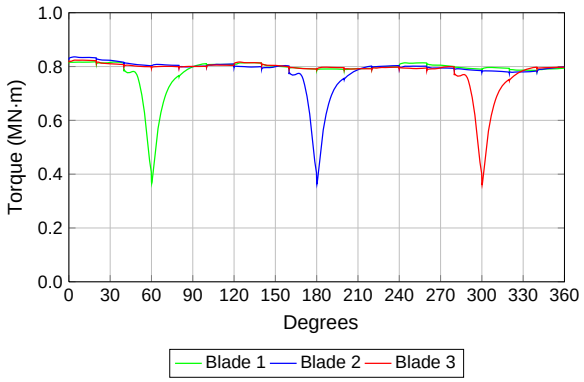


**Fig. 28** Torque for the individual blades. The figure clearly shows the expected torque drop for each blade as it passes the tower, while the other two blades maintain relatively constant torque

## 10 Two pairs of symmetrically-flapping surfaces with coordinated opening/closing actions

This section is from [138]. The 2D model was intended to resemble the left ventricle of human heart and represent the TC challenges one would face in computing the blood flow. There are two pairs of symmetrically-flapping surfaces with zero thickness, positioned and functioning like the mitral and aortic valves would be. The 2D domain changes its area like the left ventricle would change its volume. When one of the pairs closes, the domain is separated into two. Figure 29 shows the computational domain. The flow enters through the flapping pair at the inlet (we call this pair "mitral"), and exits through the pair at the outlet (we call this pair "aortic"). We

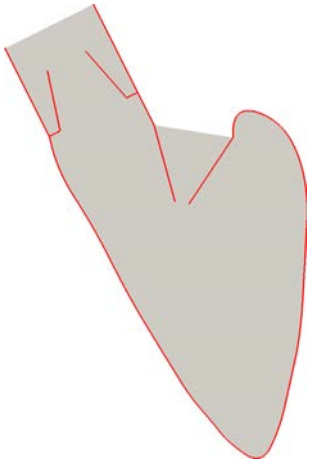

**Fig. 29** Computational domain. The 2D model was intended to resemble the left ventricle of human heart. The *red lines* represent the solid surfaces, and the rest of the domain boundaries are the inlet (near the domain center) and outlet (*upper left*). (Color figure online)

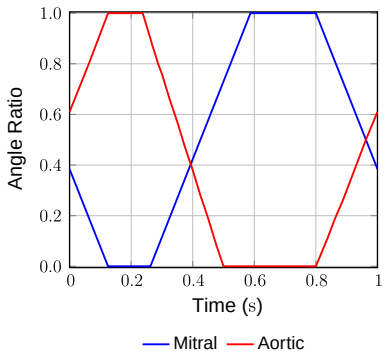


**Fig. 30** Opening/closing stages of the mitral and aortic pairs over the period 1.0 s, prescribed in terms of the angle ratio. The ratios 0.0 and 1.0 represent the closed and fully open stages

carry out the computational analysis with the convective form of the ST-VMS method and the ST-TC described in Sect. 7. In representation of the deforming parts of the domain, we use quadratic NURBS spatially, and cubic NURBS temporally. The mesh is handled with the methods described in Sect. 5, including the STNMUM. There is no remeshing in the customary sense of the word. Figure 30 shows the opening/closing stages of the mitral and aortic pairs over the period 1.0 s, prescribed in terms of the angle ratio for each pair. More information on the computational parameters and conditions can be found in [138].

Figures 31 and 32 show, from the preliminary computation reported in [138], the velocity magnitude at different instants. In those figures, the color range from blue to red corresponds to a velocity range from low to high. Figure 33 shows the velocity vectors and pressure around the mitral pair when the pair is closed. The results have all the good features expected from computations with moving-mesh methods, such as pressure jump with zero thickness and boundary-

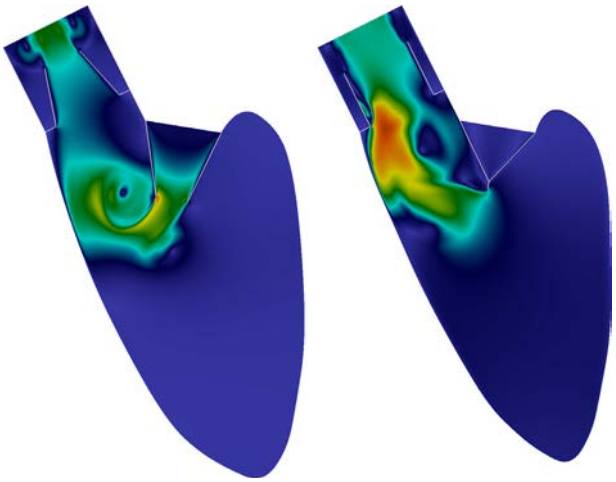

**Fig. 31** Velocity at $t = 0.002$ s and $t = 0.126$ s

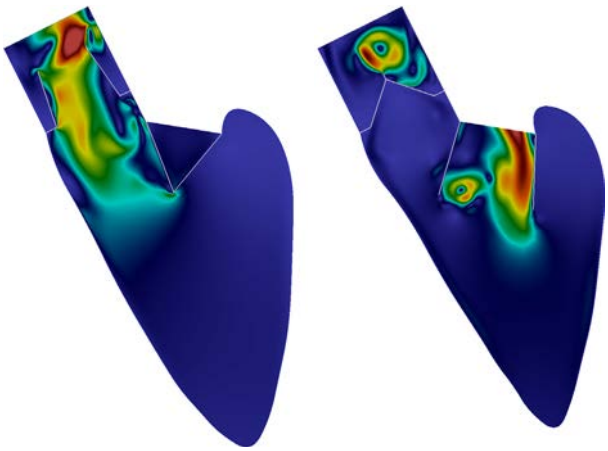

**Fig. 32** Velocity at $t = 0.290$ s and $t = 0.566$ s

layer representation. As pointed out in [138], taking this computation beyond what was reported requires a different way of dealing with the flow at the inlet when we use stress conditions there. This was not attempted in this 2D test computation, because the objective was just to show that the ST-TC method could successfully deal with the TC challenges of this class of problems.

## 11 Concluding remarks

We have presented an overview of the new directions we have taken the ST methods in bringing solution and analysis to different classes of computationally challenging engineering problems. Moving in these new directions was motivated mostly by the following three classes of problems we targeted: bio-inspired flapping-wing aerodynamics, wind-turbine aerodynamics, and cardiovascular fluid mechanics. The new directions for the ST methods include (a) the VMS version of the DSD/SST method, which is called ST-VMS,

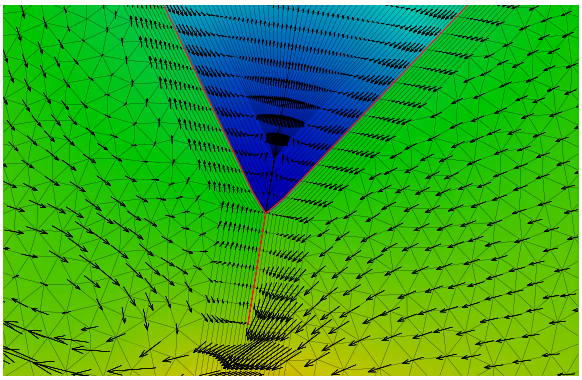

**Fig. 33** Two opening/closing pairs. Velocity vectors and pressure (*colored*) around the mitral pair when the pair is closed ($t = 0.126$ s). The red lines indicate the zero-thickness surfaces

(b) ST methods based on using NURBS basis functions in temporal representation of the unknown variables and motion of the solid surfaces and fluid meshes, including the mesh update method STNMUM, c) ST techniques with continuous representation in time, which is called ST-C, and d) ST interface-tracking with topology change, which is called ST-TC. We described the new directions, and demonstrated their power by presenting examples of the challenging problems solved from the three classes of problems targeted.

**Acknowledgments** This work was supported in part by Rice–Waseda Research Agreement.

## References

1. Hughes TJR, Liu WK, Zimmermann TK (1981) Lagrangian–Eulerian finite element formulation for incompressible viscous flows. Comput Methods Appl Mech Eng 29:329–349
2. Ohayon R (2001) Reduced symmetric models for modal analysis of internal structural-acoustic and hydroelastic-sloshing systems. Comput Methods Appl Mech Eng 190:3009–3019
3. van Brummelen EH, de Borst R (2005) On the nonnormality of subiteration for a fluid-structure interaction problem. SIAM J Sci Comput 27:599–621
4. Bazilevs Y, Calo VM, Zhang Y, Hughes TJR (2006) Isogeometric fluid-structure interaction analysis with applications to arterial blood flow. Comput Mech 38:310–322
5. Khurram RA, Masud A (2006) A multiscale/stabilized formulation of the incompressible Navier–Stokes equations for moving boundary flows and fluid-structure interaction. Comput Mech 38:403–416
6. Lohner R, Cebral JR, Yang C, Baum JD, Mestreau EL, Soto O (2006) "Extending the range of applicability of the loose coupling approach for FSI simulations", in H.-J. Bungartz and M. Schafer, editors, Fluid-structure interaction, volume 53 of Lecture notes in computational science and engineering, 82–100, Springer, Dordrecht
7. Bletzinger K-U, Wuchner R, Kupzok A (2006) "Algorithmic treatment of shells and free form-membranes in FSI", in H.-J. Bungartz and M. Schafer, editors, Fluid-structure interaction, volume 53 of Lecture notes in computational science and engineering, 336–355, Springer, Dordrecht

8. Bazilevs Y, Calo VM, Hughes TJR, Zhang Y (2008) Isogeometric fluid-structure interaction: theory, algorithms, and computations. Comput Mech 43:3–37
9. Dettmer WG, Peric D (2008) On the coupling between fluid flow and mesh motion in the modelling of fluid-structure interaction. Comput Mech 43:81–90
10. Bazilevs Y, Gohean JR, Hughes TJR, Moser RD, Zhang Y (2009) Patient-specific isogeometric fluid-structure interaction analysis of thoracic aortic blood flow due to implantation of the Jarvik (2000) left ventricular assist device. Comput Methods Appl Mech Eng 198:3534–3550
11. Bazilevs Y, Hsu M-C, Benson D, Sankaran S, Marsden A (2009) Computational fluid-structure interaction: methods and application to a total cavopulmonary connection. Comput Mech 45:77–89
12. Calderer R, Masud A (2010) A multiscale stabilized ALE formulation for incompressible flows with moving boundaries. Comput Mech 46:185–197
13. Bazilevs Y, Hsu M-C, Zhang Y, Wang W, Liang X, Kvamsdal T, Brekken R, Isaksen J (2010) A fully-coupled fluid-structure interaction simulation of cerebral aneurysms. Comput Mech 46:3–16
14. Bazilevs Y, Hsu M-C, Zhang Y, Wang W, Kvamsdal T, Hentschel S, Isaksen J (2010) Computational fluid-structure interaction: methods and application to cerebral aneurysms. Biomech Model Mechanobiol 9:481–498
15. Bazilevs Y, Hsu M-C, Akkerman I, Wright S, Takizawa K, Henicke B, Spielman T, Tezduyar TE (2011) 3D simulation of wind turbine rotors at full scale. Part I: geometry modeling and aerodynamics. Int J Numer Methods Fluids 65:207–235. doi:10.1002/fld.2400
16. Bazilevs Y, Hsu M-C, Kiendl J, Wüchner R, Bletzinger K-U (2011) 3D simulation of wind turbine rotors at full scale. Part II: fluid-structure interaction modeling with composite blades. Int J Numer Methods Fluids 65:236–253
17. Akkerman I, Bazilevs Y, Kees CE, Farthing MW (2011) Isogeometric analysis of free-surface flow. J Comput Phys 230:4137–4152
18. Hsu M-C, Bazilevs Y (2011) Blood vessel tissue prestress modeling for vascular fluid-structure interaction simulations. Finite Elem Anal Des 47:593–599
19. Nagaoka S, Nakabayashi Y, Yagawa G, Kim YJ (2011) Accurate fluid-structure interaction computations using elements without mid-side nodes. Comput Mech 48:269–276. doi:10.1007/s00466-011-0620-7
20. Bazilevs Y, Hsu M-C, Takizawa K, Tezduyar TE (2012) ALE-VMS and ST-VMS methods for computer modeling of wind-turbine rotor aerodynamics and fluid-structure interaction. Math Models Methods Appl Sci 22:1230002. doi:10.1142/S0218202512300025
21. Akkerman I, Bazilevs Y, Benson DJ, Farthing MW, Kees CE (2012) Free-surface flow and fluid-object interaction modeling with emphasis on ship hydrodynamics. J Appl Mech 79:010905
22. Hsu M-C, Akkerman I, Bazilevs Y (2012) Wind turbine aerodynamics using ALE-VMS: validation and role of weakly enforced boundary conditions. Comput Mech 50:499–511
23. Hsu M-C, Bazilevs Y (2012) Fluid-structure interaction modeling of wind turbines: simulating the full machine. Comput Mech 50:821–833
24. Akkerman I, Dunaway J, Kvandal J, Spinks J, Bazilevs Y (2012) Toward free-surface modeling of planing vessels: simulation of the Fridsma hull using ALE-VMS. Comput Mech 50:719–727
25. Minami S, Kawai H, Yoshimura S (2012) Parallel BDD-based monolithic approach for acoustic fluid-structure interaction. Comput Mech 50:707–718
26. Miras T, Schotte J-S, Ohayon R (2012) Energy approach for static and linearized dynamic studies of elastic structures containing incompressible liquids with capillarity: a theoretical formulation. Comput Mech 50:729–741
27. van Opstal TM, van Brummelen EH, de Borst R, Lewis MR (2012) A finite-element/boundary-element method for large-displacement fluid-structure interaction. Comput Mech 50:779–788
28. Yao JY, Liu GR, Narmoneva DA, Hinton RB, Zhang Z-Q (2012) Immersed smoothed finite element method for fluid-structure interaction simulation of aortic valves. Comput Mech 50:789–804
29. Larese A, Rossi R, Onate E, Idelsohn SR (2012) A coupled PFEM–Eulerian approach for the solution of porous FSI problems. Comput Mech 50:805–819
30. Bazilevs Y, Takizawa K, Tezduyar TE (2013) Computational fluid-structure interaction: methods and applications. Wiley, New York
31. Bazilevs Y, Takizawa K, Tezduyar TE (2013) Challenges and directions in computational fluid-structure interaction. Math Models Methods Appl Sci 23:215–221. doi:10.1142/S0218202513400010
32. Korobenko A, Hsu M-C, Akkerman I, Tippmann J, Bazilevs Y (2013) Structural mechanics modeling and FSI simulation of wind turbines. Math Models Methods Appl Sci 23:249–272
33. Yao JY, Liu GR, Qian D, Chen CL, Xu GX (2013) A moving-mesh gradient smoothing method for compressible CFD problems. Math Models Methods Appl Sci 23:273–305
34. Kamran K, Rossi R, Onate E, Idelsohn SR (2013) A compressible Lagrangian framework for modeling the fluid-structure interaction in the underwater implosion of an aluminum cylinder. Math Models Methods Appl Sci 23:339–367
35. Hsu M-C, Akkerman I, Bazilevs Y (2013) "Finite element simulation of wind turbine aerodynamics: Validation study using NREL Phase VI experiment", Wind Energy, published online, doi:10.1002/we.1599
36. Tezduyar TE (1992) Stabilized finite element formulations for incompressible flow computations. Adv Appl Mech 28:1–44. doi:10.1016/S0065-2156(08)70153-4
37. Tezduyar TE, Behr M, Liou J (1992) A new strategy for finite element computations involving moving boundaries and interfaces—the deforming-spatial-domain/space-time procedure: I. The concept and the preliminary numerical tests. Comput Methods Appl Mech Eng 94:339–351. doi:10.1016/0045-7825(92)90059-S
38. Tezduyar TE, Behr M, Mittal S, Liou J (1992) A new strategy for finite element computations involving moving boundaries and interfaces—the deforming-spatial-domain/space-time procedure: II. Computation of free-surface flows, two-liquid flows, and flows with drifting cylinders. Comput Methods Appl Mech Eng 94:353–371. doi:10.1016/0045-7825(92)90060-W
39. Tezduyar TE (2003) Computation of moving boundaries and interfaces and stabilization parameters. Int J Numer Methods Fluids 43:555–575. doi:10.1002/fld.505
40. Tezduyar TE, Sathe S (2007) Modeling of fluid-structure interactions with the space-time finite elements: solution techniques. Int J Numer Methods in Fluids 54:855–900. doi:10.1002/fld.1430
41. Takizawa K, Tezduyar TE (2011) Multiscale space-time fluid-structure interaction techniques. Comput Mech 48:247–267. doi:10.1007/s00466-011-0571-z
42. Takizawa K, Tezduyar TE (2012) Space-time fluid-structure interaction methods. Math Models Methods Appl Sci 22:1230001. doi:10.1142/S0218202512300013
43. Tezduyar TE, Sathe S, Pausewang J, Schwaab M, Christopher J, Crabtree J (2008) Interface projection techniques for fluid-structure interaction modeling with moving-mesh methods. Comput Mech 43:39–49. doi:10.1007/s00466-008-0261-7

44. Tezduyar T, Aliabadi S, Behr M, Johnson A, Mittal S (1993) Parallel finite-element computation of 3D flows. Computer 26:27–36. doi:10.1109/2.237441
45. Johnson AA, Tezduyar TE (1994) Mesh update strategies in parallel finite element computations of flow problems with moving boundaries and interfaces. Comput Methods Appl Mech Eng 119:73–94. doi:10.1016/0045-7825(94)00077-8
46. Tezduyar TE (2001) Finite element methods for flow problems with moving boundaries and interfaces. Arch Comput Methods Eng 8:83–130. doi:10.1007/BF02897870
47. Stein K, Tezduyar T, Benney R (2003) Mesh moving techniques for fluid-structure interactions with large displacements. J Appl Mech 70:58–63. doi:10.1115/1.1530635
48. Stein K, Tezduyar TE, Benney R (2004) Automatic mesh update with the solid-extension mesh moving technique. Comput Methods Appl Mech Eng 193:2019–2032. doi:10.1016/j.cma.2003.12.046
49. Tezduyar T, Aliabadi S, Behr M, Johnson A, Kalro V, Litke M (1996) Flow simulation and high performance computing. Comput Mech 18:397–412. doi:10.1007/BF00350249
50. Tezduyar TE, Sathe S, Schwaab M, Pausewang J, Christopher J, Crabtree J (2008) Fluid-structure interaction modeling of ringsail parachutes. Comput Mech 43:133–142. doi:10.1007/s00466-008-0260-8
51. Tezduyar TE, Takizawa K, Moorman C, Wright S, Christopher J (2010) Space-time finite element computation of complex fluid-structure interactions. Int J Numer Methods Fluids 64:1201–1218. doi:10.1002/fld.2221
52. Takizawa K, Spielman T, Tezduyar TE (2011) Space-time FSI modeling and dynamical analysis of spacecraft parachutes and parachute clusters. Comput Mech 48:345–364. doi:10.1007/s00466-011-0590-9
53. Takizawa K, Tezduyar TE (2012) Computational methods for parachute fluid-structure interactions. Arch Comput Methods Eng 19:125–169. doi:10.1007/s11831-012-9070-4
54. Takizawa K, Fritze M, Montes D, Spielman T, Tezduyar TE (2012) Fluid-structure interaction modeling of ringsail parachutes with disreefing and modified geometric porosity. Comput Mech 50:835–854. doi:10.1007/s00466-012-0761-3
55. Takizawa K, Montes D, Fritze M, McIntyre S, Boben J, Tezduyar TE (2013) Methods for FSI modeling of spacecraft parachute dynamics and cover separation. Math Models Methods Appl Sci 23:307–338. doi:10.1142/S0218202513400058
56. Takizawa K, Tezduyar TE, Boben J, Kostov N, Boswell C, Buscher A (2013) Fluid-structure interaction modeling of clusters of spacecraft parachutes with modified geometric porosity. Comput Mech 52:1351–1364. doi:10.1007/s00466-013-0880-5
57. Takizawa K, Henicke B, Puntel A, Kostov N, Tezduyar TE (2012) Space-time techniques for computational aerodynamics modeling of flapping wings of an actual locust. Comput Mech 50:743–760. doi:10.1007/s00466-012-0759-x
58. Takizawa K, Kostov N, Puntel A, Henicke B, Tezduyar TE (2012) Space-time computational analysis of bio-inspired flapping-wing aerodynamics of a micro aerial vehicle. Comput Mech 50:761–778. doi:10.1007/s00466-012-0758-y
59. Takizawa K, Tezduyar TE, Kostov N (2014) "Sequentially-coupled space-time FSI analysis of bio-inspired flapping-wing aerodynamics of an MAV", Comput Mech, to appear, doi:10.1007/s00466-014-0980-x
60. Takizawa K, Tezduyar TE, McIntyre S, Kostov N, Kolesar R, Habluetzel C (2014) Space-time VMS computation of wind-turbine rotor and tower aerodynamics. Comput Mech 53:1–15. doi:10.1007/s00466-013-0888-x
61. Akin JE, Tezduyar TE, Ungor M (2007) Computation of flow problems with the mixed interface-tracking/interface-capturing technique (MITICT). Comput & Fluids 36:2–11. doi:10.1016/j.compfluid.2005.07.008
62. Cruchaga MA, Celentano DJ, Tezduyar TE (2007) A numerical model based on the Mixed Interface-Tracking/Interface-Capturing Technique (MITICT) for flows with fluid-solid and fluid-fluid interfaces. Int J Numer Methods Fluids 54:1021–1030. doi:10.1002/fld.1498
63. Wick T (2013) Coupling of fully Eulerian and arbitrary Lagrangian–Eulerian methods for fluid-structure interaction computations. Comput Mech 52:1113–1124
64. Mittal S, Tezduyar TE (1992) Notes on the stabilized space-time finite element formulation of unsteady incompressible flows. Comput Phys Commun 73:93–112. doi:10.1016/0010-4655(92)90031-S
65. Tezduyar TE, Aliabadi SK, Behr M, Mittal S (1994) Massively parallel finite element simulation of compressible and incompressible flows. Comput Methods Appl Mech Eng 119:157–177. doi:10.1016/0045-7825(94)00082-4
66. Wren GP, Ray SE, Aliabadi SK, Tezduyar TE (1997) Simulation of flow problems with moving mechanical components, fluid-structure interactions and two-fluid interfaces. Int J Numer Methods Fluids 24:1433–1448. doi:10.1002/(SICI)1097-0363(199706)24:12<1433:AID-FLD568>3.3.CO;2-L
67. Tezduyar TE (1999) CFD methods for three-dimensional computation of complex flow problems. J Wind Eng Ind Aerodyn 81:97–116. doi:10.1016/S0167-6105(99)00011-2
68. Guler I, Behr M, Tezduyar T (1999) Parallel finite element computation of free-surface flows. Comput Mech 23:117–123. doi:10.1007/s004660050391
69. Tezduyar TE (2006) Interface-tracking and interface-capturing techniques for finite element computation of moving boundaries and interfaces. Comput Methods Appl Mech Eng 195:2983–3000. doi:10.1016/j.cma.2004.09.018
70. Mittal S, Tezduyar TE (1992) A finite element study of incompressible flows past oscillating cylinders and aerofoils. Int J Numer Methods Fluids 15:1073–1118. doi:10.1002/fld.1650150911
71. Mittal S, Tezduyar TE (1994) Massively parallel finite element computation of incompressible flows involving fluid-body interactions. Comput Methods Appl Mech Eng 112:253–282. doi:10.1016/0045-7825(94)90029-9
72. Aliabadi SK, Tezduyar TE (1995) Parallel fluid dynamics computations in aerospace applications. Int J Numer Methods Fluids 21:783–805. doi:10.1002/fld.1650211003
73. Kalro V, Aliabadi S, Garrard W, Tezduyar T, Mittal S, Stein K (1997) Parallel finite element simulation of large ram-air parachutes. Int J Numer Methods Fluids 24:1353–1369. doi:10.1002/(SICI)1097-0363(199706)24:12<1353:AID-FLD564>3.0.CO;2-6
74. Tezduyar T, Kalro V, Garrard W (1997) Parallel computational methods for 3D simulation of a parafoil with prescribed shape changes. Parallel Comput 23:1349–1363. doi:10.1016/S0167-8191(97)00057-4
75. Tezduyar T, Osawa Y (1999) Methods for parallel computation of complex flow problems. Parallel Comput 25:2039–2066. doi:10.1016/S0167-8191(99)00080-0
76. Tezduyar T, Osawa Y (2001) The Multi-Domain Method for computation of the aerodynamics of a parachute crossing the far wake of an aircraft. Comput Methods Appl Mech Eng 191:705–716. doi:10.1016/S0045-7825(01)00310-3
77. Ray SE, Tezduyar TE (2000) Fluid-object interactions in interior ballistics. Comput Methods Appl Mech Eng 190:363–372. doi:10.1016/S0045-7825(00)00207-3

78. Mittal S, Tezduyar TE (1995) Parallel finite element simulation of 3D incompressible flows—Fluid-structure interactions. Int J Numer Methods Fluids 21:933–953. doi:10.1002/fld.1650211011
79. Takizawa K, Henicke B, Puntel A, Spielman T, Tezduyar TE (2012) Space-time computational techniques for the aerodynamics of flapping wings. J Appl Mech 79:010903. doi:10.1115/1.4005073
80. Takizawa K, Henicke B, Puntel A, Kostov N, Tezduyar TE (2013) Computer modeling techniques for flapping-wing aerodynamics of a locust. Comput & Fluids 85:125–134. doi:10.1016/j.compfluid.2012.11.008
81. Behr M, Tezduyar T (1999) The Shear-Slip Mesh Update Method. Comput Methods Appl Mech Eng 174:261–274. doi:10.1016/S0045-7825(98)00299-0
82. Behr M, Tezduyar T (2001) Shear-slip mesh update in 3D computation of complex flow problems with rotating mechanical components. Comput Methods Appl Mech Eng 190:3189–3200. doi:10.1016/S0045-7825(00)00388-1
83. Takizawa K, Henicke B, Tezduyar TE, Hsu M-C, Bazilevs Y (2011) Stabilized space-time computation of wind-turbine rotor aerodynamics. Comput Mech 48:333–344. doi:10.1007/s00466-011-0589-2
84. Aliabadi SK, Tezduyar TE (1993) Space-time finite element computation of compressible flows involving moving boundaries and interfaces. Comput Methods Appl Mech Eng 107:209–223. doi:10.1016/0045-7825(93)90176-X
85. Wren GP, Ray SE, Aliabadi SK, Tezduyar TE (1995) Space-time finite element computation of compressible flows between moving components. Int J Numer Methods Fluids 21:981–991. doi:10.1002/fld.1650211015
86. Ray SE, Wren GP, Tezduyar TE (1997) Parallel implementations of a finite element formulation for fluid-structure interactions in interior flows. Parallel Comput 23:1279–1292. doi:10.1016/S0167-8191(97)00053-7
87. Takase S, Kashiyama K, Tanaka S, Tezduyar TE (2010) Space-time SUPG formulation of the shallow-water equations. Int J Numer Methods Fluids 64:1379–1394. doi:10.1002/fld.2464
88. Takase S, Kashiyama K, Tanaka S, Tezduyar TE (2011) Space-time SUPG finite element computation of shallow-water flows with moving shorelines. Comput Mech 48:293–306. doi:10.1007/s00466-011-0618-1
89. Johnson AA, Tezduyar TE (1996) Simulation of multiple spheres falling in a liquid-filled tube. Comput Methods Appl Mech Eng 134:351–373. doi:10.1016/0045-7825(95)00988-4
90. Johnson AA, Tezduyar TE (1997) Parallel computation of incompressible flows with complex geometries. Int J Numer Methods Fluids 24:1321–1340. doi:10.1002/(SICI)1097-0363(199706)24:12<1321::AID-FLD562>3.3.CO;2-C
91. Johnson AA, Tezduyar TE (1997) 3D simulation of fluid-particle interactions with the number of particles reaching 100. Comput Methods Appl Mech Eng 145:301–321. doi:10.1016/S0045-7825(96)01223-6
92. Johnson AA, Tezduyar TE (1999) Advanced mesh generation and update methods for 3D flow simulations. Comput Mech 23:130–143. doi:10.1007/s004660050393
93. Johnson A, Tezduyar T (2001) Methods for 3D computation of fluid-object interactions in spatially-periodic flows. Comput Methods Appl Mech Eng 190:3201–3221. doi:10.1016/S0045-7825(00)00389-3
94. Kalro V, Tezduyar TE (2000) A parallel 3D computational method for fluid-structure interactions in parachute systems. Comput Methods Appl Mech Eng 190:321–332. doi:10.1016/S0045-7825(00)00204-8
95. Stein K, Benney R, Kalro V, Tezduyar TE, Leonard J, Accorsi M (2000) Parachute fluid-structure interactions: 3-D Computation. Comput Methods Appl Mech Eng 190:373–386. doi:10.1016/S0045-7825(00)00208-5
96. Tezduyar T, Osawa Y (2001) Fluid-structure interactions of a parachute crossing the far wake of an aircraft. Comput Methods Appl Mech Eng 191:717–726. doi:10.1016/S0045-7825(01)00311-5
97. Stein K, Benney R, Tezduyar T, Potvin J (2001) Fluid-structure interactions of a cross parachute: numerical simulation. Comput Methods Appl Mech Eng 191:673–687. doi:10.1016/S0045-7825(01)00312-7
98. Stein KR, Benney RJ, Tezduyar TE, Leonard JW, Accorsi ML (2001) Fluid-structure interactions of a round parachute: modeling and simulation techniques. J Aircr 38:800–808. doi:10.2514/2.2864
99. Stein K, Tezduyar T, Kumar V, Sathe S, Benney R, Thornburg E, Kyle C, Nonoshita T (2003) Aerodynamic interactions between parachute canopies. J Appl Mech 70:50–57. doi:10.1115/1.1530634
100. Stein K, Tezduyar T, Benney R (2003) Computational methods for modeling parachute systems. Comput Sci Eng 5:39–46. doi:10.1109/MCISE.2003.1166551
101. Tezduyar TE, Sathe S (2004) Enhanced-discretization space-time technique (EDSTT). Comput Methods Appl Mech Eng 193:1385–1401. doi:10.1016/j.cma.2003.12.029
102. Tezduyar TE, Sathe S, Keedy R, Stein K (2006) Space-time finite element techniques for computation of fluid-structure interactions. Comput Methods Appl Mech Eng 195:2002–2027. doi:10.1016/j.cma.2004.09.014
103. Tezduyar TE, Sathe S, Stein K (2006) Solution techniques for the fully-discretized equations in computation of fluid-structure interactions with the space-time formulations. Comput Methods Appl Mech Eng 195:5743–5753. doi:10.1016/j.cma.2005.08.023
104. Torii R, Oshima M, Kobayashi T, Takagi K, Tezduyar TE (2006) Computer modeling of cardiovascular fluid-structure interactions with the Deforming-Spatial-Domain/Stabilized Space-Time formulation. Comput Methods Appl Mech Eng 195:1885–1895. doi:10.1016/j.cma.2005.05.050
105. Torii R, Oshima M, Kobayashi T, Takagi K, Tezduyar TE (2006) Fluid-structure interaction modeling of aneurysmal conditions with high and normal blood pressures. Comput Mech 38:482–490. doi:10.1007/s00466-006-0065-6
106. Tezduyar TE (2007) Finite elements in fluids: stabilized formulations and moving boundaries and interfaces. Comput & Fluids 36:191–206. doi:10.1016/j.compfluid.2005.02.011
107. Tezduyar TE (2007) Finite elements in fluids: special methods and enhanced solution techniques. Comput & Fluids 36:207–223. doi:10.1016/j.compfluid.2005.02.010
108. Tezduyar TE, Sameh A (2006) Parallel finite element computations in fluid mechanics. Comput Methods Appl Mech Eng 195:1872–1884. doi:10.1016/j.cma.2005.05.038
109. Stein K, Tezduyar TE, Sathe S, Benney R, Charles R (2005) Fluid-structure interaction modeling of parachute soft-landing dynamics. Int J Numer Methods Fluids 47:619–631. doi:10.1002/fld.835
110. Tezduyar TE, Sathe S, Cragin T, Nanna B, Conklin BS, Pausewang J, Schwaab M (2007) Modeling of fluid-structure interactions with the space-time finite elements: arterial fluid mechanics. Int J Numer Methods Fluids 54:901–922. doi:10.1002/fld.1443
111. Torii R, Oshima M, Kobayashi T, Takagi K, Tezduyar TE (2007) Influence of wall elasticity in patient-specific hemodynamic simulations. Comput & Fluids 36:160–168. doi:10.1016/j.compfluid.2005.07.014
112. Sathe S, Benney R, Charles R, Doucette E, Miletti J, Senga M, Stein K, Tezduyar TE (2007) Fluid-structure interaction modeling of complex parachute designs with the space-time finite element techniques. Comput & Fluids 36:127–135. doi:10.1016/j.compfluid.2005.07.010

113. Tezduyar TE, Sathe S, Schwaab M, Conklin BS (2008) Arterial fluid mechanics modeling with the stabilized space-time fluid-structure interaction technique. Int J Numer Methods Fluids 57:601–629. doi:10.1002/fld.1633
114. Torii R, Oshima M, Kobayashi T, Takagi K, Tezduyar TE (2007) Numerical investigation of the effect of hypertensive blood pressure on cerebral aneurysm—Dependence of the effect on the aneurysm shape. Int J Numer Methods Fluids 54:995–1009. doi:10.1002/fld.1497
115. Manguoglu M, Sameh AH, Tezduyar TE, Sathe S (2008) A nested iterative scheme for computation of incompressible flows in long domains. Comput Mech 43:73–80. doi:10.1007/s00466-008-0276-0
116. Sathe S, Tezduyar TE (2008) Modeling of fluid-structure interactions with the space-time finite elements: contact problems. Comput Mech 43:51–60. doi:10.1007/s00466-008-0299-6
117. Torii R, Oshima M, Kobayashi T, Takagi K, Tezduyar TE (2008) Fluid-structure interaction modeling of a patient-specific cerebral aneurysm: influence of structural modeling. Comput Mech 43:151–159. doi:10.1007/s00466-008-0325-8
118. Tezduyar TE, Schwaab M, Sathe S (2009) Sequentially-coupled arterial fluid-structure interaction (SCAFSI) technique. Comput Methods Appl Mech Eng 198:3524–3533. doi:10.1016/j.cma.2008.05.024
119. Torii R, Oshima M, Kobayashi T, Takagi K, Tezduyar TE (2009) Fluid-structure interaction modeling of blood flow and cerebral aneurysm: significance of artery and aneurysm shapes. Comput Methods Appl Mech Eng 198:3613–3621. doi:10.1016/j.cma.2008.08.020
120. Manguoglu M, Sameh AH, Saied F, Tezduyar TE, Sathe S (2009) Preconditioning techniques for nonsymmetric linear systems in the computation of incompressible flows. J Appl Mech 76:021204. doi:10.1115/1.3059576
121. Takizawa K, Christopher J, Tezduyar TE, Sathe S (2010) Space-time finite element computation of arterial fluid-structure interactions with patient-specific data. Int J Numer Methods Biomed Eng 26:101–116. doi:10.1002/cnm.1241
122. Tezduyar TE, Takizawa K, Moorman C, Wright S, Christopher J (2010) Multiscale sequentially-coupled arterial FSI technique. Comput Mech 46:17–29. doi:10.1007/s00466-009-0423-2
123. Takizawa K, Moorman C, Wright S, Christopher J, Tezduyar TE (2010) Wall shear stress calculations in space-time finite element computation of arterial fluid-structure interactions. Comput Mech 46:31–41. doi:10.1007/s00466-009-0425-0
124. Torii R, Oshima M, Kobayashi T, Takagi K, Tezduyar TE (2010) Influence of wall thickness on fluid-structure interaction computations of cerebral aneurysms. Int J Numer Methods Biomed Eng 26:336–347. doi:10.1002/cnm.1289
125. Manguoglu M, Takizawa K, Sameh AH, Tezduyar TE (2010) Solution of linear systems in arterial fluid mechanics computations with boundary layer mesh refinement. Comput Mech 46:83–89. doi:10.1007/s00466-009-0426-z
126. Torii R, Oshima M, Kobayashi T, Takagi K, Tezduyar TE (2010) Role of 0D peripheral vasculature model in fluid-structure interaction modeling of aneurysms. Comput Mech 46:43–52. doi:10.1007/s00466-009-0439-7
127. Takizawa K, Moorman C, Wright S, Spielman T, Tezduyar TE (2011) Fluid-structure interaction modeling and performance analysis of the Orion spacecraft parachutes. Int J Numer Methods Fluids 65:271–285. doi:10.1002/fld.2348
128. Takizawa K, Moorman C, Wright S, Purdue J, McPhail T, Chen PR, Warren J, Tezduyar TE (2011) Patient-specific arterial fluid-structure interaction modeling of cerebral aneurysms. Int J Numer Methods Fluids 65:308–323. doi:10.1002/fld.2360
129. Takizawa K, Wright S, Moorman C, Tezduyar TE (2011) Fluid-structure interaction modeling of parachute clusters. Int J Numer Methods Fluids 65:286–307. doi:10.1002/fld.2359
130. Manguoglu M, Takizawa K, Sameh AH, Tezduyar TE (2011) Nested and parallel sparse algorithms for arterial fluid mechanics computations with boundary layer mesh refinement. Int J Numer Methods Fluids 65:135–149. doi:10.1002/fld.2415
131. Torii R, Oshima M, Kobayashi T, Takagi K, Tezduyar TE (2011) Influencing factors in image-based fluid-structure interaction computation of cerebral aneurysms. Int J Numer Methods Fluids 65:324–340. doi:10.1002/fld.2448
132. Tezduyar TE, Takizawa K, Brummer T, Chen PR (2011) Space-time fluid-structure interaction modeling of patient-specific cerebral aneurysms. Int J Numer Methods Biomed Eng 27:1665–1710. doi:10.1002/cnm.1433
133. Manguoglu M, Takizawa K, Sameh AH, Tezduyar TE (2011) A parallel sparse algorithm targeting arterial fluid mechanics computations. Comput Mech 48:377–384. doi:10.1007/s00466-011-0619-0
134. Takizawa K, Spielman T, Moorman C, Tezduyar TE (2012) Fluid-structure interaction modeling of spacecraft parachutes for simulation-based design. J Appl Mech 79:010907. doi:10.1115/1.4005070
135. Takizawa K, Brummer T, Tezduyar TE, Chen PR (2012) A comparative study based on patient-specific fluid-structure interaction modeling of cerebral aneurysms. J Appl Mech 79:010908. doi:10.1115/1.4005071
136. Takizawa K, Bazilevs Y, Tezduyar TE (2012) Space-time and ALE-VMS techniques for patient-specific cardiovascular fluid-structure interaction modeling. Arch Comput Methods Eng 19:171–225. doi:10.1007/s11831-012-9071-3
137. Takizawa K, Tezduyar TE (2012) Bringing them down safely. Mech Eng 134:34–37
138. Takizawa K, Tezduyar TE, Buscher A, Asada S (2013) "Space-time interface-tracking with topology change (ST-TC)", Comput Mech, October 2013, doi:10.1007/s00466-013-0935-7
139. Brooks AN, Hughes TJR (1982) Streamline upwind/Petrov–Galerkin formulations for convection dominated flows with particular emphasis on the incompressible Navier-Stokes equations. Comput Methods Appl Mech Eng 32:199–259
140. Tezduyar TE, Mittal S, Ray SE, Shih R (1992) Incompressible flow computations with stabilized bilinear and linear equal-order-interpolation velocity-pressure elements. Comput Methods Appl Mech Eng 95:221–242. doi:10.1016/0045-7825(92)90141-6
141. Hughes TJR (1995) Multiscale phenomena: green's functions, the Dirichlet-to-Neumann formulation, subgrid scale models, bubbles, and the origins of stabilized methods. Comput Methods Appl Mech Eng 127:387–401
142. Hughes TJR, Oberai AA, Mazzei L (2001) Large eddy simulation of turbulent channel flows by the variational multiscale method. Phys Fluids 13:1784–1799
143. Bazilevs Y, Calo VM, Cottrell JA, Hughes TJR, Reali A, Scovazzi G (2007) Variational multiscale residual-based turbulence modeling for large eddy simulation of incompressible flows. Comput Methods Appl Mech Eng 197:173–201
144. Bazilevs Y, Akkerman I (2010) Large eddy simulation of turbulent Taylor–Couette flow using isogeometric analysis and the residual-based variational multiscale method. J Comput Phys 229:3402–3414
145. Takizawa K, Henicke B, Montes D, Tezduyar TE, Hsu M-C, Bazilevs Y (2011) Numerical-performance studies for the stabilized space-time computation of wind-turbine rotor aerodynamics. Comput Mech 48:647–657. doi:10.1007/s00466-011-0614-5

146. Hughes TJR, Cottrell JA, Bazilevs Y (2005) Isogeometric analysis: CAD, finite elements, NURBS, exact geometry, and mesh refinement. Comput Methods Appl Mech Eng 194: 4135–4195
147. Bazilevs Y, Hughes TJR (2008) NURBS-based isogeometric analysis for the computation of flows about rotating components. Comput Mech 43:143–150
148. Takizawa K, Tezduyar TE (2014) Space-time computation techniques with continuous representation in time (ST-C). Comput Mech 53:91–99. doi:10.1007/s00466-013-0895-y
149. Hughes TJR, Tezduyar TE (1984) Finite element methods for first-order hyperbolic systems with particular emphasis on the compressible Euler equations. Comput Methods Appl Mech Eng 45:217–284. doi:10.1016/0045-7825(84)90157-9
150. Tezduyar TE, Park YJ (1986) Discontinuity capturing finite element formulations for nonlinear convection-diffusion-reaction equations. Comput Methods Appl Mech Eng 59:307–325. doi:10.1016/0045-7825(86)90003-4
151. Hughes TJR, Franca LP, Balestra M (1986) A new finite element formulation for computational fluid dynamics: V. Circumventing the Babuška-Brezzi condition: A stable Petrov-Galerkin formulation of the Stokes problem accommodating equal-order interpolations. Comput Methods Appl Mech Eng 59:85–99
152. Tezduyar TE, Osawa Y (2000) Finite element stabilization parameters computed from element matrices and vectors. Comput Methods Appl Mech Eng 190:411–430. doi:10.1016/S0045-7825(00)00211-5
153. Tezduyar TE, Ganjoo DK (1986) Petrov–Galerkin formulations with weighting functions dependent upon spatial and temporal discretization: applications to transient convection-diffusion problems. Comput Methods Appl Mech Eng 59:49–71. doi:10.1016/0045-7825(86)90023-X
154. Le Beau GJ, Ray SE, Aliabadi SK, Tezduyar TE (1993) SUPG finite element computation of compressible flows with the entropy and conservation variables formulations. Comput Methods Appl Mech Eng 104:397–422. doi:10.1016/0045-7825(93)90033-T
155. Akin JE, Tezduyar T, Ungor M, Mittal S (2003) Stabilization parameters and Smagorinsky turbulence model. J Appl Mech 70:2–9. doi:10.1115/1.1526569
156. Akin JE, Tezduyar TE (2004) Calculation of the advective limit of the SUPG stabilization parameter for linear and higher-order elements. Comput Methods Appl Mech Eng 193:1909–1922. doi:10.1016/j.cma.2003.12.050
157. Tezduyar TE, Senga M (2006) Stabilization and shock-capturing parameters in SUPG formulation of compressible flows. Comput Methods Appl Mech Eng 195:1621–1632. doi:10.1016/j.cma.2005.05.032
158. Tezduyar TE, Senga M (2007) SUPG finite element computation of inviscid supersonic flows with YZ$\beta$ shock-capturing. Comput & Fluids 36:147–159. doi:10.1016/j.compfluid.2005.07.009
159. Tezduyar TE, Senga M, Vicker D (2006) Computation of inviscid supersonic flows around cylinders and spheres with the SUPG formulation and YZ$\beta$ shock-capturing. Comput Mech 38:469–481. doi:10.1007/s00466-005-0025-6
160. Tezduyar TE, Sathe S (2006) Enhanced-discretization selective stabilization procedure (EDSSP). Comput Mech 38:456–468. doi:10.1007/s00466-006-0056-7
161. Catabriga L, Coutinho ALGA, Tezduyar TE (2005) Compressible flow SUPG parameters computed from element matrices. Commun Numer Methods Eng 21:465–476. doi:10.1002/cnm.759
162. Catabriga L, Coutinho ALGA, Tezduyar TE (2006) Compressible flow SUPG parameters computed from degree-of-freedom submatrices. Comput Mech 38:334–343. doi:10.1007/s00466-006-0033-1
163. Onate E, Valls A, Garcia J (2006) FIC/FEM formulation with matrix stabilizing terms for incompressible flows at low and high Reynolds numbers. Comput Mech 38:440–455
164. Corsini A, Rispoli F, Santoriello A, Tezduyar TE (2006) Improved discontinuity-capturing finite element techniques for reaction effects in turbulence computation. Comput Mech 38:356–364. doi:10.1007/s00466-006-0045-x
165. Rispoli F, Corsini A, Tezduyar TE (2007) Finite element computation of turbulent flows with the discontinuity-capturing directional dissipation (DCDD). Comput & Fluids 36:121–126. doi:10.1016/j.compfluid.2005.07.004
166. Tezduyar TE, Ramakrishnan S, Sathe S (2008) Stabilized formulations for incompressible flows with thermal coupling. Int J Numer Methods Fluids 57:1189–1209. doi:10.1002/fld.1743
167. Rispoli F, Saavedra R, Corsini A, Tezduyar TE (2007) Computation of inviscid compressible flows with the V-SGS stabilization and YZ$\beta$ shock-capturing. Int J Numer Methods Fluids 54:695–706. doi:10.1002/fld.1447
168. Bazilevs Y, Calo VM, Tezduyar TE, Hughes TJR (2007) YZ$\beta$ discontinuity-capturing for advection-dominated processes with application to arterial drug delivery. Int J Numer Methods Fluids 54:593–608. doi:10.1002/fld.1484
169. Corsini A, Menichini C, Rispoli F, Santoriello A, Tezduyar TE (2009) A multiscale finite element formulation with discontinuity capturing for turbulence models with dominant reactionlike terms. J Appl Mech 76:021211. doi:10.1115/1.3062967
170. Rispoli F, Saavedra R, Menichini F, Tezduyar TE (2009) Computation of inviscid supersonic flows around cylinders and spheres with the V-SGS stabilization and YZ$\beta$ shock-capturing. J Appl Mech 76:021209. doi:10.1115/1.3057496
171. Hughes TJR, Scovazzi G, Tezduyar TE (2010) Stabilized methods for compressible flows. J Sci Comput 43:343–368. doi:10.1007/s10915-008-9233-5
172. Corsini A, Iossa C, Rispoli F, Tezduyar TE (2010) A DRD finite element formulation for computing turbulent reacting flows in gas turbine combustors. Comput Mech 46:159–167. doi:10.1007/s00466-009-0441-0
173. Hsu M-C, Bazilevs Y, Calo VM, Tezduyar TE, Hughes TJR (2010) Improving stability of stabilized and multiscale formulations in flow simulations at small time steps. Comput Methods Appl Mech Eng 199:828–840. doi:10.1016/j.cma.2009.06.019
174. Corsini A, Rispoli F, Tezduyar TE (2011) Stabilized finite element computation of NOx emission in aero-engine combustors. Int J Numer Methods Fluids 65:254–270. doi:10.1002/fld.2451
175. Corsini A, Rispoli F, Tezduyar TE (2012) Computer modeling of wave-energy air turbines with the SUPG/PSPG formulation and discontinuity-capturing technique. J Appl Mech 79:010910. doi:10.1115/1.4005060
176. Corsini A, Rispoli F, Sheard AG, Tezduyar TE (2012) Computational analysis of noise reduction devices in axial fans with stabilized finite element formulations. Comput Mech 50:695–705. doi:10.1007/s00466-012-0789-4
177. Kler PA, Dalcin LD, Paz RR, Tezduyar TE (2013) SUPG and discontinuity-capturing methods for coupled fluid mechanics and electrochemical transport problems. Comput Mech 51:171–185. doi:10.1007/s00466-012-0712-z