OpenFOAM® Journal

Volume 4 [Full Papers], Pages 41–61
ISSN: 2753-8168

A Modified Overset Method in OpenFOAM® for Simultaneous Motion and Deformation: A Case Study of a Flexible Flapping Foil

Karim Ahmed [1,*], Ahmed A. Hamada[2,*], Ludovic Chatellier [1], and Mirjam Fürth [2]

[1] Institut PPRIME UPR 3346 CNRS, Université de Poitiers, ISAE-ENSMA, 11 Bd Marie et Pierre Curie, TSA51124, 86073 Poitiers Cédex 9

[2] Department of Ocean Engineering, Texas A&M University, Haynes Engineering Building, 727 Ross St, College Station, Texas 77843

Email address: karim.ahmed@univ-poitiers.fr
Email address: ahamada@tamu.edu

DOI: https://doi.org/10.51560/ofj.v4.96
Results with version(s): OpenFOAM® v2306
Repository: https://github.com/AhmedAHamada/dynamicOversetZoneFvMesh

Abstract. The influence of wing deformation in animal propulsion and motion is an intriguing phenomenon that has contributed to a growing interest in biomimetics in both academia and industry. The study focuses on developing a robust numerical tool for analyzing the motion of biological systems in fluid flows, crucial to understanding fluid-structure interaction (FSI) phenomena. The research introduces a new mesh deformation solver, *dynamicOversetZoneFvMesh*, designed to address limitations in OpenFOAM's conventional Overset method. This solver enables the application of both rigid body motion and deformation at specified patches within certain zones, allowing for more complex motion scenarios. Through meticulous verification, parallelization, validation, and mesh convergence studies, the modified Overset solver's reliability is confirmed. Two cases are examined: a rigid-body flapping foil and a flexible foil with leading-edge deformation. Results demonstrate the adapted solver's proficiency in managing complex fluid dynamics and accommodating simultaneous motion and deformation. Furthermore, the study reveals that leading-edge flexibility improves the power extraction efficiency of flapping foils, aligning with previous literature. Overall, the research lays the foundation for advancing FSI simulations and opens avenues for addressing previously challenging problems in the field.

1. Introduction

The examination of motion-generating mechanisms found in biological systems has emerged as a prominent field of study within biomimetics. In response to the increasing demand for innovative and efficient technologies from the aviation and maritime industries, numerous researchers have dedicated their efforts to replicating these bio-inspired systems. Extensive research has been carried out in the realms of bio-inspired aerodynamics and hydrodynamics [1,2]. Furthermore, these emerging concepts have been applied to various domains, including energy harvesting [3,4], propulsion [5], transportation [6], construction and architecture [7], marine applications [8], and aviation advancements [9,10].

In the realm of animal locomotion, the phenomenon of flapping wing motion is notably distinctive due to its direct connection to the propulsion and maneuvering of insects, hummingbirds, and dolphins. Insects, such as bees and butterflies, use flapping wing motion for precise aerial maneuvering, while hummingbirds excel at hovering and agile flight with rapid wing beats. Dolphins employ a similar motion in their pectoral fins, enhancing both propulsion and steering during their graceful underwater glides. This particular motion has served as a source of inspiration for engineers seeking to develop devices for energy harvesting and propulsion, as highlighted by the work of Hamada et al. [11,12]. It is characterized as a combination of pitching and heaving motions with a deformable membrane wing.

Moving bodies are numerically handled either with a fixed or a moving mesh. This classification depends on the way the moving boundaries are handled. In moving meshes, the coordinate nodes follow the moving boundaries; i.e. the mesh is reprocessed in each time step. On the other hand, the coordinate nodes do not change in fixed meshes. In these meshes, the boundaries are defined with marker functions

* Corresponding authors, Karim Ahmed and Ahmed A. Hamada contributed equally to this work as first authors.
Received: 30 January 2023, Accepted: 20 March 2024, Published: 02 April 2024

that represent the moving boundaries. Dynamic mesh generation for flapping motion is complex. The complexity of flapping motion lies in the combination of multiple motions: rotation and heaving. Any solid body motion, such as heave, or deformation with a small amplitude can be achieved by a deformable moving mesh that follows the Arbitrary Lagrangian-Eulerian (ALE) technique [13], such as in [14–16]. Furthermore, the technique of Multiple Frames of Reference (MFR) is used to simulate rotating bodies, as in [17–19]. However, using deformable moving meshes to achieve both the rotation and heaving motions will squeeze the mesh cells, increasing the non-orthogonality in the case of small rotation motions and a numerical failure in large rotation angles and large translating motions. For example, handling the flapping foil in swing-arm mode (flapping on an arc path) is complicated as it needs to use MFR for multiple zones of motions [20]. Hamada and Fürth [21] used the fixed mesh and the Level-Set Method (LSM) to simulate the pitch motion and the plunge motion in directions $x-$ and $y-$. Despite the simplicity and efficiency of LSM, it has not been implemented on OpenFOAM yet. Thus, enhancing the Overset moving mesh technique in OpenFOAM is a necessity to simply implement such complex motions.

Over the past decades, a multi-block strategy that combines both ideas of fixed and moving meshes has been applied for simulating complex types of motion and geometries. The main idea of the strategy, which was introduced by Lee et al. [22], is to divide the working domain into smaller blocks where each block has its own individual mesh. There are several forms of the multi-block criteria; Overset is one of those methods that can handle a flapping foil motion. One of the benefits of the Overset method is that it preserves the quality of the mesh and enables random types of motions for complicated geometries [23].

The methodology of Overset meshes allows us to perform any arbitrary rigid body motion that was in the past impractical to simulate using the classical mesh deformation or motion strategies. Recently, some purpose-built codes and commercial CFD software developed successful solvers for the Overset technique [24–26]. Further development in the OpenFOAM community has been conducted to integrate the Overset mesh utility within OpenFOAM [27].

OpenFOAM-v1706 introduced the first release of a significant set of functionalities enabling users to perform Overset mesh calculations [28]. The approach performs cell-to-cell mapping between multiple disconnected mesh regions, to generate a composite domain applicable to both static and dynamic analyses. In OpenFOAM-v1812, general improvements to the pressure-velocity coupling for dynamic Overset cases were included which led to a more realistic pressure-time evolution [29]. Moreover, new Overset solvers were added to OpenFOAM-v2106 related to the multiphase flow domain, which expands the scope of Overset in OpenFOAM [30].

Several studies have been carried out to analyze the Overset function in OpenFOAM. Laws et al. [31] numerically solved a classical benchmark problem of a stationary foil using the Overset mesh technique, evaluating the congruence of their computational results with experimental data. Chandar et al. [24] introduced a comparative performance analysis for the Overset mesh solvers available in both the open source codes (OpenFOAM) and the commercial software (STAR-CCM+ [32], and ANSYS-Fluent [33]).

The aim of this paper is to illustrate the capability of the current Overset mesh in OpenFOAM-v2306 and to introduce an enhanced technique to combine mesh deformation with rigid body motion in Overset meshes. This improvement will enable the OpenFOAM community to simulate complicated moving problems with body deformation. This development can be used in multiple applications, such as blade flutter in wind turbine [34,35], variable shape wave energy converter [36–38], flexible flapping foil [39,40], fish-like swimming [41,42], and droplet formation [43,44].

2. **Overset approach**

The Overset technique (also known as Chimera) was introduced in the CFD community at the beginning of the 1980s by Atta [45]. Further developments were made by Benek et al. [46–48]. Figure (1) shows a slice of an Overset grid for a foil using a hybrid hexahedral (hex) and split-hexahedra (split-hex). The Overset mesh consists of a single background mesh and multiple front meshes for each moving object that overlap each other. Since there is no constraint at the boundaries of the different blocks, this makes the process of mesh generation significantly easier. Furthermore, the component meshes can either be structured or non-structured. One notable advantage of the Overset method is that it eliminates the need for a mesh interface, allowing the technique to effectively handle complex geometries and motions. However, this flexibility comes at the cost of requiring interpolation between overlapped zones to establish connectivity within the domain between the disconnected meshes [23].

The working principle for solving the governing equations in the disconnected background and front meshes can be summarized as follows. When solving in the background domain, the mesh elements related to the front domain are treated as holes and excluded from the computational domain. The boundary value of the Overset domain (boundary of the front mesh) is then interpolated to the background mesh.

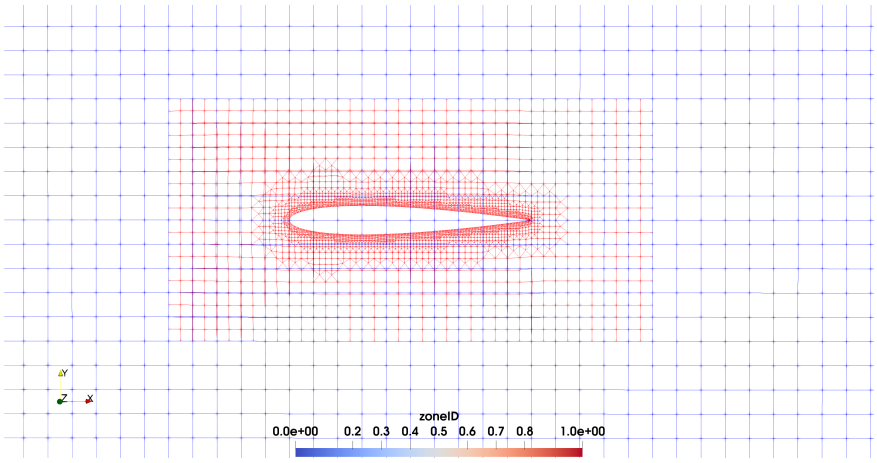


Figure 1. The Overset mesh components: the blue mesh is the background mesh, the red mesh is the front mesh that includes the motion, and interpolation is done at the frontier of the overlapped region.

Within the front mesh, the same solution procedure is performed. At each time step of the solution, the cells in the front and background mesh will follow one of the following types illustrated in Figure (2). The first type of cells is the calculated ones, for which the governing equations are solved. The second is the interpolated cells, whose values are interpolated from the neighboring cells of the other mesh (background elements for the front elements and vice versa). The third type is called holes, for which the cells are removed and no solution is sought [49].

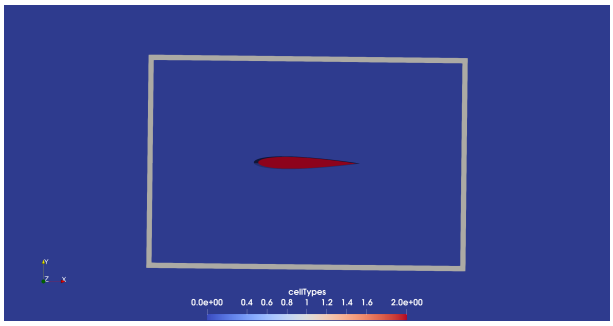


Figure 2. Different cell types: the red cells are holes, the grey cells are interpolated, and the blue are calculated.

3. Physical model

The morphing process of the flexible flapping foil is conducted by controlling the movement of the foil's membrane, which is added to the flapping motion (heaving and pitching). To show the flexibility of the flapping foil, a representation of the current foil configuration is shown in Figure (3). A NACA0012 foil in a uniform flow with a velocity U_∞ is forced to heave and pitch around a fixed axis with given kinematics (heave $h(t)$ and pitch angle $\theta_t(t)$).

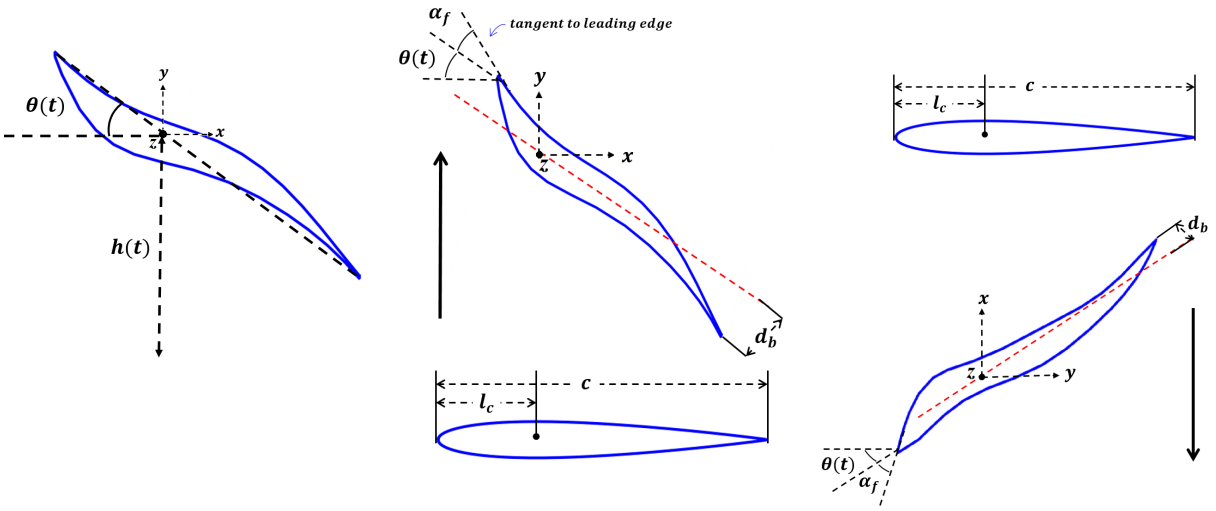
The motion of a flexible flapping foil is described on the basis of the kinematic equations developed by Liu et al. [50]. The temporal variations of heave and pitch are:

$$h(t) = h_0 \sin(\omega t) \tag{1}$$

$$\theta_t(t) = \theta_0 \sin\left(\omega t - \frac{\pi}{2}\right) \tag{2}$$

where h_0 and θ_0 are the amplitudes of heave and pitch, respectively, and $\omega = 2\pi f = 2\pi f^* U/C$ is the flapping angular velocity, where $f = f^* U/C$ is the flapping frequency, U is the free-stream velocity, f^* is the reduced frequency, and c is the foil's chord length.

To mimic the flexibility effect at the leading and trailing edge, a deformation function is specified as a function of time (t), flapping angular velocity (ω), and chord length (c) [50]. The instantaneous position



(a) Oscillation motion device **(b)** Upstroke **(c)** Downstroke

Figure 3. Schematic diagram for (a) oscillating motion device, (b) upstroke and (c) downstroke adapted from Liu et al. [50]

y of any point on the foil ($y(x_f, t)$) is defined as:

$$y(x_f, t) = a_t(x_f) \sin\left(\omega t - \frac{\pi}{2}\right) \tag{3}$$

where x_f is the local coordinate of foil relative to the pitching axis, $a_t(x_f)$ is the lateral amplitude, which is defined as a piece-wise function:

$$a_t(x_f) = \begin{cases} -l_c \left(\frac{x_f}{l_c}\right)^n \sin\alpha_f(t) & x_f < 0 \\ (c - l_c)\left(\frac{x_f}{c-l_c}\right)^n d_b & x_f \geq 0 \end{cases} \tag{4}$$

where l_c is the distance between the leading edge and the fixation point of flapping, d_b is the foil displacement at the trailing edge (m), n is the flexibility coefficient, and α_f is the local Angle of Attack (AoA) at the leading edge in the body-fixed coordinate system (deg) which is expressed as:

$$\alpha_f(t) = \theta_t(t) - \tan^{-1}\left(\frac{dh(t)/dt}{U_\infty}\right) \tag{5}$$

The instantaneous power coefficient (C_{op}) is determined by

$$C_{op}(t) = \frac{p(t)}{0.5\rho U_\infty^3 c} = \frac{1}{U_\infty}\left[C_l(t)\frac{dh(t)}{dt} + C_m(t)\frac{d\theta(t)}{dt}\right] \tag{6}$$

where ρ and p are the density and pressure of the flow, respectively, and $C_l(t)$, $C_d(t)$, and $C_m(t)$ are the instantaneous lift, drag, and pitching moment coefficients of the flexible flapping foil at the flapping point ($c/3$). Then, the power extraction efficiency (η_P) in one flapping cycle is

$$\eta_P = \frac{1}{N_f}\int C_{op}(t)dt = \frac{1}{N_f U_\infty}\int\left[C_l(t)\frac{dh(t)}{dt} + C_m(t)\frac{d\theta(t)}{dt}\right]dt \tag{7}$$

where N_f is the number of flapping cycles.

4. Current overset in OpenFOAM

The present Overset methodology employed in OpenFOAM demonstrates the ability to execute a solid body flapping motion involving both rotation and heave of the front mesh. This motion is mathematically represented by equations (1) and (2). To showcase the capability of the Overset approach, an initial analysis is conducted focusing solely on the solid-body motion without deformation. A two-dimensional test case is presented in Figure (4), featuring a foil undergoing flapping motion with a heaving amplitude of $0.5m$ and a rotation angle of 30° at a frequency of 0.1Hz. The flapping motion occurs seamlessly, where the front mesh experiences both rotational and heaving movements, while the background mesh remains stationary.

Generating deformations at the leading edge and trailing edge of the mesh involves an active motion process, which necessitates the use of a custom-coded displacement motion solver to deform the mesh cells. Thus, the *coded* motion solver in OpenFOAM is selected to compute mesh motion. The prescribed motion of a boundary patch is defined in the *0/pointDisplacement* directory, as specified in OpenFOAM's documentation [27]. To enable the flexibility of the flapping foil, a coded boundary condition for the deformed patch is presented in Listing 1. The implementation of the deformation equation (4) is carried out within the directory *system/codeDict*, see Listing 2, which is invoked with the *codedFixedValue* option in the *0/pointDisplacement* directory.

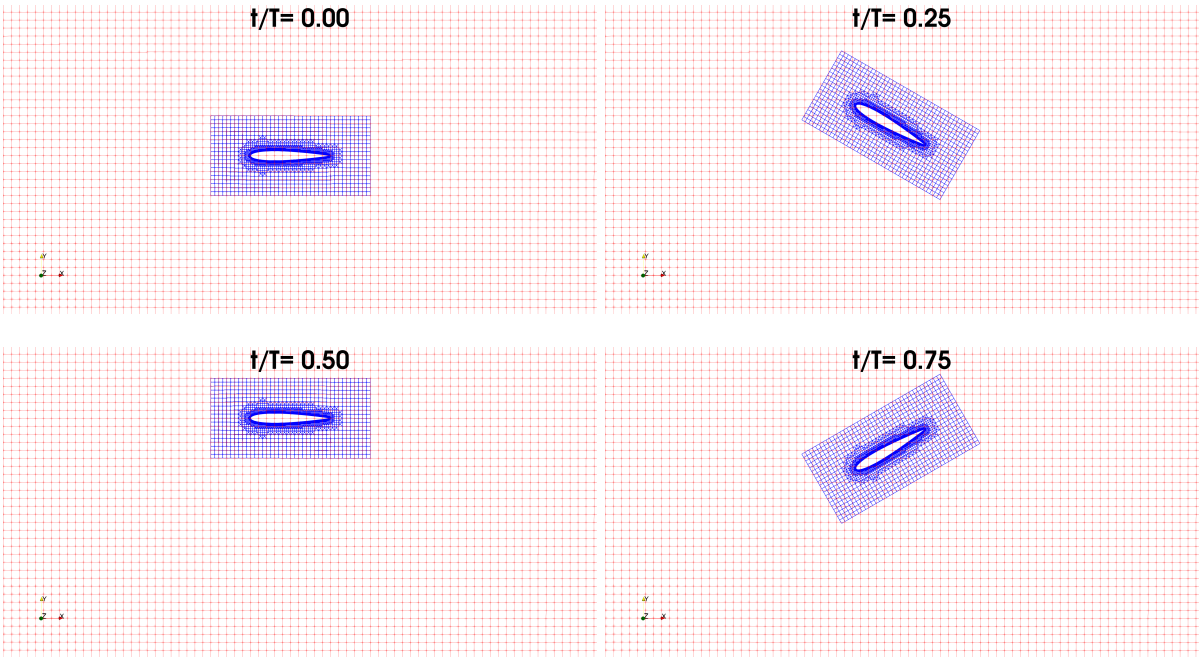


Figure 4. Overset mesh deformation for a rigid flapping airfoil at different time steps.

```
wing   //The deforming patch
   {
        type            codedFixedValue;
        name          panelDisplacement;
        value           uniform (0 0 0);
   }
```

Listing 1. Point Displacement Dictionary

```
panelDisplacement
{
    codeInclude #{ #include <cmath> #};
    code #{
    // Put your deformation code here
         #};
}
```

Listing 2. Code Dictionary

A straightforward 2D simulation is conducted to test the deformation motions, involving both leading-edge and trailing-edge displacements, alongside a flapping motion, encompassing heaving and pitching motions. The motion solver employed in this investigation is the *solidBodyDisplacementLaplacian*, which is capable of modeling both displacement and solid body phenomena [51]. To initiate the simulation, it is paramount to configure the *dynamicMeshDict* directory, focusing on and customizing the two pivotal components: the *dynamicFvMesh* and the *motionSolver*, as shown in Listing 3.

```
dynamicFvMesh    dynamicOversetFvMesh;
motionSolver     solidBodyDisplacementLaplacian;
```

Listing 3. Dynamic Mesh Dictionary

The subsequent step involves commencing the simulation to examine the motion of the flexible flapping foil. To ensure the precision of the mesh motion before proceeding to compute the flow dynamics around the flexible flapping foil, the solver *moveDynamicMesh* is executed to simulate the mesh motion. The simulation results are shown in Figure (5), indicating that the default dynamic Overset mesh in OpenFOAM accurately applies the deformation to the leading edge and the trailing edge. However, a

limitation emerges when incorporating rigid body motion with deformation, as both the front and the background meshes move and deform concurrently. The issue lies in the current Overset procedure's inability to distinguish between different zones. This leads to the application of both rigid body motion and deformation to the entire mesh instead of solely the front mesh. Furthermore, this issue imposes significant constraints when endeavoring to conduct a numerical simulation involving multiple deformable moving bodies using Overset methodology. Consequently, it becomes imperative to enhance the Overset library within OpenFOAM to enable selective deformation solely in the front mesh while maintaining the background mesh in a stationary state.

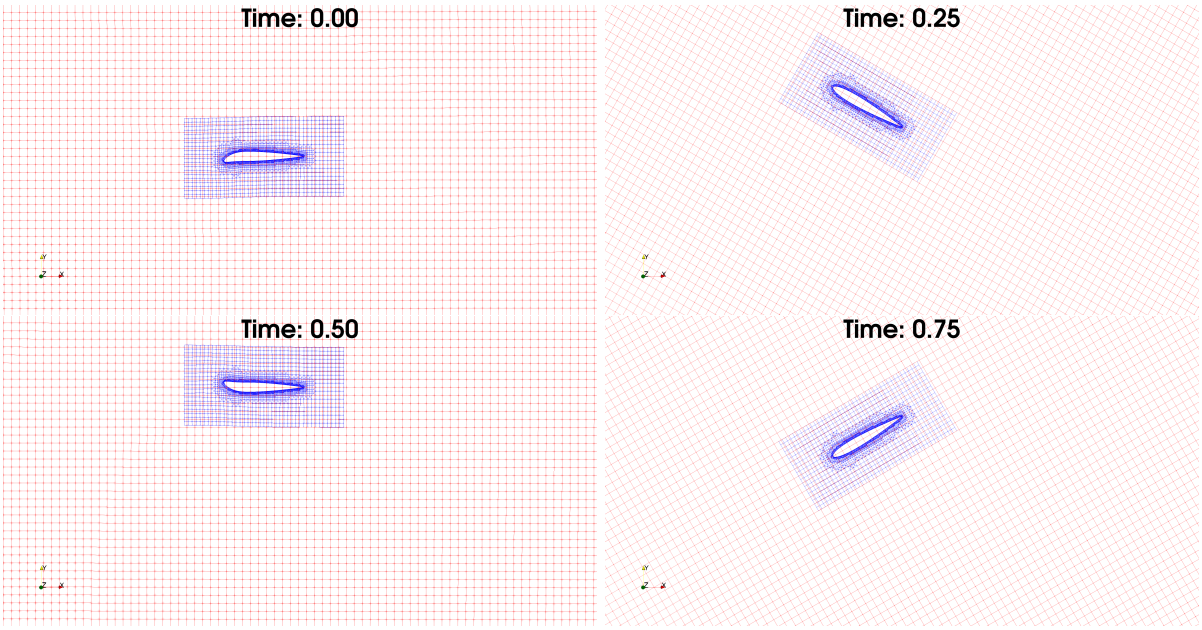


Figure 5. Dynamic mesh motion using Overset technique. The Overset deficiency in different zone identification appears clearly at $t/T = 0.25$, $t/T = 0.5$, and $t/T = 0.75$, where $T = 1.92$ is the periodic flapping cycle. Both the background and the front mesh are in motion.

5. Overset with mesh zones

To address the aforementioned problem, a thorough understanding of the functionality of the Overset code in OpenFOAM is crucial. This understanding will enable the modification of the code to incorporate zone identification within the Overset mesh module. The OpenFOAM API map of *dynamicOversetFvMesh* [52] provides an overview of the underlying process, as depicted in Figure (6). The *dynamicOversetFvMesh* sub-class in the Overset library is linked to the *dynamicFvMesh* library to perform the body motion. Additionally, *dynamicOversetFvMesh* is built upon a certain mesh solver in the *dynamicFvMesh* library, named *dynamicMotionSolverListFvMesh* [52], as shown in Figure (7)A. Upon examination of the C++ code and the header files associated with the solver, namely *dynamicMotionSolverListFvMesh.c* and *dynamicMotionSolverListFvMesh.h*, it becomes apparent that the existing implementation lacks a mechanism for zone identification, due to the glaring absence of the *zoneID* variable definition. Consequently, when deformation is combined with rigid body motion, the current Overset method in OpenFOAM fails to differentiate between the front and background meshes. This issue is visually represented in Figure (5).

In response to this limitation, a practical solution that comes to the fore is the replacement of the current *dynamicMotionSolverListFvMesh* solver with an alternative solver located within the *dynamicFvMesh* directory, as shown in Figure (7)B. This alternative solver has the capability to differentiate between distinct zones while retaining the original mesh motion functionality. By adopting such a solver, the impediment to applying zone-specific deformations is effectively overcome, thereby enhancing the versatility and adaptability of the Overset methodology in OpenFOAM for a broader range of applications with flexible moving objects.

In a comprehensive overview, a comparison of the member functions defined for both solvers, namely *dynamicMotionSolverListFvMesh* and *dynamicMultiMotionSolverFvMesh*, is presented in listings (4) and

(5). In the first solver, it is observed that the only mentioned function is *motionSolvers*, which is not originally associated with any zone identification (*zoneID* variable) [53]. On the contrary, the second solver includes the indexing variable *zonei*, which depends on the *zoneID* array variable and allows for differentiation between different zones [54]. In its member function, the variable *pointIDs* is defined based on the *zoneID* variable [55], enabling the application of a specific motion solver to the points of each defined zone. The *dynamicMultiMotionSolverFvMesh* possesses the desired capabilities of zone differentiation and mesh motion.

The *dynamicMultiMotionSolverFvMesh* unequivocally embodies the desired capabilities of zone differentiation coupled with seamless mesh motion integration. Consequently, the logical progression involves the creation of a new dynamic solver subclass within *oversetFvMesh*. This is achieved by duplicating the existing solver, thus preserving the original solver for comparative purposes. Subsequently, the default solver (*dynamicMotionSolverListFvMesh*) in *dynamicOversetZoneFvMesh* is supplanted by the more adept *dynamicMultiMotionSolverFvMesh*, as shown in Figure (8).

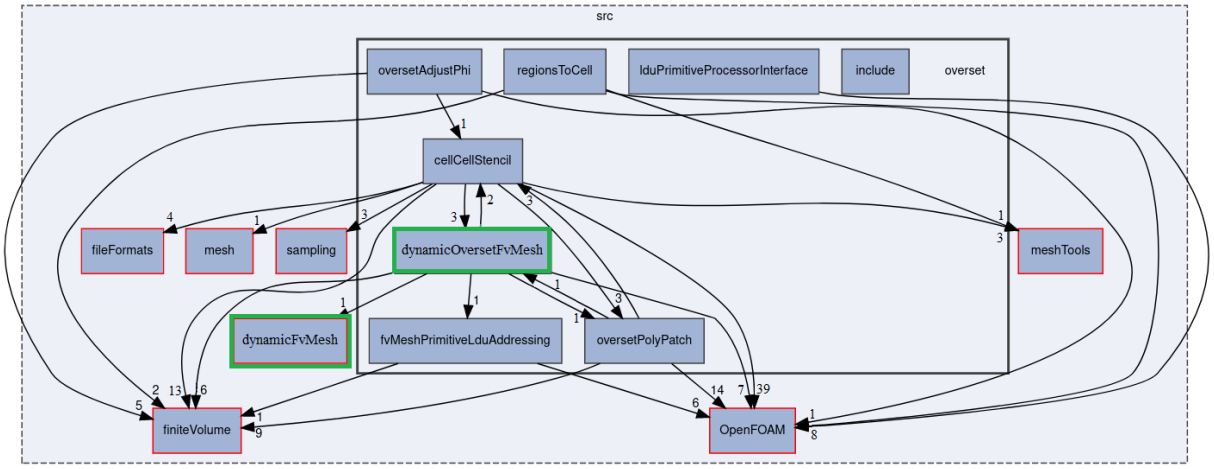
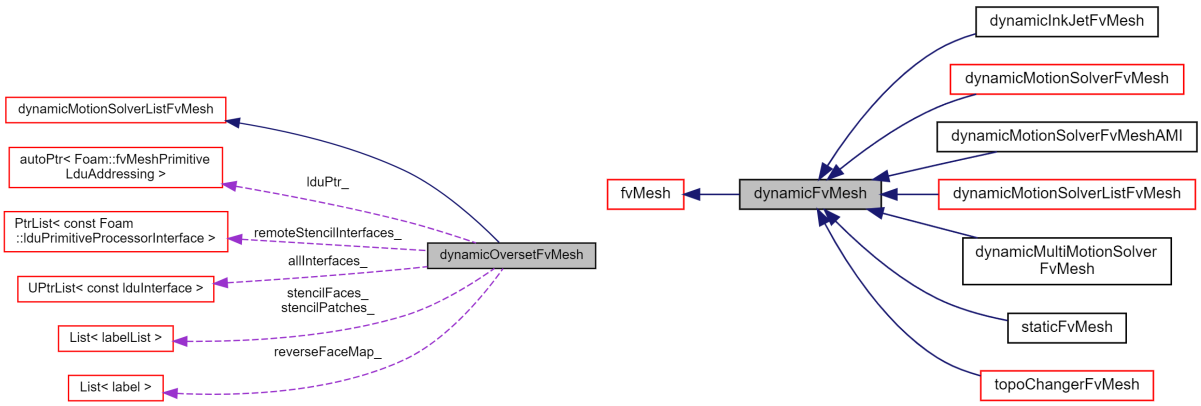


Figure 6. Directory dependency graph for Overset [52], showing the library dependence of Overset on dynamic mesh libraries. Boxes outlined in black signify the classes of the Overset library, while those outlined in red represent other classes (outside the Overset library) for which not all inheritance/containment relations are displayed. Arrows are utilized to depict public inheritance relations between two classes. Numbers over the arrows indicate the number of files included between the classes. Highlighted green boxes specifically identify classes within the Overset library that handle dynamic motions.



(a) *dynamicOversetFvMesh* sub-classes [56]. **(b)** *dynamicFvMesh* sub-classes [57].

Figure 7. Inheritance diagrams for the sub-classes of the *dynamicOversetFvMesh* [56] and *dynamicFvMesh* [57] libraries, showing the dependency of the default Overset technique on the *dynamicMotionSolverListFvMesh* sub-class and the different dynamic motion sub-classes existed in the *dynamicFvMesh*.

```
//* * * * Member Functions  * * * * //
bool Foam::dynamicMotionSolverListFvMesh
    ::update()
{
    if (motionSolvers_.size())
    {
        // Accumulated displacement
        pointField disp(motionSolvers_
    [0].newPoints() - fvMesh::points());
        for (label i = 1; i <
    motionSolvers_.size(); i++)
        {
            disp += motionSolvers_[i].
    newPoints() - fvMesh::points();
        }
        fvMesh::movePoints(points() +
    disp);
        volVectorField* Uptr =
    getObjectPtr<volVectorField>("U");
.......
.......
}
```

Listing 4. Member function of *dynamicMotionSolverListFvMesh* (default solver)

```
//* * * * Member Functions  * * * * //
bool Foam::
    dynamicMultiMotionSolverFvMesh::
    update()
{
    pointField transformedPts(points());
    forAll(motionPtr_, zonei)
    {
        const labelList& zonePoints =
    pointIDs_[zonei];
        const pointField newPoints(
    motionPtr_[zonei].newPoints());
        for (const label pointi :
    zonePoints)
        {
            transformedPts[pointi] =
    newPoints[pointi];
        }
    }
  ......
  ......
}
```

Listing 5. Member function of *dynamicMultiMotionSolver-FvMesh* (The added solver)

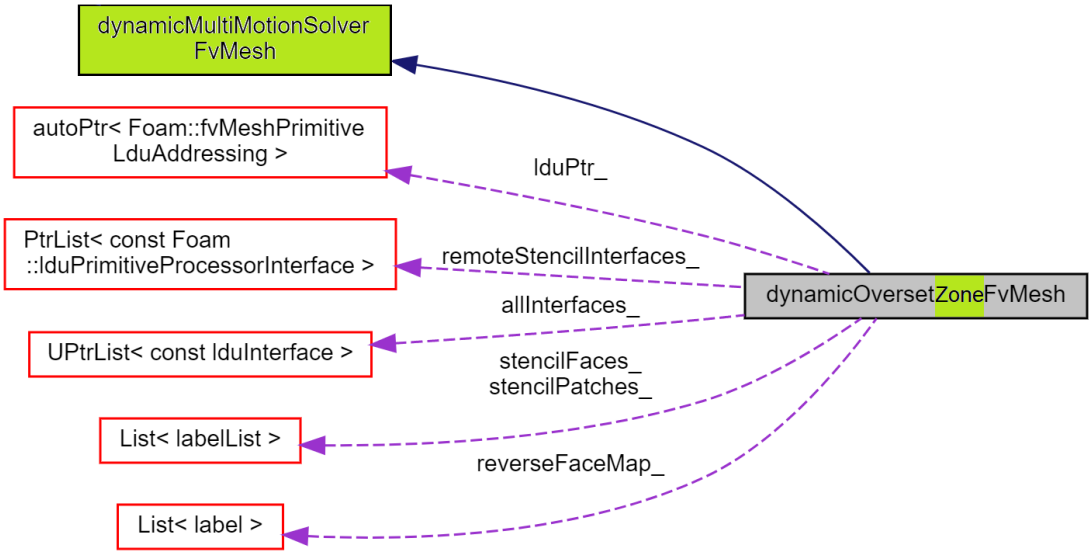


Figure 8. Inheritance diagrams for the *dynamicOversetZoneFvMesh* library, showing the dependency of the modified Overset technique on the *dynamicMultiMotionSolver-FvMesh* library, existed in the *dynamicFvMesh*.

This process enables the retention of the original solver, facilitating side-by-side evaluations and comparisons with the newly implemented solver (*dynamicOversetZoneFvMesh*). By keeping the old solver intact, a comprehensive assessment of the enhancements and effectiveness brought about by the new solver is made feasible, ensuring a robust and well-informed evaluation process.

6. Results and discussion

The modified solver's functionality is validated through 2D and 3D simulations involving a flexible flapping foil. Moreover, parallelization ability and proficiency are assessed with strong scaling in the 2D simulation. A meticulous examination of mesh convergence and validation is performed as a precursor to a numerical investigation of the impact of active membrane flexibility on flow dynamics and power extraction in flapping foil operations. Thus, these meticulously designed simulations served as an invaluable test-bed for the enhanced Overset solver, ensuring its robustness and reliability in tackling complex fluid dynamics scenarios.

Across all simulations, the foil flaps in a simple mode in a high Reynolds number flow ($Re = 10^6$) with heaving and pitching amplitudes (h_0 and θ_0) of $0.5c$ and $40.23°$, respectively, and a flapping frequency (ω) of $2rad/s$ around the flapping point from the leading edge (x_{0f}) of $c/3$. The flexibility of the flapping foil is regulated by active control of its leading edge deformation, incorporating a local Angle of Attack (α_f) set at $15°$, as defined by Equations (3) and (4) (with consideration limited to $x_f < 0$). Furthermore, in the 3D simulation, the foil undergoes deformation along the z-direction by applying the 2D deformation multiplied by the absolute value of the normalized z coordinate with the wing half-span ($S_{pan}/2$), denoted as $y(x_f, z, t) = y(x_f, t)|2z/S_{pan}|$. Consequently, a full 2D deformation occurs across the airfoil section at the wing's tip, whereas no deformation is observed across the airfoil section at the wing's root. The interpolation of deformations between these two locations is executed linearly. The flapping and flexibility parameters are shown in Table (1).

Table 1. Flow, flexibility, and flapping parameters in the current study.

Parameter	**value**
Reynolds number (Re)	1×10^6
Foil chord (c)	$1.0m$
Flapping point from the leading edge (x_{0f})	$c/3$
local Angle of Attack (α_f)	$15°$
2D flexibility function ($y(x_f, t)$)	Equation (3)
3D flexibility function ($y(x_f, z, t)$)	$y(x_f, t)\|2z/S_{pan}\|$
Reduced frequency (f^*)	0.15
Flapping frequency ($\omega = 2\pi f^* U/C$)	$2rad/s$
Heaving amplitude (h_0)	$0.5c$
Pitching amplitude ($\theta_0 = \alpha_f + \arctan(\omega h_0/U)$)	$40.23°$

6.1. **Numerical setup.** The simulations model an incompressible turbulent flow ($Re = 1 \times 10^6$) passing over flapping foil/wing. Thus, incompressible Reynolds Averaged Navier-stokes (RANS) equations are solved with the k-ω SST turbulence model.

The *overPimpleDyMFoam* solver, a transient module designed for incompressible turbulent flow on a moving mesh (overset), was selected for simulations. The *pimple* solvers are a combination of PISO (Pressure Implicit with Splitting of Operator) and SIMPLE (Semi-Implicit Method for Pressure-Linked Equations) [58]. The PISO algorithm is an efficient method to solve the Navier-Stokes equations in unsteady problems, while the SIMPLE algorithm is used for steady ones. Thus, the used solver relies on the PISO mode to simulate the unsteady flow around the flapping foil. The solver accounts for the incompressibility effect and the uncoupling between the continuity and momentum equations by solving the momentum equation and the Poisson pressure equation as a pressure correction [59].

As shown in Figure (9), the computational domain takes the form of a rectangular region with length and height of $20c$ and $12c$, respectively. The origin point of the domain is located at the flapping center of the foil ($c/3$) at its initial condition and divides the y-direction equally. The foil flaps in the $x - y$ plane. The inlet boundary is far from the inlet by $8c$. The domain was discretized with a Cartesian grid (hexahedral cells). Two levels of refinements were implemented in both the front and background meshes. Additionally, two layers close to the foil were included in the front mesh during the mesh generation process using *refineMesh* and *snappyHexMesh*. This meshing strategy was employed to capture the turbulent boundary layer and the flow behavior around the flapping foil. The sizes of the cells in the interpolation regions between the front and background meshes were kept the same to ensure a smooth transition of the solution between the two meshes. In the 3D case, the z-direction aligns with the span of the wing, which was discretized with equal spaces. Table (2) shows the initial and boundary conditions of the computational domain for incompressible turbulent flow over a flapping foil problem. The simulations for both cases were carried out for a duration of 25 seconds to ensure the attainment of a steady-state solution. A variable time step was applied with a maximum Courant–Friedrichs–Lewy (CFL) number of 0.5 and a maximum time step of 0.01.

When configuring our simulation files (*fvScheme* and *fvSolution*), we adhere to the best practices recommended by both the OpenFOAM community and Wolf Dynamics [60], as shown in Table (3). This involves employing cell-limited Gauss linear schemes for gradient computations, utilizing *Gauss linearUpwindV* for divergence in Overset mesh scenarios, and adjusting relaxation factors to finely control convergence. The PIMPLE algorithm is configured to improve computational stability and accuracy. Enabling the *momentumPredictor* parameter anticipates velocity field corrections, improving simulation accuracy.

Activation of *oversetAdjustPhi* accommodates complexities in Overset mesh configurations. The number of outer and inner correctors is set to one, balancing computational efficiency and accuracy. Setting *nNonOrthogonalCorrectors* to two refines solutions in non-orthogonal mesh regions. *moveMeshOuterCorrectors* dynamically adjusts the mesh during outer corrector iterations. Moreover, enabling *ddtCorr* and *checkMeshCourantNo* enhances stability and accuracy by considering time derivative term corrections and monitoring the Courant number. This meticulous PIMPLE configuration underscores a robust simulation methodology.

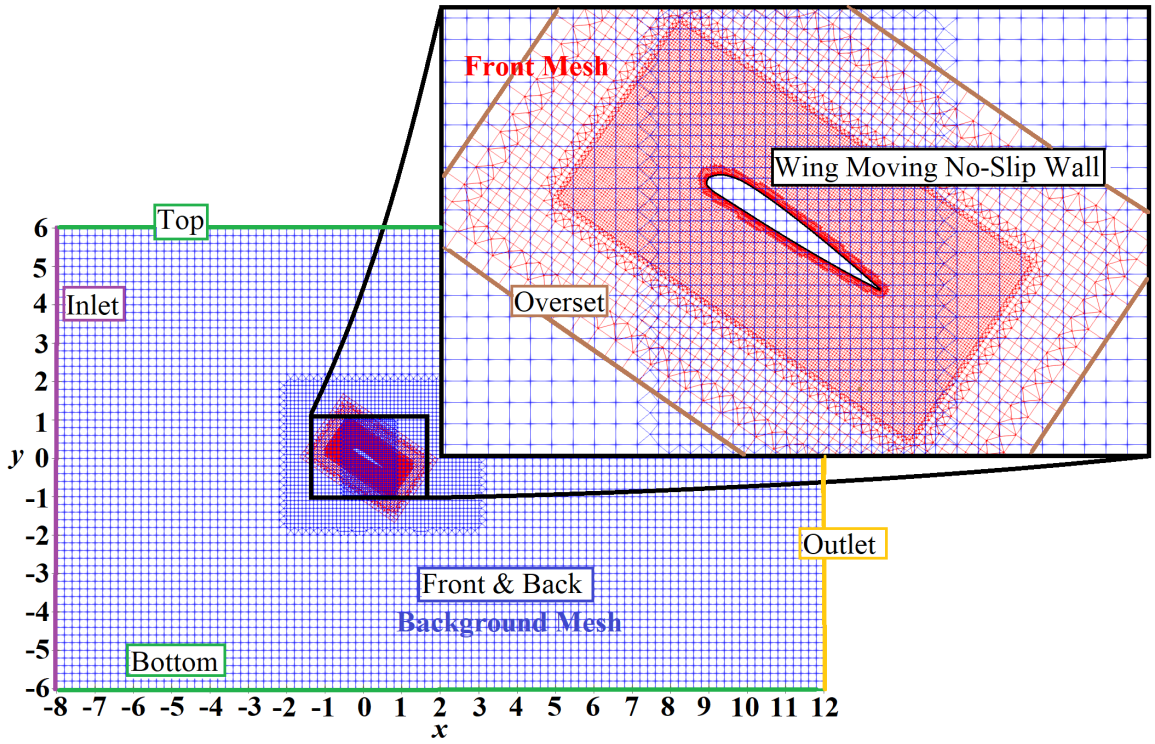


Figure 9. The computational domain for the flapping foil problem, showing the mesh, and the boundaries. The grid is refined around the foil to capture the turbulence.

Table 2. Boundary conditions of the computational domain for the flapping foil problem.

	p	U	k	ω	ν_t	*pointDisplacement*
Wing	zeroGradient	movingWallVelocity	(kqR, omega, nut)WallFunction			codedFixedValue
Inlet		fixedValue			calculated	uniformFixedValue
Outlet	fixedValue	inletOutlet				
Top & Bottom	zeroGradient					
Overset	overset					
Front & Back	empty for 2D symmetryPlane for 3D					

6.2. **Verification.** To ascertain the efficacy of the enhanced Overset solver, 2D and 3D simulations are executed utilizing a coarse mesh. Verification of the enhanced Overset solver involves executing the OpenFOAM mesh motion solver, *moveDynamicMesh*. This step assesses the effectiveness of the modified solver in properly handling both front and background meshes. Subsequently, *overPimpleDyMFoam* is executed to confirm the compatibility of communication between the modified overlay mesh solver and the flow solver.

The intricate dynamics of the movement and deformation of the mesh are vividly illustrated in Figures (10) and (11), providing a comprehensive description of the flawless performance of the deformation of

Table 3. *overPimpleDyMFoam* solver configuration: key parameters and settings

Parameter	Setting
Gradient	Cell-limited Gauss linear
Divergence	Gauss linearUpwindV
PIMPLE Algorithm	
momentumPredictor	Enabled
oversetAdjustPhi	Enabled
Correctors	
Outer and Inner	1
nNonOrthogonal	2
moveMeshOuter	Enabled
ddtCorr	Enabled
checkMeshCourantNo	Enabled

the leading edge region throughout the entire flapping cycle. A comparison with the mesh of a rigid-body flapping foil is shown in the figure to visualize the deformation effect on the mesh in the flexible case. In addition, the flapping foil coordinates are displayed for the 3D simulation to demonstrate the variation of the foil deformation along the third coordinate (z). It is worth noting the synchronized display of both rigid-body flapping motion and deformation in the front mesh, juxtaposed against the stationary state of the background mesh. Furthermore, the flow field surrounding the flexible flapping foil and wing, as shown in Figure (12), serves as evidence of the effective communication between the modified Overset solver and the flow solver. These contours show the smooth transition of the solution between the background and front meshes. Therefore, the cell volume changes during the flapping cycle, accounting for the front mesh deformation due to active flexibility. These results validate the functionality of the modified solver, successfully surmounting prior limitations and precisely realizing the targeted motion behavior. The subsequent simulations are conducted in a two-dimensional (2D) framework owing to the substantial computational demands associated with three-dimensional (3D) simulations.

6.3. **High performance computation.** In this section, the ability of the modified Overset solver to run in parallel computations is examined through strong scaling. In strong scaling, the mesh size (N) was kept constant and the number of processors (N_p) was increased. The study was carried out on three meshes (coarse, medium, and fine) of $7,525$, $23,838$, and $90,234$ cells using two workstations. The selected decomposing method for parallelization was *hierarchical*. The CPU architecture and memory information of the workstations is presented in Table (4). The speedup (S) and efficiency (E) of the parallel computations were calculated for the strong scaling, and they are expressed as:

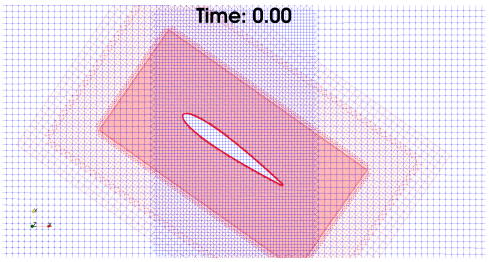
$$S = T_s/T_p \tag{8}$$

$$E = S/N_p = \frac{T_s}{N_p T_p} \tag{9}$$

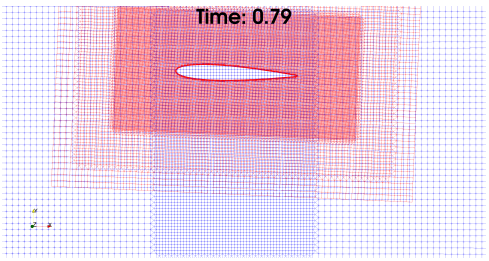
where T_s and T_p are the serial and parallel time to execute the program, respectively.

Table 4. CPU architecture and memory information of the used workstations.

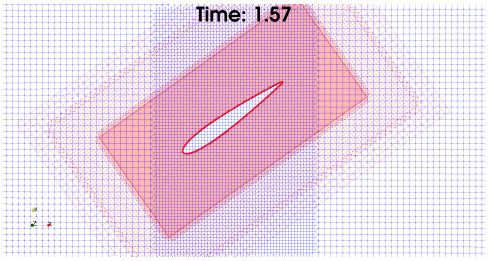
	Workstation 1	Workstation 2
Model name	Intel(R) Xeon(R) W-2295	Intel(R) Xeon(R) Gold 6240
CPUs (processors)	36	36
Threads per socket	2	1
Cores per socket	18	18
Sockets	1	2
CPU Base MHz	3000	2600
CPU max MHz	4800	3900
L1d cache	576 KiB	1.1 MiB
L1i cache	576 KiB	1.1 MiB
L2 cache	18 MiB	36 MiB
L3 cache	24.8 MiB	49.5 MiB
Memory size	128 GB	256 GB



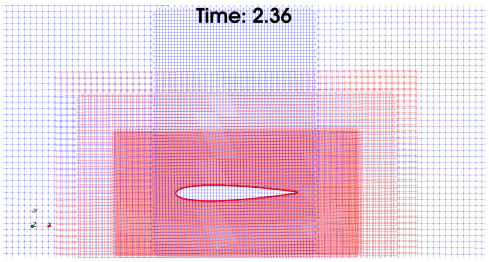
(a) Rigid flapping foil at $t/T = 0.00$.



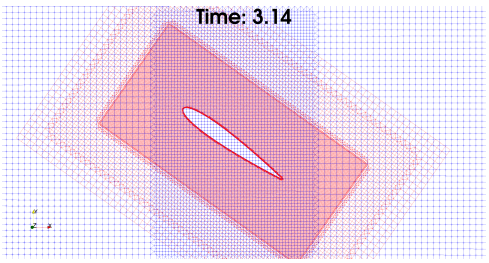
(c) Rigid flapping foil at $t/T = 0.25$.



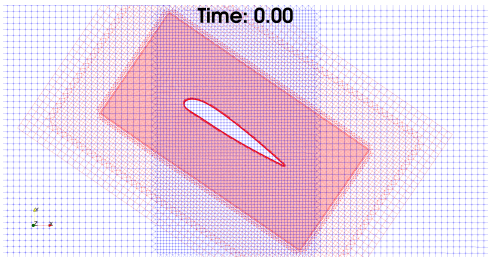
(e) Rigid flapping foil at $t/T = 0.50$.



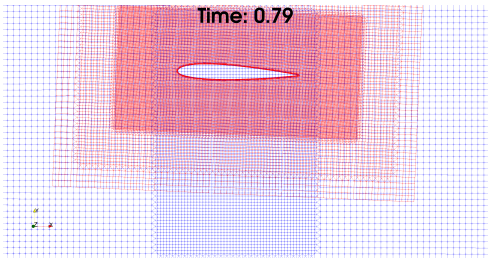
(g) Rigid flapping foil at $t/T = 0.75$.



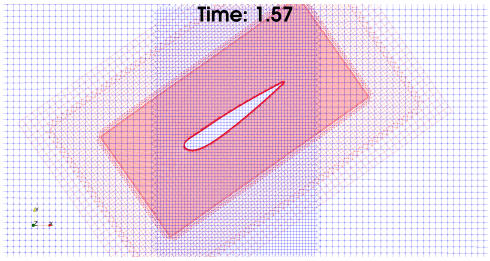
(i) Rigid flapping foil at $t/T = 1.00$.



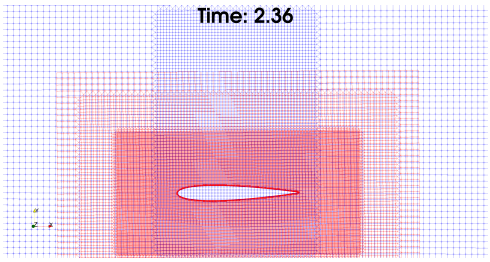
(b) Leading edge flexible flapping foil at $t/T = 0.00$.



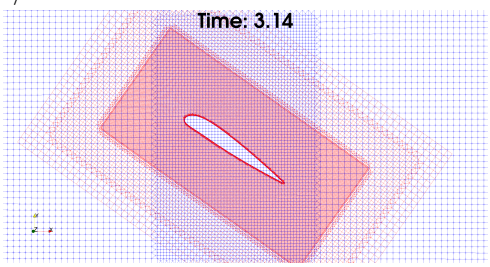
(d) Leading edge flexible flapping foil at $t/T = 0.25$.



(f) Leading edge flexible flapping foil at $t/T = 0.50$.

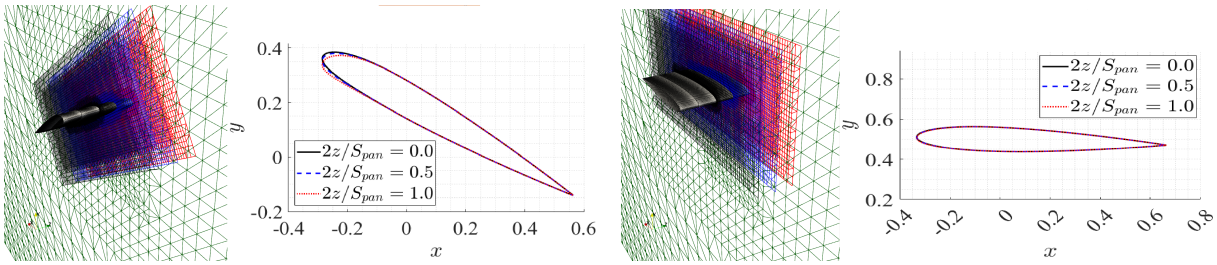


(h) Leading edge flexible flapping foil at $t/T = 0.75$.

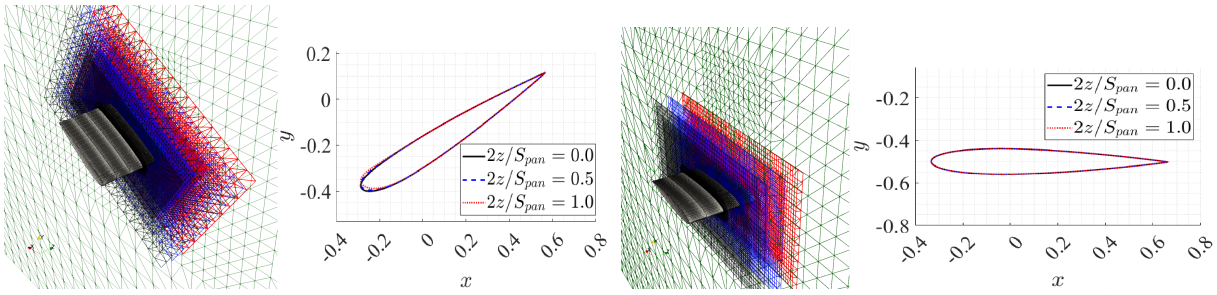


(j) Leading edge flexible flapping foil at $t/T = 1.00$.

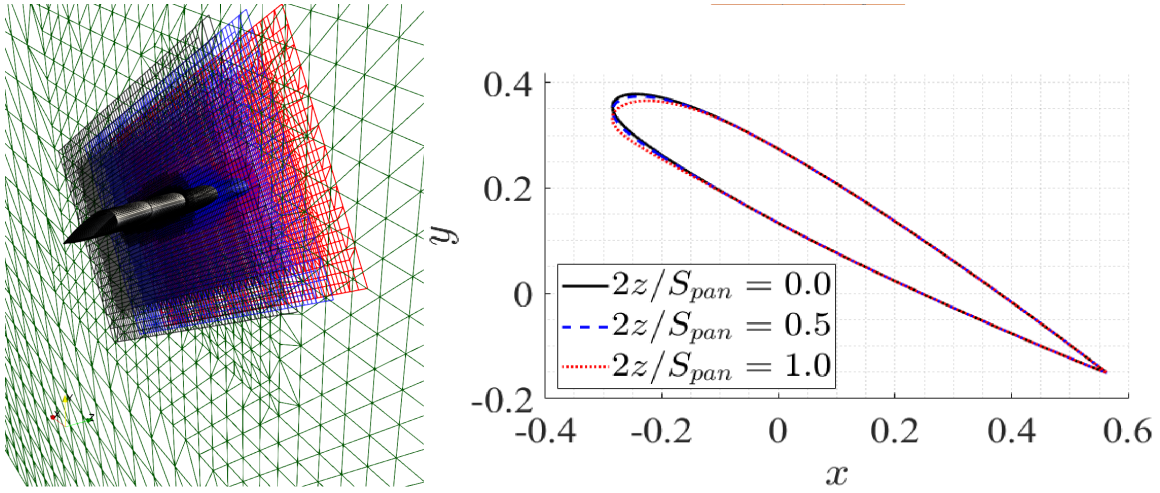
Figure 10. Mesh motion results for the cases of a rigid and flexible foil during a flapping cycle. The modified solver *dynamicOversetZoneFvMesh* applies the cell deformation to the front mesh only around the leading edge region. In addition, a rigid body motion is applied to the whole front mesh boundaries while the background mesh is kept stationary.



(a) Leading edge flexible flapping wing at $t/T = 0.00$. **(b)** Leading edge flexible flapping wing at $t/T = 0.25$.

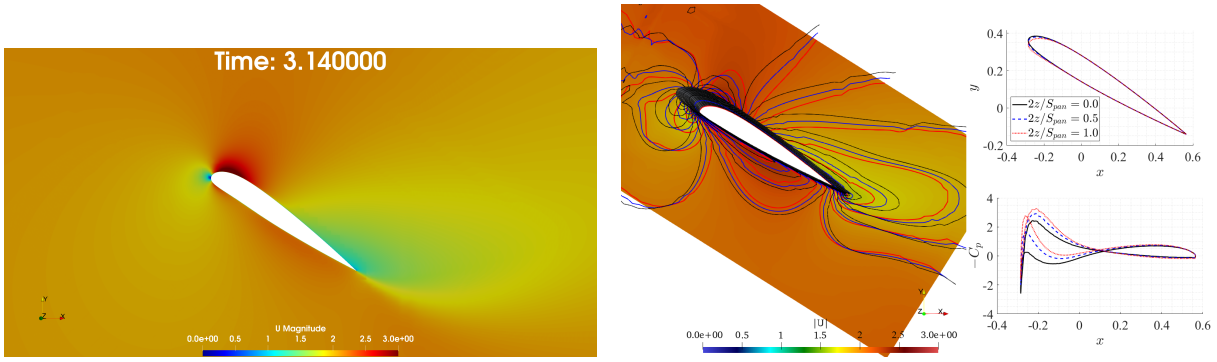


(c) Leading edge flexible flapping wing at $t/T = 0.50$. **(d)** Leading edge flexible flapping wing at $t/T = 0.75$.



(e) Leading edge flexible flapping wing at $t/T = 1.00$.

Figure 11. Mesh motion results for the case of a flexible flapping wing during a flapping cycle. The modified solver *dynamicOversetZoneFvMesh* applies the cell deformation to the front mesh only around the leading edge region. In addition, a rigid body motion is applied to the whole front mesh boundaries while the background mesh is kept stationary.



(a) 2D simulation: Leading edge flexible flapping foil. **(b)** 3D simulation: Leading edge flexible flapping wing.

Figure 12. Flow field results for the cases of flexible foil and wing at $t/T = 1.00$ during a flapping cycle, confirming the successful communication between the modified Overset solver and the flow solver.

Figure (13) shows the impact of parallel simulation using the modified Overset solver on computational expenses. These results are obtained by averaging the execution time of a 100-time step after ensuring the periodic converged solution (after a simulation time of 25 seconds). The results show that when the number of cores increases, the time required for a single time step significantly decreases, especially for larger mesh sizes. However, parallelization is ineffective for smaller mesh sizes, resulting in a notable drop in computational efficiency even if the execution time is decreased, such as in the coarse and medium meshes. Opting for 18 processors proves to be a suitable choice for the fine mesh (90,234 cells), striking a balance between computation cost and speed. In serial computing (1 processor), the simulation time amounts to 4.11 seconds per time step. However, leveraging parallel processing with 18 cores remarkably reduces this time to 0.3 seconds, maintaining the computational efficiency close to 1 and the speedup curve close to the unity-slope line. This demonstrates the scalability and efficiency gains achieved through parallelization in our simulations, in addition to validating the capability of the modified Overset solver for parallel computation.

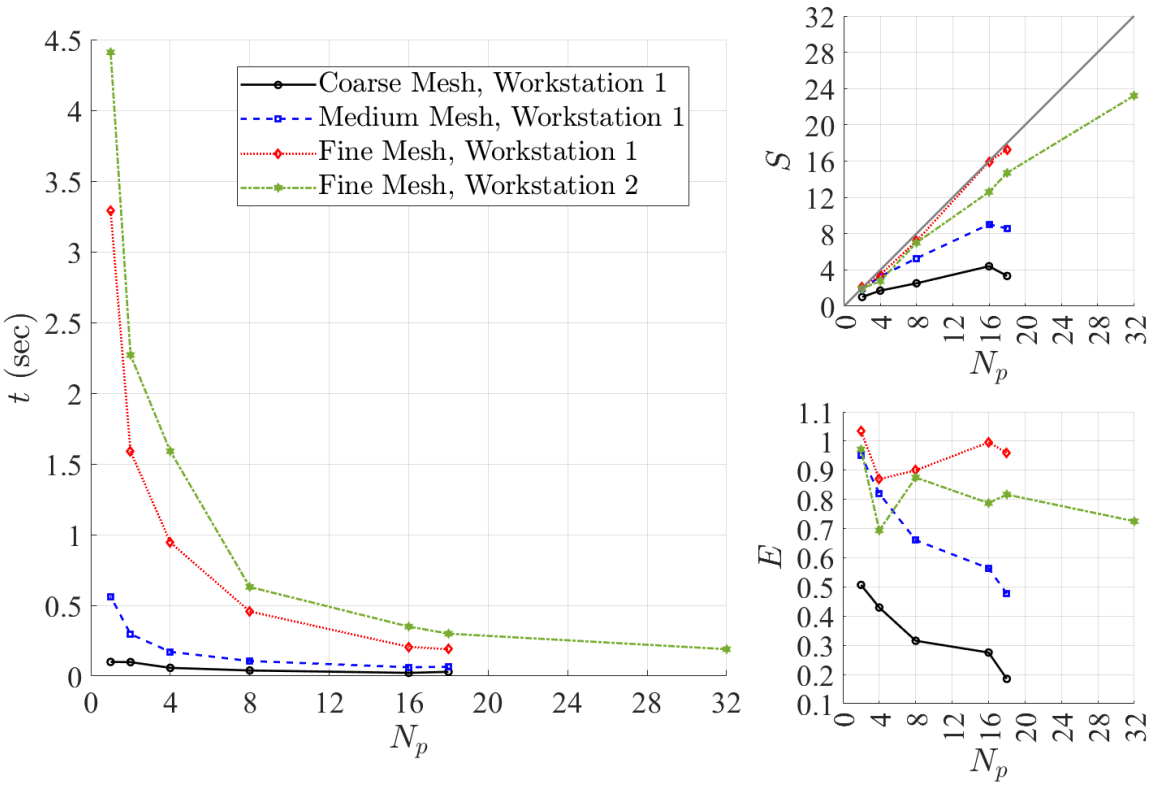


Figure 13. The execution time, speedup, and efficiency of parallel simulations with varying numbers of processors.

6.4. **Validation and mesh convergence studies.** The validation of the mesh and flow solvers draws upon the work of Liu et al. [50], who numerically simulated a 2D rigid flapping foil. The setup involved a Reynolds number (R_e) of 10^6 under sea level conditions, with a velocity (U) of $14.6m/s$. In the simulation, the $k-\omega$ SST turbulence model is employed. The motion characteristics encompassed active heaving with an amplitude (h_o) of $1m$ and pitching motion with an amplitude (θ_o) of 61.487^o. The reduced frequency (f^*) was set to 0.2. Figure (14) shows the temporal variation of the lift coefficient ($(C_l(t/T))$, where T is the time for one flapping cycle), compared with the literature data. The temporal results generally align with the findings reported in the literature during the flapping cycle, with minor discrepancies noted in the transition from the upstroke to the downstroke. This correspondence underscores the predictive accuracy inherent in the employed numerical configuration.

To ensure the independence of the results on the resolution of the grid, a mesh convergence study was performed using three meshes (coarse, medium, and fine) of $7,525$, $23,838$, and $90,234$ cells ($2,067$, $8,214$, and $32,970$ cells in the background mesh and $5,458$, $15,624$, and $57,264$ cells in the front mesh, respectively). The grid convergence analysis is conducted with a 2D flexible flapping foil configured with predefined flapping and flexibility parameters, see Table (1). Figure (15) shows the temporal behavior of the lift, drag, pitching moment, and power coefficients for the mesh convergence assessment.

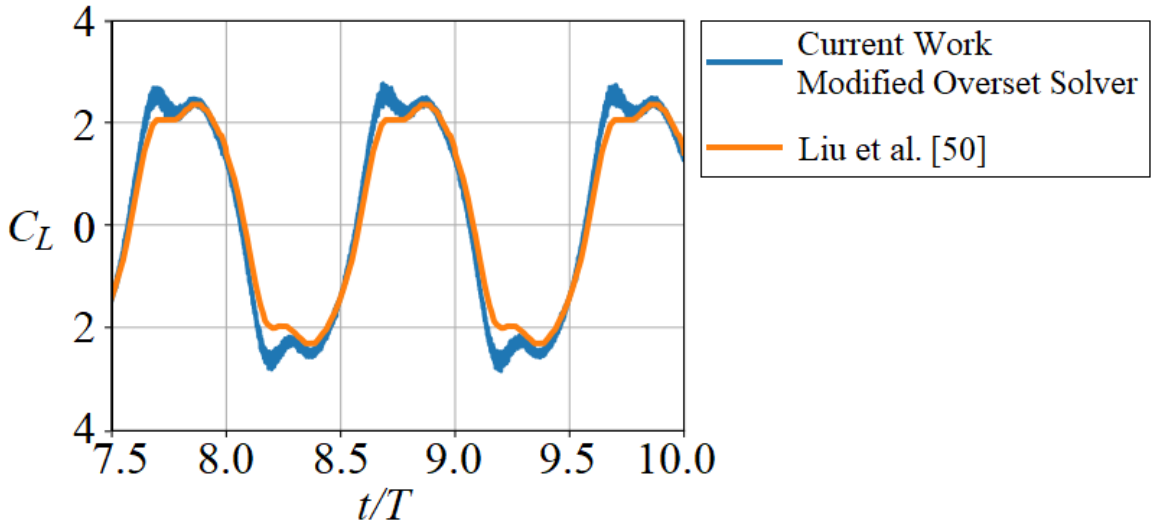


Figure 14. Validation study: temporal variation of lift coefficient($C_l(t/T)$) demonstrating overall alignment of the current work with literature data [50].

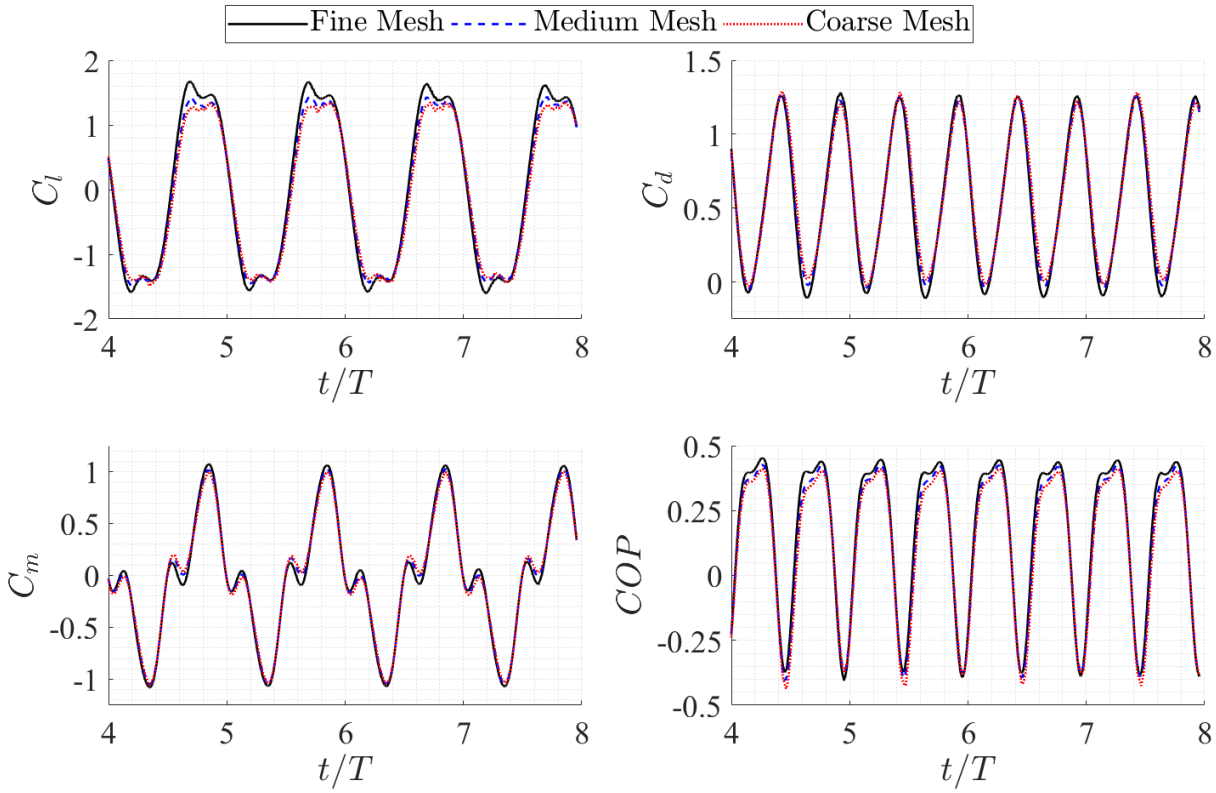


Figure 15. Mesh convergence study: temporal variation of lift, drag, pitching moment, and power coefficients ($C_l(t/T)$, $C_d(t/T)$, $C_m(t/T)$, and $COP(t/T)$, respectively) for three meshes.

Furthermore, first-order statistics of the coefficients were calculated, including the mean drag coefficient ($\overline{C_d(t/T)}$) and the root mean square (RMS) of the lift and pitching moment coefficients ($rms\,(C_l(t/T))$ and $rms\,(C_m(t/T))$, respectively), as shown in Table (5). The order of convergence was determined through Richardson's extrapolation [61] with refinement ratios of $r_{21} = \frac{h_2}{h_1} = r_{32} = \frac{h_3}{h_2} = 2$, following the guidelines of Celik et al. [62] for the lowest refinement limit ratio ($r > 1.3$). The extrapolated values for the mean drag coefficient and the RMS of the lift and pitching moment coefficients were 0.5594, 1.2615, and 0.6157, respectively, with the corresponding values of the Grid Convergence Index (GCI) of 5.27%, 5.79%, and 5.28%. Table (5) presents the mesh error percentages relative to the fine mesh and relative

to the extrapolated values obtained from the Richardson extrapolation method [61]. The extrapolated errors are within the 4% range for the fine mesh, and the results of the last two fine meshes are adequate. This indicates the convergence of the simulation. Consequently, the implementation of an even finer mesh beyond the fine one was ignored, considering the associated increase in computational expenses with mesh refinement (a simulation of 25 seconds on the fine mesh takes approximately a week with 18 processors). The finest mesh of $90,234$ cells was thus selected for the subsequent flow analysis.

Table 5. Independent convergence study: First order statistics of instantaneous lift, drag, and pitching moment coefficients during a flapping cycle for a flexible flapping foil in simple mode.

Mesh size (N)	Numerical values			Extrapolated values		
	$\overline{C_d}$	RMS(C_l)	RMS(C_m)	$\overline{C_d}$	RMS(C_l)	RMS(C_m)
$7,525$	0.6212	1.0904	0.5632	0.5594	1.2615	0.6157
$23,838$	0.5997	1.1179	0.5721			
$90,234$	0.5840	1.2057	0.5908	GCI = 5.27%	GCI = 5.79%	GCI = 5.28%
	Error% w.r.t. the finest mesh			Error% w.r.t. the extrapolated mesh		
$7,525$	6.3699	9.5629	4.6716	11.0476	13.5632	8.5269
$23,838$	2.6884	7.2821	3.1652	7.2042	11.3833	7.0814
$90,234$	− − −	− − −	− − −	4.3976	4.4233	4.0442

6.5. **Flexibility effect on flapping foil.** Two 2D simulations are conducted to investigate the effect of foil flexibility on both flow dynamics and power extraction in a flapping foil system. The first simulation involves a rigid body flapping foil, while the second one introduces a flexible flapping foil, characterized by parameters detailed in Table (1). The sinusoidal deformation of the leading edge is deliberately designed to modulate the cambering of the flapping foil with the flapping stroke, whether upward or downward. This deformation choice is informed by recommendations from Li et al. [63] and Hamada and Fürth [64], advocating for flexible flapping foil cambering to enhance the flapping foil's performance from propulsion and energy harvesting perspectives, respectively. This investigation not only underscores the imperative of employing the modified Overset solver for resolving flow dynamics around a flexible flapping foil but also delves into the flexibility of the foil's camber from a power extraction perspective. Consequently, these meticulously designed simulations served as an invaluable testbed for the enhanced Overset solver, ensuring its robustness and reliability in tackling complex fluid dynamics scenarios.

The simulations for both cases were carried out with fine mesh for a duration of 25 seconds, ensuring the attainment of a steady-state solution. The velocity contours during one cycle, shown in Figure (16), compare the flow dynamics between the rigid and flexible cases at different time intervals. This serves as a testament to the functionality of the modified solver, which adeptly handles both flow dynamics and the necessary deformation within the front mesh. In addition, the flexibility affects the leading-edge vortex as its size increases compared to the rigid case as shown in Figure (16) at $t/T = 0$, 0.5, and 1.0. This happens because the leading edge deformation increases the camber in the direction of flapping (i.e., cambering up during the upstroke phase and cambering down during the downstroke phase). This is in line with the results of Li et al. [63]; they explored the flexibility of the foil's camber with a focus on its implications for propulsion and the recommendations provided by Hamada and Fürth [64] after investigating the effect of the foil shape on energy harvesting using a flapping foil in swing-arm mode. In addition, the variation in the pressure distribution over the foil's surface is shown in Figure (17), reflecting the flexibility effect. The envelope size of the pressure coefficient distribution at $t/T = 0.0$, 0.5, and 1.0 is larger in the flexible foil than that of the rigid one. This reflects the increase in the strength of the leading-edge vortex in the flexible case. Although the pressure coefficient distribution's envelope size decreases at $t/T = 0.25$, and 0.75, its impact on power extraction is negligible. This insignificance arises because the foil takes a horizontal orientation during these instances, where power generation is minimal, even in the rigid case. These observed flow dynamics demonstrate a remarkable enhancement in power extraction in the flexible case, as indicated by the temporal variations of key parameters, including lift, drag, moment, and power coefficients (C_l, C_d, C_m, and COP, respectively), shown in Figure (18). Furthermore, the statistics for these crucial coefficients ($\overline{C_d}$, RMS(C_l), RMS(C_m), $\overline{COP}$, and η_p) were calculated to provide further insight.

The temporal lift coefficient $C_l(t)$ of the flexible flapping foil experiences a tendency to be flattened by possessing an oscillation in its peaks, resulting in decreasing its root-mean-square and increasing its temporal average with respect to the rigid case by $\Delta RMS(C_l) = 0.287$ and $\Delta\overline{C_l} = 0.016$, respectively. The temporal drag coefficient $C_d(t)$ is enhanced as its sinusoidal amplitude is reduced compared to the rigid case. This results in a decrease in its mean value by $\Delta\overline{C_d} = 0.058$. Moreover, the strength of the moment coefficient non-linearity (small oscillation) that happens near the mid-stroke ($t/T = 0.0$, $t/T = 0.5$, and $t/T = 1.0$) is decreased due to the flexibility. Furthermore, the degree of non-linearity in the moment coefficient, characterized by small oscillations near the mid-stroke ($t/T = 0.0$, $t/T = 0.5$, and $t/T = 1.0$), is reduced as a result of the foil's flexibility. However, the moment coefficient amplitude increases slightly, resulting in a negligible increase in its root-mean-square with respect to the rigid case by $\Delta RMS(C_m) = 0.004$. This culminates in an increase in the temporal averaged power coefficient by $\Delta\overline{COP} = 0.072$. Hence, the power extraction efficiency (η_P) is enhanced from 0.073 for the rigid case to 0.156 for the flexible case, representing a boosting factor of 2.144 with the leading-edge (camber) flexibility.

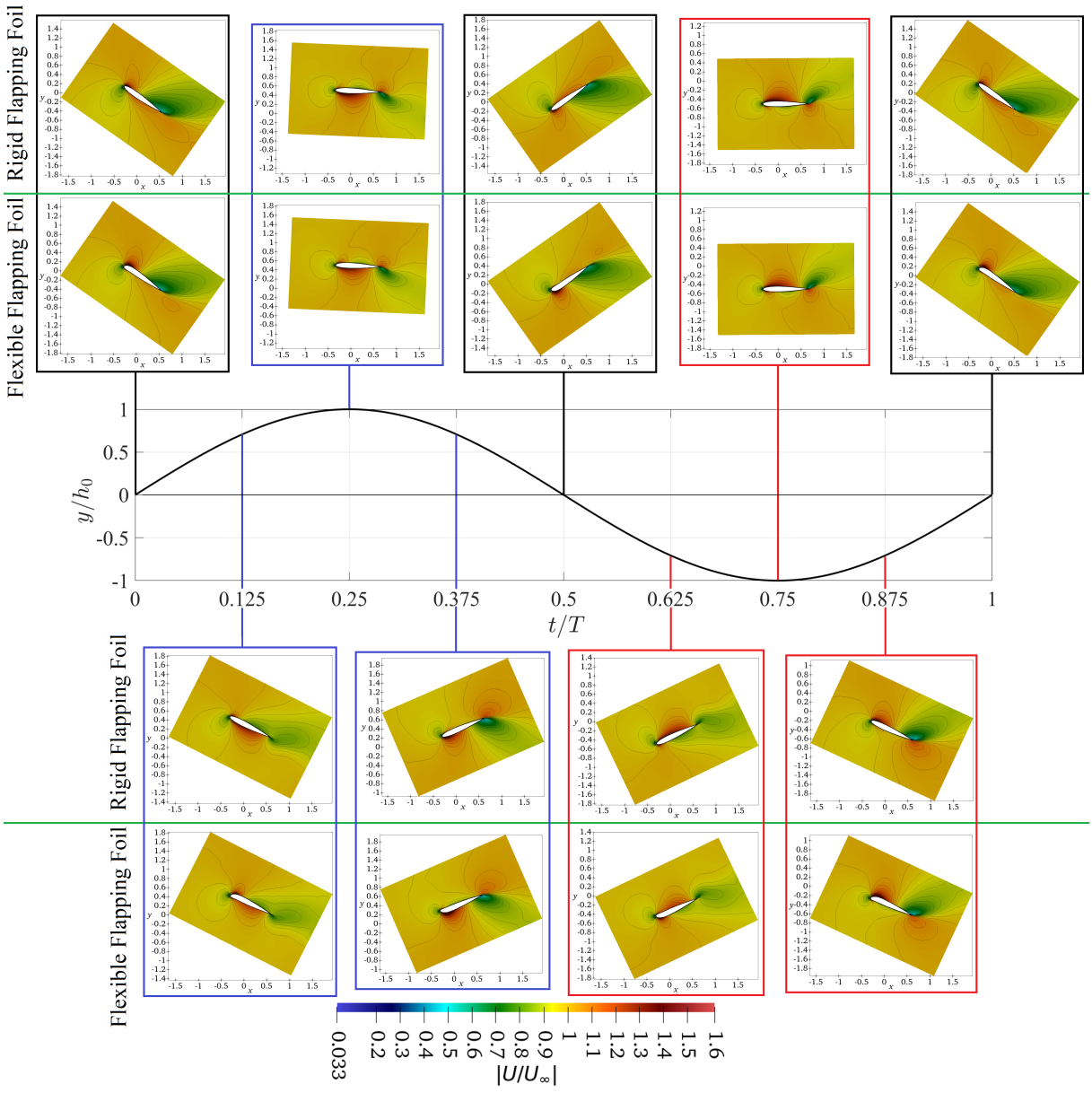


Figure 16. Flow-field velocity contours using the modified Overset solver for both the rigid and flexible airfoils.

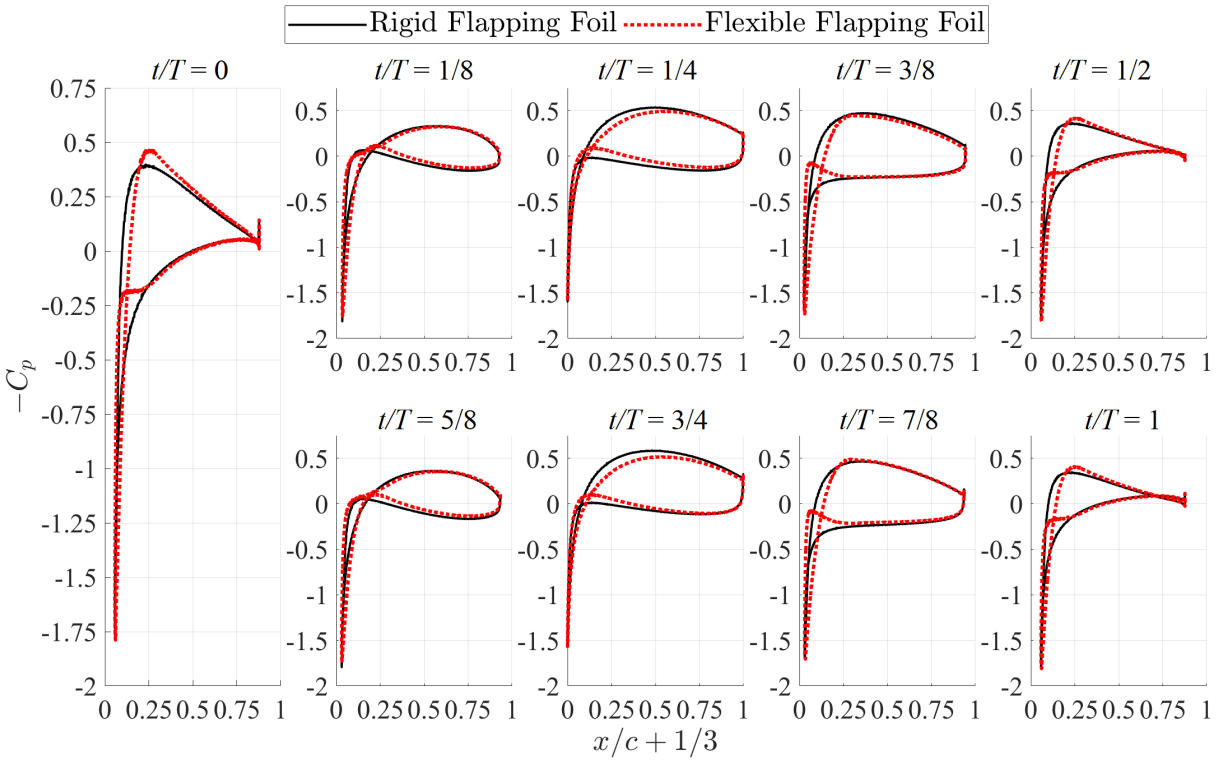


Figure 17. Pressure distribution using the modified Overset solver for both the rigid and flexible airfoils.

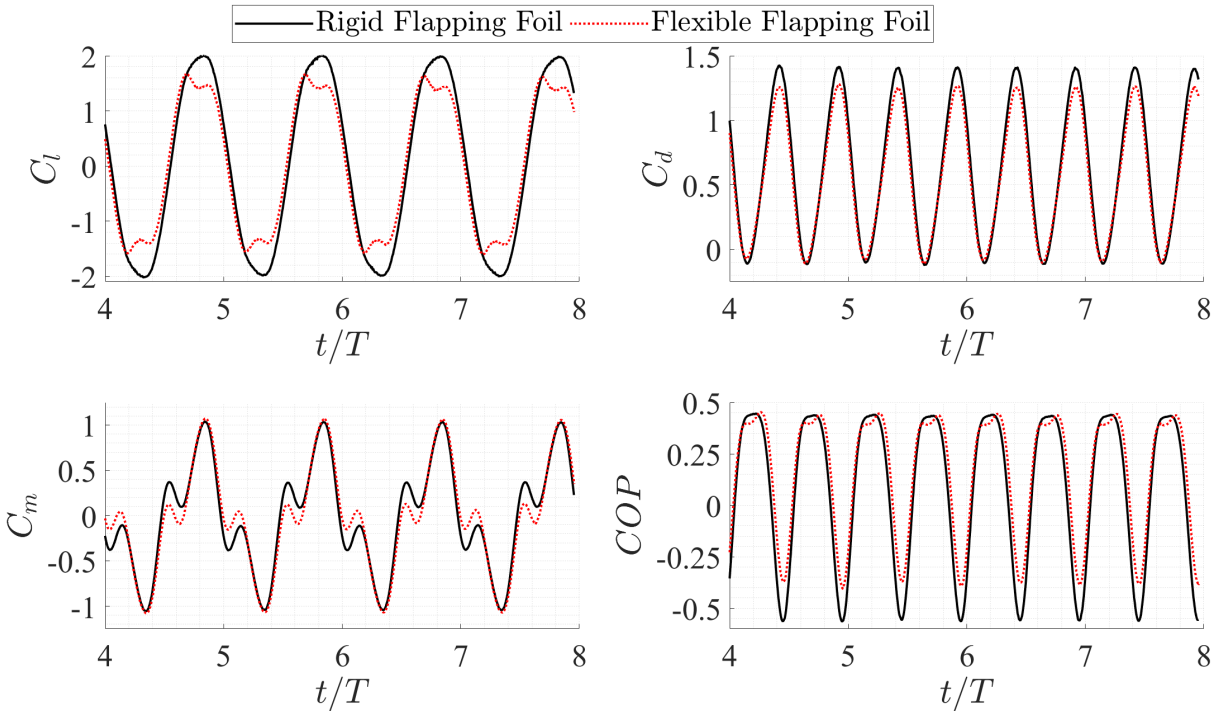


Figure 18. Temporal lift, drag, moment, and power coefficients distribution using the modified Overset solver for both the rigid and flexible airfoils.

7. Conclusion

The process of numerically studying the motion of some biological systems in fluid flows starts with having an adequate numerical tool to solve these complex FSI problems. This numerical tool can be

divided into mesh motion utility, fluid solver, and structure solver. In the presented work, the Overset technique was selected to be the mesh motion utility, addressing the dynamic mesh motion. Although OpenFOAM's Overset method was widely used, there are still some types of motion that it cannot address. The conventional Overset method allowed for solid body motion for the front mesh or the background mesh. In addition, it allowed for the displacement of any of the patches in the front mesh. However, it was not possible to have a combined motion that consists of displacement for a given patch in the front mesh while having a solid-body motion at the same time. For this purpose, a new mesh deformation solver was introduced. The objective of this study was to demonstrate the ability of *dynamicOversetZoneFvMesh* to apply both rigid body motion and deformation at specified patches within certain zones. Additionally, it aimed to conduct a numerical investigation of a leading-edge flexible flapping foil, aligning the foil's camber with the flapping direction. Verification, parallelization, validation, and mesh convergence studies were conducted prior to the numerical investigation, verifying the modified Overset solver. Two cases were meticulously simulated, allowing for a comparative analysis between the original and modified versions of the *dynamicOverset* libraries. The first case was a rigid body flapping foil with no deformation, while the second case featured shape flexibility during the flapping cycle characterized by leading-edge deformation. The results obtained with the modified *dynamicOversetZoneFvMesh* solver stood as compelling evidence of the proficiency of the adapted solver, as it adeptly managed complex fluid dynamics while effectively accommodating the requisite motion and deformations within the front mesh and keeping the background stationary as intended. Furthermore, the numerical investigation showed that leading-edge flexibility improves the power extraction efficiency of flapping foil, in agreement with the conclusions of Li et al. [63] and Hamada and Fürth [64]. The presented work can be a cornerstone for many problems in the field of fluid-structure interaction which may have been impossible to simulate so far.

Acknowledgements

This project has received funding from the European Union's Horizon 2020 research and innovation programme under Grant agreement No 101035803 : EC2U.

Author Contributions: Conceptualisation, K.A., A.A.H., L.C. and M.F.; methodology, K.A. and A.A.H.; software, K.A. and A.A.H.; validation, K.A. and A.A.H.; formal analysis, K.A. and A.A.H.; investigation, K.A. and A.A.H.; resources, L.C. and M.F.; data curation, K.A., A.A.H., L.C. and M.F.; writing—original draft preparation, K.A. and A.A.H.; writing—review and editing, K.A., A.A.H., L.C. and M.F.; visualisation, K.A. and A.A.H.; supervision, L.C. and M.F.; project administration, L.C. and M.F.; funding acquisition, L.C. and M.F. All authors have read and agreed to the published version of the manuscript.

References

[1] J. Rose, S. G. Natarajan, and V. Gopinathan, "Biomimetic flow control techniques for aerospace applications: a comprehensive review," *Reviews in Environmental Science and Bio/Technology*, vol. 20, no. 3, pp. 645–677, 2021.

[2] F. Xie, Q. Zuo, Q. Chen, H. Fang, K. He, R. Du, Y. Zhong, and Z. Li, "Designs of the biomimetic robotic fishes performing body and/or caudal fin (bcf) swimming locomotion: a review," *Journal of Intelligent & Robotic Systems*, vol. 102, no. 1, pp. 1–19, 2021.

[3] H. Alqaleiby, M. Ayyad, M. R. Hajj, S. A. Ragab, and L. Zuo, "Effects of piezoelectric energy harvesting from a morphing flapping tail on its performance," *Applied Energy*, vol. 353, p. 122022, 2024.

[4] W. Ibrahim, M. Mohamed, R. Ismail, P. Leung, W. Xing, and A. Shah, "Hydrokinetic energy harnessing technologies: A review," *Energy Reports*, vol. 7, 2021.

[5] E. Farrell Helbling and R. J. Wood, "A review of propulsion, power, and control architectures for insect-scale flapping-wing vehicles," *Applied Mechanics Reviews*, vol. 70, no. 1, 2018.

[6] K. Hargroves and M. H. Smith, "Innovation inspired by nature biomimicry," *Ecos*, no. 129, pp. 27–30, 2006.

[7] E. Jamei and Z. Vrcelj, "Biomimicry and the built environment, learning from nature's solutions," *Applied Sciences*, vol. 11, no. 16, p. 7514, 2021.

[8] M. Rufo and M. Smithers, "Ghostswimmer™ auv: Applying biomimetics to underwater robotics for achievement of tactical relevance," *Marine Technology Society Journal*, vol. 45, no. 4, 2011.

[9] B. M. Kulfan and A. J. Colozza, "Biomimetics and flying technology," *Biomimetics: Nature-Based Innovation, edited by Bar-Cohen, Y., CRC Press, Taylor & Francis Group, Boca Raton, FL*, pp. 525–674, 2011.

[10] S. Budholiya, A. Bhat, S. A. Raj, M. T. Hameed Sultan, A. U. Md Shah, and A. A Basri, "State of the art review about bio-inspired design and applications: An aerospace perspective," *Applied Sciences*, vol. 11, no. 11, p. 5054, 2021.

[11] A. A. Hamada and M. Fürth, "Development of a finite element solver including a level-set method for modeling hydrokinetic turbines," in *ASME Fluids Engineering Division Summer Meeting*, vol. 87877. American Society of Mechanical Engineers, 2022.

[12] ——, "Numerical investigation of the energy harvesting capabilities of naca series flapping foil turbines in swing-arm mode," *SSRN Electron. J.*, 2022.

[13] C. W. Hirt, A. A. Amsden, and J. Cook, "An arbitrary lagrangian-eulerian computing method for all flow speeds," *Journal of computational physics*, vol. 14, no. 3, pp. 227–253, 1974.

[14] A. A. Hamada and M. Fürth, "Ground effect on current energy harvesting from a freely-oscillating circular cylinder at low reynolds number," in *International Conference on Offshore Mechanics and Arctic Engineering*, vol. 85192. American Society of Mechanical Engineers, 2021, p. V009T09A003.
[15] T. Jardin, A. Farcy, and L. David, "Three-dimensional effects in hovering flapping flight," *Journal of fluid mechanics*, vol. 702, pp. 102–125, 2012.
[16] H. Jasak and Ž. Tuković, "Dynamic mesh handling in openfoam applied to fluid-structure interaction simulations," in *Proceedings of the V European Conference on Computational Fluid Dynamics ECCOMAS CFD 2010*, 2010.
[17] M. Page, M. Beaudoin, and A.-M. Giroux, "Steady-state capabilities for hydroturbines with openfoam," *International Journal of Fluid Machinery and Systems*, vol. 4, no. 1, pp. 161–171, 2011.
[18] B. Sangolola and S. Lakey, "Commercial cfd software application to unsteady aerodynamics of oscillating aerofoils," in *24th AIAA Applied Aerodynamics Conference*, 2006, p. 3851.
[19] L. Li, S. Sherwin, and P. W. Bearman, "A moving frame of reference algorithm for fluid/structure interaction of rotating and translating bodies," *International Journal for Numerical Methods in Fluids*, vol. 38, no. 2, pp. 187–206, 2002.
[20] B. Zouzou, I. Dobrev, F. Massouh, and R. Dizene, "Experimental and numerical analysis of a novel darrieus rotor with variable pitch mechanism at low tsr," *Energy*, vol. 186, p. 115832, 2019.
[21] A. A. Hamada and M. Fürth, "Modeling and analysis of the leading-edge vortex on flapping foil turbines in swing-arm mode," *Journal of Fluids Engineering*, vol. 145, no. 6, p. 061105, 2023.
[22] K. Lee, M. Huang, N. Yu, and P. Rubbert, "Grid generation for general three-dimensional configurations," *NASA. Langley Research Center Numerical Grid Generation Tech.*, 1980.
[23] T. J. Baker, "Mesh generation: Art or science?" *Progress in Aerospace Sciences*, vol. 41, no. 1, pp. 29–63, 2005.
[24] D. D. Chandar, B. Boppana, and V. Kumar, "A comparative study of different overset grid solvers between openfoam, starccm+ and ansys-fluent," in *2018 AIAA Aerospace Sciences Meeting*, 2018, p. 1564.
[25] J. Cai, H. M. Tsai, and F. Liu, "A parallel viscous flow solver on multi-block overset grids," *Computers & fluids*, vol. 35, no. 10, pp. 1290–1301, 2006.
[26] M. Jarkowski, M. Woodgate, G. Barakos, and J. Rokicki, "Towards consistent hybrid overset mesh methods for rotorcraft cfd," *International Journal for Numerical Methods in Fluids*, vol. 74, no. 8, pp. 543–576, 2014.
[27] OpenCFD, "Openfoam official website," http://www.openfoam.com, 2022, accessed: 2022-09-23.
[28] ——, "Opencfd release openfoam v1706," https://www.openfoam.com/news/main-news/openfoam-v1706, 2017, accessed: 2022-09-23.
[29] ——, "Overset extended features," https://www.openfoam.com/news/main-news/openfoam-v1812/numerics#numerics-overset, 2018, accessed: 2022-09-23.
[30] ——, "Overset new solvers," https://www.openfoam.com/news/main-news/openfoam-v2106/solver-and-physics, 2021, accessed: 2022-09-23.
[31] P. Laws, J. S. Saini, and A. Kumar, "A study on openfoam's overset mesh support using flow past naca 0018 airfoil," 2019, preprint on webpage at https://www.preprints.org/manuscript/201907.0217/download/final_file.
[32] Siemens, "Starccm+ official website," https://mdx.plm.automation.siemens.com/star-ccm-plus, 2022, accessed: 2022-09-23.
[33] ANSYS, "Ansys fluent official website," http://www.ansys.com/Products/Fluids/ANSYS-Fluent, 2022, accessed: 2022-09-23.
[34] S. R. Vatne, "Aeroelastic instability and flutter for a 10 mw wind turbine," Master's thesis, Institutt for energi-og prosessteknikk, 2011.
[35] D. Ramdenee*, A. Ilinca, S. l. Minea, and I. Hussein, "Modelling of aerodynamic flutter on a naca 4412 airfoil with application to wind turbine blades," *International Journal of Simulation and Process Modelling*, vol. 8, no. 1, pp. 79–87, 2013.
[36] M. A. Shabara, S. Zou, and O. Abdelkhalik, "Numerical investigation of a variable-shape buoy wave energy converter," in *International Conference on Offshore Mechanics and Arctic Engineering*, vol. 85192. American Society of Mechanical Engineers, 2021, p. V009T09A013.
[37] M. A. Shabara and O. Abdelkhalik, "Dynamic modeling and vibrations of variable-shape wave energy converters," 2022. [Online]. Available: https://arxiv.org/abs/2210.17297
[38] ——, "On the dynamics of variable shape wave energy converters," 2022. [Online]. Available: https://arxiv.org/abs/2201.08942
[39] Y. Zhang, Y. Wang, Y. Xie, G. Sun, and J. Han, "Effects of flexibility on energy extraction performance of an oscillating hydrofoil under a semi-activated mode," *Energy*, vol. 242, p. 122940, 2022.
[40] J. Wu, C. Shu, N. Zhao, and F.-B. Tian, "Numerical study on the power extraction performance of a flapping foil with a flexible tail," *Physics of Fluids*, vol. 27, no. 1, p. 013602, 2015.
[41] Y. Luo, Q. Xiao, G. Shi, G. Pan, and D. Chen, "The effect of variable stiffness of tuna-like fish body and fin on swimming performance," *Bioinspiration & biomimetics*, vol. 16, no. 1, p. 016003, 2020.
[42] M. S. U. Khalid, J. Wang, I. Akhtar, H. Dong, M. Liu, and A. Hemmati, "Larger wavelengths suit hydrodynamics of carangiform swimmers," *Physical Review Fluids*, vol. 6, no. 7, p. 073101, 2021.
[43] W. Wang, Z. Liu, Y. Jin, and Y. Cheng, "Lbm simulation of droplet formation in micro-channels," *Chemical Engineering Journal*, vol. 173, no. 3, pp. 828–836, 2011.
[44] K. Mohammadi, M. R. Movahhedy, and S. Khodaygan, "A multiphysics model for analysis of droplet formation in electrohydrodynamic 3d printing process," *Journal of Aerosol Science*, vol. 135, pp. 72–85, 2019.
[45] E. Atta, "Component-adaptive grid interfacing," in *19th Aerospace Sciences Meeting*, 1981, p. 382.
[46] J. Benek, J. Steger, and F. C. Dougherty, "A flexible grid embedding technique with application to the euler equations," in *6th computational fluid dynamics conference Danvers*, 1983, p. 1944.
[47] J. Benek, P. Buning, and J. Steger, "A 3-d chimera grid embedding technique," in *7th Computational Physics Conference*, 1985, p. 1523.

[48] J. Benek, T. Donegan, and N. Suhs, "Extended chimera grid embedding scheme with application to viscous flows," in *8th Computational Fluid Dynamics Conference*, 1987, p. 1126.
[49] P. Tisovska, "Description of the overset mesh approach in esi version of openfoam," *Proceedings of the CFD with OpenSource Software*, 2019.
[50] W. Liu, Q. Xiao, and F. Cheng, "A bio-inspired study on tidal energy extraction with flexible flapping wings," *Bioinspiration & biomimetics*, vol. 8, no. 3, p. 036011, 2013.
[51] OpenCFD, "Openfoam: solidbodydisplacementlaplacianfvmotionsolver.h file reference," https://www.openfoam.com/documentation/guides/latest/api/solidBodyDisplacementLaplacianFvMotionSolver_8H.html, 2021, accessed: 2022-09-23.
[52] ——, "Openfoam: overset directory reference," https://www.openfoam.com/documentation/guides/latest/api/dir_130ea3eb54766613bed36c4f96e55285.html, 2021, accessed: 2022-09-23.
[53] ——, "Openfoam: dynamicmotionsolverlistfvmesh class reference," https://www.openfoam.com/documentation/guides/latest/api/classFoam_1_1dynamicMotionSolverListFvMesh.html, 2021, accessed: 2022-09-23.
[54] ——, "Openfoam: dynamicmultimotionsolverfvmesh.h file reference," https://www.openfoam.com/documentation/guides/latest/api/dynamicMultiMotionSolverFvMesh_8H.html, 2021, accessed: 2022-09-23.
[55] ——, "Openfoam: Api guide v2112 - interpolated faces, zoneid," https://www.openfoam.com/documentation/guides/latest/api/interpolatedFaces_8H.html#ae7f6f504738f059d42ea1822a9f09447, 2021, accessed: 2022-09-23.
[56] ——, "Openfoam: dynamicoversetfvmesh class reference," https://www.openfoam.com/documentation/guides/latest/api/classFoam_1_1dynamicOversetFvMesh.html, 2021, accessed: 2022-09-23.
[57] ——, "Openfoam: dynamicfvmesh class reference," https://www.openfoam.com/documentation/guides/latest/api/classFoam_1_1dynamicFvMesh.html, 2021, accessed: 2022-09-23.
[58] S. Patankar, *Numerical heat transfer and fluid flow*. Taylor & Francis, 2018.
[59] J. H. Ferziger, M. Perić, and R. L. Street, *Computational methods for fluid dynamics*. Springer, 2002, vol. 3.
[60] W. Dynamics, "Dynamic meshes in openfoam," https://www.wolfdynamics.com/training/movingbodies/OF2021/dynamicmeshes_2021_OF8.pdf, 2021, accessed: 2023-12-27.
[61] L. F. Richardson, "Ix. the approximate arithmetical solution by finite differences of physical problems involving differential equations, with an application to the stresses in a masonry dam," *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, vol. 210, no. 459-470, pp. 307–357, 1911.
[62] I. B. Celik, U. Ghia, P. J. Roache, and C. J. Freitas, "Procedure for estimation and reporting of uncertainty due to discretization in cfd applications," *Journal of fluids Engineering-Transactions of the ASME*, vol. 130, no. 7, 2008.
[63] Y. Li, Z. Pan, and N. Zhang, "Propulsive properties of a flexible oscillating wing with time-varying camber deformation," *Ocean Engineering*, vol. 235, p. 109332, 2021.
[64] A. A. Hamada and M. Fürth, "Effect of flapping foil shape on power extraction from hydrokinetic turbines in swing-arm mode," 2024, manuscript submitted for publication.