OpenFOAM® Journal

Volume 1 [Full Papers], Pages 47–61
ISSN: 2753-8168

OpenFOAM FOR FRANCIS TURBINE TRANSIENTS

SAEED SALEHI* AND HÅKAN NILSSON

Division of Fluid Dynamics, Department of Mechanics and Maritime Sciences, Chalmers University of Technology, Gothenburg SE-412 96, Sweden
Email address: saeed.salehi@chalmers.se

Division of Fluid Dynamics, Department of Mechanics and Maritime Sciences, Chalmers University of Technology, Gothenburg SE-412 96, Sweden
Email address: hakan.nilsson@chalmers.se

DOI: 10.51560/ofj.v1.26
Version(s): OpenFOAM v1912
Repo: https://doi.org/10.7910/DVN/31JGOM

Abstract. The flexibility and fast responsiveness of hydropower systems make them a reliable solution to overcome the intermittency of renewable energy resources and balance the electrical grid. Therefore, investigating the complex flow fields during such operation is essential to increase the reliability and lifetime of future hydropower systems. The current article concerns the utilization of OpenFOAM for the numerical study of Francis turbines during transient load change operations. The details of employed models and numerical schemes are thoroughly explained. The Laplacian smoothing algorithm is applied for the deformation of the guide vane domain. For the first time, The impact of different mesh diffusivity parameters on both load rejection and acceptance operations is studied. It also is shown that general slip boundary conditions cannot be used for slipping points on the guide vane upper and lower surfaces. Instead, different alternatives are introduced and compared. The developed framework is tested on a high-head Francis turbine. Different transient operations are simulated and results are compared to the experimental data. It is shown that OpenFOAM can be employed as a trustworthy CFD solver for numerical investigation of Francis turbines transient operations.

1. Introduction

Intermittency of renewable energy resources, such as wind and solar, could potentially result in an unstable electricity grid. Flexibility and fast responsiveness of the hydropower systems have made them a sustainable solution for stabilizing the power [1, 2]. Therefore, the fast growth of wind and solar electrical energy production caused hydraulic turbines to work in transient operations more often. The transient operation in hydraulic turbines usually refers to the change in operating conditions. For instance, if there is a need for producing more power, the guide vanes open up, the flow rate grows, and thus the turbine output power increases which is usually known as load acceptance operation. In contrast, the load rejection operation closes down the guide vanes to decrease the flow rate and the output power.

The transients operations usually generate large fluctuations in local and integrated flow quantities such as pressure and runner torque that could potentially deteriorate turbine lifetime [3, 2]. Although the importance of understanding transient phenomena of hydraulic turbines is ever-increasing, complex flow fields during transient procedures are still not well perceived and require more in-depth studies.

Accurate experimental measurements are often expensive and time-consuming. Computational Fluid Dynamics (CFD) is a reliable alternative to study the transient flow field of hydraulic turbines during load change operations. A few authors have employed proprietary CFD codes to investigate load change operation of Francis turbines [4, 5, 6, 7, 8, 9, 10, 11]. However, to the best of our knowledge, no study is reported in the literature on the investigation of load change operation using an open-source CFD code, except for our previous studies [12, 13, 14]. For more information, the readers are referred to Salehi et al. [12].

Employing OpenFOAM as a reliable and flexible open-source CFD tool for studying transient flow fields during load changes can be quite advantageous. The current paper signifies that OpenFOAM can be an accurate and trustworthy alternative to proprietary CFD codes for the assessment of complex

* Corresponding author
Received: 24 June 2021, Accepted: 29 October 2021, Published: 17 November 2021

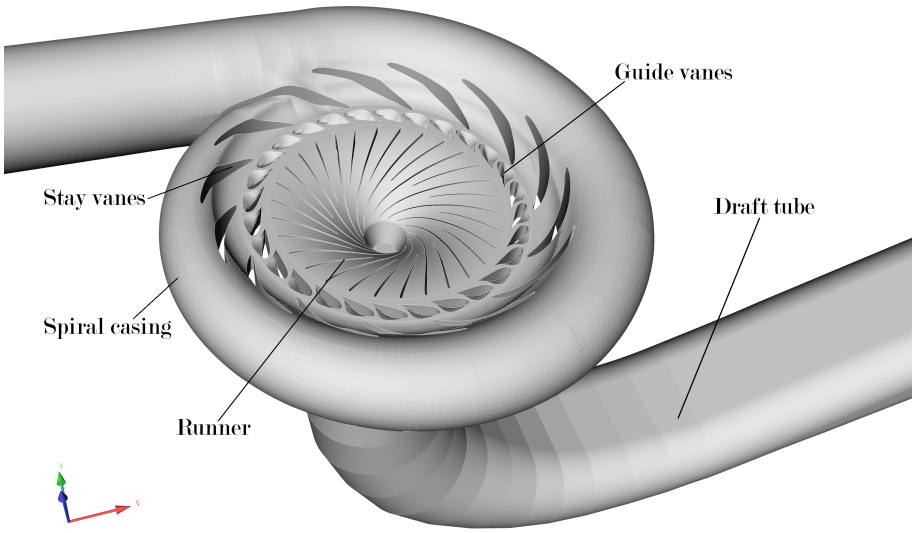


FIGURE 1. The full assembly of Francis-99 model turbine

transient load change phenomena in Francis turbines. Detailed descriptions on OpenFOAM simulation of Francis turbine load change are provided. For the first time, the effect of different diffusivity parameters on the mesh morphing procedure of the guide vane domain during both load rejection and acceptance is studied in detail. Additionally, the slipping of the mesh points on the lower and upper flat surfaces of the guide vane domain is carefully investigated and discussed. The high-head Francis-99 turbine is employed as a test case to evaluate the performance of the employed numerical framework.

2. Investigated test case

The Francis-99 high-head model turbine, provided by the Francis-99 workshop series [15] is utilized as a test case to study the performance of the adopted numerical framework for predictions of Francis turbine load changes. The model is integrated with 14 stay vanes in the spiral casing, 28 guide vanes, 30 runner blades (15 main blades and 15 splitters). The model head at BEP condition is about $H \approx 12$ m. The full model of the turbine, including spiral casing, guide vanes, runner, and draft tube is considered in this study (Fig. 1).

The computational mesh consists of four different domains whose corresponding mesh are generated separately. The spiral casing, runner, and draft tube domains are meshed in ICEM-CFD, while the guide vane mesh is generated by TurboGrid. The meshes are converted into OpenFOAM format and then merged together to construct the full model shown in Fig. 2. The model contains nearly 16 million hexahedral cells. More information about the produced mesh is provided by Salehi et al. [12].

Fig. 3 displays the y-normal section of the Francis-99 turbine. The vaneless space pressure is measured at the VL2 probe, while in-plane velocity components are measured at the PIV plane on three different lines.

Two load change operations are investigated in this work, namely, load rejection, Best Efficiency Point (BEP) to Part Load (PL), and load acceptance, BEP to High Load (HL). Both transient procedures start from the BEP condition. The guide vane angles are changed (close down or open up) and consequently the flow rate changes. The guide vane angles at PL, BEP, and HL are $\alpha_{\mathrm{PL}} = 6.72°$, $\alpha_{\mathrm{BEP}} = 9.84°$, and $\alpha_{\mathrm{HL}} = 12.43°$.

3. Mathematical formulation

The Unsteady Reynolds-Averaged Navier-Stokes (URANS) equations, which governs transient incompressible turbulent flows, are stated as

$$\frac{\partial U_j}{\partial x_j} = 0, \tag{1}$$

$$\frac{\partial U_i}{\partial t} + \frac{\partial (U_i U_j)}{\partial x_j} = \frac{-1}{\rho}\frac{\partial p}{\partial x_i} + \frac{\partial}{\partial x_j}\left(\nu \frac{\partial U_i}{\partial x_j} - \overline{u_i u_j}\right). \tag{2}$$

The current study employs the Shear Stress Transport (SST) based Scale-Adaptive Simulation (SAS) URANS model [16, 17] (i.e., `kOmegaSSTSAS`) to compute of the unknown Reynolds stress tensor $-\rho\overline{u_i u_j}$. SST-SAS is a turbulence-resolving URANS model, used for the simulation of practical transient flows.

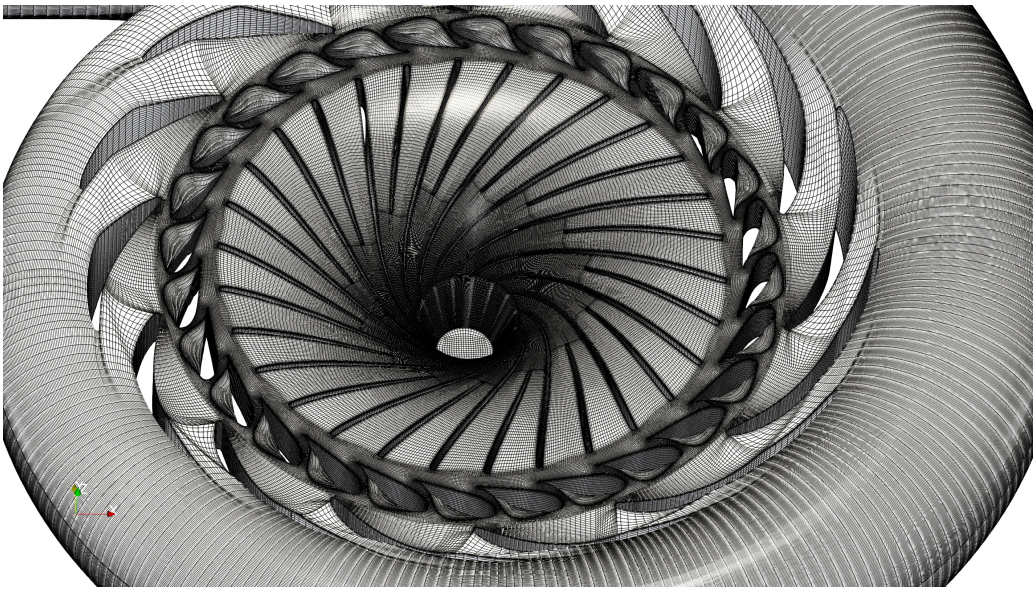


FIGURE 2. Zoomed view of the computational mesh at BEP condition, showing the spiral casing, the guide vanes and the runner (the draft tube mesh is now shown here).

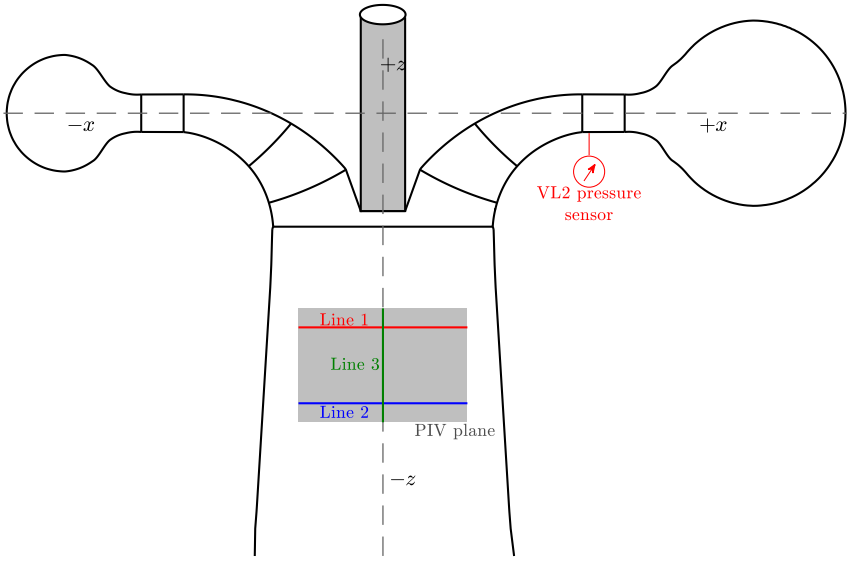


FIGURE 3. Two sections of the Francis-99 model, showing PIV plane, velocity lines, pressure sensors

The formulation of this model introduces an additional source term in the turbulent eddy frequency (ω) equation, which decreases the local eddy viscosity to resolve the turbulent spectrum and break-up of large eddies, providing LES-like solutions. A variety of studies investigated and verified the performance of the SST-SAS model for simulations of hydraulic machinery flows [18, 19, 20, 21, 22, 9, 10].

4. Computational framework

This section explains the details of the employed computational framework, i.e., discretization schemes, pressure-velocity coupling, boundary conditions, linear solvers, and parallel processing.

4.1. **Discretization schemes.** The utilized discretization schemes are elaborated upon in the following subsections.

4.1.1. *Temporal schemes.* The temporal derivatives in all transport equations are approximated using the implicit second-order backward scheme (i.e., `backward`). The time-step is chosen to $\Delta t = 1.25 \times 10^{-4}$ s, based on the rotational speed of the runner, which is common practice in CFD simulations

of turbomachinery flows. In the current test case, the runner is rotating at a constant angular speed of 332.59 rpm. Accordingly, the runner rotates 0.25° in each time-step, which is sufficiently small to capture most physical phenomena. Moreover, the guide vanes change their angle by 1.625×10^{-4} ° each time-step. The blade passing frequency (rotational frequency of the runner times number of the blades) is $f_{\mathrm{b}} = 166.3$, while the time-step frequency is 8 kHz. Hence, the high-frequency pulsations due to rotor-stator interaction should be easily captured.

The mean and maximum Courant-Friedrichs-Lewy (CFL) [23] numbers at BEP are 0.025 and 55, respectively. The high CFL values correspond to a very small portion of the domain where that cells are very small because of the complexity of the geometry. In fact, 99.38% of the cells have CFL numbers less than 2.

4.1.2. *Gradient schemes.* The second-order `linear` scheme is usually employed in OpenFOAM for the discretization of gradient schemes. Here, the `cellLimited` approach is utilized to improve the stability and boundedness of the gradient terms, due to the high skewness of the cells in some regions, which is inevitable because of the complex geometry.

4.1.3. *Divergence schemes.* The Linear-Upwind Stabilised Transport (LUST) scheme [24] is used to discretize the convective terms in the momentum equation, which is an unbounded discretization scheme that blends the central and second-order linear upwind schemes (i,e., `linear` and `linearUpwind`) using a weighted averaging, as

$$\phi_{\mathrm{LUST}} = \gamma\phi_{\mathrm{c}} + (1-\gamma)\phi_{\mathrm{lu}}. \tag{3}$$

In Eq. (3), ϕ_{c} and ϕ_{lu} represent the face-centered values approximated using the central and linear upwind schemes, respectively. The blending factor, γ , is hardcoded 0.75 in the OpenFOAM `LUST` scheme, indicating that the face values are computed blending 75% second-order central and 25% second-order linear upwind schemes. Therefore, the method enhances the numerical stability significantly. In most practical engineering flows with complex geometries the pure central scheme cannot be used due to numerical instabilities. Hence, LUST provides a proper alternative for turbulence resolving simulation of such flow fields.

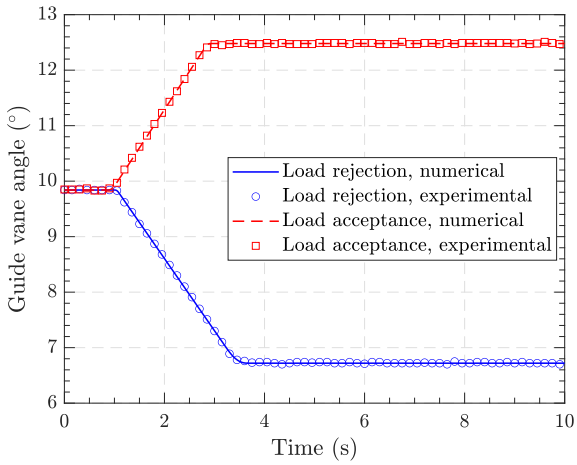
The second-order upwind discretization scheme is employed for approximations of other convective terms (i.e., k and ω).

4.1.4. *Laplacian schemes.* The second-order scheme with explicit non-orthogonal correction is employed for the calculation of the Laplacian terms. Here again, a limiter is applied because of the mesh quality in some regions. A limiter with a coefficient of 0.333 is utilized for the non-orthogonal correction part. Therefore, all the Laplacian terms are approximated using `Gauss linear limited corrected 0.333` scheme.

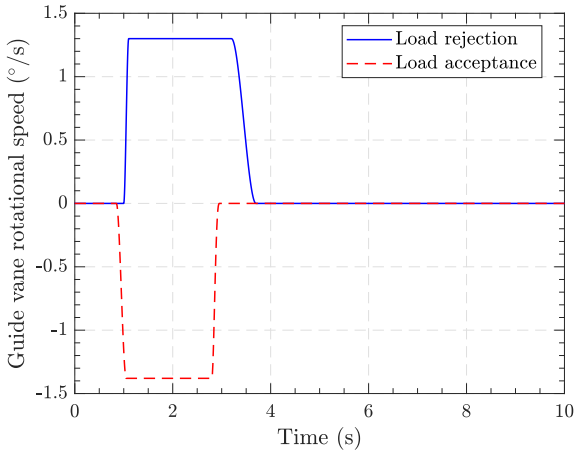
4.2. **Pressure-velocity coupling.** The CFD simulations are performed using the `pimpleFoam` solver, suggesting a correction of the pressure field through the PIMPLE algorithm. PIMPLE combines the two well-known pressure correction algorithms, SIMPLE [25] and PISO [26]. The SIMPLE algorithm works as an outer loop while the PISO algorithm performs inner correction loops. The maximum number of outer corrections is set to 10. Residual criteria of 10^{-5} and 10^{-6} for p and U, respectively, control the number of performed outer loops. In most time steps the pimple algorithm converges after five outer loops in the present case. Two inner corrections are carried out in each outer loop. In each inner loop, one additional non-orthogonal correction is considered to ensure convergence of all explicit parts in the discretized equations such as non-orthogonal correction. The calculations of the turbulence transport equations are carried out at the end of the final outer correction loop.

4.3. **Boundary conditions.** In the Francis-99 case, the guide vanes change their angle by 1.3°/s during the load change procedures. Fig. 4a compares the variation of the measured guide vane angle in the experiments with the applied ad-hoc smooth function for the numerical simulations. The smooth variations reduce the numerical instabilities at the starting and stopping processes of the guide vane rotation. A new boundary condition for the `pointDisplacement` field is implemented that receives the guide vane rotational speed as a `Func1` variable, which enables time-varying values for boundary conditions. The rotational speed is calculated as the time-derivative of the guide vane angle and is shown in Fig. 4b.

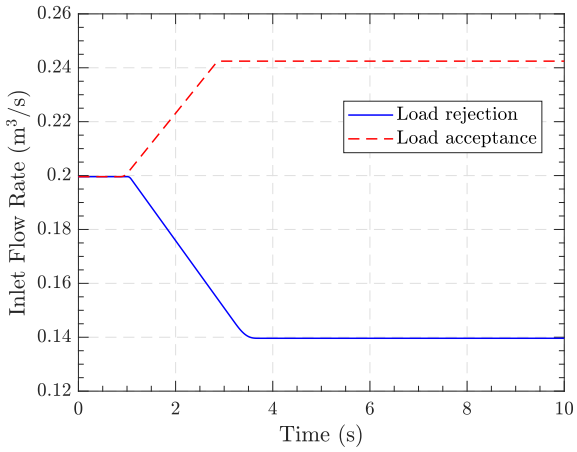
The inlet volume flow rate of the turbine is assumed to vary linearly with respect to the guide vane angle, as recommended by the Francis-99 workshops [15]. Consequently, a time-varying but spatially uniform velocity is imposed at the inlet of the spiral casing using a `csvFile` option in the `uniformFixedValue` boundary condition. A similar approach is applied for the inlet values of the turbulent kinetic energy, k, and the specific rate of dissipation, ω, assuming a fixed turbulence intensity ($I = 10\%$) and viscosity



(A) Guide vane angle



(B) Guide vane rotational speed



(C) Volume flow rate

FIGURE 4. Variation of (a) guide vane angle, (b) guide vane rotational speed, and (c) turbine volume flow rate for both investigated cases

ratio ($\nu_t/\nu = 100$). The `inletOutlet` condition is imposed at the outlet for the velocity, with an inlet value of `uniform (0 0 0)` to prevent incoming backflow.

The couplings between the domains (spiral casing, guide vane, runner, and draft tube) are realized through the Cyclic Arbitrary Mesh Interface (`cyclicAMI`) boundary condition [27, 28].

Both transient cases start from the stationary BEP condition. Therefore, before the commencement of the transient procedures, a statistically stationary flow condition at BEP should be achieved. For a faster convergence, a steady potential flow solution was computed and used as an initial condition. Afterward, the unsteady turbulent flow at the BEP condition is simulated for 2 s flow time, corresponding to 11 runner rotations. Subsequently, the procedures plotted in Fig. 4 started.

4.4. **Linear solvers.** The Geometric Agglomerated Algebraic Multi-Grid (`GAMG`) linear solver with the combined DIC/Gauss-Seidel smoother is employed to solve the pressure and Laplacian dynamic mesh equations. The solutions of the linear systems of velocity, turbulent kinetic energy, and specific dissipation rate are obtained with the iterative `smoothSolver` with the Gauss-Seidel smoother.

4.5. **Parallel processing.** The scotch [29] domain decomposition approach is used to split the computational domain and distribute roughly equal loads to the processors while minimizing their interconnections. The job is submitted to a Linux cluster using 320 CPU cores. Each simulation took around 18 days (440 hours) to complete. Therefore, the computational cost of each transient simulation was more than 140,000 core hours.

5. Mesh motion framework

Simulations of Francis turbine load change procedures involve two different mesh motions, i.e, rotation of the runner domain and mesh deformation of the guide vane domain (please have a look at Figs. 1 and 2 to see the runner and guide vane domains of the current case study). Such type of simultaneous mesh motions can be carried out in OpenFOAM using the `dynamicMultiMotionSolverFvMesh` class, which enables utilization of different `fvMotionSolver`s on each cell zone. It applies all the prescribed mesh motions on the corresponding cell zones in a sequence using a `forAll` loop. Accordingly, the mesh in the runner domain performs a solid-body rotation around the turbine axis using a combination of the `solidBodyMotionSolver` and `rotatingMotion` classes. Besides, a mesh morphing approach is employed to spread and smoothen the guide vane motion into the guide vane region mesh. For this purpose, the `displacementLaplacian` mesh motion solver is applied to this cell zone.

After calculation of the new locations of the mesh points, the points are moved and then the face fluxes are corrected using the face velocity which is computed based on the face swept volume. More details on the dynamic mesh implementation in OpenFOAM is provided by Jasak and Tuković [30] and Jasak [31].

In the OpenFOAM PIMPLE loop all the dynamic mesh calculations are performed in the first outer correction loop using the `controlledUpdate()` function of the `dynamicFvMesh` class.

5.1. **Mesh morphing.**

5.1.1. *Displacement Laplacian solver.* The `displacementLaplacian` mesh motion solver performs the mesh morphing procedure using the Laplacian smoothing equation

$$\nabla \cdot (\Gamma \nabla \boldsymbol{\delta}_{\text{cell}}) = \mathbf{0}, \tag{4}$$

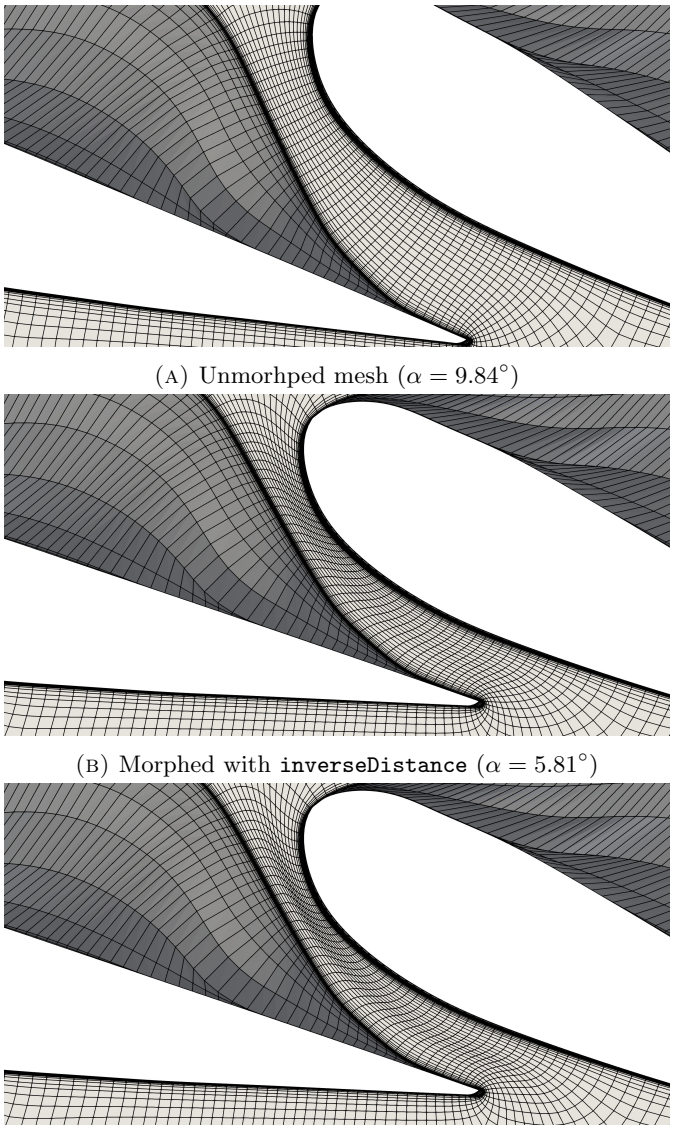
using a cell-centered finite volume approach. Herein, Γ is the motion diffusivity and $\boldsymbol{\delta}_{\text{cell}}$ is a vector field representing the spatial displacement of the cell centers (`cellDisplacement`). However, to update the mesh at each time step, the location of the new points are needed, meaning that one needs the displacement vector of the mesh points $\boldsymbol{\delta}_{\text{point}}$ (`pointDisplacement`). After obtaining the solution for $\boldsymbol{\delta}_{\text{cell}}$, the inverse distance weighting approach is adopted to interpolate the cell-centered displacement vectors onto the point locations and calculate $\boldsymbol{\delta}_{\text{point}}$. In contrast, the mesh motion boundary conditions (in this case the rotation of the guide vanes shown in Fig. 4b) are specified by the user on the `pointDisplacement` and they are automatically adapted as boundary conditions for the `cellDisplacement` equation. Finally, the new point locations are calculated by

$$\mathbf{x}^{t+\Delta t}_{\text{point}} = \mathbf{x}^{t}_{\text{point}} + \boldsymbol{\delta}^{t}_{\text{point}}. \tag{5}$$

5.1.2. *Motion diffusivity.* The motion diffusivity, Γ, controls how far the boundary movement is spread into the internal mesh. Different approaches are available in OpenFOAM to calculate this diffusivity, e.g., uniform and inverse distance. In order to have a smooth mesh morphing, the small cells close to the moving boundary (here guide vanes), should be able to move with the boundary with the least deformation, while the larger cells away from the surface can deform noticeably. Consequently, Γ should decrease with distance from the moving patch (here guide vanes), indicating that points close to the guide vanes move at nearly the same rate as the guide vane boundary points, while the points far away from the boundary surface are less affected. Hence, the inverse distance (`inverseDistance`) approach with respect to the guide vane surfaces seems to be a proper choice for the diffusion parameter.

Additionally, one can enhance the decay rate of Γ through the diffusivity manipulators that are available in OpenFOAM (for instance, quadratic and exponential). The `inverseDistance` and the `quadratic inverseDistance` diffusivities work as $1/y$ and $1/y^2$ functions, respectively, where y is the minimum distance from the guide vane surfaces. Obviously, the `quadratic` manipulator significantly increases the rate of diffusivity reduction with distance from a surface. Here, a study is carried out to investigate the effect of the diffusivity parameter decay rate on the mesh during turbine transient operations. A Francis turbine load change procedures involve either closing down (load rejection) or opening up (load acceptance) of the guide vanes. The mesh motion in both types of load changes is here simulated with and without the `quadratic` manipulator.

The numerical tests are performed using the `moveDynamicMesh` utility, which only conducts the dynamic mesh calculations without solving the flow field while monitoring the mesh quality aspects. First, the load rejection procedure (i.e, guide vane closure) is examined. Fig. 5a demonstrates the unmorphed mesh at the BEP condition with guide vane angle $\alpha = 9.84°$, while Figs. 5b and 5c illustrate morphed meshes at $\alpha = 5.81°$ utilizing the `inverseDistance` and `quadratic inverseDistance` diffusivity parameters, respectively. α is the guide vane angle computed with respect to the fully closed position. The figure shows the mesh between the guide vanes on the upper surface, which is the region with the highest



(A) Unmorhped mesh ($\alpha = 9.84^\circ$)

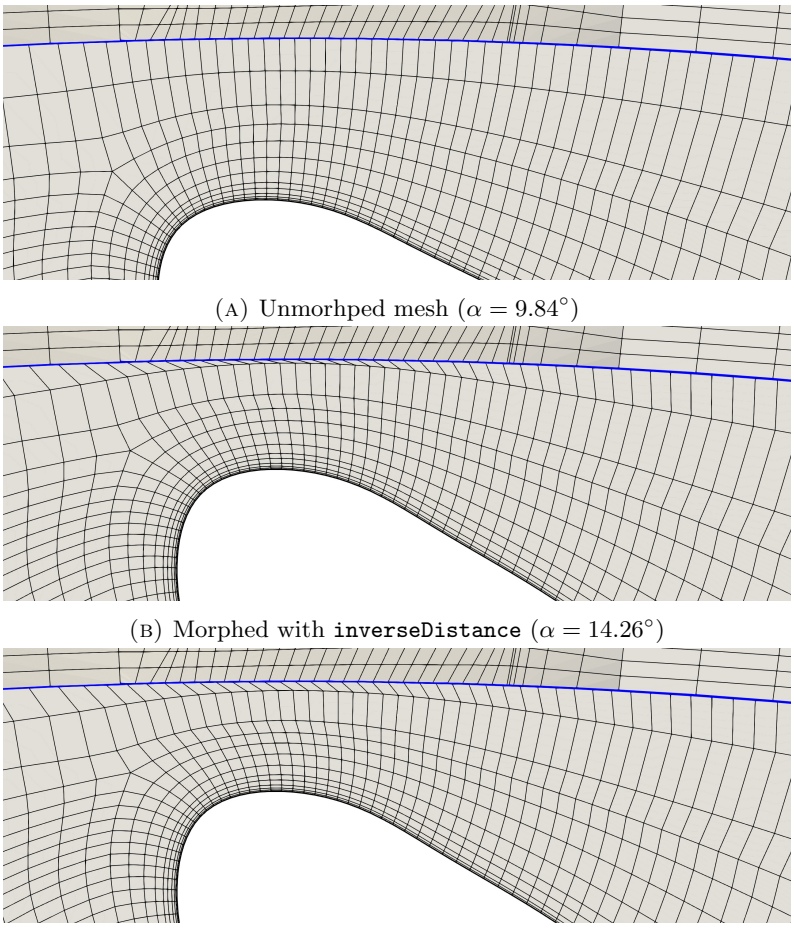
(B) Morphed with `inverseDistance` ($\alpha = 5.81^\circ$)

(C) Morphed with `quadratic inverseDistance` ($\alpha = 5.81^\circ$)

FIGURE 5. Mesh morphing in the load rejection operation on the upper surface of the guide vane domain

mesh deformation. The guide vanes are closing down. Consequently, the cells between the guide vanes are squeezed and the cells that are in the middle of this region sustain the largest deformation. One can see that adding the `quadratic` manipulator is pushing the deformation more towards the middle as the cells near to the guide vanes becomes more rigid. Therefore, the `inverseDistance` diffusivity is more appropriate for the guide vane closure procedure. The `inverseDistance` produces the first negative-volume cell after 4.79° guide vane rotation, whereas `quadratic inverseDistance` can only rotate the guide vanes for 4.07° without any mesh quality problems.

A similar comparison is made for the guide vane opening procedure (i.e., load acceptance) in Fig. 6. Once more, a zoomed view of the region with the largest deformation is shown, which in this case corresponds to the cells near the leading edge on the middle surface of the guide vane domain. The undeformed interface with the upstream domain (spiral casing) is highlighted with a blue curve. In the opening situation, the cells are pushed towards the upstream and downstream domains. Since the locations of the points on the `cyclicAMI` surfaces are kept fixed, the first cells at the interface experience the highest deformation. The first negative-volume cell is detected after a guide vane opening of 4.42° and 5.02° for `inverseDistance` and `quadratic inverseDistance` diffusivities, respectively. For this reason, the `quadratic` manipulator which increases the decay rate of the diffusivity improves the mesh morphing procedure and enables larger guide vanes openings without any mesh quality problems.

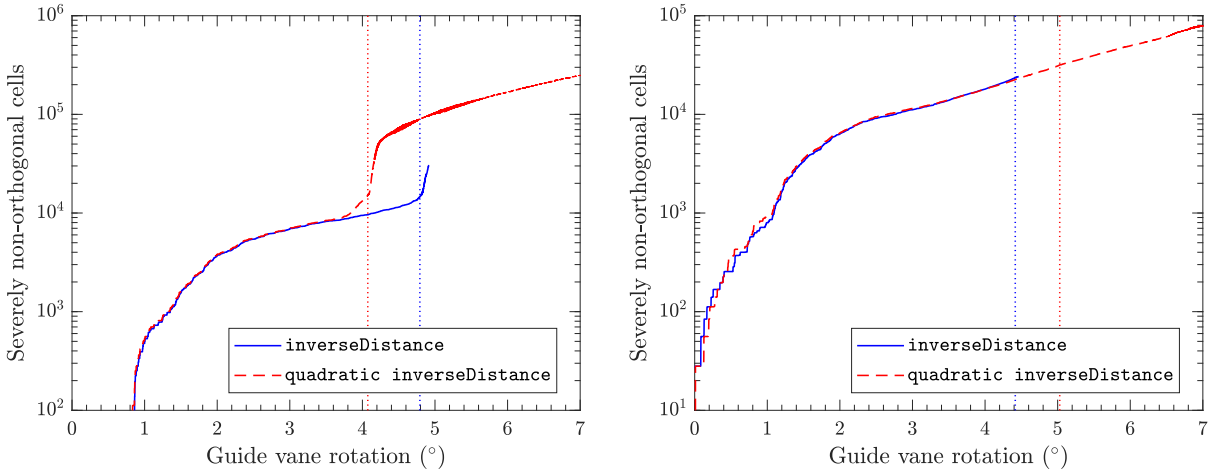


(A) Unmorhped mesh ($\alpha = 9.84°$)

(B) Morphed with `inverseDistance` ($\alpha = 14.26°$)

(C) Morphed with `quadratic inverseDistance` ($\alpha = 14.26°$)

FIGURE 6. Mesh morphing in the load acceptance operation on the middle plane of the guide vane domain



(A) Load rejection

(B) Load acceptance

FIGURE 7. Number of severely non-orthogonal faces created during the mesh deformation calculations. The dotted lines indicate appearance of the first negative-volume cell. Colors of the dotted lines are similar to the legends.

The effects of the considered diffusivity parameters are further examined by assessing the mesh quality measures during the dynamic mesh calculations. The `moveDynamicMesh` utility checks and reports the mesh quality at each time step. Therefore, the quality measures are extracted from the output `log` file,

```
const vectorField& nHat = this->patch().pointNormals();

tmp<Field<Type>> tvalues =
(
    (
        this->patchInternalField()
        + transform(I - 2.0*sqr(nHat), this->patchInternalField())
    )/2.0
);
```

LISTING 1. `evaluate` fucntion of `basicSymmetryPointPatchField`

utilizing the `foamLog` script and an ad-hoc query database `foamLog.db`. Here the number of severely non-orthogonal faces ($> 70°$) created during the mesh deformation is explored. In load rejection (Fig. 7a), the number of severely non-orthogonal faces shows a sudden rise for the `quadratic` manipulator after 3.7° rotation whereas `inverseDistance` produces a significantly lower number of highly skewed faces. For load acceptance operation (Fig. 7b), both diffusivity parameters create nearly similar number of non-orthogonal faces. However the `inverseDistance` encounters negative-volume cells sooner.

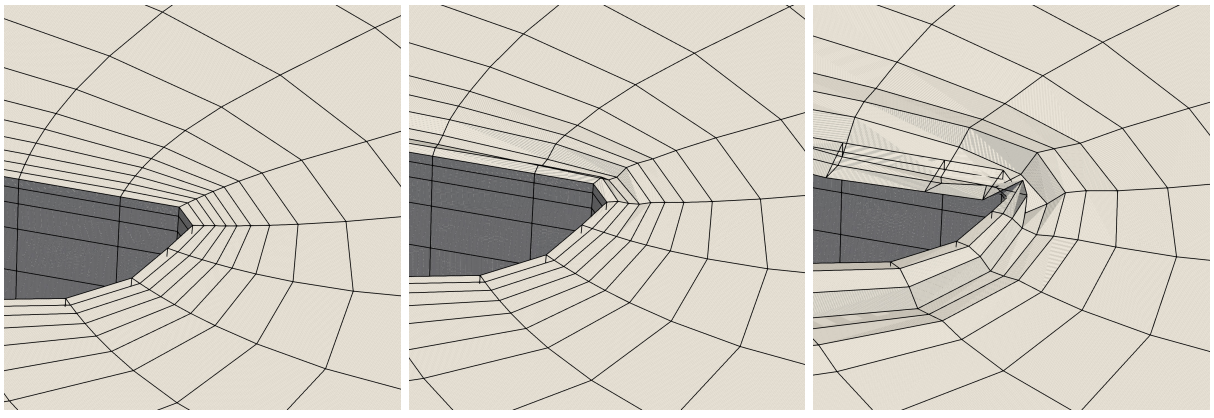
One should note that the selection of a proper diffusivity parameter in turbine transient operations becomes more important when dealing with significant load changes (e.g., shutdown and startup). It can provide more robust mesh deformation that results in a lower number of mesh changes. In the current test case, the `inverseDistance` and `quadratic inverseDistance` are adopted for load rejection and acceptance, respectively.

5.1.3. *Slip condition.* The mesh points corresponding to the lower and upper surfaces of the guide vane domain are required to slip on this flat surface. The general point slip boundary condition in OpenFOAM (`slipPointPatchField`) is identical to `basicSymmetryPointPatchField`, which works as an explicit correction. After obtaining a mesh displacement solution, the normal component of the quantity of interest (here `pointDisplacement`) with respect to the boundary surface is removed and only the tangential components are kept, i.e.,

$$\boldsymbol{\delta}_{\text{point},\parallel} = \boldsymbol{\delta}_{\text{point}} - (\boldsymbol{\delta}_{\text{point}} \cdot \hat{n}) \cdot \hat{n}. \tag{6}$$

Accordingly, the `evaluate` function in `basicSymmetryPointPatchField` removes the normal component as shown in Listing 1.

Fig. 8 presents the mesh deformation on the upper surface near the trailing edge of a guide vane, employing the `slip` boundary condition for the upper and lower surfaces of the guide vane domain. The explicit correction in the `slip` condition is very unstable and very sensitive to small distortions. Since the local normal vectors of the points are calculated at each time step (see the first line in the previous code listing), very small imperfections will accumulate and cause sudden divergence. The figure shows that the points on the upper surface of the guide vanes are notably distorted after a very small rotation of the guide vanes (0.16°) and the mesh is totally destroyed with numerous negative-volume cells after



(A) $\alpha = 9.84°$ (BEP) (B) $\alpha = 9.68°$ (C) $\alpha = 9.59°$

FIGURE 8. Performance of the general slip condition on the upper surface of the guide vane domain

```
scalar zValue = 0.0298040;
const label patchID = boundaryMesh().findPatchID("gv_upper");
labelList patchPoints = boundaryMesh()[patchID].meshPoints();
forAll(patchPoints,pointI)
{
        transformedPts[patchPoints[pointI]].z() = zValue;
}
```

Listing 2. Example code for correction of flat surface points

only a 0.25° rotation. Here it should be noted that some/most mesh generators never give perfectly flat patches, and those tiny distortions will grow extremely fast. Even round-off errors will wrinkle the surface quickly. Such round-off errors may typically appear at edges on the slip surface, where it is not ideal to calculate the point normal. The `slip` condition by itself has no information about the actual geometry, and it should therefore always be avoided (or removed as an alternative).

In order to tackle this problem, one needs to make sure that the points stay on the requested geometrical surface while slipping. For this purpose, different alternatives can be suggested. One way is to use the `fixedNormalSlip` condition, which does not calculate the local point normals, but receives a user-specified normal and keeps the tangential slipping direction fixed. Obviously, this alternative can only be used for flat surfaces where the point normals are the same over the entire patch. Again it should be noted that some/most mesh generators never give perfectly flat surfaces, and those surfaces will remain non-flat when using the `fixedNormalSlip` condition.

Another remedy, that could also work for curved surfaces, is the `surfaceSlipDisplacement` boundary condition which slips the points while keeping them on a specified STL surface using a point projection algorithm. The resolution of the triangles in the STL surface may however impact the accuracy due to the fact that STL surfaces are discretized with triangles. That discretization also yields a non-continuous surface normal direction that will could affect the slipping process.

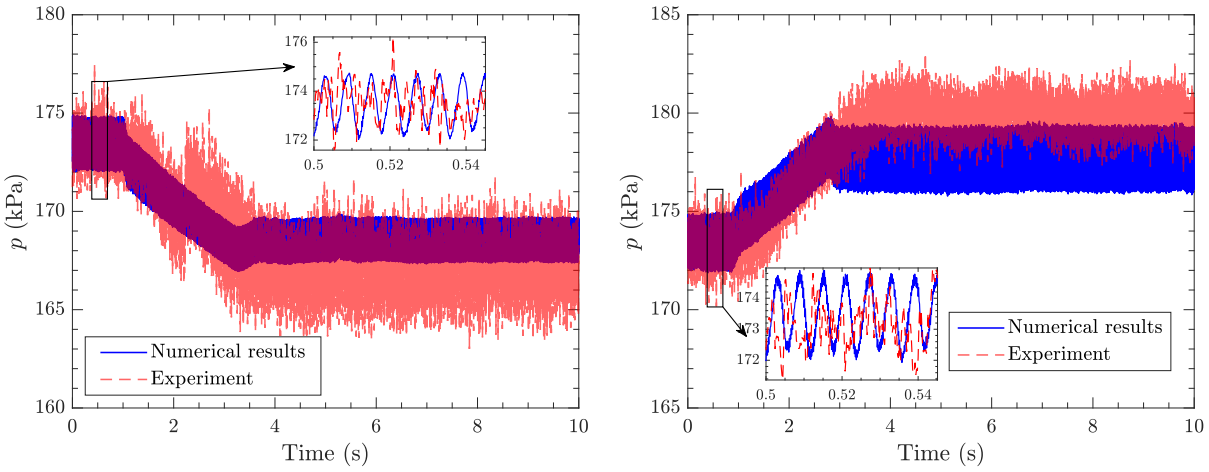
A more exact treatment would be to add an explicit correction step after the slip condition to make sure that the points do not leave the geometrical surface. As an example, one can implement a new `dynamicFvMesh` class that corrects the point locations according to the exact representation of the geometrical surface. This correction step should be performed after calculating the new point locations and just before calling the `fvMesh::movePoints` function. For instance, the user could provide the location of the flat surface and the boundary condition should make sure that the patch points stay at that surface. An example of such a procedure is shown in Listing 2. This alternative is not limited to flat surfaces, but the exact geometry must somehow be introduced inside the dynamic mesh solver. One could correct the points inside the `forAll` loop to make them stay on any geometrical surface. It can be shown that all the explicit corrections alternatives only work on surfaces with small curvatures and thus Salehi et al. [14] developed a new dynamic mesh framework in OpenFOAM that can be applied on surfaces with high curvatures. The framework was then successfully applied to the transient operation of a Kaplan turbine.

In the currently investigated test case, the points slip on flat surfaces and thus all three aforementioned alternatives can be successfully applied. It should just be remembered that the `slip` condition will not work. Here, all the computations are performed using the `surfaceSlipDisplacement` approach.

6. Results and validation

In this section, the numerical results and discussion are presented. The main intention of the current section is to assess the accuracy of the utilized numerical framework in OpenFOAM for predicting transient operations of Francis turbines. Therefore, the focus is on comparisons with the experimental data to validate the numerical results, leaving out a detailed analysis of the physics of the flow.

The pressure in the vaneless space (between guide vanes and runner blades) is monitored throughout both the load rejection and load acceptance operations. Fig. 9 compares numerical and experimental pressure at the VL2 probe (see Fig. 3) for both load change operations. The comparison reveals that the pressure variation is adequately predicted by the numerical simulations. The maximum relative error ($|p_{\text{num}} - p_{\text{exp}}|/p_{\text{exp}} \times 100$) is 3.96% and 3.76%, respectively. When the transient initiates, the pressure decreases during load rejection and increases during load acceptance and settles at a new stationary condition. The numerical pressure is slightly lower than the experimental values at the PL condition and slightly higher than the experimental values at the HL condition. It should be noted that the reference pressure is set at the outlet of the draft tube, so, much of the differences due to losses show up in the



(A) Load rejection (B) Load acceptance

FIGURE 9. Time-variation of static pressure in vaneless space

pressure at VL2. The zoomed view of the pressure at the BEP condition denotes that the high-frequency pressure pulsations due to Rotor-Stator Interaction (RSI) are well-captured.

Three PIV lines in the draft tube are defined in the experiments [15] (see Fig. 3), and the experimental and numerical velocity components are monitored on these lines. Figs. 10 and 11 compare the numerical and experimental variation of axial velocity on the three PIV lines during load rejection and acceptance, respectively.

An overview of the presented results reveals quite similar trends for the time-variation of the axial velocity on all lines at both operating conditions. The velocity contours on Line 1 and 2 (horizontal lines) demonstrate a quasi-steady condition at BEP ($t < 1$ s). The load rejection results on the first two lines (Fig. 10) show the formation and development of a large central recirculation (stagnant) flow region. The sudden change in the flow direction of Line 3 also denotes the creation of a reversed flow region. Strong low-frequency oscillations are observed after $t = 6$ s, indicating the establishment of a Rotating Vortex Rope (RVR) due to a large positive residual swirl at the runner outlet.

In the load acceptance operation, the flow leaves the runner with a residual negative swirl. Both the experimental and numerical results show the creation of a rather thin and straight central stagnant region, which in contrast to the load rejection procedure is quite stable and does not rotate.

To further assess these phenomena, the flow structures in the draft tube are visualized using the λ_2 vortex identification method [32]. The OpenFOAM function object `Lambda2` is utilized to compute the λ_2 field. Fig. 12 presents vortical structures inside the draft tube using iso-surfaces of $\lambda_2 = 750$ s^{-2} for both PL and HL conditions in red color. Additionally, the stagnant region is illustrated with iso-surfaces of zero axial velocity $W = 0$ in blue. Clearly, at PL the RVR is formed helically wrapped around the central stagnant region and rotates around the turbine axis. In contrast, at the HL condition, a large counter-rotating stable vortex is established around the slender reversed flow region at the center of the draft tube.

7. Conclusion

The current article presented a detailed description of OpenFOAM features for numerical investigation of Francis turbines transient operations. Simulation of load change operations in Francis turbines requires concurrent solid-body rotation of the runner domain and mesh deformation of the guide vane domain, which was carried out employing the Laplacian mesh morphing approach. A study was conducted on the impact of diffusivity parameters in different load change operations and it was revealed that `inverseDistance` diffusivity performs a smoother mesh deformation at load rejection while `quadratic` manipulator enhances the mesh morphing procedure at load acceptance. The general slip condition was shown to be unstable and highly sensitive to small distortions and should be avoided for slipping of the mesh points on guide vane upper and lower surfaces. Three different alternatives, i.e., slip with a fixed normal vector, slip on a specified surface, and correction of the points inside the dynamic mesh solver were presented and their advantages were discussed. The high-head Francis turbine model, Francis-99, was chosen as a case study, and both load rejection (BEP to PL) and load acceptance (BEP to HL)

operations were examined. Numerical results were validated against experimental data and the developed methodology was shown to be a reliable and accurate framework for simulation of Francis turbine transients.

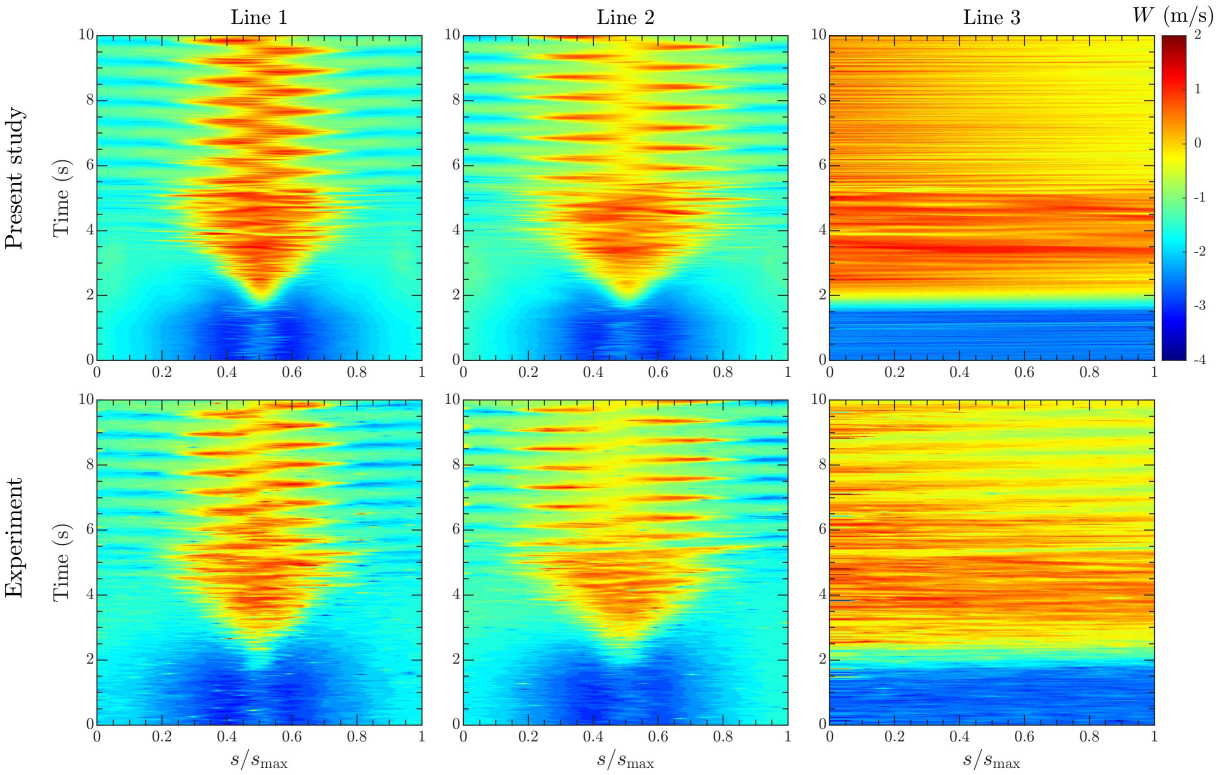


FIGURE 10. Time-variation of axial (W) velocity during load rejection operation

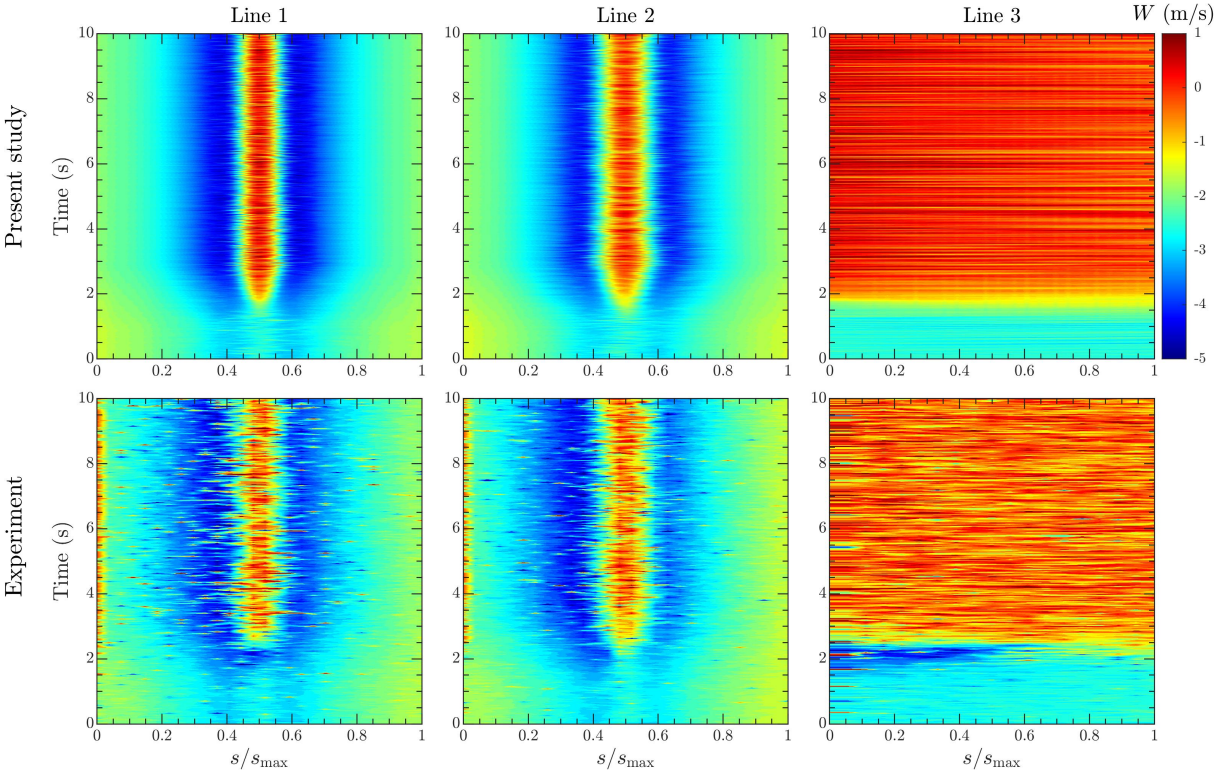
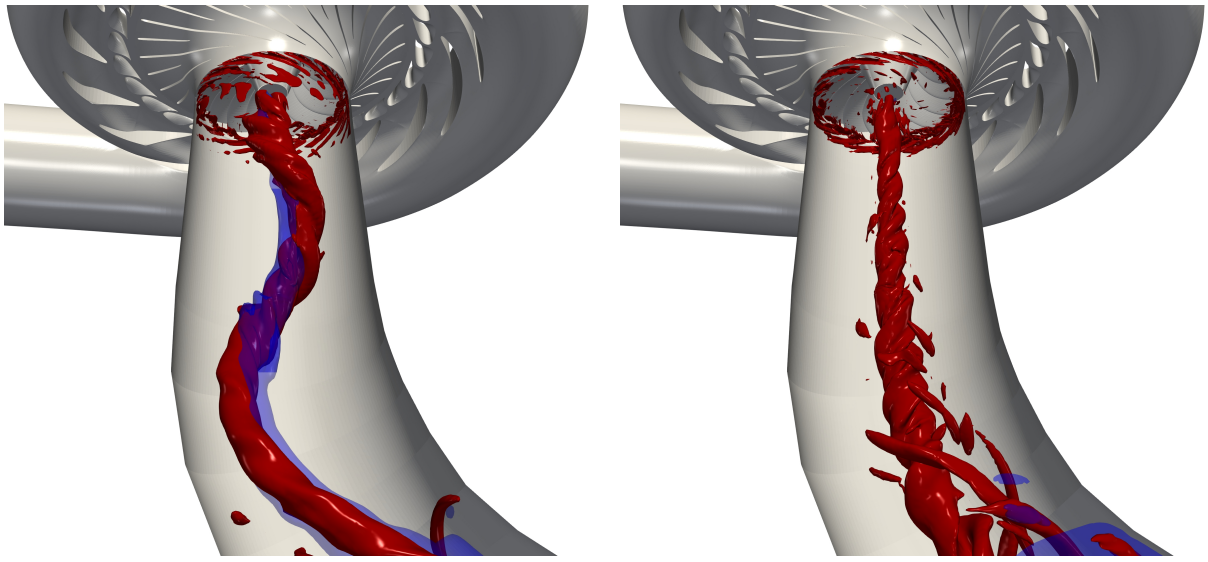


FIGURE 11. Time variation of axial (W) velocity during load acceptance operation



(A) Part load (B) High load

FIGURE 12. Illustration of draft tube vortical structures with iso-surface of $\lambda_2 = 750\ \mathrm{s}^{-2}$ (red) and stagnant region using the zero axial velocity iso-surface (blue)

ACKNOWLEDGEMENTS

The current research was carried out as a part of the "Swedish Hydropower Centre - SVC". SVC is established by the Swedish Energy Agency, EnergiForsk and Svenska Kraftnät together with Luleå University of Technology, The Royal Institute of Technology, Chalmers University of Technology and Uppsala University, `www.svc.nu`.

The computations were enabled by resources provided by the Swedish National Infrastructure for Computing (SNIC) at NSC partially funded by the Swedish Research Council through grant agreement no. 2018-05973.

The investigated test-case is provided by NTNU – Norwegian University of Science and Technology under the Francis-99 workshop series.

Author Contributions: Conceptualisation, S.S and H.N.; methodology, S.S and H.N.; software, S.S.; investigation, S.S.; resources, H.N.; writing—original draft preparation, S.S.; writing—review and editing, S.S. and H.N.; visualisation, S.S.; supervision, H.N.; funding acquisition, H.N. All authors have read and agreed to the published version of the manuscript.

NOMENCLATURE

Acronyms

BEP	Best Efficiency Point
CFD	Computational Fluid Dynamics
CFL	Courant-Friedrichs-Lewy
HL	High Load
LUST	Linear-Upwind Stabilised Transport
PL	Part Load
RSI	Rotor-Stator Interaction
RVR	Rotating Vortex Rope
SAS	Scale-Adaptive Simulation
URANS	Unsteady Reynolds-Averaged Navier-Stokes

English symbols

$\hat{n}$	Patch normal vector	
k	Turbulent kinetic energy	$\mathrm{m}^2/\mathrm{s}^2$
p	Pressure	Pa
U_i	Cartesian velocity vector	m/s

$\mathbf{x}_{\text{point}}$ Points positions ..... m

Greek symbols

α Guide vane angle ..... °
$\boldsymbol{\delta}$ Point or cell displacement ..... m
Γ Motion diffusivity
λ_2 `Lambda2` variable ..... s^{-2}
ν Fluid kinematic viscosity ..... m^2/s
ω Specific dissipation rate ..... s^{-1}
ρ Fluid density ..... kg/m^3

References

[1] J. Hell, "High flexible hydropower generation concepts for future grids," *Journal of Physics: Conference Series*, vol. 813, p. 012007, apr 2017.

[2] R. Goyal and B. K. Gandhi, "Review of hydrodynamics instabilities in Francis turbine during off-design and transient operations," *Renewable Energy*, vol. 116, pp. 697 – 709, 2018.

[3] C. Trivedi, B. Gandhi, and C. J. Michel, "Effect of transients on Francis turbine runner life: a review," *Journal of Hydraulic Research*, vol. 51, no. 2, pp. 121–132, 2013.

[4] J. Nicolle, J. F. Morissette, and A. M. Giroux, "Transient CFD simulation of a Francis turbine startup," *IOP Conference Series: Earth and Environmental Science*, vol. 15, no. 6, p. 062014, nov 2012.

[5] P. Mössinger, R. Jester-Zürker, and A. Jung, "Francis-99: Transient CFD simulation of load changes and turbine shutdown in a model sized high-head Francis turbine," *Journal of Physics: Conference Series*, vol. 782, p. 012001, jan 2017.

[6] Y. Dewan, C. Custer, and A. Ivashchenko, "Simulation of the Francis-99 hydro turbine during steady and transient operation," *Journal of Physics: Conference Series*, vol. 782, p. 012003, jan 2017.

[7] K.-R. G. Jakobsen and M. A. Holst, "CFD simulations of transient load change on a high head francis turbine," *Journal of Physics: Conference Series*, vol. 782, p. 012002, jan 2017.

[8] C. Trivedi, "Time-dependent inception of vortex rings in a Francis turbine during load variation: large eddy simulation and experimental validation," *Journal of Hydraulic Research*, vol. 58, no. 5, pp. 790–806, 2020.

[9] N. Sotoudeh, R. Maddahian, and M. J. Cervantes, "Investigation of rotating vortex rope formation during load variation in a Francis turbine draft tube," *Renewable Energy*, vol. 151, pp. 238 – 254, 2020.

[10] J. Unterluggauer, V. Sulzgruber, E. Doujak, and C. Bauer, "Experimental and numerical study of a prototype Francis turbine startup," *Renewable Energy*, vol. 157, pp. 1212 – 1221, 2020.

[11] L. Sun, P. Guo, and J. Yan, "Transient analysis of load rejection for a high-head Francis turbine based on structured overset mesh," *Renewable Energy*, vol. 171, pp. 658–671, 2021.

[12] S. Salehi, H. Nilsson, E. Lillberg, and N. Edh, "An in-depth numerical analysis of transient flow field in a francis turbine during shutdown," *Renewable Energy*, vol. 179, pp. 2322–2347, 2021.

[13] ——, "Numerical simulation of hydraulic turbine during transient operation using OpenFOAM," *IOP Conference Series: Earth and Environmental Science*, vol. 774, no. 1, p. 012060, jun 2021.

[14] ——, "Development of a novel numerical framework in OpenFOAM to simulate kaplan turbine transients," *IOP Conference Series: Earth and Environmental Science*, vol. 774, no. 1, p. 012058, jun 2021.

[15] "Francis-99," https://www.ntnu.edu/nvks/francis-99, 2020, accessed: 2020-01-28.

[16] F. Menter and Y. Egorov, "A scale adaptive simulation model using two-equation models," in *43rd AIAA Aerospace Sciences Meeting and Exhibit*, 2005, p. 1095.

[17] Y. Egorov and F. Menter, "Development and application of SST-SAS turbulence model in the DESIDER project," in *Advances in Hybrid RANS-LES Modelling*, S.-H. Peng and W. Haase, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 261–270.

[18] T. Krappel, A. Ruprecht, S. Riedelbauch, R. Jester-Zuerker, and A. Jung, "Investigation of Francis turbine part load instabilities using flow simulations with a hybrid RANS-LES turbulence model," *IOP Conference Series: Earth and Environmental Science*, vol. 22, no. 3, p. 032001, mar 2014.

[19] A. Javadi and H. Nilsson, "A comparative study of scale-adaptive and large-eddy simulations of highly swirling turbulent flow through an abrupt expansion," *IOP Conference Series: Earth and Environmental Science*, vol. 22, no. 2, p. 022017, mar 2014.

[20] J. Nicolle and S. Cupillard, "Prediction of dynamic blade loading of the Francis-99 turbine," *Journal of Physics: Conference Series*, vol. 579, p. 012001, jan 2015.

[21] C. Trivedi, "Investigations of compressible turbulent flow in a high-head Francis turbine," *Journal of Fluids Engineering*, vol. 140, no. 1, 09 2017, 011101.

[22] J. Decaix, V. Hasmatuchi, M. Titzschkau, and C. Münch-Alligné, "CFD investigation of a high head Francis turbine at speed no-load using advanced urans models," *Applied Sciences*, vol. 8, no. 12, p. 2505, 2018.

[23] R. Courant, K. Friedrichs, and H. Lewy, "Über die partiellen differenzengleichungen der mathematischen physik," *Mathematische Annalen*, vol. 100, no. 1, pp. 32–74, Dec 1928.

[24] H. Weller, "Controlling the computational modes of the arbitrarily structured C grid," *Monthly Weather Review*, vol. 140, no. 10, pp. 3220–3234, 10 2012.

[25] S. Patankar and D. Spalding, "A calculation procedure for heat, mass and momentum transfer in three-dimensional parabolic flows," *International Journal of Heat and Mass Transfer*, vol. 15, no. 10, pp. 1787 – 1806, 1972.

[26] R. Issa, "Solution of the implicitly discretised fluid flow equations by operator-splitting," *Journal of Computational Physics*, vol. 62, no. 1, pp. 40 – 65, 1986.

[27] P. Farrell and J. Maddison, "Conservative interpolation between volume meshes by local galerkin projection," *Computer Methods in Applied Mechanics and Engineering*, vol. 200, no. 1, pp. 89 – 100, 2011.

[28] H. J. Aguerre, S. Márquez Damián, J. M. Gimenez, and N. M. Nigro, "Conservative handling of arbitrary non-conformal interfaces using an efficient supermesh," *Journal of Computational Physics*, vol. 335, pp. 21 – 49, 2017.

[29] F. Pellegrini and J. Roman, "Scotch: A software package for static mapping by dual recursive bipartitioning of process and architecture graphs," in *High-Performance Computing and Networking*, H. Liddell, A. Colbrook, B. Hertzberger, and P. Sloot, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 1996, pp. 493–498.

[30] H. Jasak and Z. Tukovic, "Automatic mesh motion for the unstructured finite volume method," *Transactions of FAMENA*, vol. 30, no. 2, pp. 1–20, 2006.

[31] H. Jasak, "Dynamic mesh handling in openfoam," in *47th AIAA Aerospace Sciences Meeting including The New Horizons Forum and Aerospace Exposition*, 2009, p. 341.

[32] J. Jeong and F. Hussain, "On the identification of a vortex," *Journal of Fluid Mechanics*, vol. 285, p. 69–94, 1995.