Numerical Modeling of Oryon Watermill

Literature Study

on

Treatment of non-smooth dynamics and approaches for efficiently obtaining solutions to ordinary differential equations with periodic boundary conditions

by

Amey Vasulkar
(4750500)

TU Delft

List of Figures

Contents

1 Introduction

The Oryon Water Mill (OWM) is an innovative water turbine concept developed by Deep Water Energy BV (DWE). The company need to develop a reliable and efficient numerical model which will aid in the engineering design process. This project, thus, aims at developing the numerical model of the OWM. Work has previously been carried out on developing a 2D numerical model for OWM but it has its own limitations. In this chapter, the OWM is introduced along with its experimentally obtained characteristics which followed by a brief discussion of the previous work with its shortcomings. The chapter concludes with the problems that will be addressed in the current report.

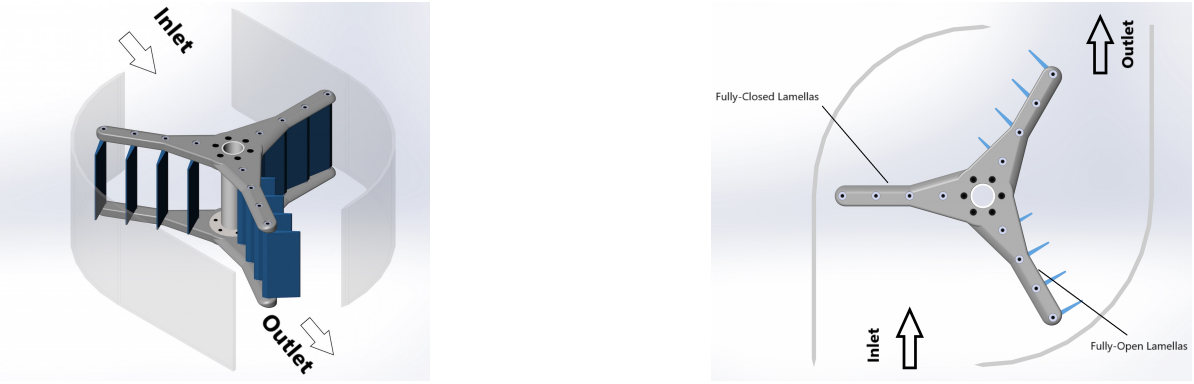
1.1. Oryon Water Mill

The OWM is a water turbine which generates power from moving water, typically, rivers, small streams and even open water channels with currents of $1-2\mathrm{m/s}$. A few important characteristics of the turbine are:

- Has a modular build making it economically adaptable to local flow conditions.
- Has ability to operate under low pressure head conditions (rivers, etc.)
- Is eco-friendly as it operates at low speeds and has large clearances which do not adversely affect the morphology and wildlife (like fishes) of the local water body.
- Has a innovative design (described later) of the rotor arm which increases the torque and power delivered by the turbine.

1.1.1. The details of the construction of OWM

The turbine primarily consists of a three-armed rotor with its axis mounted vertically and perpendicularly to the flow. Each arm of the rotor has 4 equally sized movable flaps (referred to as *lamellas*). These lamellas are allowed to rotate with respect to the rotor arm. The angle of rotation of individual lamella depends on the stoppers effected in the design. A detailed description of the stoppers and their mechanism is provided in the next chapter. This entire assembly is encased in a housing. The rotor shaft is connected to an electric generator. Figure 1.1 is an illustration of the OWM as given by DWE.



(a) (b)

Figure 1.1: Figure (a) shows the iso-metric view of the OWM with inlet and outlet for water. Figure (b) shows the top view of the OWM with the *lamellas* in fully-open and fully-closed positions.

1.1.2. The working of OWM

The flowing water entering the turbine causes the rotor arms along with their lamellas to move. The independent motion of the lamellas with respect to the rotor is a result of the changing angle of attack of the water on the lamellas. Here, a convention for the lamellas is defined, which, from now on will serve as a standard. The situation when the lamellas are aligned with the rotor arm (i.e. 0° angle), is referred to as-*fully-closed*, and when they are at the outermost position aligned with the flow, is referred to as-*fully-open*. Figure 1.1(b) illustrates these positions.

The turbine's working principle is of a three-armed *drag-based* rotor turbine. In a *drag-based* turbine, a drag force is exerted by the flowing water which propels the rotor arms. Likewise, the incoming water transfers momentum to the lamellas which in turn, based on their position with respect to the rotor, transfer momentum to the rotor arm and this results in motion of the rotor arm. It is imperative to note that the direct transfer of momentum from water to the rotor arm is negligible and nearly all the momentum is transferred from the water to the rotor arms through the lamellas. The mechanism can be explained in the following way.

The incoming water transfers momentum to the lamellas which are in the fully-open position. In this case, the angular momentum of the water closes the lamellas while the linear momentum is transferred from the lamellas to the rotor arm. Once the lamellas are in the fully-closed position, the entire momentum of the water is transferred to the rotor arm. As the rotor arm passes near the outlet, the moving water impinges at an angle such that it causes the lamellas to open again. In this case too, only linear momentum is transferred while the angular momentum results in the movement of the lamellas with respect to the rotor. When the lamellas reach the fully open position, they align themselves with the flow. This means that the area of the rotor obstructing the upstream flow is minimized, effectively minimizing the drag in the recovery phase. Thus, the total torque and power delivered by the water is increased. The rotor arm moves towards the inlet and the cycle continues. The design of the housing ensures that the upstream water impinges suitably on the lamellas and leaves at the right point. The housing and the lamella motion facilitates the existence of the morphology or the wild life in the water body.

1.1.3. The experimentally obtained characteristics of OWM

For the OWM, like any other turbine, characteristic curves like the torque v/s the speed curve are essential for design optimization and performance analysis. To obtain these results, experimental tests were performed on the OWM by MARIN[1]. Experimental results are also needed as a benchmark for the numerical results. Once the numerical results are in accordance with the experimental results, further efforts in design optimization of the OWM and testing of various lamella design and configuration can be done numerically as opposed to experimentally. This can be cost-effective and faster.

In [19], the author has mentioned the experimental tests performed on the turbine which were used as a benchmark to evaluate his numerical model. In this project too, these, experimental tests, will be used as a benchmark. The report from MARIN also elucidates the tests performed on the turbine.

The tests were conducted in a controlled laboratory environment in the "Shallow Water Basin" (Binnenvaart) of MARIN. The test setup consisted of a six component measurement frame with an attached mounting frame for an electric generator and an attached drive shaft. The shaft of the OWM contains a torque sensor which measures forces in the x, y and z directions, with the positive direction of turbine rotation defined as clockwise when viewed along the shaft from above the water surface. Additionally, the setup included flow meters installed at several locations at the inlet and outlet of the turbine housing. This setup was then mounted under the towing carriage of the 'shallow water basin' facility. The model tests were conducted for several variants of the turbine housing and for a number of different towing speeds. This description of the model setup is reproduced from [19].

The current report will include a review of the variant, amongst the five variants tested by MARIN, with the highest evaluated power generation co-efficient (described later), referred to as *Type IIA*, at a single selected towing velocity of 2.5m/s (requested by DWE). The *Type IIA* variant of the OWM consists of only the curved housing segments and is illustrated schematically in figure 1.2.

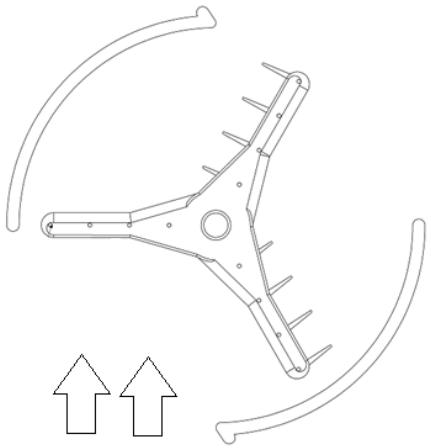


Figure 1.2: Schematic Representation of the OWM (*Type IIA*) with the arrows indicating the inlet for water

[1]Maritime Research Institute in Netherlands

The nomenclature for the turbine characteristics will be discussed and to have coherency the same is used in the remainder of the report.
The amount of Power, P, generated by a turbine is given by:

$$P = 2\pi QN,$$

where Q=Net torque generated by the turbine; N=Frequency of rotations in Hertz (Hz). Furthermore, the available power in unrestricted water (no obstructions in the flow of water) based on the total kinetic energy of the incoming water in the rotor is defined by:

$$P_{available} := \frac{1}{2}\rho A v^3,$$

where, ρ is the density of water; A is the projected area perpendicular to the inlet; and v is the velocity of incoming water.
This leads to the definition of power coefficient in unrestricted water as:

$$C_p := \frac{P}{P_{available}} = \frac{2\pi QN}{\frac{1}{2}\rho A v^3}.$$

The Tip Speed Ratio (TSR) is the ratio of the tangential velocity of the tip of the rotor arm and the velocity of the incoming water. TSR is mathematically represented as:

$$TSR := \frac{v_{tip}}{v} = \frac{2\pi RN}{v},$$

where, v_{tip} is the velocity of the rotor tip; and R is the radius of the turbine.

The performance characteristic curve for the OWM is the C_p v/s TSR plot.

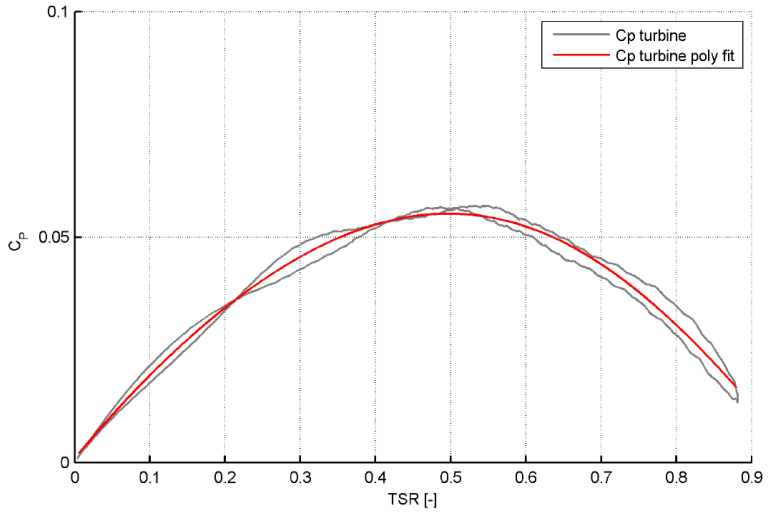


Figure 1.3: Flow Model Test Results: C_p v/s TSR- at 2.5m/s for the OWM Type IIa

Figure 1.3 shows the C_p v/s TSR relation, obtained from experimental tests performed by MARIN, shows three curves. One of the grey colored curve is obtained by plotting the C_p by increasing the TSR from 0 to around 0.9, while the other is obtained by decreasing the TSR from maximum till 0. The red curve is then obtained by fitting an appropriate polynomial of degree 4. This curve depicts a *concave*

downward parabola which implies that there is an optimum TSR and thus, a turbine operating speed for which the power generated from the turbine is maximized. An increase of the TSR beyond this optimum reduces the power obtained from the turbine.

The performance can also be expressed with the Q v/s N curve. This curve is derived from the previous plot and is just an alternative way to visualize the same results. Observation of figure 1.4 shows that the frequency versus torque relation is linear. The torque reduces as the frequency increases. Extrapolating the curve beyond N = 1 the torque produced becomes negative, implying that the turbine will consume power instead of producing. The section of the curve between 0 to 0.1 is omitted because the Q v/s N curve becomes non-linear as $N \to 0$ with $Q \to \infty$. This is not true in reality and Q is finite. This discrepancy can be attributed to the fact that the experiment is started with the turbine being stationary when the water impinges on the rotor arm. Thus implying that at $C_p = 0$ the TSR = 0. But the curve fitting polynomial in figure 1.3 doesn't pass through the point $(0, 0)$ which leads to the non-linear behavior of the derived Q v/s N curve. Thus, the rotational speeds are restricted to less than one and greater than 0.1.

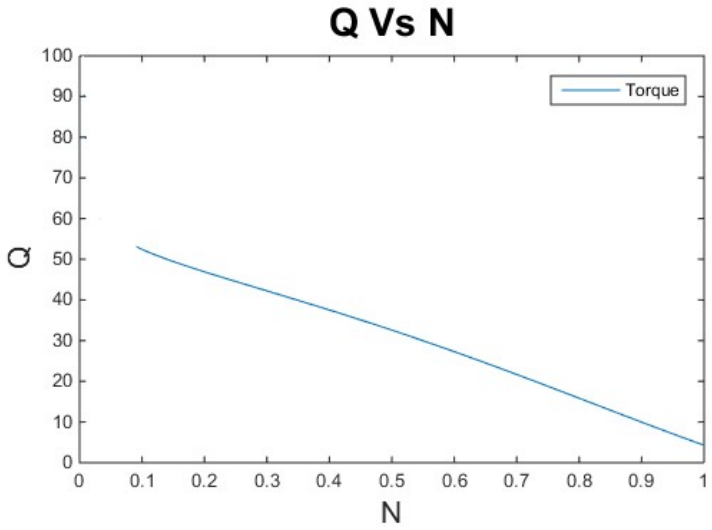


Figure 1.4: Flow Model Test Results:Q v/s N- at 2.5m/s for OWM IIa

1.2. The baseline 2D model for the OWM developed by Maniyara

In this section, the previous work done on the numerical modeling of the OWM is briefly discussed. The numerical model of the OWM developed by Maniyara will be referred to as the baseline 2D model. The baseline model is 2D as the problems faced in 2D case will also be faced in the 3D one, so correctly modeling a 2D case is a first step towards modeling of the 3D. Also, the 2D model already incorporates most of the essential parts of the 3D model and has lesser complexity. The limitations of this baseline 2D model are noted which will lead the way for further literature study.

It must be clear by now that modeling of the lamellas will be the most crucial task as they are the link for momentum transfer to the rotor arm, and thus, will be important for correct prediction of the torque. OpenFOAM® is used as a solver for the baseline 2D numerical model.

A few of the important characteristics of the model are as follows:

- The fact that the fluid is water and the velocity of water during operation is around 2.5m/s (<< speed of sound in water), the model assumes it to be an incompressible continuum following the Newtonian behavior. Thus, the Navier-Stokes equations for incompressible flow are adopted.
- The operation of turbine is a fluid-structure interaction problem.
- The dynamics of the lamella is modeled with the help of Newton's Second Law. The motion of the lamellas is driven by the flow induced force distribution.
- A sliding mesh technique is adopted as the lamellas are in motion and the mesh around them is also moving. The lamellas rotate about their hinge point which is at their leading edge. The mesh around them also rotates about the hinge point. This motion of the mesh leads the sliding mesh technique to fail. To tackle this problem, the concept of *virtual hinging* of the lamellas is used. The lamellas are modeled as hinged at the center as opposed to the hinging at their leading edge. This means the mesh around a lamella can be centered and the sliding mesh can be applied. Such virtual hinging is assumed to have little impact on the predicted torque from the numerical model. Virtual hinging is just for numerical simplicity!
- The motion of the lamellas is modeled using the event-driven approach of non-smooth dynamics. More on this approach will be discussed in the next chapter.
- The turbine torque is based on effective torque induced on the rotor shaft by individual lamellas. As discussed in section 1.1.2, the lamella motion makes the momentum transfer phenomenon intricate. When the lamella is moving (not in fully-closed or fully-open position) only linear momentum from water is transferred to the rotor arm. This means that the forces caused by the momentum transfer times the radial distance of the lamella hinge point from the axis of the rotor will give the torque induced. When the lamella is in the fully-closed condition all the momentum is transferred. This means the torque because of angular momentum on the lamella plus the torque because of the force times the radial distance will give the total torque on the lamella which is transferred completely to the rotor. In the fully-open condition there is no torque because of angular momentum on the lamellas as they are aligned with the flow. Only the torque induced by the transfer of linear momentum exists and is calculated as discussed before. Eventually, summing up the net torques calculated in the above manner for each individual lamella will give the turbine torque.

1.2.1. Results and Discussion

It is understood that the developed 2D model produces results that significantly differ from the experimental observations.

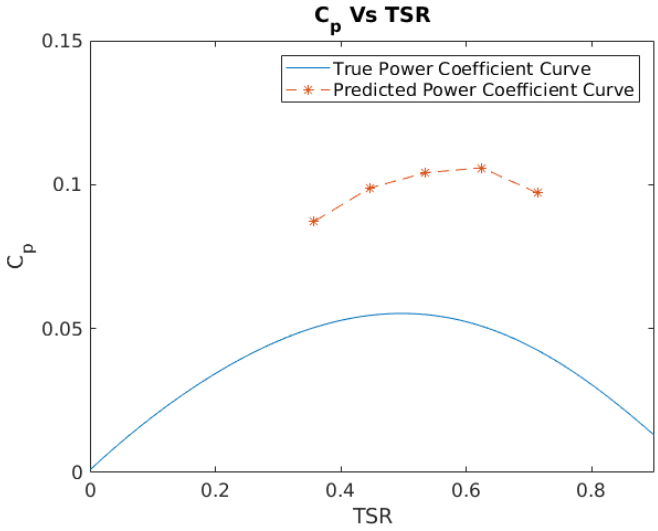


Figure 1.5: The C_p v/s TSR curve observed experimentally (blue) and predicted numerically (red dotted).

Figure 1.5 shows the C_p v/s TSR curve observed experimentally and calculated numerically. This clearly illustrates that the model has some deficiencies. The conclusions and recommendations in [19] need to be carefully studied to make the model more accurate.

The C_p vs TSR comparison has shown over-prediction in torque. The author hypothesizes this as a result of the fact that the gap between the turbine and casing in the axial direction in 3D which wasn't accounted for in the 2D model. Also, the shift in the optimal turbine performance is accredited to the difference in the velocities used to calculate the TSR in the experiments and the numerical results. In the experiments the average inlet velocity is considered while in numerical results the velocity just in front of the rotor is considered. It should be noted that the incorrect non-smooth dynamics modeling will also affect the turbine torque calculation.

1.2.2. The limitations of the baseline 2D model

- A laminar flow model is used, as it is cited in [19] that the Reynolds number of the flow is well below the turbulent regime. But lamellas when closing and opening move very fast. The motion is recorded in the videos of the experiments. So, the fast motion of lamellas will lead to separation of the fluid near the lamellas. Thus, a low-Reynolds Turbulence model would be more appropriate as it will model the separation in the turbine dynamics more precisely.

- The torque curve (figure 1.6) shows sudden spikes in the solution.

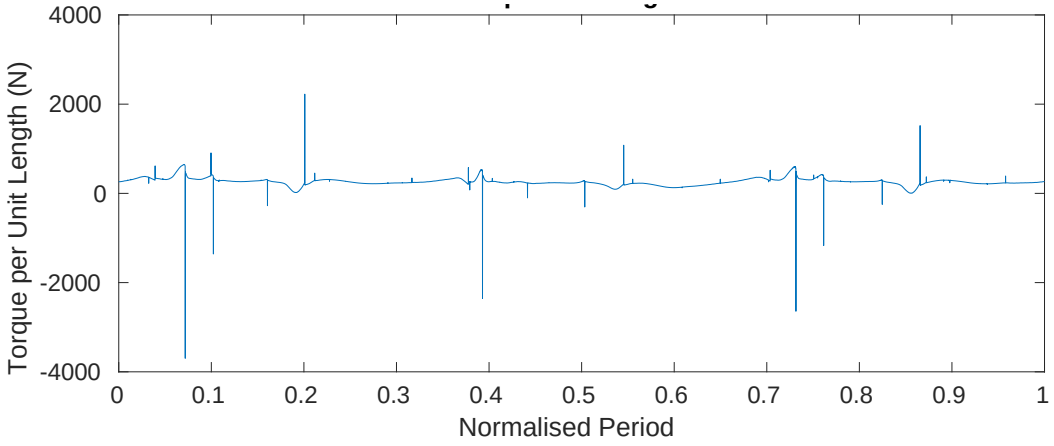


Figure 1.6: Torque Time Signal curve showing spikes

These are presumed to be numerical artifacts as such high torque spikes are unphysical. The spikes are actually because of the coupling (or the lack thereof) between the numerical model of the fluid flow and the lamella dynamics. This implies the coupling between the fluid flow and the lamella dynamics should be enforced more correctly.

- The developed model solver is currently serial which takes it very long for computations. Parallelization of the solver is needed to speed up the analysis.

- The lamellas in each period of the rotor rotation should have the same angular position throughout the period. This means the lamellas should open and close at the same time instant. This defines the lamella periodicity. The lamella dynamics predicted from the numerical model show a lack of periodicity.

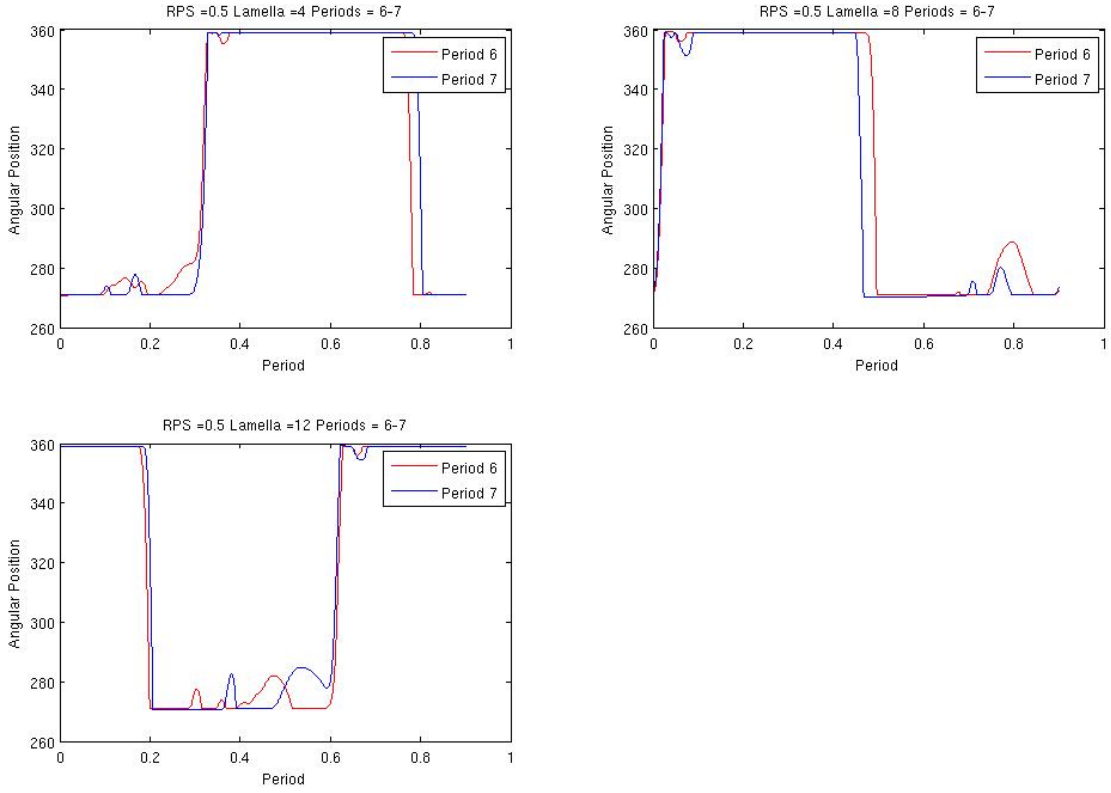


Figure 1.7: The figure depicts lamella periodicity for three lamellas located at same position on the three rotor arms for two different periods. Here, the lamella periodicity for lamellas 4, 8, 12 are considered in periods 6 and 7. The figure is reproduced from [19]

Figure 1.7 clearly illustrates that the respective lamellas of each arm show a small shift in their periodicity curves. A clear lack of overlap in their period of operation versus the angle of operation curve is visible. This discrepancy is a result of the fact that the turbine dynamics has not adequately converged as mentioned in [19].

1.3. Conclusion

From all the considerations, two target areas are identified that are expected to lead to a more accurate and efficient numerical model.

- Improvement of the modeling of the non-smooth dynamics of the lamellas to handle the contact phenomenon more realistically as opposed to the current inelastic contact.
- Improvement of the time-integration method to speed up the convergence to a time-periodic solution of the OWM problem.

With this context, in the following chapters the literature study performed to tackle these two target areas is presented with the appropriate conclusions and research questions to be answered during the course of the project.

2

Modeling of non-smooth dynamics

In this chapter the literature study performed to tackle the non-smooth dynamics modeling problem is presented. The literature study is carried out with the following goals in mind. The lamella dynamics should be modeled with sufficient accuracy to predict the torque on the rotor as per the experimental results without the unphysical spikes of the baseline 2D model. The structural model should not affect the accuracy of the flow model. Finally, the accuracy of motion of the lamella should not be more than required as the ultimate aim is to calculate the torque and the unconstrained motion will have little effect on the torque. This chapter starts with an elaborate explanation of the motion of the lamella followed by briefly discussing the current event-driven approach in the 2D baseline model and its problems. A major part of the chapter deals with the theory about non-smooth dynamics related to the problem of lamella dynamics under consideration and proposition of approaches to solve the problem.

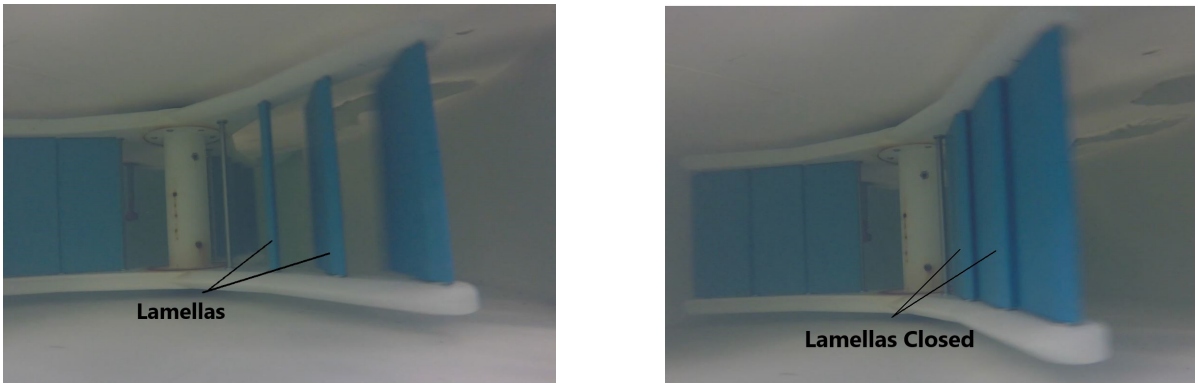
2.1. The impact of the lamella stoppers in the OWM

In the modeling of the lamella dynamics it is assumed that the lamellas act as rigid bodies. This assumption stands valid as the lamellas do not undergo any form of realizable deformation during their operation. The rigid body assumption simplifies our numerical model. By now it is clear that the lamellas operate with nearly a quarter of a rotation with stoppers at fully-closed and fully-open position. The stoppers should be clearly understood as the lamella motion is constrained by them and the unphysical spikes are observed around these positions.

2.1.1. Stoppers at fully-closed and fully-open positions

Observing the motion of the lamellas, it is clear that the lamella motion is restricted at the fully-closed position by a mechanical stop. For the fully-open position the lamellas are aligned with the flow because of which there is no torque induced by the flow on the lamellas to cause the rotation of the lamellas. Figure 2.1 illustrates the orientation of lamellas through the closing process.

The OWM videos elucidate the working of the lamellas. At the start with the fully-open position, when the rotor arm passes from the outlet towards the inlet the lamellas are aligned with flow. As the arm passes through the inlet, the incoming flow imparts momentum to the lamellas which turn anti-clockwise with respect to the rotor arm and go to the fully-closed position. The lamellas are stopped at the fully-closed posi-



(a) Lamellas Just Before Closing (b) Closed Lamellas

Figure 2.1: Snapshots reproduced from the testing of the OWM taken with a camera mounted near its inlet.

tion by a stopper as shown in figure 2.1. The trailing edge of the rotating lamellas is stopped by the hinge point of their neighboring lamellas on the same rotor arm. These hinge points act as *stopper*. The incoming water still keeps on pushing the lamellas which are now in the fully-closed position. In this situation, all the drag force and the torque from the water is imparted to the rotor through the closed lamellas.

After the lamellas pass near the outlet they rotate again as the angle of attack of the flowing water leads to opening of the fully-closed lamellas. The lamellas are opened to the extent that they are aligned with the flow. Here, the motion of the lamellas is not stopped by a mechanical *stopper* but by the flow. Figure 2.2 illustrates this operation.

Figure 2.2: Snapshots reproduced from the testing of the OWM depicting the lamellas at the fully-open condition aligned with the flow

In a gist, the numerical model should accurately model the behavior of the lamellas between the fully-open and fully-closed condition. Currently, the standard FSI solvers do not have the collision (between the lamella and its stopper) dynamics modeling and thus, it might be required to extend the capability of the FSI solver to be used.

2.2. Lamella modeling in the 2D baseline model

In the model presented in [19], the stopper functionality is incorporated by defining a proximity region near the limiting angular positions of the lamellas, which behaves as a perfect momentum sink. The momentum sink character makes it possible to define a region where the momentum instantaneously reduces to zero and the lamella is restrained from rotating clockwise or anti-clockwise depending on its position.

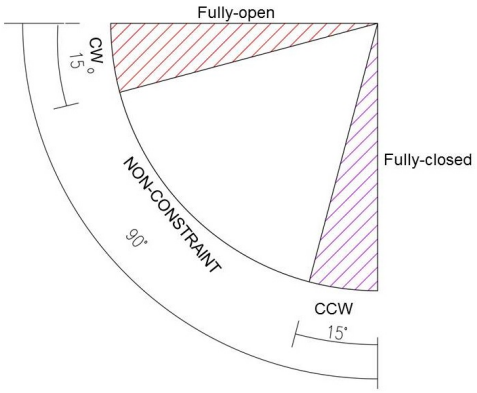


Figure 2.3: The curve depicting the motion of the lamella with different events as per the baseline 2D model

The author implements this in the model using an event selection approach, wherein three events are defined,

1. Counter-Clockwise (CCW) constraint
2. Non-constraint
3. Clockwise (CW) constraint

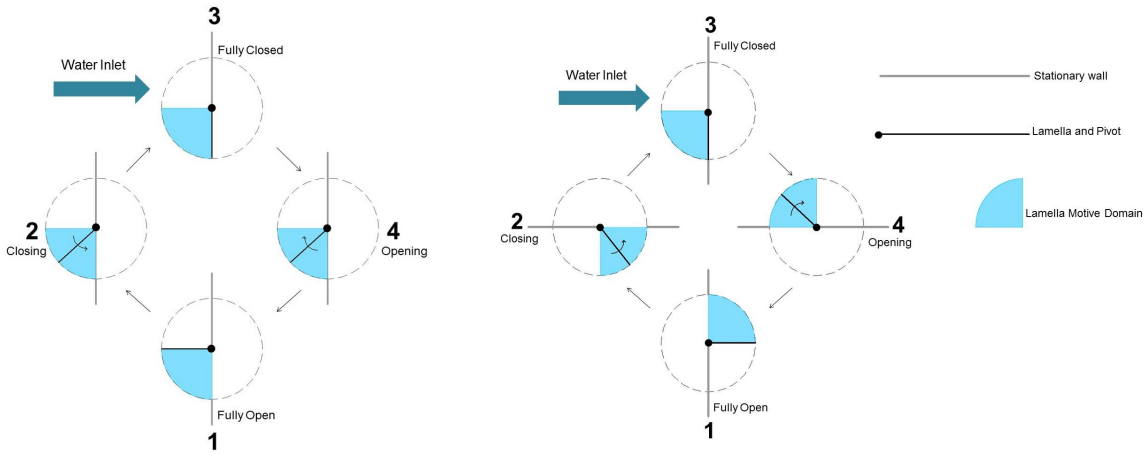
Figure 2.3 depicts the three different constraints used in the modeling. The event selection is on the basis of the lamella position. In the proximity regions, the CW or CCW constraint is applied depending on lamella position.

This approach serves as a good starting point. But it is seen that there are spikes in the results of the solutions as shown in figure 1.6. The unphysical spikes lead to the conclusion that the current momentum sink assumption is definitely a hypothetical one, and in reality, there is a momentum transfer. The implications of different lamella positions need to be studied from the rigid body dynamics point of view considering collision dynamics.

2.3. The details of lamella dynamics for numerical modeling of OWM

Figure 2.4 shows the schematic of the lamella motion in one full rotation of the rotor.

1. **Lamella Fully-Open**
   In this phase the lamellas are in fully-open position, and as discussed before there is no rotational torque acting on the lamella. The modeling equation for the lamella dynamics should accommodate this aspect. In this case, the torque on the lamella is represented only by the external forces.



(a) Rotor Frame of Reference (b) Stationary Frame of Reference

Figure 2.4: Schematic of a single lamella motion in two different frames of references for better understanding.

2. **Lamella Closing**
   As the rotor moves towards the inlet, the incoming water exerts a force on the lamellas which result in the closing of the lamellas. This can be modeled by the water exerting drag force on the lamellas in the direction of the rotor which results in the closing action.

3. **Lamella Fully-Closed**
   The analysis from closing to the fully closed period is tantamount in the modeling of the lamella dynamics. As the water pushes the lamellas in the closing direction, the lamellas gain momentum. This momentum is suddenly changed as the lamellas are stopped by the mechanical stopper discussed in section 2.1. This stopping procedure, physically, can be compared to rigid body collision. As the lamella clashes with the stopper for the first time, it rebounds in the similar way as a ball bounces of the ground. With some restitution coefficient, between the lamella and its corresponding stopper, the lamella keeps bouncing of the stopper under the action of the water force until all its oscillations are damped and the lamella is fully-closed. The fully-closed position continues for a considerable period of rotor rotation. It is essential to model the contact between the lamella and the stopper to correctly estimate the lamella dynamics.

4. **Lamella Opening**
   The moving rotor with the closed lamellas is the major torque generating phase of the turbine. When the rotor is at an angle where the water impingement is no longer orthogonal to the lamellas, the lamellas start opening. This phase can again be modeled as the external forces in the Newton's second law.

Having an elaborate understanding of the lamella motion and the underlying dynamics that should be incorporated to model the lamella behavior, the next step is to study the rigid body equation of motion and collision dynamics. Note that, in the rotor frame of reference the lamella motion is a pure rigid body rotation problem.

2.4. The rigid body equation of motion

This section introduces to the dynamics of rigid bodies which are described by their equation of motion. This equation of motion describes the relation between acceleration of and the forces acting on the body.

In classical mechanics, Newton's Laws of Motion (mostly, Second Law) describes the behavior of the physical system. Newtonian mechanics can be a possible option to develop the equations of motion, but Lagrangian mechanics is preferred as it gives a more elegant way of approaching the equations of motion if there are any constraints involved in the motion. The reason being, the constraints can be made implicit in the Lagrangian formulation as opposed to the Newtonian formulation, where the constraints are explicit. Solution procedures for complying with explicit constraints are generally complicated.

Now the rigid body equations of motion developed using the Lagrangian mechanics are discussed. At time t, $\mathbf{x}(t)$ is a vector defining the position co-ordinates of the Center of Gravity (COG) of the rigid body and $\mathbf{v}(t)(= \dot{\mathbf{x}}(t))$ is the vector defining the velocity of COG of rigid body. The *Lagrangian function* (L) is defined as the excess of kinetic energy over the potential energy. This Lagrangian function is the most fundamental quantity in the mathematical analysis of mechanical problems.
Mathematical formulation of Lagrangian function:

$$\mathsf{L}(\mathbf{x}, \mathbf{v}, t) = \mathsf{T} - \mathsf{U} - \lambda(\text{algebraic constraints}), \tag{2.1}$$

$$\mathsf{T} = \frac{1}{2}\mathbf{v}^{\mathsf{T}}\boldsymbol{M}\mathbf{v},$$

where, T is the Kinetic Energy of the system, $\boldsymbol{M}$ is a symmetric positive definite mass matrix (essentially considered constant), λ is called the Lagrangian multiplier, and U is the Potential Energy associated with the system.

The Euler-Lagrange equations of motion for rigid bodies are given by:

$$\frac{\mathsf{d}}{\mathsf{dt}}\left(\frac{\partial \mathsf{L}}{\partial \mathbf{v}}\right) - \frac{\partial \mathsf{L}}{\partial \mathbf{x}} = 0. \tag{2.2}$$

It should be noted that the Euler-Lagrange equations of motion when evaluated, essentially follow Newton's Second Law. The formulation can be represented in the form of-*Force=Mass x Acceleration*. To illustrate the application of this equation, a simple example without considering the algebraic constraints is discussed.

Consider a rigid ball in the $\mathsf{x} - \mathsf{y}$ plane. It is dropped from a height Y, to the ground. Model the equation of motion for the ball until it hits the ground which is located at $\mathsf{y} = 0$.

The position co-ordinates of the ball at time 't' are $\mathbf{x}(t) = (x(t), y(t))$.
The Lagrangian function ($\mathbf{L}$) is thus, given by:

$$\mathsf{L}(\mathbf{x}, \mathbf{v}, t) = \frac{1}{2}\mathbf{v}^{\mathsf{T}}\boldsymbol{M}\mathbf{v} - \mathbf{g}^{\mathsf{T}}\mathcal{M}\mathbf{x},$$

where $\mathbf{g}$ is the gravitational acceleration.

Note that the ball has no algebraic constraints to its motion so there is no term with Lagrangian multiplier (λ).
Using the Euler-Lagrange equations of motion (2.2):

$$\begin{aligned} m_1\ddot{x}(t) &= 0, \\ m_2\ddot{y}(t) + m_2 M|\mathbf{g}| &= 0, \\ m_1 &= (\boldsymbol{M})_{1,1}, \\ m_2 &= (\boldsymbol{M})_{2,2}, \end{aligned} \tag{2.3}$$

where $\ddot{x}(t)$ and $\ddot{y}(t)$ are the accelerations in x and y directions respectively.
The above system of equations clearly describes the motion of the ball before it hits the ground. It can be seen that the equations are in accordance with Newton's second Law. During impact (collision), the above equation of motion will have constraints associated with it. The next section deals with these constraints.

2.5. Smooth/Non-Smooth dynamics in rigid body motion

In rigid body dynamics it is generally assumed that position and velocity are smooth functions of time when the bodies involved do not touch each other. This is referred as *smooth dynamics.*

When the rigid bodies in the system are interacting (i.e. accelerating or colliding), there are discontinuities in velocity at the time of collision. In literature [23][8], this is referred as *non-smooth dynamics.* A classic example of this phenomenon is a rigid ball hitting the ground as considered previously. Just before hitting the ground, the ball has a pre-impact velocity which is brought to a halt instantaneously by the ground. This creates the discontinuity in the velocity function of the ball.

The field of non-smooth dynamics is very vast and to not digress from the main aim of modeling the motion of lamella only a few relevant aspects of this field are considered.

As discussed above, lamella collision is similar to ball bouncing on the ground. Studying the dynamics of the ball motion or any other similar physical phenomena can aid in applying similar analysis to the case of lamella motion. As mentioned in [23] the ball bouncing dynamics is a case of unilateral contacts. Furthermore, frictionless contact, between the lamella and its corresponding mechanical stopper, is assumed for simplification by considering that the presence of water will make the contact frictionless.

The phenomenon of impact can be modeled by impact laws. There are two such laws as given in [23]-*Newton's kinematic impact law* and *Poisson's impact law.* Poisson's impact law is more useful for collisions with deformation of the bodies (flexible bodies). In this case the bodies are considered rigid and thus, Newton's kinematic impact law is chosen. Newton's kinematic Impact law for rigid bodies is given by:

$$\mathbf{v_{rel}}^{-} = \epsilon \mathbf{v_{rel}}^{+},$$

where $\mathbf{v_{rel}}^{-}$ is the relative velocity before impact, $\mathbf{v_{rel}}^{+}$ is the relative velocity after impact and ϵ is the coefficient of restitution between the impacting bodies.

So, the equations of motion along with the impact dynamics analysis will give the exact description of the motion of a physical phenomena similar to lamella motion.

2.5.1. Impact Dynamics

Formulation of the generalized equations of motion for rigid body system reproduced from [8] with own notations:

$$\boldsymbol{M}\mathbf{a}_{\mathbf{p}}(t) = \mathbf{h}(\mathbf{p}(t), \mathbf{v}_{\mathbf{p}}, t) - \mathbf{w}(\mathbf{p}(t), t)\Lambda = 0, \tag{2.4}$$

where $\mathbf{p}(t)$ and $\mathbf{v}_{\mathbf{p}}$ are the position vector and velocity vector associated with the point of contact P, $\mathbf{a}_{\mathbf{p}}(t)$ is the acceleration vector associated with point P, $\boldsymbol{M}$ is a symmetric positive definite mass matrix. $\mathbf{h}$ is the vector accounting for external forces, Coriolis forces, centrifugal forces, etc., $\mathbf{w}$ accounts for the generalized force direction of the contact force whose magnitude is given by Λ. $\mathbf{w}$ is essentially a unit vector.

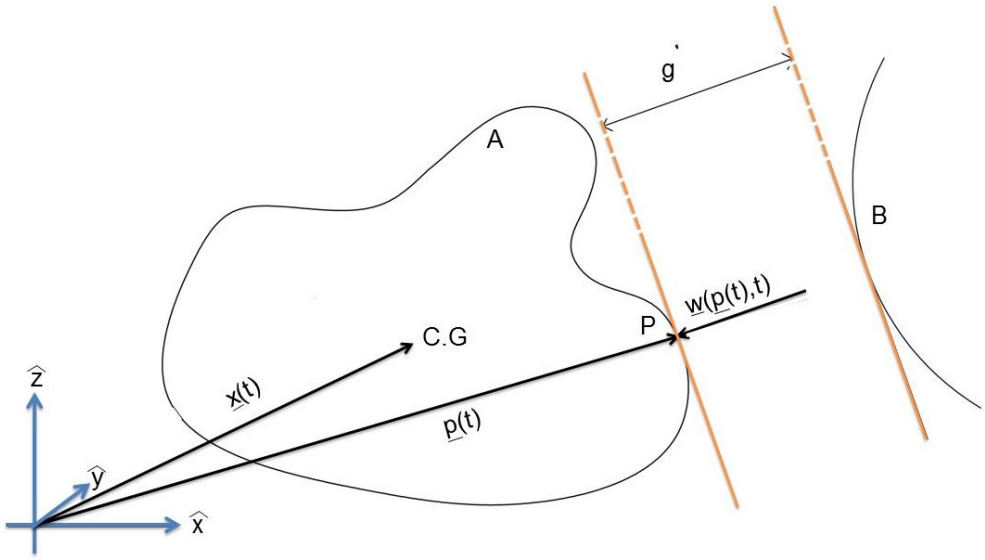


Figure 2.5: Schematic showing an interaction of 2 rigid bodies A and B and depicting the nomenclature used in the rigid body equation of motion (2.4)

Equation (2.4) is also to referred as the *projected Newton-Euler equation*. The next concern is to determine $\mathbf{w}\Lambda$ for which the contact constraints need to be understood.

Unilateral Contact Constraint

The gap function ($g(\mathbf{p}, t)$) of the contact points is defined as:

- $g > 0$ the bodies are separated.
- $g = 0$ the bodies are in contact.
- $g < 0$ the bodies are penetrating.

This implies that the *unilateral constraint equation* between two impacting bodies is $g \geq 0$. It is important to note that equation (2.4) along with the constraint equation form a system of Differential Algebraic Equation (*DAE*). Further, $\gamma (= \dot{g})$ defined as the normal relative velocity of the point of contact of one body with respect to the other, is given by $\gamma = \mathbf{w}^{\mathsf{T}}\mathbf{v}_{\mathbf{p}}$ [8].

As per [8], on integrating the equation (2.4) over singleton (t_o) the following impact equation is obtained:

$$\begin{aligned} \boldsymbol{M}(\mathbf{v}^+ - \mathbf{v}^-) &= \mathbf{w}\Lambda, \\ \gamma^{\pm} &= \mathbf{w}^T\mathbf{v}^{\pm}. \end{aligned} \tag{2.5}$$

here, the + and – signs indicate velocities after and before the impact. Next important step is to determine Λ. For this, the Newton's impact law comes into picture. Applying the law for two model conditions, $g = 0$ and $g > 0$:

$$\begin{aligned} g > 0: \quad & \boldsymbol{M}(\mathbf{v}^+ - \mathbf{v}^-) = 0, \\ \\ g = 0: \quad & \mathcal{M}(\mathbf{v}^+ - \mathbf{v}^-) = \mathbf{w}\Lambda; \qquad \gamma^{\pm} = \mathbf{w}^T\mathbf{v}^{\pm}; \\ & \gamma^+ + \epsilon\gamma^- \geq 0; \quad \Lambda \geq 0; \quad (\gamma^+ + \epsilon\gamma^-)\Lambda = 0, \end{aligned} \tag{2.6}$$

here again ϵ is the coefficient of restitution between the impacting bodies.
The first line of (2.6) is same as (2.5). The second line represents the set of all the complementarity conditions as referred in [8][23].

The impulsive force, if there is any, should act as a compressive magnitude, $\Lambda \geq 0$. This is clear from the physics of the motion. In the case of non-vanishing impulse ($\Lambda > 0$) apply the Newton's impact law as usual, i.e. $\gamma^+ = -\epsilon\gamma^-$, which is expressed in the third condition. Suppose now that, for any reason the contact does not participate in the impact, i.e. that the value of the impulsive force is zero, although the contact is closed. (Note: This happens generally for multi-contact situations). In such a situation, allow the post-impact relative velocities to be higher than prescribed by Newton's impact law in the case of non-vanishing impulse, $\gamma^+ \geq -\epsilon\gamma^-$, in order to express that the contact is superfluous and could be removed without changing the contact-impact process.

To solve the complementarity problem numerically, quadratic programming is one of the possible alternatives as cited in [8].

2.6. Numerical Solution of the Lamella Motion Problem

After studying the mathematical formulation of the lamella dynamics the next question is-*How to solve this problem numerically?* In the literature [23][21][8], methodical approaches to solve such problems are given.
There are two approaches to integrate the resulting DAEs-

- Event Driven Integration
- Time Stepping Integration

The two approaches are briefly discussed in the following sub-sections.

2.6.1. Event-Driven approach

Historically, this was the first approach to solve non-smooth dynamical equations involving the linear complementarity problem (LCP). In the event driven approach, one must separate the non-smooth motion into piecewise smooth parts and switching points (points of discontinuity). These smooth parts and switching points are

referred as 'states' or 'events' (σ). So, the approach specifies to evaluate the smooth and non-smooth parts separately. This means to integrate till the point of collision in the usual smooth way using an arbitrary DAE integrator ([20]) or an ODE integrator (in the case the DAE for the smooth part reduces to an ODE). At the switching points, or discontinuities as referred before, solve the impact problem as referred in section 2.5.1 and determine the new underlying DAE. Note that, it is important to determine the post-impact velocities which will serve as the initial conditions for future DAE/ODE integration into the smooth part after the discontinuity. The lamella motion is a case of pure rotation and thus, ODE integrators will be required. Figure 2.6 shows the flow chart for the event-driven approach.

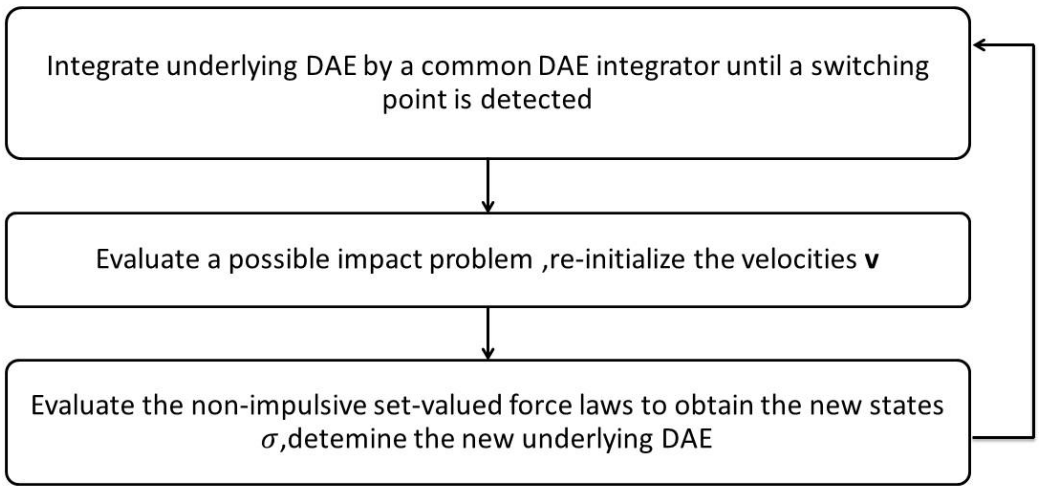


Figure 2.6: Flowchart of event-driven approach

This approach solves the problem accurately, but is a cumbersome process as it requires a lot of administrative effort in determining the discontinuity points, solving the impact law and then, re-initializing the problem for the following smooth part. So it is not preferred when there is an accumulation of a large number of discontinuities like a collection of balls repeatedly colliding with each other and bouncing on the ground.

2.6.2. Time-Stepping Approach

In the time-stepping approach, the entire DAE is discretized with the impact problem inequalities. Therefore, no discontinuity detection is required. For each time step, an appropriate discrete state σ^D is determined. This discrete state (σ^D) is assumed to hold over the entire time step. At each time step, the algorithm decides whether the whole time step is computed by the assumption of a closed or of an open contact. As a consequence, the discrete scheme can only switch its discrete state (σ^D) at discrete time points, i.e. at the start or end point of a time step. A small step size has to be chosen in order to resolve discontinuities properly. Note that, only for small step sizes the evolution of σ and σ^D can become similar. The method is well suited for accumulative discontinuities.

In the following, a general discretization technique is mentioned which is used in most of the time stepping methods. Consider a time step with size Δt, for which the start and end points are denoted S and E, respectively.

The equation (2.4) in the discretized form looks like:

$$\boldsymbol{M}_{\mathrm{D}}(\mathbf{v}_{\mathrm{E}} - \mathbf{v}_{\mathrm{S}}) - \mathbf{h}_{\mathrm{D}}\Delta t - \mathbf{W}_{\mathrm{D}}\mathrm{P} = 0;$$

where, $\boldsymbol{M}_{\mathrm{D}}$, $\mathbf{h}_{\mathrm{D}}$ and $\mathbf{W}_{\mathrm{D}}$ are approximations of mass matrix, external forces and contact forces. The term P represents the magnitude of an impulse. The velocities, position and acceleration of all the bodies together form a particular discrete state.

There are a lot of time stepping schemes like Moreau's Midpoint scheme [21], Paoli and Schatzman scheme [22], Funk [15], Foerg [7] etc., which deal with the above discretized equation. Each scheme has its own advantages and disadvantages with specific applications. Time stepping methods suffer from the major drawback that the solution to the discretized equation obtained from these schemes lacks accuracy. For the modeling of the lamella, it would require solving an impact problem, so the literature study will focus on how time-stepping schemes in general provide results for impact phenomena.

When the algebraic constraints in the system are solely stated in terms of displacements, then this is called a 'formulation on displacement level', while if also velocities are present in the restrictions (constraints) it is called a 'formulation on velocity level'. In the case of the motion of lamella, there is a constraint on displacement as the lamella shouldn't move past the stopper while there is a restriction on velocity when the lamella moves along with the rotor where the relative velocity of lamella should be zero. These constraints are clear from the impact equation (2.5) discussed before. Such a contact is referred in literature on non-smooth dynamics as the *sustaining contact*.

In [23], there is a detailed discussion on how time stepping methods deal with formulations on displacement and velocity level. Here a discussion on impact treatments with formulation on displacement and velocity level follows.

Formulation on Displacement Level

Discrete set-valued laws (eg. impact or friction law) which are formulated on displacement level do provide good accuracy in time stepping schemes. But this formulation for impacts laws has a major drawback as it is only capable of handling fully *inelastic* contact. The case of lamella motion is a partial elastic scenario. The author in [23] proposes some future ideas for formulation on displacement level which incorporate partially elastic impact behavior. But the author notifies that it cannot be implemented for a *sustaining contact*. In the lamella motion, the *sustaining contact* is one of the most important phases of torque generation so it should be modeled correctly.

Formulation on Velocity Level

Formulations on velocity level can model elastic impact behavior, but suffer from a lack of accuracy in time stepping schemes.

The advantages and disadvantages of time-stepping schemes can be summarized as follows:

Advantages

- Less administrative effort as the discontinuities are solved together with the ODE.
- Schemes are robust and simple.

- Good for modeling accumulative discontinuities like in the case of modeling a collection of balls bouncing or colliding.

Disadvantages

- A very small time step size is required, as the discontinuities and smooth parts are solved together.
- Poor accuracy as compared to the event-driven approach.
- Inability to model the partial elastic behavior correctly with sufficient accuracy.

Due to the last point above, the traditional event-driven integration approach is preferred for modeling partially elastic impacts. These disadvantages of time stepping schemes would lead to an increase in the computation time with reduced accuracy. Already the 2D baseline model of OWM requires a lot of time for simulation, increasing it further wouldn't be advisable unless necessary. Thus, in the case of modeling of the dynamics of the lamella *event-driven approach* will be used.

2.7. Solution using the event driven approach

In the previous section it was decided to use the event-driven approach for lamella dynamics modeling. The event-driven approach used in the field of animation and graphics was referred as a part of literature study. This approach is discussed in the following sections.

It is clear, that the motion of the lamella is just pure rotation, followed by collision. There is no friction involved. For such a case, the equations are developed, but solving numerically is challenging. The field of animation also requires a similar collision simulation. Referring to literature available for such simulations, it can be judged for application of similar methods for the current problem.

In [3], there is a mention of collision of rigid bodies where the author tries to solve the rigid body equation of motion for collisions (impacts, as referred before) and develop algorithms which predict the motion of the bodies in the system. The approach eventually tries to solve the same impact equations as referred to in section 2.5.1 but the equations are implicit in this approach. This approach involves frictionless collision and the numerical algorithms are already available for reference. The next section discusses the process described in [3] and [12]

By now, it is known that the equation of motion alone can be a simple ODE but additional constraints add the contact forces and constraint equations which make it difficult to solve. These constraints are referred in literature as *Nonpenetration Constraints*[3]. For such contact constraints, there are two types of contacts: colliding contact and resting contact.

Colliding contact means that the two bodies colliding against each other will have a discontinuity in velocity after the collision (i.e. instantaneous change in velocity) and after collision move apart with different velocities.
Resting contact means that the two bodies after collision are resting on each other i.e. they have zero relative velocity with respect to each other. This is the same as the *sustaining contact* mentioned in the previous section. The case of colliding contact is discussed first.

2.7.1. Colliding Contact

As the position and velocity are not smooth functions of time, in colliding contact, continue the integration of the equation until the time when collision occurs (t_c). Stop at the collision, and compute how the velocities of the bodies involved will change. It requires the application of Newton's impact law. This will give the velocities post collision which will be used to continue the integration. Thus, the problem of impact discussed in section 2.5.1 is taken care off. But there are still a few concerns like how to compute velocity changes and computing the contact forces that prevent the inter-penetration of the two bodies.

The foremost thing is computing the time of collision (t_c). Consider the position of the body at different time steps through the integration of the ODE. Call these time steps $t_n; 1 \leq n \leq N$. Consider that t_c lies between t_n and t_{n+1} as shown in the figure

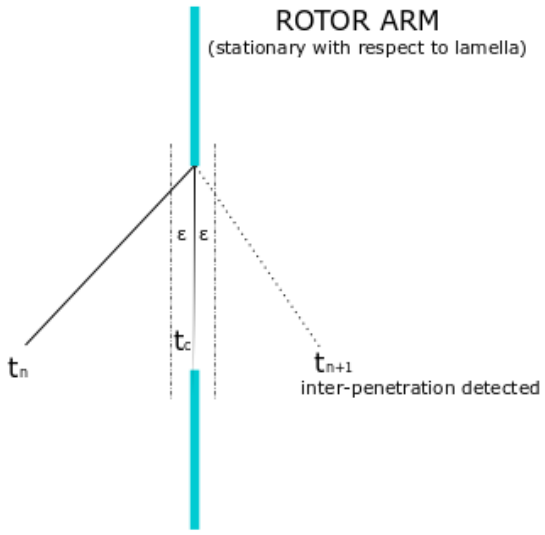


Figure 2.7: Shows the position of lamella at different time instants. The solid line shows position at t_n and dotted line shows position at t_{n+1}.

2.7. Then, use a numerical root finding algorithm to determine t_c. The method is as follows:

- As t_c lies between t_n and t_{n+1}, it means that we have detected inter-penetration at time instant t_{n+1}. In the case, simply determine when the lamella has crossed the rotor arm as the inter-penetration condition.
- Inform the ODE solver that restart at time t_n and proceed by $\frac{\Delta t}{2}$ time interval. Note that, Δt is the time step between two instants. In this case, $\Delta t = t_{n+1} - t_n$.
- If we encounter no collision (or inter-penetration) that means, t_c lies between $t_n + \frac{\Delta t}{2}$ and t_{n+1}. And, then restart the ODE solver with the starting point as $t_n + \frac{\Delta t}{2}$ and time step $\frac{\Delta t}{4}$. Otherwise, t_c is less than $t_n + \frac{\Delta t}{2}$ and try to simulate from t_n to $t_n + \frac{\Delta t}{4}$.
- Repeating the above step, the collision time is computed with some suitable numerical tolerance.

With the time of impact determined, the next step is determining the velocities post collision and the contact forces.

Before going into the details of this step, a small detour is taken to clarify a few concepts which will help determine whether the contact between two bodies is colliding, resting or no contact at all.

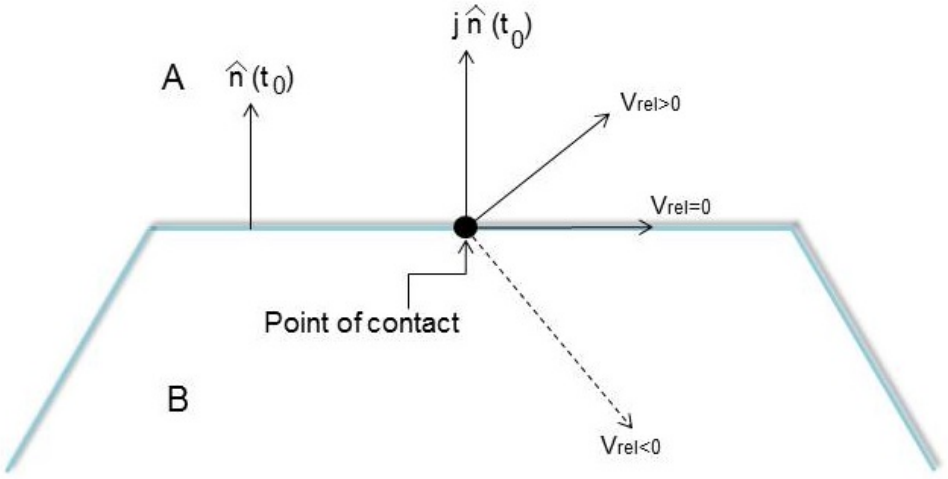


Figure 2.8: Two bodies A and B and the depiction of the nomenclature used in describing the contacts between the bodies.

Figure 2.8 shows two bodies (A and B) with different individual velocities at time instant (t_o). The vector ($\hat{\mathbf{n}}(t_o)$) is a unit normal vector acting from body B towards body A. Reference this as the positive direction. It is defined that $\mathbf{v_{rel}}$ is the relative velocity of body A with respect to body B at t_o. Then γ, as defined earlier in section 2.5.1, will now be equal to $\mathbf{v_{rel}}^T\hat{\mathbf{n}}$. The value and sign of this γ determines the type of contact.

- $\gamma > 0$ (pointing outwards (towards A)) → The bodies are not contacting after t_o.
- $\gamma < 0$ (pointing inwards (towards B)) → The bodies are in colliding contact after t_o.
- $\gamma = 0$ → The bodies are in resting contact after t_o.

The vector $j\hat{\mathbf{n}}$ represents the impulse required to avoid the penetration of the bodies whose magnitude is j and direction given by the unit normal vector.

If the bodies are in colliding contact it is necessary to compute the relative velocity post impact and the contact force responsible for the instantaneous change in the relative velocity. For computing the relative velocity post impact, Newton's impact law is used for reasons specified before. The equation states:

$$\gamma^+ = -\epsilon\gamma^-;$$

where, γ^- is the relative velocity before impact and γ^+ is the relative velocity post impact. ϵ is the coefficient of restitution as before.

Next, it is necessary to determine the contact force. When two bodies collide there is an instantaneous change in the velocity. As per mechanics, this change is caused by an impulse. The aim would be to find this impulse so that the integration of the ODE can be continued after the collision instant with new initial velocities and external force conditions. Calculating the impulse involves a lengthy derivation which is skipped here. In the case of lamella motion the stoppers are stationary in the rotor frame of reference.

Applying the condition when one of the interacting bodies is fixed, the formulation for the impulse magnitude j as given in [3] is:

$$j = \frac{-(1+\epsilon)\gamma^{-}}{\frac{1}{M_a} + \hat{\mathbf{n}}.(\boldsymbol{I}_{\boldsymbol{a}}^{-1}(t_0)(\mathbf{r_a} \times \hat{\mathbf{n}})) \times \mathbf{r_a}}$$

where, M_a is the mass of the body A, $\boldsymbol{I}_{\boldsymbol{a}}^{-1}(t_0)$ is the inertia tensor associated with the body, $\mathbf{r_a}$ is position vector of the point of contact with respect to COG of body A.

To conclude this section, it is now known how to estimate the time of collision, calculate the changes in relative velocity and finding the underlying contact forces in case of colliding contact. The next section will deal with resting contact.

2.7.2. Resting Contact

It is clear that resting contact occurs when the relative velocities between the impacting bodies are zero. For the case of numerical modeling achieving this is an ideal case and it is assumed that the relative velocity is within a numerical tolerance (e.g. $|\gamma| \leq \epsilon_r$) instead of being exactly zero. In the case of lamella motion, this will occur when all the lamella oscillations are damped and the lamellas are stationary with respect to the rotor arm in the fully-closed position.

Like in the case of colliding contact, it is imperative to reset the integration of the ODEs post collision. If the relative velocity post collision is within the numerical tolerance as specified above, it is concluded that the case we are considering is a resting contact. This is a fairly complicated problem to solve. As is the case with colliding contact, for resting contact too, there is a contact force at the contact point which functions to keep the bodies intact. This contact force has three conditions:

1. It must prevent inter-penetration.
2. It must be repulsive; it shouldn't act as 'glue' and hold bodies together.
3. It should vanish if the bodies begin to separate.

Solving for these contact forces again results in a linear complementary problem as discussed in [3]. Linear complementary problem can be solved using quadratic programming. Quadratic programming is not a built-in feature in OpenFOAM®. Thus, it would be required to considerably modify the structural solver if this approach is to be used.

A further research on this front is necessary to solve the problem as the lamellas will be in resting contact position for a larger part of rotor rotation and this is the phase when maximum torque is transferred to the rotor. In [12], the resting contact

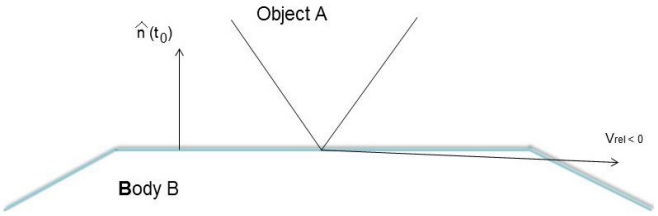
is treated in a more intuitive way which is used along with the physical understanding of the torque transfer in the lamella motion to propose a new method. The idea of this method is: Let the motion of lamellas be simulated as mentioned in [12], and then, in the resting contact situation override the structural solver by giving the condition that when in resting contact entire transfer of torque from water happens to the rotor arm. To elucidate this in detail, it is necessary to understand the approach given in [12].

This approach is simple, let the bodies in resting contact move as if they were not influenced by each other. At t_o, it is seen that the bodies are in resting contact, then $|\gamma| \leq \epsilon_r$. If $\gamma > 0$ but under the numerical tolerance (ϵ_r), the bodies are simulated till next time step ($t_o + \Delta t$) as individual without the effect of the neighboring bodies. This case of $\gamma > 0$ is easy to identify and doesn't affect the course of simulation as the bodies are seen to be separating. The case of $\gamma < 0$ is of concern as it can even give a condition that objects are colliding. The inter-penetration at the next time instant would then be small, less or equal to $\epsilon_r \Delta t$ which will show it is colliding. [12] provides an algorithm to solve this problem. The algorithm is as below:

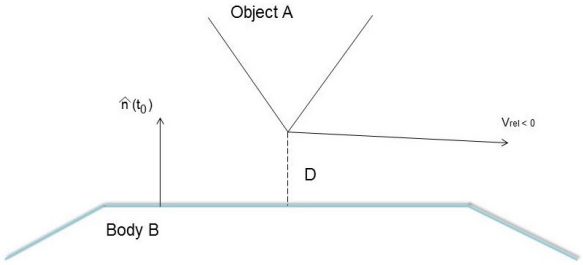
Suppose we simulate time period (t_0, t_1),

1. $t_s = t_o$ (t_s is the start time)
2. Use the ODE solver to simulate between this time instant and determine if collision occurs.
3. **If** collision is reported at time $t_c \in (t_0, t_1)$ **then** stop the simulation at time t_c, separate the objects to distance D and tell the ODE solver of the new positions after the separation.
4. **else return**
5. $t_s = t_c + f(D)$
6. **if** $t_s \geq t_1$ **then return**
7. go to step 2

The separation by distance D as shown in figure 2.9 would mean the bodies are no longer in contact and thus, the ODE solver can proceed normally. Two things are important here. Firstly, the separation distance means to translate the body A (or lamella in our problem) by vector $D\hat{\mathbf{n}}(t_c)$. Keep in mind that the body B is considered fixed. Choosing parameter D is a question of tuning. Higher D will lead to faster execution, but more coarse steps. Another thing, is the function f. This function will determine the number of separations at each step (Δt). It must be that the separations be bounded else the loop will be infinite. The author in [12] proposes that a simple linear function of D should be fine.



(a) Resting Contact Situation



(b) After Separation to distance D

Figure 2.9: Depicts the approach to solve resting contact given by Kavan.

Understanding this method and knowing that when the bodies are in resting contact the entire force and torque are transferred to the rotor arm the final algorithm looks like:

1. **IF** at t_0 bodies are in resting contact, **then** inform the solver that both the force from the water and torque are transferred to the rotor arm till the next time instant. In tandem, solve the body motion as mentioned in the previous algorithm.

2. **else return**

This way the calculation of contact force is eliminated and we directly achieve our goal. It should be noted that the physical understanding of the lamella dynamics is exploited here. Nonetheless, this approach is not suitable. As the contact force is not calculated, the resulting torque is not as accurate. The paper [12] targets applications in graphics and animations. In this field, the forces are not of importance but the motion of the bodies before and after the impact is important. Whereas our aim is to correctly model the torque on the rotor and modeling of the exact motion of lamellas is not very significant. This makes it difficult to adopt the above discussed approach.

2.8. The basics of quadratic programming to solve LCP

The equation (2.6) has complementarity conditions as discussed before. Solving for these conditions is known as the linear complementarity problem (LCP). This problem also arose when the resting contact problem was treated. As discussed in [3][8], this problem can be solved using quadratic programming. This section will deal with the very basics of quadratic programming, the available algorithms and how the LCP can be formulated as a quadratic problem.

Quadratic programming is a special case of Non-Linear programming which deals with quadratic equations. A quadratic program asks to minimize a *quadratic function* of several variables subject to a set of linear constraints on these variables [5]. In its most general form it can be written as follows:

$$\begin{aligned} \text{(QP)} \quad \min \quad & \mathbf{x}^T \boldsymbol{D}\mathbf{x} + \mathbf{x}^T\mathbf{c} \\ \text{s.t.} \quad & \boldsymbol{A}\mathbf{x} \geq \mathbf{b}, \\ & \mathbf{x} \geq \mathbf{0}, \end{aligned} \tag{2.7}$$

where $\boldsymbol{A} \in \mathbb{R}^{m\times n}$, $\boldsymbol{D} \in \mathbb{R}^{n\times n}$, $\mathbf{b} \in \mathbb{R}^m$, $\mathbf{c} \in \mathbb{R}^n$ and $\mathbf{x}$ is the solution vector which consists of variables to be found for optimal values. $\boldsymbol{A}$ is called as the constraint matrix.

If $\boldsymbol{D} = 0$, then it is called a *linear program*, else when $\boldsymbol{D} \neq 0$ and is *positive semi-definite* (i.e. $\mathbf{x}^T\boldsymbol{D}\mathbf{x} \geq \mathbf{0}$, holds $\forall \mathbf{x}$) then the problem 2.7 is called a *convex quadratic optimization problem*. In that case any local minimum of the quadratic equation above is also a global minimum. The solutions for such convex optimization problems exist and are unique [5].

2.8.1. Karush-Kuhn-Tucker Conditions

The basis for checking whether a feasible solution is optimal are the Karush-Kuhn-Tucker (KKT) conditions [14][11], which are derived for general convex optimization problems by requiring that the gradient vanishes at an optimal point. For convex quadratic programs the KKT conditions are necessary and sufficient.

Theorem (KKT conditions for 2.7). A feasible solution $\mathbf{x}^* \in \mathbb{R}^n$ to convex quadratic program 2.7 is optimal if and only if there exists $\lambda \in \mathbb{R}^n$ and $\mu \in \mathbb{R}^n$, $\mu \leq 0$, such that,

$$\mathbf{c}^T + 2\mathbf{x}^{*T}\boldsymbol{D} = -\lambda^T\boldsymbol{A} + \mu^T,$$

$$\mu^T\mathbf{x}^* = 0.$$

The proof is elaborate and complicated and thus, not discussed here. For details the reader is referred to [11][14].

2.8.2. List of program routines to solve QP

The field of quadratic programming is been under development for over 40 years. A plethora of algorithms and subsequent routines are available to solve the quadratic programs. The following is a list of these which are particularly useful for solving the convex quadratic optimization problems.

- LSSOL this is from SOL optimization software. It is implemented in FORTRAN 77 and is available as a source code which can be used in FORTRAN, C or MATLAB. This is not an open source software.
- CPLEX is from IBM. This is an optimization software specially designed for Quadratic Programming. This is also a paid software.
- HOPDM is a package developed by Jacek Gondzio and Anna Altman. It is an open source package written in FORTRAN.
- QuadProg++ is a C++ solver for convex optimization problems written by Luca Di Gaspero. It is an open source code available on github.

- ALGLIB is a library that supports C++, Delphi. It is partially free and some of its functionalities are paid.

It should be noted that the discussion of quadratic programming is presented for the completeness of the literature study report. It is later seen that the quadratic programming can be avoided.

2.8.3. The formulation of impact problem as a quadratic optimization problem

The basic understanding of quadratic programs will help to re-write the LCP as quadratic optimization problem.

In the section 2.7.2 the nature of contact forces to solve the resting contact problem is discussed. This nature gives rise to three equations which on solving, will give the contact forces exactly satisfying these conditions. Obtaining the contact forces, it is then possible to simulate the resting contact situation.

In the case of lamella motion with resting contact, there is no velocity normal to the surface so a constraint force needs to be computed that maintains the non-interpenetration constraint. The constraint force is $\mathbf{w}(t_0)\Lambda$ as discussed before. In the following discussion for simplicity it is represented as $\mathbf{f}(t_0)$. The conditions on $\mathbf{f}(t_0)$ can be stated as follows. First, since the constraint force must be repulsive, we require $\mathbf{f}(t_0) \geq \mathbf{0}$. Second, the constraint force must be strong enough to prevent the lamella from accelerating into the stopper. Let $\mathbf{a_N}(t_0)$ denote the acceleration of the lamella with respect to the stopper in the normal direction at the point of contact at time t_0, then to prevent interpenetration, it is required $\mathbf{a_N}(t_0) \geq \mathbf{0}$ as well. If $\mathbf{a_N}(t_0) = \mathbf{0}$, the lamella remains on the surface of stopper, otherwise $\mathbf{a_N}(t_0) > \mathbf{0}$ indicates the lamella breaks contact with the surface. Lastly, since constraint forces must be workless, the constraint force must be zero if the lamella is breaking contact. This condition is written as $\mathbf{f}^T(t_0)\mathbf{a_N}(t_0) = 0$ which ensures that if $\mathbf{a_N}(t_0)$ is positive, $\mathbf{f}(t_0)$ is zero. For the simple lamella stopper example, it turns out that $\mathbf{a_N}(t_0)$ is a linear function of $\mathbf{f}(t_0)$ and can be expressed as $\mathbf{a_N}(t_0) = \boldsymbol{A}\mathbf{f}(t_0) + \mathbf{b}$. Thus, solving the system:

$$\mathbf{f}(t_0) \geq \mathbf{0}; \quad \boldsymbol{A}\mathbf{f}(t_0) + \mathbf{b} \geq \mathbf{0}; \quad \text{and} \quad \mathbf{f}^T(t_0)(\boldsymbol{A}\mathbf{f}(t_0) + \mathbf{b}) = 0,$$

gives the solution for the contact forces.

For frictionless contacts, $\boldsymbol{A}$ is a positive definite matrix and the code to calculate $\boldsymbol{A}$ and $\mathbf{b}$ is given in [3]. The above system is then represented as a Quadratic problem.

$$\begin{aligned} \text{(QP)} \qquad \min \quad & \mathbf{f}^T\boldsymbol{A}\mathbf{f} + \mathbf{f}^T\mathbf{b} \\ \text{s.t.} \quad & \boldsymbol{A}\mathbf{f} \geq \mathbf{b}, \\ & \mathbf{f} \geq \mathbf{0}, \end{aligned} \tag{2.8}$$

Thus, the representation of the LCP as a quadratic problem which can be solved in any QP solver will give the solution for contact forces. The contact forces will then be used in the ODE solvers and the simulation for lamella motion will continue.

2.9. Formulation of resting contact in the case of lamella

The lamella dynamics is a case of single contact in 2D. Thus, the $\boldsymbol{A}$ and the $\mathbf{b}$ from the previous section would just be constants for lamella dynamics.
As per [3], acceleration in normal direction ($\mathbf{a_N}(t_0)$) can be represented in terms of general acceleration as:

$$\mathbf{a_N}(t_0) = \hat{\mathbf{n}}(t_0).(\ddot{\mathbf{p}}_\mathbf{a}(t_0)) + 2\dot{\hat{\mathbf{n}}}(t_0).(\dot{\mathbf{p}}_\mathbf{a}(t_0)) \tag{2.9}$$

where $\mathbf{p_a}(t_0)$ is the position vector of the point of contact between body A and body B at time t_0. The right-most term $(2\dot{\hat{\mathbf{n}}}(t_0).(\dot{\mathbf{p}}_\mathbf{a}(t_0))$ in equation 2.9 is a velocity dependent term (i.e. one can immediately calculate it without knowing the forces involved), and will be part of (constant) $\mathbf{b}$ from the previous section.

In the case of motion of the lamellas The next part is understanding of the first term from equation 2.9. From the rigid body dynamics:

$$\ddot{\mathbf{p}}_\mathbf{a}(t) = \dot{\mathbf{v}}_\mathbf{a}(t) + \dot{\omega}_\mathbf{a}(t) \times \mathbf{r_a}(t) + \omega_\mathbf{a}(t) \times (\omega_\mathbf{a}(t) \times \mathbf{r_a}(t))$$

where $\mathbf{v_a}(t)$ is the velocity of the C.G of the body A, $\omega_\mathbf{a}(t)$ is the angular velocity of the body A with axis passing through the C.G. of the body. Initially, the development of the matrix $\boldsymbol{A}$ is discussed.

The vector $\ddot{\mathbf{p}}_\mathbf{a}(t)$ will depend on the contact forces $\mathbf{f}$ and the external forces on the body A. The dependence will be of the form *Force=Mass* × *Acceleration*. If only contact forces are considered and $\ddot{\mathbf{p}}_\mathbf{a}(t)$ is evaluated using the above equation, then the constant in front of the contact force will form the matrix $\boldsymbol{A}$ which in the case of lamellas will be just a constant A'. The constant as evaluated in [3] is given by,

$$A' = \frac{\hat{\mathbf{n}}(t_0)}{M_a} + (I_a^{-1}(t_0)(\mathbf{r_a} \times \hat{\mathbf{n}}(t_0)) \times \mathbf{r_a}).$$

For (constant) b, the second term as discussed before and the evaluation of $\ddot{\mathbf{p}}_\mathbf{a}(t)$ using the external forces are summed up. Thus, (constant) b as given in [3], is of the form,

$$b = \hat{\mathbf{n}}(t_0).\mathbf{q}(t_0) + 2\dot{\hat{\mathbf{n}}}(t_0).(\dot{\mathbf{p}}_\mathbf{a}(t_0)),$$
$$\mathbf{q} = \mathbf{ext} + \mathbf{cons},$$
$$\mathbf{ext} = \frac{(\mathbf{F_a})}{M_a} + \left(I_a^{-1}(t_0)\tau_\mathbf{a}(t_0)\right) \times \mathbf{r_a},$$
$$\mathbf{cons} = \omega_\mathbf{a}(t_0) \times (\omega_\mathbf{a}(t_0) \times \mathbf{r_a}) + \left(I_a^{-1}(t_0)(\mathbf{L_a}(t_0) \times \omega_\mathbf{a}(t_0))\right) \times \mathbf{r_a},$$

where $\mathbf{F_a}$ is the external forces acting on the body A while $\tau_\mathbf{a}$ represents the external torque acting on the body A. $\mathbf{L_a}(t_0)$ is the angular momentum associated with the body A. Thus, we have the dependence we require and we can proceed with the minimization problem as discussed in the previous section.

This completes the mathematical description of the problem of modeling the lamella dynamics. In the case of the motion of the lamellas, the resting contact problem is very simple as it is a case of pure rotation with single contact. Using the physical understanding of the lamella problem, the contact force during resting contact for the lamella is equal to the external force from water on the lamella. This simplification can be used to eliminate the need of quadratic programming. To sum it up,

the lamella dynamics are evaluated using an event-driven approach where the collision between the lamella and the mechanical stopper is evaluated using colliding or resting contact based on the criteria discussed before. This gives a lucid approach to correctly evaluate the lamella motion.

2.10. Conclusions

It is thus concluded that:

- To solve the problem of unphysical spikes, the torque transfer mechanism was studied. It was concluded that one needs to model the contact phenomenon applying the impact dynamics equations from the field of non-smooth dynamics.
- Similar to the baseline 2D model, the event-driven approach should be used. But, the approach should be adapted to model the partially elastic collisions of the lamellas. This type of collision is proposed to be modeled through the consideration of colliding and resting contacts.
- It was seen that the apparently complex problem of modeling lamella motion was reduced to simple ODE integration but with appropriate conditions applied at every time instant the lamella reaches its extreme positions.

3

Survey of numerical methods for system of ODEs with time periodic solutions

3.1. Introduction

In the first chapter it was pointed out that the baseline 2D numerical model took very long time for computations. This is attributed to the fact that the direct time integration schemes (leap-frog) employed in the model require very small time steps to correctly predict the motion of the rotor-lamella configuration and the solution to converge. The unsteady transient simulations require simulations of 10 periods of rotor rotations to achieve sufficient periodicity in the solutions. This can be another reason for longer computation times. Thus, it is our aim to try and develop a computationally more efficient model than the 2D baseline model.

With a defined aim, the literature study was carried out to search for the methods providing faster solutions for the OWM CFD model. In this chapter, these surveyed methods are briefly described with their properties and determine their applicability to the modeling of OWM. The applicability, for acceleration of the solution algorithm for the OWM model, is judged on certain requirements for the method to be used in the numerical model of OWM. These requirements are discussed in the next section.

3.2. Requirements for the numerical methods

The problems of the 2D baseline model have been studied thoroughly. These problems will influence the requirements for the numerical method used in the new numerical model. The requirements are as follows:

- The method should be the faster alternative amongst all the suitable options including the current implementation of leap-frog time integration schemes .
- The method should have low memory requirements. This is because the intention is to develop a model that is potentially extendable to 3D. The current baseline 2D model itself requires a lot of memory to store data for at least 10 iterations. The problem intensifies with simulations in 3D. Also, the memory requirements increase if an iterative procedure is used to solve the governing equations of OWM.
- The numerical model is realized with the OpenFOAM solver. The time available is decisive in what modifications can be done in the solver. With less than

six months available for the project, the method used in the new numerical model should be easy to implement on the OpenFOAM solver with minimum modifications to the solver.

It is imperative to know the equations involved in the numerical model of the OWM. The form of these equations and the boundary conditions will define the survey of methods for applicability to the OWM problem. The next section describes these equations which leads the way to the description of methods.

3.3. Equations defining the numerical model for OWM

In the previous chapter, it was noted that the lamella motion is governed by an ODE with conditions applied at the extreme positions of fully-closed and fully-open. The ODE from equation (2.4) is written in simplified form as:

$$\boldsymbol{M}\ddot{\mathbf{x}}(t) = \mathbf{h}(t) + \mathbf{w}(t); \tag{3.1}$$

where $\boldsymbol{M}$ is the mass matrix associated with the lamella, $\mathbf{x}(t)$ is the position vector of the lamella, $\mathbf{h}$ is the vector of external forces and $\mathbf{w}$ is the vector of contact forces. In the case of lamella motion, the vector of external forces is essentially the forces exerted by water on lamellas. This is a second order ODE which can be reduced to two first order ODEs. Thus, we have two first order ODEs for the structural part.

The incompressible Navier-Stokes equations (N-S) are the governing equations for fluid flow. It is a well known fact that the momentum equation from the N-S on semi-discretization in space leads to a system of non-linear ODEs. The N-S equations for incompressible flow, in Cartesian coordinates, are written in semi-discrete form as:

$$\frac{d\mathbf{u}}{dt} + \mathbf{R}(\mathbf{u}) = 0; \tag{3.2}$$

where $\mathbf{u}$ is the vector of conservative variables

$$\mathbf{u}(t) = [\rho, \rho\mathbf{u}, \rho\mathbf{v}, \rho\mathbf{w}]^T.$$

$\mathbf{R}(\mathbf{u})$ is the residual vector resulting from the spatial discretization of the convective and viscous fluxes. The pressure and viscous forces terms from the N-S give input for the external forces vector in the ODE governing the lamella motion. The equations (3.1) and (3.2) together will form a huge system of coupled non-linear ODEs.
The form of equations is in place, to solve these uniquely one needs boundary conditions. There are two possibilities - initial conditions or boundary conditions. The type of boundary condition depends on the problem. For any general turbine or rotor, it can be seen that after the initial transients die out a periodicity in the flow is obtained. This periodicity can be achieved by some periodic boundary conditions. The OWM is essentially a turbine and from its operation it can be concluded that the flow and the motion of lamella are periodic i.e. after one full rotation of the rotor the same flow conditions and lamella configurations are obtained. This implies the boundary conditions in this case are periodic.

Thus, the literature study should focus on methods solving a system of non-linear ODEs with periodic boundary conditions and satisfying the requirements from section 3.2. It is also hypothesized that such periodicity is useful to develop faster solutions. This hypothesis will be tested for validity in the discussions of methods in the following sections.

3.4. Standard methods for solving non-linear ODEs with periodic solutions

The problem of non-linear ODEs with periodic solutions occurs in many other engineering applications like electric circuits, pendulum systems, optimal control problems, and many more. There are different numerical methods developed to solve such systems. In this section the following three standard methods will be discussed:

1. Shooting method
2. Collocation method
3. Finite-difference method

3.4.1. Shooting method

The shooting method is mainly employed in circuit analysis. It is a way of solving the boundary value problem by reformulating it as an initial value problem. It is a generic method not necessarily used only for problems with periodic boundary conditions. A classic application of the shooting method in fluid dynamics, where non-periodic boundary conditions exist, is the solution to the Blasius equation obtained using the shooting method. The solution to this equation defines the Blasius boundary layer. For brief description of the shooting method, consider a system of non-linear ODEs represented as:

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), t), \tag{3.3}$$

where $\mathbf{x}$ and $\mathbf{f} \in \mathbb{R}^n$ and $\mathbf{f}$ is periodic in time t with period T. The shooting method is employed when an ODE is represented as a 2-point boundary value problem (BVP). The problem of obtaining periodic solutions of (3.3) is expressed as a 2-point BVP

$$\mathbf{x}(0) = \mathbf{x}(T). \tag{3.4}$$

With initial state $\mathbf{x}(0) = \mathbf{x_0}$, let $\mathbf{x}(t; \mathbf{x_0})$ be the solution of (3.3). Thus, integrating both sides of (3.3) from 0 to T,

$$\mathbf{x}(T; \mathbf{x_0}) = \int_0^T \mathbf{f}(\mathbf{x}(t; \mathbf{x_0}, t)dt + \mathbf{x_0}.$$

The right hand side of the equation depends only on $\mathbf{x_0}$. Now, define a function:

$$\mathbf{F}(\mathbf{x_0}) = \int_0^T \mathbf{f}(\mathbf{x}(t; \mathbf{x_0}, t)dt + \mathbf{x_0}. \tag{3.5}$$

The relation (3.4) can be is expressed as:

$$\mathbf{x_0} = \mathbf{F}(\mathbf{x_0}). \tag{3.6}$$

The above equation (3.6) can be solved by the Newton-Raphson method. The function $\mathbf{F}(\mathbf{x_0})$ can be obtained by numerical integration. The trapezoidal rule is applied in this discussion as it is one of the most widely used schemes for non-linear schemes. (ref)

For simplicity, choose a constant step size with N grid points then, $h \equiv \frac{T}{N}$. Substitute $t_i = ih$ and represent the approximate value of $\mathbf{x}(t_i)$ as $\mathbf{x_i}$. The function (3.5) on application of trapezoidal rule, looks like:

$$\mathbf{F}(\mathbf{x_0}) \approx \mathbf{x_0} + \sum_{i=1}^{N} \frac{h}{2}[\mathbf{f}(\mathbf{x_i}, t_i) + \mathbf{f}(\mathbf{x_{i-1}}, t_{i-1})]. \tag{3.7}$$

The Newton-Raphson method to solve (3.6) is given by:

$$\boldsymbol{J_F}(\mathbf{x_n})(\mathbf{x_{n+1}} - \mathbf{x_n}) = -\mathbf{F}(\mathbf{x_n}), \tag{3.8}$$

where $\boldsymbol{J_F}(\mathbf{x_n})$ is the Jacobian matrix for the function (3.5). One can either solve the linear algebraic system to compute $\mathbf{x_{n+1}}$ or compute the inverse of the Jacobian and perform matrix multiplication. This completes the description of the shooting method. The method though simple to implement, comes with its own short-comings and is rejected on the following grounds:

- Even though this method converges quadratically, it often becomes unstable because of the discretization error [13].
- The algorithm is stable but has problems in convergence if the step size of numerical integration is chosen too large. [13] Whereas, too small step sizes increase the computational time. Thus, a better and faster approach is needed.

Thus, it is concluded that more methods need to be studied for computations of non-linear ODEs with periodic solutions.

3.4.2. The collocation method

The collocation method is another method for finding periodic solutions of non-linear differential equations. This method requires that the approximate solution satisfies the ordinary differential equations at certain preselected *collocation points*. In the collocation method for a non-linear case one forms an equivalent linear problem whose solution will give the solution to the non-linear problem. Thus, the collocation method for non-linear case is reduced to solving a linear problem using a collocation method. This will be made more clear in the coming discussion. The concept of collocation points is described by figure 3.1.

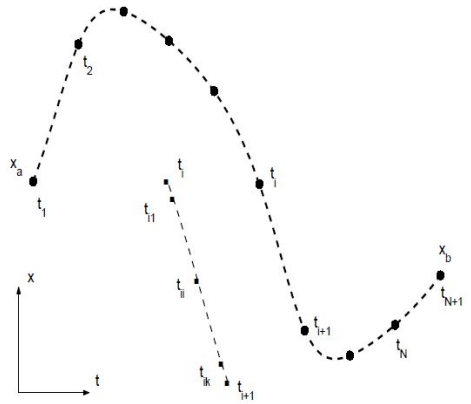


Figure 3.1: Shows the n intervals between x_a and x_b. In the collocation method, the interval $[t_i, t_{i+1}]$ is further sub-divided into k sub-intervals called as the collocation points.

For the illustration of the method, adopted from [18], a non-linear ODE defined by:

$$\mathbf{x}' = \mathbf{f}(\mathbf{x}) \quad \text{with} \quad \mathbf{x}(0) = \mathbf{x}(T), \tag{3.9}$$

is considered in dimensionless form:

$$\frac{d\mathbf{x}}{d\tau} = T\mathbf{f}(\mathbf{x}) \quad \text{with} \quad \mathbf{x}(0) = \mathbf{x}(1), \quad \tau = \frac{t}{T} \quad \text{and} \quad \tau \in [0, 1]. \tag{3.10}$$

The quasi-linearization method is employed for rewriting the solution of the non-linear BVP into a sequence of solutions of a linear BVP. To achieve this, the method uses Taylor series expansion of the non-linear term for small perturbations and truncate the second and higher order terms. Thus, for small perturbations $\mathbf{w}$ on variable $\mathbf{x}$ and small perturbation dT on the period time T from an initial guess $[\mathbf{x^0}(\tau), T^0]^T$, the m^{th} step is given by:

$$(\mathbf{x^m}(t) + \mathbf{w})' = \mathbf{f}(\mathbf{x^m}(t)) + \frac{\partial \mathbf{f}}{\partial x}(\mathbf{x}^m(t))\mathbf{w},$$

$$\Rightarrow \frac{d\mathbf{w}}{d\tau} = \boldsymbol{A}(\tau)\mathbf{w} + \mathbf{f}(\mathbf{x^m}(\tau))dT + \mathbf{q}(\tau) = \mathbf{f}^*(t, \mathbf{w}),$$

$$\text{with} \quad \boldsymbol{A}(\tau) = T^m \frac{\partial \mathbf{f}}{\partial x}(\mathbf{x^m}(\tau)) \quad \text{and} \tag{3.11}$$

$$\mathbf{q}(\tau) = T^m \mathbf{f}(\mathbf{x^m}(\tau)) - (\mathbf{x^m}(\tau))',$$

and the boundary conditions as

$$\mathbf{x}(0) - \mathbf{x}(1) = \mathbf{0} \rightarrow \mathbf{w_1} - \mathbf{w_{N+1}} = \mathbf{x^m_{N+1}} - \mathbf{x^m_1}.$$

The above equation (3.11) with the boundary conditions is a linear ODE. This method is similar to the Newton-Raphson method, a residual $\mathbf{q}(\tau)$ in the form of difference between the ODE and the derivative of the solution and a Jacobian operator ($\boldsymbol{A}$) acting on the correction step $\mathbf{w}$. The residual in this case is zero if the solution satisfies the ODE at collocation points. After some lengthy derivations, which are skipped here, one obtains the following set of equations for the computation of the periodic solution.

$$\mathbf{w_{i+1}} = \mathbf{w_i} + h_i \sum_{l=1}^{k} \alpha_{jl} \mathbf{f}^*(t_{il}, \mathbf{w_{il}})$$

$$= \mathbf{w_i} + h_i[\beta_1 \boldsymbol{I} \ldots \beta_k \boldsymbol{I}][\boldsymbol{W_i}^{-1}\boldsymbol{V_i}\mathbf{w_i} + \boldsymbol{W_i}^{-1}\mathbf{U_i}dT + \boldsymbol{W_i}^{-1}\mathbf{q_i}] \tag{3.12}$$

$$= \boldsymbol{\Gamma_i}\mathbf{w_i} + \boldsymbol{\Lambda_i}dT + \boldsymbol{r_i},$$

where the description of the above terms is:

- h_i is the step size of the i^{th} collocation step.
- $\alpha_{jl} = \int_0^{\rho_j} L_l(\rho)d\rho$ with $L_l(\rho)$ being a Lagrangian interpolation polynomial of $(k-1)$ degree. Similarly, $\beta_l = \int_0^1 L_l(\rho)d\rho$. It can be seen that the equation (3.12) has an identical form to the Runge-Kutta method. To calculate the values of α and β one has to decide on the quadrature rule for the numerical integration which will give the values of k and ρ. For example, for Simpson's quadrature scheme $k = 3, \rho_1 = 0, \rho_2 = 0.5, \rho_3 = 1$.
  It should be noted that ρ, β, α, together, form a butcher array of R-K method.

The Butcher array looks like,

ρ_j	α_{11}	...	α_{13}	
0	0	0	0	α_{13}
0.5	5/24	1/3	–1/24	⋮
1	1/6	2/3	1/6	α_{33}
β_l	1/6	2/3	1/6	

(3.13)

- $\boldsymbol{W_i} \in \mathbb{R}^{nk\times nk}$, where it is assumed that it is a system of n non-linear differential equations. $\boldsymbol{V_i} \in \mathbb{R}^{nk\times n}$ and $\mathbf{q_i} \in \mathbb{R}^{nk}$. The individual matrices are defined by:

$$\boldsymbol{W_i} = \boldsymbol{I} - h_i \begin{bmatrix} \alpha_{11}\boldsymbol{A}(t_{i1}) & \dots & \alpha_{1k}\boldsymbol{A}(t_{i1}) \\ \vdots & & \vdots \\ \alpha_{1k}\boldsymbol{A}(t_{ik}) & \dots & \alpha_{kk}\boldsymbol{A}(t_{ik}) \end{bmatrix}, \quad \boldsymbol{V_i} = \begin{bmatrix} \boldsymbol{A}(t_{i1}) \\ \vdots \\ \boldsymbol{A}(t_{ik}) \end{bmatrix}, \quad \mathbf{q_i} = \begin{bmatrix} \mathbf{q}(t_{il}) \\ \vdots \\ \mathbf{q}(t_{ik}) \end{bmatrix}. \tag{3.14}$$

- $\boldsymbol{U_i} \in \mathbb{R}^{nk}$ is defined by:

$$\boldsymbol{U_i} = \begin{bmatrix} \mathbf{f}(\mathbf{x}^m(t_{i1})) \\ \vdots \\ \mathbf{f}(\mathbf{x}^m(t_{ik})) \end{bmatrix}. \tag{3.15}$$

- $\boldsymbol{\Gamma_i} = I + h_i \boldsymbol{D}\boldsymbol{W_i}^{-1}\boldsymbol{V_i}$, $\mathbf{r_i} = h_i \boldsymbol{D}\boldsymbol{W_i}^{-1}\mathbf{q_i}$, and $\boldsymbol{\Lambda_i} = h_i \boldsymbol{D}\boldsymbol{W_i}^{-1}\boldsymbol{U_i}$ with $\boldsymbol{D} = [\beta_1 \boldsymbol{I} \dots \beta_k \boldsymbol{I}]$.

The system (3.12) together with the boundary conditions cannot be solved uniquely as it consists of n(N+1) equations in n(N+1)+1 unknowns. To overcome this problem a phase-start condition, also known as an anchor equation, is added to the system of equations. Generally, an anchor equation is the orthogonality condition $\mathbf{f}(\mathbf{x_1})^T\mathbf{w_1} = 0$. The final linear set of equations which must be solved sequentially looks like:

$$\begin{bmatrix} -\boldsymbol{\Gamma}_1 & \boldsymbol{I} & 0 & 0 & -\boldsymbol{\Lambda}_1 \\ 0 & \ddots & \ddots & 0 & \vdots \\ 0 & 0 & -\boldsymbol{\Gamma}_N & \boldsymbol{I} & -\boldsymbol{\Gamma}_N \\ \boldsymbol{I} & 0 & 0 & -\boldsymbol{I} & 0 \\ \mathbf{f}(\mathbf{x_1}^m) & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{w_1} \\ \vdots \\ \vdots \\ \mathbf{w_{N+1}} \\ dT \end{bmatrix} = \begin{bmatrix} \mathbf{r_1} \\ \vdots \\ \mathbf{r_N} \\ \mathbf{x}^m_{N+1} - \mathbf{x}^m_1 \\ 0 \end{bmatrix}. \tag{3.16}$$

After each iteration the approximate solution $\mathbf{x}^m(t)$ must be updated as, $\mathbf{x}^{m+1}(t_i) = \mathbf{x}^m(t_i) + \mathbf{w_i}$ while $\mathbf{f_i}$ and the time period T as

$$\mathbf{f}_i^{m+1} = \mathbf{f}_i^m + \boldsymbol{W}_i^{-1}\boldsymbol{V}_i\mathbf{w_i} + \boldsymbol{W}_i^{-1}\boldsymbol{U}_i dT + \boldsymbol{W}_i^{-1}\mathbf{q_i},$$

$$T^{m+1} = T^m + dT.$$

This completes a brief description of the collocation method. The above implementation of the collocation method was compared against the shooting method and the finite-difference method in [18]. It is concluded in [18] that the solution s obtained using the collocation method when compared to the true solution at all the mesh points was more accurate than the solution from the shooting method and the finite-difference method. The accuracy was compared in terms of the relative error of the computed solution versus the true solution at all mesh points.

It is also concluded in [18] that the collocation method is more robust as compared to the other two methods-the shooting method and the finite-difference method. Robustness here, is defined by the speed of convergence of the method with a bad initial

guess. An initial guess which is not reasonably close to the true solution is termed as a bad initial guess. The step size required for the collocation method to converge with sufficient accuracy is more than the shooting and finite-difference method, implying faster convergence. Hence, the collocation method is more accurate, more robust and faster than the shooting and the finite-difference method.

Nonetheless this method suffers from drawbacks. The computations of Jacobian matrices are done analytically. If numerically determined approximations are used the CPU time increases but the accuracy and robustness is not hampered. It is important to consider the efficacy of this method for the application to case of OWM. In this case, the application of collocation method will require a significant modification in the existing solver which violates one of the requirements discussed in section 3.2. Thus, it is concluded that further research into better and faster methods is necessary.

3.4.3. The finite-difference method

The finite-difference method for solving partial differential equations can be used to solve non-linear ODEs represented as 2-point boundary value problems instead of initial value problem. It can also be extended to solve non-linear ODEs with periodic boundary conditions. The method applied to a non-linear ODE is described below. Consider a scalar non-linear ODE for the ease of representation:

$$\dot{x}(t) = f(x(t), t), \tag{3.17}$$

along with the boundary condition,

$$x(0) = x(T). \tag{3.18}$$

As the name suggests, this method uses the *difference formulae* from the usual finite-difference method applied to partial differential equations. The difference formulas are not discussed here, and, one is referred to any standard finite-difference method books for further details. The step by step procedure for the finite-difference method is as follows:

STEP 1:
Divide the interval $[0, T]$ into N equal subintervals, which gives:

$$h = \frac{T}{N},$$

with end points at $t_i = ih$ for $i = 0, 1, 2, \ldots, N$.

STEP 2:
Using the forward difference formula gives:

$$\frac{x(t_{i+1}) - x(t_i)}{h} = f(x_i, t_i). \tag{3.19}$$

STEP 3:
The periodic boundary conditions are given by:

$$x_0 = x_T = \alpha \text{ (phase start)}.$$

STEP 4:
On defining the boundary conditions, an $N \times N$ nonlinear system, $\mathbf{F}(\mathbf{x})$ is produced using the finite-difference method as:

$$\begin{aligned} x_1 - x_N - hf(x_N, t_0) &= 0 \\ x_2 - x_1 - hf(x_1, t_1) &= 0 \\ &\vdots \\ x_N - x_{N-1} - hf(x_{N-1}, t_{N-1}) &= 0 \end{aligned} \tag{3.20}$$

(3.21)

STEP 5:
Apply Newton's iterative method to approximate the solution of the system $\mathbf{F}(\mathbf{x}) = \mathbf{0}$. The method is:

$$\begin{aligned} \mathbf{x_i^k} &= \mathbf{x_i^{k-1}} + \mathbf{v_i}, \quad \text{for} \quad i = 1, 2, ..., N. \\ \boldsymbol{J}(x_1, ..., x_N)(v_1, ...v_N)^T &= -\mathbf{F}(x_1, ..., x_N). \end{aligned} \tag{3.22}$$

This gives one iteration of Newton's iterative method with the initial approximation $\mathbf{x}^0 = \alpha\mathbb{1}$. $\boldsymbol{J}$ is the Jacobian matrix associated with the function $\mathbf{F}$.

The Jacobian matrix is a sparse matrix of the form:

$$\boldsymbol{J}(\mathbf{x}) = \begin{bmatrix} 1 & 0 & ... & 0 & b_{N,0} \\ b_{1,1} & 1 & 0 & ... & 0 \\ 0 & \ddots & \ddots & 0 & \vdots \\ \vdots & 0 & b_{N-2,N-2} & 1 & 0 \\ 0 & ... & 0 & b_{N-1,N-1} & 1 \end{bmatrix}, \tag{3.23}$$

where $b_{ij} = -1 - hf'(x_i, t_j)$. With the Jacobian matrix one can find $\mathbf{v_i}$ and thus, iterate according to (3.22) until $\mathbf{x^k}$ converges to the solution of $\mathbf{F}(\mathbf{x}) = 0$. The stopping criterion for this iteration is $||\mathbf{x^k} - \mathbf{x^{k-1}}||_2 \leq \epsilon$, where ϵ is the required accuracy of the solution. As compared to the collocation method, it is easier to implement the finite difference method on the existing solver. The form of the Jacobian hints that it is needed to store an n– dimensional vector to develop the Jacobian matrix so it requires less memory storage as compared to the collocation method. Newton's method being an iterative procedure it is clear that at every step we need to store the vector corresponding to the Jacobian, the vector $\mathbf{v_i}$ and finally, the vector corresponding to the solution at the previous time step i.e. $\mathbf{x^k}$. This memory storage requirement will be compared to the memory storage required for methods discussed in the next sections.

The three basic methods introduced at the start of this section have been discussed and it can be concluded that collocation method is the most accurate and robust of them all. But the finite difference method is more suited as it is a better fit for our requirements (3.2). In the next section, specific methods developed and applied to problems similar to flow in the OWM are discussed.

3.5. Dedicated methods for fast analysis of periodic flows

Simulations for turbine and rotor flows have been done for a few decades. It is necessary to study what kind of approaches are used in such simulations and if there are special techniques which fasten the solution process. This section deals with modern methods applied for rotor simulations like the helicopter rotor simulations and turbomachinery. It should be noted that all these methods employ periodicity of the flow or domain to obtain fast and efficient solutions. In generic sense, these are designed for PDEs with time periodic flow solutions. Some of these methods have their roots to the standard methods discussed in the previous section. The methods discussed in this section are:

1. Multitime multigrid method
2. Time linearization method
3. Time spectral method

3.5.1. Multitime multigrid method for time-periodic flow simulations

This method is essentially an algorithm discussed in [24], which extends the multigrid method to multitime. The algorithm exploits the periodic nature of the rotor flow. Here, the dynamics are not solved time step after time step, but are solved for all the time steps at once. The basic idea is to treat time like space and the equations are solved on a 4-dimensional mesh (instead of 3-dimensions) using the multigrid techniques. This means the 3D spatial grid is changed to 4D grid. The method as proposed by the author is discussed.

The Euler equations for fluid flow in 2-dimensions are considered for ease of representation. In 3-dimensions only an extra dimension will be added without any modifications to the method. The equations are given by:

$$\frac{\partial \mathbf{U}}{\partial t} + \frac{\partial \mathbf{F}}{\partial x} + \frac{\partial \mathbf{G}}{\partial y} = 0; \tag{3.24}$$

where **U** is the vector of conservation variables and **F** and **G** are called as the flux vectors. These are given as:

$$\mathbf{U} = \begin{bmatrix} \rho \\ \rho u \\ \rho v \end{bmatrix}; \quad \mathbf{F} = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \end{bmatrix}; \quad \mathbf{G} = \begin{bmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \end{bmatrix}; \tag{3.25}$$

where u and v are velocities in the x and y directions respectively, ρ is the density of the fluid, p is the pressure acting at the grid point under consideration.
In a non-standard way, these equations can be represented as:

$$\tilde{\nabla}.\boldsymbol{F} = 0; \quad \text{where}$$

$$\boldsymbol{F} = \begin{bmatrix} \rho & \rho u & \rho v \\ \rho u & \rho u^2 + p & \rho uv \\ \rho v & \rho uv & \rho v^2 + p \end{bmatrix}, \quad \tilde{\nabla} = \begin{bmatrix} \frac{\partial}{\partial t} \\ \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \end{bmatrix}. \tag{3.26}$$

Now the author applies a discontinuous Galerkin finite element approximation to (3.26) to obtain the discretized equation of the form:

$$\boldsymbol{L}_h(\mathbf{U}_h) = 0. \tag{3.27}$$

Note that a 3D grid will be required in this case, as it is 2D problem with the extra time dimension. The author in [24] uses the multigrid algorithm to treat the above system. Once this system is achieved we can solve it using pseudo-time stepping or Newton's method as discussed in the finite-difference approach. In fact, this approach can essentially be regarded as similar to finite-difference approach if, instead of finite element discretization the finite difference discretization is used.

4D grids in this case are generated 'by hand'. In the most simple way one can construct a 4D grid by collating a series of spatial 3D grids generated for each time instant. The 3D grids (spatial grids) are generated in the conventional way. In this way, each space-time element in the 4D mesh is the linear interpolation in time of 3D element at a time $t = t_n$ and a 3D element at time $t = t_{n+1}$.

The author claims that one order of magnitude reduction in time compared to the classic multigrid method could be achieved at the expense of one order of magnitude increase in memory usage. In essence, the author proposes to solve the problem for all time steps simultaneously, recognizing that periodic problems can be considered as steady state problems in the space–time domain.

As discussed before, this method is similar to the finite difference method. It is indeed an efficient method to solve, but is rejected on the grounds of increases memory usage, as referred before, and difficult to implement on an existing solver, both of which are important requirements as discussed in section 3.2.

3.5.2. The time linearization method

This method was proposed by Hall, et al. [9] in 1992 and is predominantly used in the investigation of the onset of flutter in turbomachinery. As OWM is essentially a turbomachine, the method was studied.

The method involves linearization of Euler equations for the analysis of unsteady flows. The linearization implies that the results are only valid for small perturbations of the geometry and the solution.
The equation (3.24) is linearized by splitting the solution variables i.e. $\mathbf{U}$ into the mean flow part $\bar{\mathbf{U}}$ over a known period and a perturbation $\tilde{\mathbf{U}}$;

$$\mathbf{U}(t) = \bar{\mathbf{U}} + \tilde{\mathbf{U}}(t)$$

The linearized form of Euler equations (3.24) using the above representation of conservative variables gives:

$$\begin{aligned} \frac{\partial \mathbf{F}(\bar{\mathbf{U}})}{\partial x} + \frac{\partial \mathbf{G}(\bar{\mathbf{U}})}{\partial y} &= 0; \\ \frac{\partial \tilde{\mathbf{U}}}{\partial t} + \frac{\partial (\boldsymbol{A}\tilde{\mathbf{U}})}{\partial x} + \frac{\partial (\boldsymbol{B}\tilde{\mathbf{U}})}{\partial y} &= 0; \end{aligned} \tag{3.28}$$

where the first is the mean flow equation and $\boldsymbol{A}$ and $\boldsymbol{B}$ are the Jacobian matrices evaluated using the mean flow solution,

$$\boldsymbol{A} = \frac{\partial \mathbf{F}}{\partial \bar{\mathbf{U}}}; \quad \boldsymbol{B} = \frac{\partial \mathbf{G}}{\partial \bar{\mathbf{U}}};$$

The unsteadiness in the flow is temporally harmonic because of the periodicity of the flow. This gives the opportunity to model the perturbations as harmonic amplitudes of the conservation variables. The perturbation $\tilde{\mathbf{U}}$ is thus, expanded in Fourier series as:

$$\tilde{\mathbf{U}}(t) = \sum_{k=-\infty}^{\infty} \hat{\mathbf{U}}_k e^{ik\omega t}. \tag{3.29}$$

The procedure requires to first solve the mean flow equation for $\bar{\mathbf{U}}$. This makes it possible to solve for any k by using the mean flow solution and calculate the Jacobian matrices $\boldsymbol{A}$ andd $\boldsymbol{B}$. Because of the linearity of the equation (3.28), all the temporal modes are decoupled. Thus, they can be calculated independently of each other.

The above briefly described method is applied for flows with airfoils under pitching and some inviscid and viscous cases as given in [16]. The method is efficient and relatively easy to implement in the solver [16]. This method provides results in good agreement with the time-marching analysis, when the unsteady part is of the nature of small perturbations about a mean solution like in the case of airfoil pitching about its mean position. The method is computationally efficient and the memory overhead is very modest [16]. But for cases when the unsteady part is not associated with small perturbations about some mean solution like in the case of the motion of lamella, the method is known to have difficulties to capture the flow features. On this ground, time linearization method is rejected for its application to the numerical model of OWM.

3.5.3. The time spectral method

The time spectral method developed by Jameson and Gopinath [2] in 2005, is a method for fast and efficient computations of time periodic turbulent flow calculations for flows past 2- and 3-dimensional bodies. This is a generalization of an initial approach proposed by Hall, et al., [10] in 2001 referred by the name of *harmonic balance approach* for unsteady periodic flows. The time spectral method is a larger class of methods under which the harmonic balance technique falls. The difference between the two approaches is that one is a frequency domain method and the other is time domain. It will be discussed later in this section. There have been many developments in the original harmonic balance approach as proposed by Hall, et al. In the following, time spectral method applied to N-S equation is discussed and improvements to the method proposed by few other researchers are presented.

The basic concept of this method is - modify the time derivative term as a source term and thus, form a set of steady state equations. So the harmonic balance method solves a coupled set of steady state equations, where the temporal term is modeled as an additional source term and the periodicity is externally enforced. As the periodicity is enforced it is not required to simulate for many periods until periodicity is achieved, as is the case with standard transient unsteady simulations.

The technique was initially developed for modeling unsteady non-linear periodic flows in turbomachinery applications and airfoil pitching, but has now found many other applications. The main requirement for this method is the periodic nature of the flow. As already mentioned before, the case of rotor-lamella motion is periodic in nature and so, this method is investigated for suitability.

As the solutions are periodic in nature, they are represented in Fourier series. This is one of the main aspects of the time spectral method. One can say that this method derives its concept from the collocation method, where the solutions are represented as polynomials.

The description of the method given here is adopted from [4]. From the equation (3.2) the vectors **u** and **R** are represented as a Fourier series in time.

Fourier series expansion of **u**(t) with n harmonics reads:

$$\mathbf{u}(t) = \sum_{j=0}^{n} \mathcal{u}_j e^{ij\omega t}, \tag{3.30}$$

and the expansion for R(t) reads:

$$\mathbf{R}(t) = \sum_{j=0}^{n} \mathcal{R}_j e^{ij\omega t}, \tag{3.31}$$

where $\mathcal{u}_j$ and $\mathcal{R}_j$ are the Fourier co-efficients given by:

$$\begin{aligned} \mathcal{u}_j &= u_{s_j} - iu_{c_j}; \quad j > 0; \\ \mathcal{u}_0 &= u_0; \qquad\quad j = 0; \end{aligned} \tag{3.32}$$

$$\begin{aligned} \mathcal{R}_j &= R_{s_j} - iR_{c_j}; \quad j > 0; \\ \mathcal{R}_0 &= R_0; \qquad\quad j = 0; \end{aligned} \tag{3.33}$$

Inserting equations (3.30) and (3.31) in equation (3.2):

$$\omega \sum_{j=0}^{n} ij\mathcal{u}_j e^{ij\omega t} + \sum_{j=0}^{n} \mathcal{R}_j e^{ij\omega t}.$$

The above equation can be expanded using (3.32) and (3.33) in terms of the sine and cosine terms. The respective sine and cosine terms when equated to zero gives a set of equations which look like:

$$\text{n equations for sine} \begin{cases} -1\omega u_{c_1} + R_{s_1} = 0; \\ -2\omega u_{c_2} + R_{s_2} = 0; \\ \vdots \\ -n\omega u_{c_n} + R_{s_n} = 0; \end{cases} \tag{3.34}$$

$$\text{center-} \qquad R_0 = 0 \tag{3.35}$$

$$\text{n equations for cosine} \begin{cases} 1\omega u_{s_1} + R_{c_1} = 0; \\ 2\omega u_{s_2} + R_{c_2} = 0; \\ \vdots \\ n\omega u_{s_n} + R_{c_n} = 0; \end{cases} \tag{3.36}$$

A Fourier series expansion up to n harmonics involves 2n + 1 terms and so, the above set of equations should also be 2n + 1. It is seen that this is indeed true. These equations can be represented in a matrix form as:

$$\omega \boldsymbol{A} \mathrm{u} + \mathrm{R} = 0. \tag{3.37}$$

The matrices above will have the following form:

$$\boldsymbol{A} = \begin{bmatrix} & & & & 0 & -1 & 0 & \dots & 0 \\ & & & & 0 & \ddots & -2 & \ddots & \vdots \\ & & 0 & & \vdots & & \ddots & \ddots & 0 \\ & & & & \vdots & & & 0 & n \\ 0 & 0 & \dots & 0 & 0 & 0 & \dots & 0 & 0 \\ 1 & \ddots & & & \vdots & & & & \\ 0 & 2 & \ddots & & \vdots & & & & \\ \vdots & \ddots & \ddots & 0 & 0 & & 0 & & \\ 0 & \dots & 0 & n & 0 & & & & \end{bmatrix}; \quad \mathrm{u} = \begin{bmatrix} u_{s_1} \\ u_{s_2} \\ \vdots \\ u_{s_n} \\ u_0 \\ u_{c_1} \\ u_{c_2} \\ \vdots \\ u_{c_n} \end{bmatrix}; \quad \mathrm{R} = \begin{bmatrix} R_{s_1} \\ R_{s_2} \\ \vdots \\ R_{s_n} \\ R_0 \\ R_{c_1} \\ R_{c_2} \\ \vdots \\ R_{c_n} \end{bmatrix}. \tag{3.38}$$

Equation (3.37) gives a set of 2n + 1 equations which require to store 2n + 1 Fourier coefficients for each flow variable (**u** and **R**). The variables have been transformed to frequency domain. This was the proposition in the harmonic balance approach by Hall et. al [10]. Gopinath et. al [2] went further to have the variables in time domain. This led the method to be referred as time spectral method.

The time spectral method now proposes to reconstruct the Fourier coefficients of $\mathcal{u}$ and $\mathcal{R}$ from knowledge of the temporal behavior of **u**(t) and **R**(t) at 2n + 1 equally spaced points over one temporal period. Let $\hat{\mathbf{u}}$(t) and $\hat{\mathbf{R}}$(t) be the values at the 2n + 1 points over the temporal period. A discrete Fourier transform operator ($\boldsymbol{E}$) is required to connect these time domain and frequency domain vectors. The dimensions of this operator are (2n + 1 × 2n + 1). Thus, the transformation looks like:

$$\mathrm{u} = \boldsymbol{E}\hat{\mathbf{u}}(\mathrm{t}).$$

The operator $\boldsymbol{E}$ is given by:

$$\boldsymbol{E} = \frac{2}{2n+1} \begin{bmatrix} \sin(\omega t_1) & \sin(\omega t_2) & \sin(\omega t_3) & \dots & \sin(\omega t_{2n+1}) \\ \sin(2\omega t_1) & \sin(2\omega t_2) & \sin(2\omega t_3) & \dots & \sin(2\omega t_{2n+1}) \\ \vdots & \vdots & \vdots & & \vdots \\ \sin(n\omega t_1) & \sin(n\omega t_2) & \sin(n\omega t_3) & \dots & \sin(n\omega t_{2n+1}) \\ \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & \dots & \frac{1}{2} \\ \cos(\omega t_1) & \cos(\omega t_2) & \cos(\omega t_3) & \dots & \cos(\omega t_{2n+1}) \\ \cos(2\omega t_1) & \cos(2\omega t_2) & \cos(2\omega t_3) & \dots & \cos(2\omega t_{2n+1}) \\ \vdots & \vdots & \vdots & & \vdots \\ \cos(n\omega t_1) & \cos(n\omega t_2) & \cos(n\omega t_3) & \dots & \cos(n\omega t_{2n+1}) \end{bmatrix}; \tag{3.39}$$

The matrix $\boldsymbol{E}$ is a Fourier transform operator, thus, its inverse will be the inverse Fourier transform operator. The inverse operator looks like:

$$\boldsymbol{E}^{-1} = \begin{bmatrix} \sin(\omega t_1) & \dots & \sin(n\omega t_1) & 1 & \cos(\omega t_1) & \dots & \cos(n\omega t_1) \\ \sin(\omega t_2) & \dots & \sin(n\omega t_2) & 1 & \cos(\omega t_2) & \dots & \cos(n\omega t_2) \\ \sin(\omega t_3) & \dots & \sin(n\omega t_3) & 1 & \cos(\omega t_3) & \dots & \cos(n\omega t_3) \\ \vdots & & \vdots & \vdots & \vdots & & \vdots \\ \vdots & & \vdots & \vdots & \vdots & & \vdots \\ \sin(\omega t_{2n+1}) & \dots & \sin(n\omega t_{2n+1}) & 1 & \cos(\omega t_{2n+1}) & \dots & \cos(n\omega t_{2n+1}) \end{bmatrix}; \tag{3.40}$$

This allows the representation of the equation (3.37) as:

$$\omega \boldsymbol{AE}\hat{\mathbf{u}} + \boldsymbol{E}\hat{\mathbf{R}} = 0.$$

Multiplying from left with the inverse transform operator ($\boldsymbol{E}^{-1}$):

$$\omega(\boldsymbol{E}^{-1}\boldsymbol{AE})\hat{\mathbf{u}} + \hat{\mathbf{R}} = 0. \tag{3.41}$$

Using the trigonometric sum identity (sin(A + B) = sin(A)cos(B) + cos(A)sin(B)), the expressions in operator ($\boldsymbol{E}^{-1}\boldsymbol{AE}$) are simplified. Ultimately, the operator looks like:

$$\boldsymbol{E}^{-1}\boldsymbol{AE} = \frac{2}{2n+1}\begin{bmatrix} 0 & B_1 & B_2 & B_3 & \dots & \dots & B_{2n} \\ -B_1 & 0 & B_1 & B_2 & B_3 & & B_{2n-1} \\ -B_2 & -B_1 & 0 & B_1 & B_2 & & \vdots \\ -B_3 & -B_2 & -B_1 & 0 & B_1 & & \vdots \\ \vdots & & & & \ddots & & B_2 \\ \vdots & & & & & \ddots & B_1 \\ -B_{2n} & \dots & \dots & -B_3 & -B_2 & B_1 & 0 \end{bmatrix}; \tag{3.42}$$

where $B_i = \sum_{k=1}^{n} k\sin(k\omega j t_1)$; $\quad j = \{1, \dots, 2n\}$.

Thus, the time derivative term in the momentum equation of Navier-Stokes has been transformed to look as a source term and a set of 2n + 1 steady state equations are required to be solved. Note that the continuity equation is not affected and remains the same. The above set of 2n + 1 equations are solved simultaneously in a steady state solver. As already mentioned, solving steady state equations is faster than solving unsteady equations. The general numerical procedure can now be summarized as:

- 2n + 1 computational grids are generated using a suitable grid generator.
- At each node of these 2n + 1 grids, store the value of the **u**.
- The equation (3.41) is now developed by using a suitable Finite Volume discretization technique for the residual operator.
- Suitable boundary conditions are applied. If there is a spatial periodicity, then such boundary conditions should be applied taking appropriate care.
- Now the equations are solved using pseudo time marching (ref) until the solutions converge. As steady state solutions are desired, it is advisable to use local time stepping and multi-grid acceleration techniques for speeding the convergence.

The equations (3.42) and (3.41) can be solved using a pseudo time marching method. It should be noted here, that the operator $\hat{\mathbf{R}}$ is actually a function of $\hat{\mathbf{u}}$. Applying the pseudo time marching to equation (3.41):

$$\frac{d\hat{\mathbf{u}}}{d\tau} + \omega \boldsymbol{B}\hat{\mathbf{u}} + \hat{\mathbf{R}} = 0;$$
$$\text{where} \quad \boldsymbol{B} = \boldsymbol{E}^{-1}\boldsymbol{AE}. \tag{3.43}$$

This equation is solved using an implicit method for which one step is written as:

$$\frac{\hat{\mathbf{u}}^{k+1} - \hat{\mathbf{u}}^{k}}{\Delta\tau} = -[\omega \boldsymbol{B}\hat{\mathbf{u}} + \hat{\mathbf{R}}(\hat{\mathbf{u}}^{k+1})]. \tag{3.44}$$

Note that the time level for the source term $\omega \boldsymbol{B}\hat{\mathbf{u}}$ is not yet defined as it can be explicit or implicit. This discussion will be done later in the section. As $\hat{\mathbf{R}}(\hat{\mathbf{u}}^{k+1})$ is unknown, it is linearized using a Taylor Series expansion. The linearization is then:

$$\hat{\mathbf{R}}(\hat{\mathbf{u}}^{k+1}) = \hat{\mathbf{R}}(\hat{\mathbf{u}}^{k}) + \boldsymbol{J_{ts}}\Delta\hat{\mathbf{u}} + \mathcal{O}(\Delta\hat{\mathbf{u}}^{2}), \tag{3.45}$$

where $\boldsymbol{J_{ts}}$ is the Jacobian matrix of the residual vector. The block diagonal form of the Jacobian matrix is given by:

$$\boldsymbol{J_{ts}} = \begin{bmatrix} \left.\frac{\partial \mathbf{R}}{\partial \mathbf{u}}\right|_{t_1} & 0 & \dots & 0 \\ 0 & \left.\frac{\partial \mathbf{R}}{\partial \mathbf{u}}\right|_{t_2} & \ddots & \vdots \\ \vdots & & \ddots & 0 \\ 0 & \dots & 0 & \left.\frac{\partial \mathbf{R}}{\partial \mathbf{u}}\right|_{t_{2n+1}} \end{bmatrix} \tag{3.46}$$

There are two options of dealing with the source term $\omega \boldsymbol{B}\hat{\mathbf{u}}$.

Explicit

This is a quite straightforward approach and the equation (3.44) using the linearization (3.45) and explicit source term can be written as:

$$\left[\frac{\mathrm{V}\boldsymbol{I}}{\Delta\tau} + \boldsymbol{J_{ts}}\right]\Delta\hat{\mathbf{u}} = -\hat{\mathbf{R}}^{k} - \omega \boldsymbol{B}\hat{\mathbf{u}}^{k}, \tag{3.47}$$

where V is the volume of the cell. The right hand side of the equation is the standard residual operator calculated at the kth snapshot plus the approximation of the source term. The left hand side has an augmented matrix composed of the Jacobian at the kth snapshot and the ratio of volume and pseudo time step. The form of the operator is also block diagonal like the Jacobian operator. The block diagonal nature of the matrix makes it possible to solve independently for each of the 2n + 1 stationary solutions . The 2n+1 steady flows are only coupled through the Fourier approximation of the unsteady residual of the discretized system. This method has a clear advantage as only the kth snapshot of the Jacobian has to be stored; thus, no extra memory is required for the linear solver over the standard method. But, the explicit treatment of the source term restricts the size of the Courant-Friedrich-Lewy (CFL) number, and the problem becomes intensive with increase in the number of Fourier modes (harmonics). The restriction on CFL number restricts the time step and thus, the time for computations [16]. Nonetheless, this method is easy to implement on a standard solver and can be easily parallelized as noted in [1]

Implicit

The next option is to treat the source term $\omega \boldsymbol{B}\hat{\mathbf{u}}$ implicitly. This would require linearization of the source term. As the operator $\boldsymbol{B}$ is linear, the source term linearization looks like:

$$\boldsymbol{B}\hat{\mathbf{u}}^{k+1} = \boldsymbol{B}\hat{\mathbf{u}}^{k} + \boldsymbol{B}\Delta\hat{\mathbf{u}}. \tag{3.48}$$

Substituting equations (3.45) and (3.48) in (3.44) one obtains:

$$\left[\frac{\mathrm{V}\boldsymbol{I}}{\Delta\tau} + \boldsymbol{J_{ts}}\right]\Delta\hat{\mathbf{u}} + \mathrm{V}\boldsymbol{B}\Delta\hat{\mathbf{u}} = -\hat{\mathbf{R}}^{k} - \omega \boldsymbol{B}\hat{\mathbf{u}}^{k}; \tag{3.49}$$

In this case the operator on the left hand side is not block diagonal instead it is a full matrix of the form:

$$\boldsymbol{B_{imp}} = \left[\frac{V\boldsymbol{I}}{\Delta\tau} + \boldsymbol{J_{ts}} + \boldsymbol{B}\right] = \begin{bmatrix} E_1 & H_{1,2} & H_{1,3} & \cdots & H_{1,2n+1} \\ H_{2,1} & E_2 & & \cdots & \vdots \\ \vdots & & \ddots & & \vdots \\ & & & E_{2n} & H_{2n,2n+1} \\ H_{2n+1,1} & \cdots & \cdots & & E_{2n+1} \end{bmatrix}; \tag{3.50}$$

where $E_i = \frac{V}{\Delta\tau_i} + \boldsymbol{J_{ts_{i,i}}}$ and $H_{i,j} = V\omega\boldsymbol{B}_{i,j}$. Being a full matrix it poses difficulties to solve the problem. The memory requirements are more as compared to the explicit approach. Also, this approach is heavy on the CPU time as compared to explicit approach. There are different ways to solve the system (3.49). Sicot et al. [6] proposes to use an iterative block-Jacobi strategy for solving while Woodgate et al. [17] proposes to solve using Krylov subspace method with BILU factorization.

The implicit approach doesn't suffer from the problem of restriction on the CFL number implying there are no restrictions on the time step size. Even though implicit approach seems to be a better one, it would require a significant modification in the existing solver. Additionally, parallelization of the method is still under development. It is noted by many authors that the choice of explicit or implicit approach is purely problem dependent. Thus, in the OWM it remains to see which of the two approaches are better.
Nevertheless, the time spectral method has some problems. It is seen that the number of harmonics required vary according to the type of problem. If the problem requires higher number of harmonics to show sufficient agreement with transient simulations, then there is a chance that the method might take more time for computations than the transient simulations. In such a case, it is disadvantageous to use this method. To compute problems where mesh is moving, the method proposes to compute $2n + 1$ computational grids. This can be a cumbersome task! But this approach is widely used by researchers and industry experts while computing rotor flows and flows in turbomachinery like turbines.

3.6. Conclusion and research questions

It is concluded that the time spectral method is most suited for the current problem as it is the best fit out of all the methods considered in this chapter. In the case of rotor-lamella motion, the use of time spectral method is advisable as the current time-marching implementation requires long computation times. And even if 3 harmonics are considered, the model can still be faster than the transient computations for 10 rotations of rotor. This brings to the final formulation of research questions.

3.6.1. Research questions

- As discussed in section 3.3, the entire problem reduces to a large system of coupled non-linear ODEs with a periodic solution. The time spectral method is a particularly efficient approach developed for solving such systems. The current applications of these methods have been for fluid flow solutions. There are no applications for a coupled system like the OWM problem. It is necessary to find out whether it is possible, and how to formulate the time spectral method for the coupled fluid-structure problem. Does the formulation need any modification or, can it be applied in its current form?

- The system of algebraic equation resulting from the application of time spectral method is non-linear. In section 3.5.3 implicit and explicit approaches for splitting of the source term were discussed. Which will be the most appropriate approach to solve this problem? Is it easy to implement the implicit approach? Is the time step restriction using the explicit approach relaxed enough for its applicability to our problem? Thus, the project research should be aimed at understanding which is the viable option out of the two.

4

Conclusions and research questions

A thorough literature study is performed addressing two different aspects for the numerical modeling of the OWM. The first aspect is the treatment of non-smooth dynamics which defines the modeling of motion of the lamellas. Obtaining faster time periodic solution of the OWM problem is the second aspect addressed in this literature study. In this chapter, the highlights and research questions from the entire study are concisely formulated.

4.1. Initial research questions

The company, DWE, commissioned to develop a numerical model for the OWM to aid the engineering design process. A baseline 2D numerical model was developed by Maniyara [19] as a part of his master's project. The study of the baseline 2D numerical model leads to the formation of initial research questions.

- The 2D baseline model indicated unphysical spikes in the torque time signal. The research should aim at determining the cause of these spikes and what are the solutions or ways to suppress them?
- The non-smooth dynamics approach to model the contact dynamics phenomenon of the lamellas in the current 2D baseline model uses perfect inelastic collision. This lead to unphysical spikes in the torque time signal graph as discussed in section 1.2.2. The current project should focus on how to develop a more realistic model of partially elastic collision of lamellas? It should also address the question of whether it is possible for the new model to be implemented on a standard ODE solver? The new model should not have the unphysical spikes as observed in the baseline 2D model.
- The baseline 2D model is computationally demanding. It takes as much as 10 revolutions of the turbine to achieve sufficient periodicity if the model is formulated as an initial value problem. Knowing that solving for periodic solutions of the flow is necessary in many engineering applications like flow around rotors, are there computationally more efficient ways of modeling the OWM problem?
- The time integration scheme used in the 2D baseline model uses very small time steps to correctly model the lamella motion. This is a traditional approach but a memory intensive one, as lot of data needs to be stored. Are there better approaches to tackle this problem and thus, help reduce the memory requirements?

4.2. Conclusions from the literature study

After the literature study, the scope of the initial research questions was reduced to some conclusions and preliminary research questions. The conclusions are as follows:

- To solve the problem of unphysical spikes, the torque transfer mechanism was studied. It was concluded that one needs to model the contact phenomenon applying the impact dynamics equations from the field of non-smooth dynamics.
- Similar to the baseline 2D model, the event-driven approach should be used. But, the approach should be adapted to model the partially elastic collisions of the lamellas. This type of collision is proposed to be modeled through the consideration of colliding and resting contacts.
- It was seen that the apparently complex problem of modeling lamella motion was reduced to simple ODE integration but with appropriate conditions applied at every time instant the lamella reaches its extreme positions.
- It is concluded that the time spectral method is most suited for the current problem as it is the best fit out of all the methods considered in this chapter. In the case of rotor-lamella motion, the use of time spectral method is advisable as the current time-marching implementation requires long computation times. And even if 3 harmonics are considered, the model can still be faster than the transient computations for 10 rotations of rotor.

This brings to the final formulation of preliminary research questions.

4.3. Preliminary research questions

The research questions put forth throughout the course of this report are reiterated here. The questions to be addressed during the course of the project are as follows:

- As discussed in section 3.3, the entire problem reduces to a large system of coupled non-linear ODEs with a periodic solution. The time spectral method is a particularly efficient approach developed for solving such systems. The current applications of these methods have been for fluid flow solutions. There are no applications for a coupled system like the OWM problem. It is necessary to find out whether it is possible, and how to formulate the time spectral method for the coupled fluid-structure problem. Does the formulation need any modification or, can it be applied in its current form?
- The system of algebraic equation resulting from the application of time spectral method is non-linear. In section 3.5.3 implicit and explicit approaches for splitting of the source term were discussed. Which will be the most appropriate approach to solve this problem? Is it easy to implement the implicit approach? Is the time step restriction using the explicit approach relaxed enough for its applicability to our problem? Thus, the project research should be aimed at understanding which is the viable option out of the two.

In the course of the project, the aim is to address these research questions.

Bibliography

[1] M. Baba-Ahmadi A. Jackson, M. Campobasso. On the parallelization of a harmonic balance compressible navier-stokes for wind turbine aerodynamics. *ASME*, 2011.

[2] A. Jameson A. K. Gopinath. Time spectral method for periodic unsteady computations over two- and three- dimensional bodies. *AIAA Aerospace Sciences Meeting and Exhibit*, 2005.

[3] David Baraff. An introduction to physically based modeling: Rigid body simulation 2-nonpenetration constraints. *SIGGRAPH Course Notes*, 1997.

[4] Gregor Cvijetic. Analysis and implementation of the harmonic balance method in computational fluid dynamics. *Master thesis at University of Zagreb*, 2015.

[5] Brise Yves D. Integral methods for quadratic programming theory and implementation. *Doctoral Thesis, ETH Zurich*, 2013.

[6] M. Montagnac F. Sicot, G. Puigt. Block-jacobi implicit algorithms for the time spectral method. *AIAA Journal Vol. 46*, 2008.

[7] Ulbrich H. Foerg M., Pfeiffer F. Simulation of unilateral constrained systems with many bodies. *Multibody System Dynamics*, 2005.

[8] Christoph Glocker. On frictionless impact models in rigid-body systems. *The Royal Society*, 359, 2001. doi: 10.1098/rsta.2001.0857.

[9] C. B. Lorence K. C. Hall. Calculation of three-dimensional unsteady flows in turbomachinery using the linearized harmonic euler equations. *ASME*, 1992.

[10] W. S. Clark K. C. Hall, J. P. Thomas. Computation of unsteady nonlinear flows in cascades using a harmonic balance technique. *AIAA journal Vol. 40*, 2002.

[11] W. Karush. Minima of functions of several variables with inequalities as side constraints. *M.Sc. Dissertation Dept. of Mathematics, University of Chicago*, 1939.

[12] Ladislav Kavan. Rigid body collision response. *7th Central European Seminar on Computer Graphics*, 2003.

[13] Kazuo Horiuchi Kiyotaka Yamamure. Accurate shooting methods for computing periodic solutions of nonlinear systems. *Electronics and Communications in Japan, Vol. 72*, 1989.

[14] Tucker A. Kuhn H. Nonlinear programming. *Proceedings of 2nd Berkeley Symposium*, 1951.

[15] Jean M. The non-smooth contact dynamics method. *Computer Methods in Applied Mechanics and Engineering*, 1999.

[16] G.N. Barakos M. A. Woodgate. Implicit computational fluid dynamics methods for fast analysis of rotor flows. *AIAA Journal Vol.50*, 2012.

[17] K.J. Badcock M. A. Woodgate. Implicit harmonic balance solver for transonic flow with forced motions. *AIAA Journal Vol.47*, 2009.

[18] N.J. Mallon. Collocation-a method for computing periodic solutions of ordinary differential equations. *Report produced at the Dynamics and Control group of Technical University of Eindhoven*, 2002.

[19] Vineeth Maniyara. Computational modeling of the oryon watermill. *Master Thesis-TU Delft*, 2017.

[20] Volker Mehrmann and Peter Kunkel. Differential algebraic equations. *EMS Text-books in Mathematics*, 2006.

[21] J.J. Moreau. Unilateral contact and dry friction in finite freedom dynamics. *Non-smooth mechanics and applications*, 1988.

[22] Schatzman M. Paoli L. A numerical scheme for impact problems i and ii. *SIAM journal on Numerical Analysis*, 2002.

[23] Christian Studer. Numerics of unilateral contacts and friction. *Lecture notes in applied and computational mechanics*, 47.

[24] H. van der Ven. An adaptive multitime multigrid algorithm for time-periodic flow simulations. *Journal of Computational Physics*, 2008.