Investigating Synergy Between Vertical Axis Wind Turbines Using the Actuator Line Model

by

Ji Hao Zhang

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Applied Science
in
Mechanical and Mechatronics Engineering

Waterloo, Ontario, Canada, 2022

Author's Declaration

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

In the past decade, numerous studies have confirmed the presence of beneficial, power-enhancing effects in the interactions of multiple vertical axis wind turbines (VAWTs). This effect is especially important for the advancement of VAWT technology in the area of farm layout arrangement because the ability to augment power in tightly packed arrays is inconceivable for the widely adopted horizontal axis wind turbines. This thesis explores the application of a reduced-order, fast simulation technique called the Actuator Line Model (ALM) towards the simulation of 2- and 3-turbine clusters in an effort to understand VAWT "synergy" and whether ALM is capable of predicting it. Due to the relative inaccessibility of ALM simulation codes (it is not a standard feature of CFD tools), this thesis describes in detail the development process and source code of an implementation using ANSYS Fluent User-Defined Functions. The current development draws from methodologies aggregated and distilled from key literature in this field, and has been found to be reasonably valid by cross-comparison with an open-source code, prior results in literature, a full-order CFD simulation, and experimental results. An open-source ALM code for OpenFOAM (published and used in peer-reviewed articles) has been used to simulate 2-/3-turbine cluster configurations, demonstrating that ALM is fundamentally capable of predicting turbine synergy. Additionally, useful functional relationships between the configuration parameters and the power ratio (power coefficient of clustered turbines relative to isolated turbines) are identified, confirming and expanding on the findings of literature in this field. Notable patterns include higher performance enhancements experienced by the downwind turbines, power degradation of upwind turbines due to blockage, and correlation of higher downwind turbine synergy with positioning within high velocity zones bordering the wake of the upwind turbine. The V, Reverse V, and Line 3-turbine configuration shapes are demonstrated to be successful synergy-exploiting designs, where the former two, which feature horizontally flipped arrangements, produce the same array power synergy and performance. This thesis also proposes a novel pairwise superimposition scheme to approximate the power ratios of turbines in a 3-turbine configuration using a simple summation of pairwise interactions involving the same relative positions. This scheme was determined to be highly accurate in the current cases, with sub-1% errors, though the general extensibility to higher turbine counts remains to be determined. The current work concludes by summarizing the key findings and offering a set of objectives that could be undertaken by future work in order to validate and improve ALM for the purpose of general VAWT micro-siting.

Acknowledgements

I would like to express my utmost gratitude to my supervisors Prof. Fue-Sang Lien and Prof. Eugene Yee for their unwavering guidance and encouragement throughout my graduate study. This thesis would not have been possible without their commitment to supporting me by providing invaluable assistance, advice, and wisdom. I sincerely thank Prof. Lien and Prof. Yee for enabling me to pursue this MASc opportunity and dedicating their valuable time towards enduring mentorship.

I am also immensely grateful to my colleagues and friends Ying Wu, Zhi Cheng, and Ryley McConkey for their continuous support and assistance, which are indispensable for the completion of my work.

Dedication

This is dedicated to my family and friends.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

ALM	Actuator Line Model
BEM	Blade Element Momentum
CCW	Counter-Clockwise
CFD	Computational Fluid Dynamics
CW	Clockwise
HAWT	Horizontal Axis Wind Turbine
TSR	Tip-Speed Ratio
UDF	User-Defined Function
URANS	Unsteady Reynolds-Averaged Navier-Stokes
VAWT	Vertical Axis Wind Turbine

List of Symbols

Re_c	Chord-based Reynolds number
α	Angle of attack
C_p	Turbine power coefficient
c	Blade chord length
C_L, C_D	Lift and drag coefficients (airfoil section)
ϵ	Gaussian kernel width parameter
η	Gaussian regularization kernel, analogous to a weight factor to be multipled with ALM forces
λ	Tip-speed ratio
D	Turbine diameter
u_0, u_∞	Freestream velocity
u_{rel}	Relative velocity encountered by a rotating blade section at some point defined by the ALM implementation (e.g., quarter-chord)

Chapter 1

Introduction

Wind energy is an increasingly important area of study due to its sustainable nature, and significant strides have been and are still being made in advancing theoretical and engineering knowledge. The most commonly seen category of wind turbines, which are used to extract wind energy and generate power, is horizontal axis wind turbines (HAWTs). Such turbines are installed to harvest wind energy globally and account for the bulk of wind power produced today, thus enticing massive research efforts. In contrast, vertical axis wind turbines (VAWTs) are a category of turbines that are characterized by the axis of rotation of the rotor being oriented vertically with respect to the ground, as opposed to horizontally in the case of HAWTs. Although VAWTs have been reasonably well-established and technically feasible for some period of time, this type of wind turbine is not commercially competitive compared to HAWTs and remains relegated to fulfilling niche energy generation demands.

The reason for this lies in inherent operational characteristics of VAWTs. Since the axis of rotation is oriented perpendicular to oncoming wind flow, a VAWT innately possesses omnidirectional capabilities for orientation. That is, unlike a HAWT which must have the rotor disk aligned fairly closely with the wind direction, an individual VAWT is capable of generating power regardless of wind direction. However, this very trait that gives a VAWT its comparative advantage hinders power generating efficiency, which occurs due to each of the many blades being constrained to rotate into adverse conditions to produce "resistance" torques for some part of its trajectory. This is certain to occur for VAWTs in general, thus detracting from the desired power-generating torque. As such, assuming all else equal, it is not economically sensible to scale up VAWT units to replace the more efficient, mature HAWT technology. Nevertheless, there remains niche areas of demand better addressed with smaller-scale VAWTs, such as where the bulky structure of HAWTs are undesirable or infeasible and/or where the wind direction is highly volatile. Examples of suitable locations include remote areas, urban zones, and even high-

rise rooftops. Hence, VAWTs remain a topic of interest for research that aim at developing more aerodynamically favorable variants or techniques.

The focus of this thesis will be on the study of a so-called "synergy" effect between VAWTs, discovered roughly within the last decade, which enhances the power performance of closely placed turbines under certain configurations and conditions. This was a groundbreaking realization because such a phenomenon is not observed for the extensively installed HAWT type, which means HAWT farms had to be designed to space turbines to mitigate negative/interfering interactions between units within a land area constraint. In other words, HAWT farms require large plots of land for efficient operation, which is a troublesome downside. Based on literature, this synergy effect has been validated for a wide range of conditions and now opens new opportunities for VAWT farm design to increase the *power density* almost universally, thus compensating for any individual turbine-level performance deficiencies and enabling a potentially competitive alternative to HAWTs. To explore these opportunities, synergy as a phenomenon must be carefully characterized and studied in regards to the range of permitting conditions for its appearance, the magnitude of the benefits, and how these vary with turbine geometry and operating scenarios. Numerous studies have made substantial progress in this area, but a common challenge is the expensiveness of *full-order* CFD simulations when modelling several turbines, which will become a greater issue for design exercises. For such purposes, engineering models based on low-order / reduced-order formulations are typically desired to appreciably cut down on computational costs while maintaining adequate accuracy in capturing the pertinent physics.

This thesis opts to use a low-order simulation technique called the Actuator Line Model (ALM) to study the aforementioned synergy effect between VAWTs. A brief idea of the concept of ALM (to be expanded on later) is that the blades of a turbine are not present in the mesh, but their effects on the flow field are modelled with momentum source terms in the Navier-Stokes equations which are solved using CFD. These momentum source terms represent forces of the blades acting on the flow field and are calculated using the blade element momentum (BE-M) theory, which models blade aerodynamics using airfoil performance data tables. There is limited intersection between ALM and VAWT synergy in literature, which motivated the undertaking of the current study. Therefore, the objectives of this thesis is two-fold:

1. Ascertain the ability of ALM to predict the presence of synergy effects between multiple VAWTs

2. Investigate the effect of VAWT cluster arrangements on synergy

To achieve these objectives, the structure of the thesis is organized as follows. Chapter 2 will review the literature that are relevant to developing this thesis, which will serve to pro-

vide evidence and context regarding fundamental VAWT operating principles, synergistic inter-turbine effects, low-order models for VAWTs, and the ALM approach. Next, in Chapter 3, the detailed development of a from-scratch implementation of ALM for the study of VAWTs will be presented. The proof-of-concept implementation is done in the ANSYS Fluent software using User-Defined Functions with the logic extracted / deduced from a myriad of ALM literature. In Chapter 4, the established framework along with a more robust, open-source library for OpenFOAM will be validated using ALM literature results, experimental measurements, and an unsteady-RANS full-order simulation of a turbine. This chapter establishes the validity of proceeding with the OpenFOAM ALM library for application towards the study of synergy of 2- and 3-turbine configurations in Chapters 5 and 6, respectively. Finally, Chapter 7 concludes this thesis by summarizing key findings on VAWT synergy and recommendations for future work.

Chapter 2

Background

2.1 Fundamentals of Vertical Axis Wind Turbines

Before delving into VAWT synergy and the study thereof via reduced-order models such as the ALM, it is important and necessary to first lay the foundations of VAWT design and operating principles. The goal of this and the ensuing sections within this chapter is not to provide a comprehensive literature review across all the relevant topics, but rather to establish crucial research context and technical underpinnings to support the investigation of VAWT synergy.

Despite the familiarity of VAWTs being quite obscure outside of academia, they are indeed a long-established category of wind harvesters. The usage of VAWTs can be traced back to at least as early as the 9th century, with the modern, lift-based "Darrieus" VAWTs being first invented by Georges Darrieus in the 1920s [1].The curved-bladed and straight-bladed variants are shown in Figure 2.1. Another type of VAWT is the "Savonius" turbine, which was invented in 1922 by Sigurd J. Savonius but may be traced back to older roots [11]. In contrast to Darrieus turbines, the Savonius design is drag-based and thus is incapable of rotating faster than the wind speed (tip-speed ratio remains less than or equal to 1). Hence, this archetype's fundamental limitations with aerodynamic efficiency renders it undesirable for individual application towards wind power generation [11, 12, 13].

It is important to emphasize that VAWTs are not widely implemented nor are they an economically competitive choice in the market of wind energy harvesters today. Historically, there have only been about 30 notable, $\gtrsim$100-kW VAWT installations since 1975. As surveyed by Möllerström et al. [1], most of the early constructions of Φ-configuration (or curved-bladed) Darrieus turbines suffered from structural / blade fatigue, power transmission problems, and other

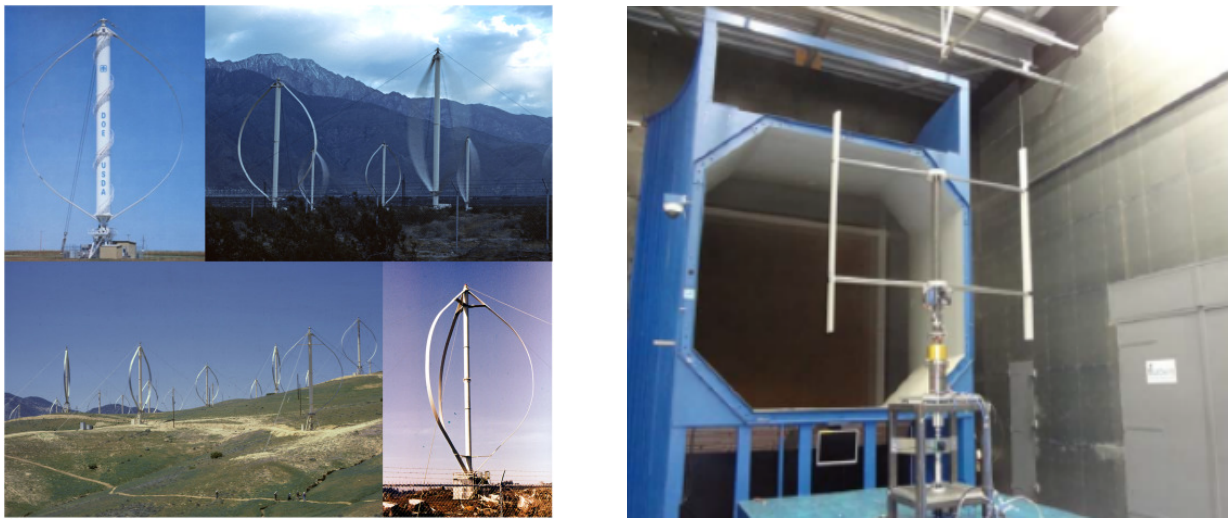


Figure 2.1: Sample images of curved-bladed Darrieus VAWTs (left) [1] and an H-rotor (straight-bladed) Darrieus VAWT (right) [2].

mechanical issues. From Figure 2.2, a resurgence of interest in the H-rotor concept took place after 2010, likely due to the relative ease-of-manufacturing of the straight blades and the suitability of the concept's lower center of gravity for offshore siting [1]. It is difficult to conclusively determine why VAWTs are not massively adopted to the extent of HAWTs, but evidence are found for the following factors: lower aerodynamic efficiency (due to blade trajectories encountering rotor wake, dynamic stall) [12, 14, 15], self-starting issues at low wind speeds [12, 14, 16, 17], prohibitive fatigue/durability problems for large-capacity units [1], and lack of investment interest (deriving from aforementioned factors) [1].

To establish a basic understanding of VAWT dynamics, Figure 2.3a shows a diagram of a cross-section of a VAWT, illustrating a blade subjected to various forces and velocities from the flow. A typical VAWT consists of a rotating shaft with a span of 2 or more blades (shaped as airfoils) all oriented perpendicular to the ground. The view plane in Figure 2.3a is oriented parallel to the ground and thus perpendicular to the rotor span, and it is situated at some spanwise location, making it useful for visualizing blade dynamics within both curved-bladed and H-rotor VAWTs. Given an arbitrary wind direction with freestream velocity u_0, the blades are induced to (and thus the turbine) rotate in a clockwise (CW) or counter-clockwise (CCW) direction at some rotational velocity. In fact, the presented schematic is a simplification / aggregation of velocities and forces encountered by a blade according to the *blade element momentum* (BE-M) theory [3, 18]. In this simplified method, the instantaneous state of a blade at each azimuthal

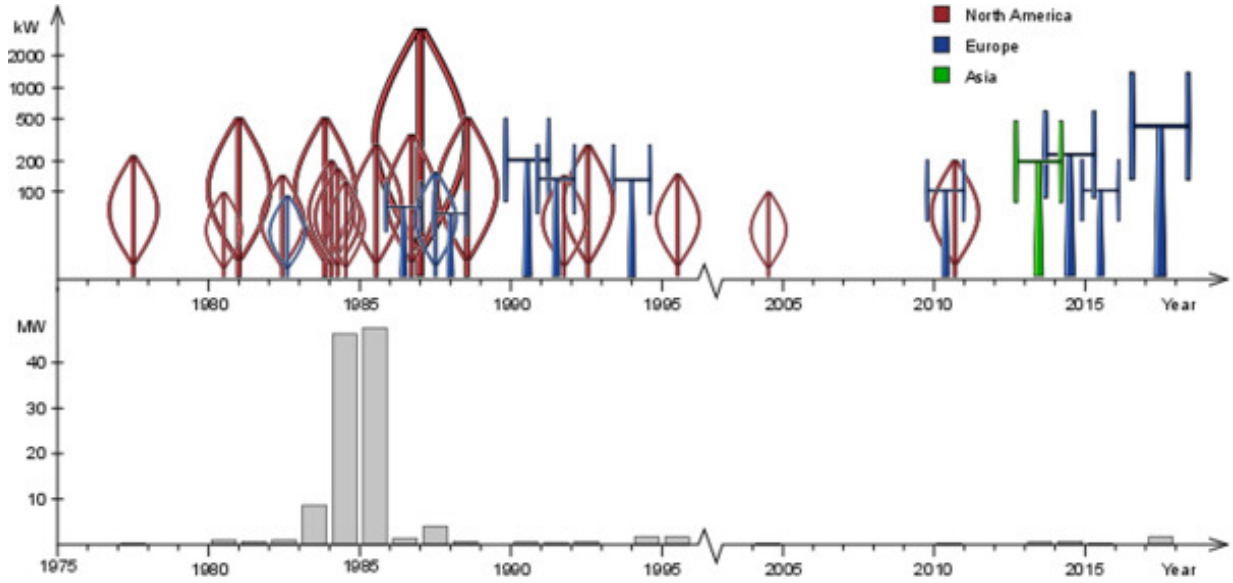


Figure 2.2: A timeline of $\gtrsim$ 100-kW VAWT installations and installed capacity from 1975 to date [1].

angle θ is described by a single relative velocity experienced by the blade $\vec{u_{rel}}$, which is a vector difference of the local flow field $\vec{u}$ and the velocity due to rotation $\vec{u_{bl}}$. Each blade can be pitched about the end of the strut that connects it to the shaft at an angle β, and the angle of $\vec{u_{rel}}$ relative to the chord-line of the airfoil is defined as the angle of attack α. Together, these two angles define some relative angle of θ_{rel}. According to aerodynamics, a force is induced on the airfoil due to $\vec{u_{rel}}$ and can be considered as two orthogonal components: lift F_L and drag F_D. They are oriented by convention to be normal to and along the direction of $\vec{u_{rel}}$, respectively [18]. A useful quantity to define is the non-dimensional value of the *tip-speed ratio* (TSR) of a turbine, commonly denoted as λ:

$$\lambda = \frac{\omega R}{U_\infty} \tag{2.1}$$

where ω is the turbine rotational velocity, R is the turbine radius, and U_∞ is the freestream wind velocity [18]. This quantity is important to track since the performance and behavior of a VAWT varies as a function of TSR. Another useful quantity is the power coefficient, C_p, which is a non-dimensional indicator of the amount of power a VAWT is capable of producing:

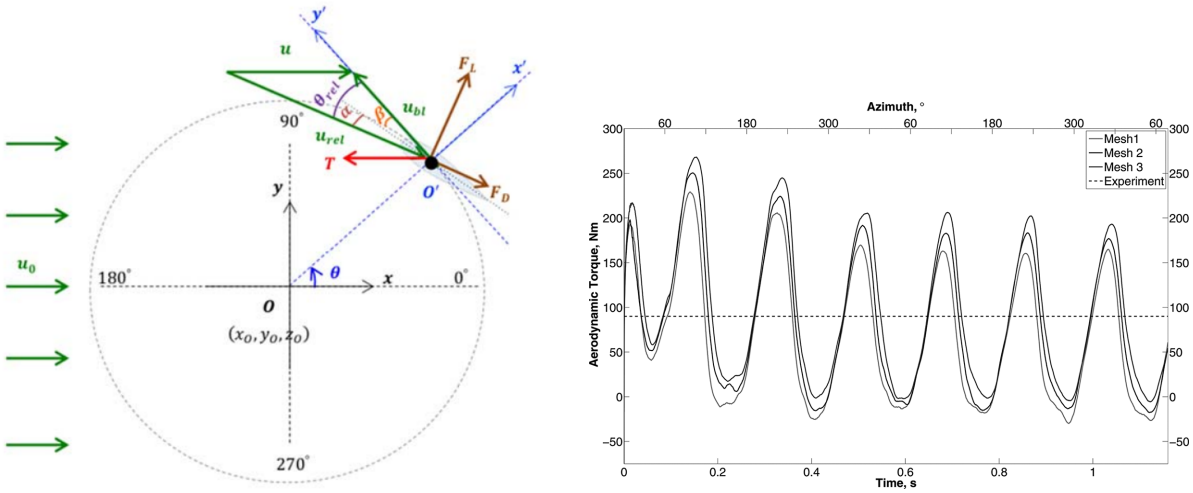
$$C_p = \frac{P}{0.5\rho D L U^3} \tag{2.2}$$

where P is power, ρ is air density, D is the turbine diameter, L is the blade span length, and U is some arbitrary reference velocity typically taken as the freestream velocity [6].

A more detailed review of the BE-M methodology as applied to ALM will be discussed in a later section. For now, it suffices to illustrate the general idea surrounding VAWT dynamics—as each blade rotates, it experiences a cyclic variation in force due to flow interactions, a tangential part of which contributes to the moment / torque of the turbine, driving rotation in that direction. Thus, the aggregate effects of all blades in a turbine generates a sinusoidal oscillation in aerodynamic torque over time, which is demonstrated in Figure 2.3b. It is this torque and the angular velocity that drive power generation in a VAWT. Note that for some azimuthal portions of each revolution, the beneficial torque can be little to none, or the torque may even detract from power generation as it is acting in the opposite direction. Such unsteady aerodynamics is an intrinsic property of VAWT operation [18].

As evidenced by the operating principles of a VAWT, a turbine's performance remains invariant when subject to any arbitrary wind direction. This is because each blade on the turbine must rotate through both beneficial flow conditions (in the upwind path) and adverse conditions (in the downwind path), independent of the particular wind direction. This inherent omni-directional capability combined with its inherent technical feasibility at smaller scales (e.g., 10-m tall VAWTs vs. 100-m tall HAWTs [6]) are of interest for niche wind generation applications. Therefore, this technology is still being actively studied. Computational Fluid Dynamics (CFD) is a common and standard method for simulating the aerodynamic performance and wake characteristics of VAWTs, which are useful for both theoretical understanding and testing proof-of-concepts of new designs. Typical CFD techniques involve solving the unsteady Reynolds-Averaged Navier-Stokes (URANS) equations with a turbulence model [17, 19, 20] or the use of a more expensive approach called Large Eddy Simulation (LES) [17, 21]. Ghasemian et al. [17] provide a review of numerous literature involving VAWT CFD, summarizing key components to an accurate simulation, such as grid resolution and the specific choice of a turbulence modelling approach. It is of interest to note that the power coefficient (describing power generation capabilities) of a VAWT increases with the TSR until some maximum, before decreasing with further increases to TSR [17]. Also, 3-dimensional (3-D) effects such as secondary flows and tip effects are important for the study of VAWT self-starting behavior, indicating situational necessity for performing 3-D CFD as opposed to the more common and less expensive 2-D simplification [17].

Meana-Fernández et al. [19] compared the validity of different turbulence models for use with URANS, Scale-Adaptive Simulation, and Detached Eddy Simulation approaches, finding that k-ω-based models performed the best in accurately predicting aerodynamic forces of VAWTs. There are several key works on VAWTs using URANS [22, 23, 24, 25, 26] and LES [27, 9, 28], though LES is employed less frequently, presumably because of extensive computational costs and advantages only in theoretical characterization. For example, Posa & Balaras



(a) (b)

Figure 2.3: (a) A cross-sectional diagram of a VAWT blade's operating conditions [3]; (b) A sample aerodynamic torque time series simulated using CFD [4].

[21] utilized high-fidelity LES to study coherent structures in the asymmetric VAWT wakes. Notably, it is found that at higher TSRs, blades interact more strongly with their shed vortices, while at lower TSRs, the blade-generated structures encounter upwind stall and affect the same blade even in the downwind trajectory. Furthermore, complex patterns in separation / re-attachment emerge as a function of TSR [21]. There are also unique literature that apply methods such as the Arbitrary Lagrangian Eulerian Variational Multiscale technique, originally developed for fluid-structure interaction (FSI) studies of HAWTs, to VAWT analysis [4]. Du et al. [16] also provides a more comprehensive review of CFD-based numerical VAWT studies and wind tunnel measurement studies.

Among the complex, unsteady characteristics of VAWTs, the most important is dynamic stall. Dynamic stall is a unique phenomenon that may be encountered by VAWTs during operation at lower TSRs, which causes significant losses in aerodynamic efficiency and thus performance degradations [22, 29]. Buchner et al. [24] states dynamic stall "consists of a separation of the boundary layer from the suction-side surface of the blade and subsequent roll-up into a leading edge vortex" and employs a combined approach of particle image velocimetry (PIV) measurement and 2-D URANS with the Menter-SST turbulence model to model this dynamic. This low-Re regime is also studied by [5], which observed complex blade-vortex interactions owing to dynamic stall. For instance, disturbances caused by blades in the upwind half spreads downstream and leads to the decoupling of the flow impingement point with the angle of attack,

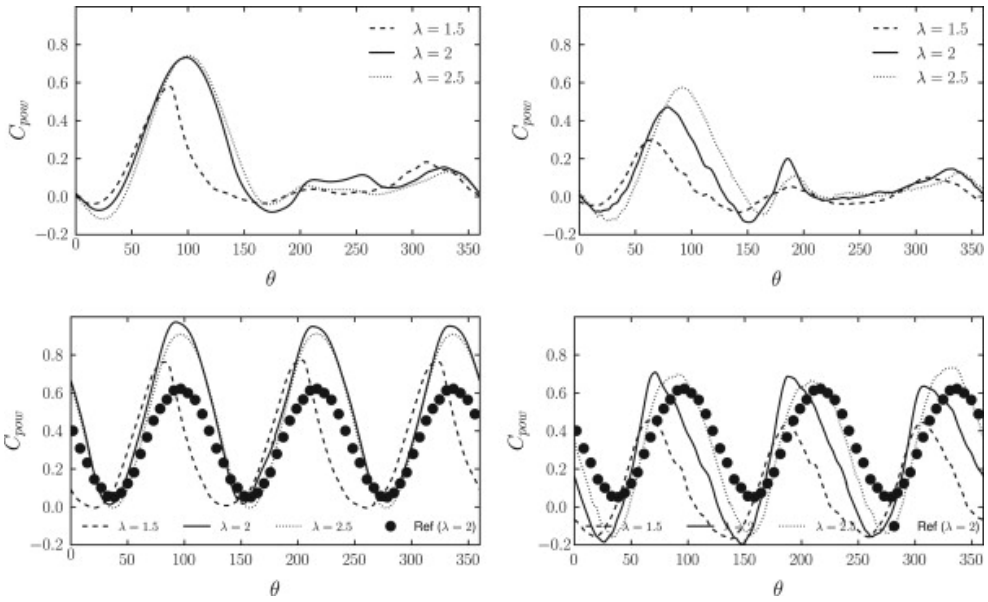


Figure 2.4: Power coefficient of a single blade (top) and the entire turbine (bottom) using the k-ω-SST (left) and k-ω-SST LRE (right) models as a function of azimuthal angle for different TSRs, from the work of McNaughton et al. [5].

leading to the variations in aerodynamic loading shown in Figure 2.4. Note that for a single blade, the performance degrades in the downwind azimuthal range due to velocity deficit in the wake. While blade-vortex interactions and dynamic stall cause perturbations in the C_p of an individual blade [5], the entire turbine still exhibits regular, cyclical variations in C_p.

Finally, a brief sample of works attempting to improve the aerodynamic efficiency and power performance of individual VAWTs is provided. Liu et al. [12] developed a hybrid Darrieus-modified-Savonius turbine that possesses improved self-starting capabilities and decent efficiencies at lower TSRs, but it cannot be scaled up efficiently despite being able to suppress dynamic stall. Additional structures can be installed to augment the operation of VAWTs, such as guide vanes that could increase the average power coefficient by 30-35% [17, 30]. Dynamically pitching the blades given a certain flow condition to enhance performance has also been extensively studied [28, 31, 32] along with a novel approach of intra-cycle rotational rate modulation that could increase power by up to 59% [33]. Therefore, even for a single turbine unit, research is ongoing to exploit novel designs, modifications, and control measures (which are not necessarily mutually exclusive) towards power / efficiency enhancement.

2.2 Synergy Between VAWTs

Having established basic concepts of a VAWT and the research landscape that seeks to better characterize its dynamics and/or exploit its power generation capabilities, this section will be dedicated to describing the evidence and current understanding of multi-turbine "synergy" phenomena. While it is useful to explore the power-generating characteristics of an individual VAWT, it is impractical that a singular VAWT be relied upon for providing the electricity to a locale. It is thus desirable to group multiple turbines in as close proximity as possible to maximize the power generated for an occupied land area, or in other words, maximize the *power density* of a wind farm. Many HAWT wind farms are in operation today with the goal of generating large supplies of electricity, but they are designed so as to mitigate the negative interference of units on each other, particularly those due to velocity deficit in long-running wakes [34, 35]. In order to realize practical or even optimal wind farm designs for VAWTs, similar problems must be navigated.

In a field experiment by Kinzel et al. [36] involving 18 VAWTs, it was found surprisingly that at only $6D$ (6 times the rotor diameter in distance) downstream, the streamwise velocity already recovered to 95% of the freestream, which is significantly shorter than the $14D$ required of HAWTs. Some evidence shows that at the lower operating heights of VAWTs, there exists enhanced turbulence and high planform kinetic energy fluxes that aid the momentum replenishment of the wind [36]. This is already a massive advantage entailing the possibility of tighter VAWT array packing in a farm, but in a 2010 field test involving 6 VAWTs operating for thousands of hours, Dabiri found that closely placed VAWTs in an array could operate at higher C_p than an isolated VAWT [6]. This work is one of the earliest to report the existence of a type of performance enhancement phenomenon between several VAWTs. For ease of communication, this thesis will employ the term "synergy" to refer to the mutual-interaction phenomenon that causes one or more turbines in a cluster to increase in performance beyond the isolated benchmark. Note that this work does not claim to have coined this term, which has already been used by some literature on this topic (e.g., by Hezaveh et al. [37]). A visualization of synergy can be found in Figure 2.5, which highlights the presence of synergy for several wind directional ranges where $C_p^{norm} > 1.0$. Given the presence of synergy, Dabiri optimistically anticipates an "order-of-magnitude increase in wind farm power density relative to existing HAWT farms" [6].

This initial discovery has since been reinforced with new, independent evidence over the past decade from both experimental and numerical studies. Some notable studies that confirm the presence of multi-VAWT synergy and attempt to elucidate its causes are reviewed hereafter, though this thesis does not claim to have comprehensively inventoried all relevant literature.

In 2016, Ahmadi-Baloutaki et al. [38] conducted wind tunnel experiments using 2-/3-turbine arrays of 5-bladed, straight blade VAWTs. The measurements confirmed synergy being present

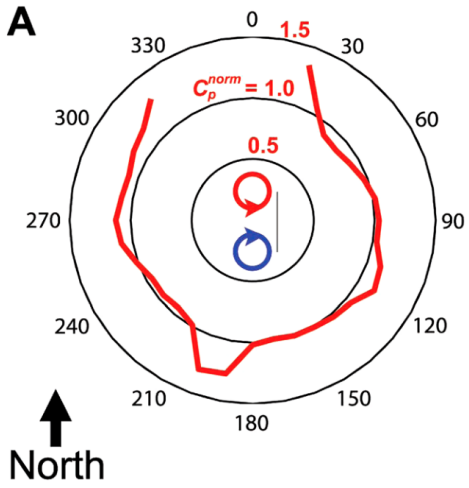


Figure 2.5: Measurements of the normalized power coefficient (with respect to two isolated turbines) of a pair of closely operating VAWTs for different incident wind directions, from the work of Dabiri [6].

in *adjacent* pairs (turbines are oriented side-by-side normal to the wind direction), though in *staggered* situations (turbines are arranged with a streamwise gap), the trailing turbine benefited more substantially. The mechanism responsible for performance enhancement are tentatively attributed to a channeled and sped-up flow field between closely placed turbines. Furthermore, it is found that co-rotating / counter-rotating (whether the rotational direction of the turbines align) pairs exhibited similar synergistic interactions. It is noted here that this is an important insight, meaning it is not required that pairs be set counter-rotating to induce a beneficial flow field.

Zanforlin & Nishino [39] studied a pair of 3-bladed VAWTs using URANS and the k-ω SST turbulence model, also confirming synergy. This study (and others using CFD to investigate synergy) was successful in demonstrating that URANS with the aid of a turbulence model is capable of predicting and modelling synergy, establishing the precedent for more widespread numerical studies on this topic. Interestingly, the streamlines show a contraction of the wake cross-section in the inner channel between the two VAWTs, which enhances power generation in the downwind path of blades. Aside from an enhanced velocity, synergy is found to correspond to an extension of the azimuthal range in which torque is generated in the upwind path. This was found to correlate with a "suppression" of y-velocity components. For a trailing turbine, a higher C_p is attributed to upstream turbine blockage leading to increased flow rates. At the same time, the blockage of a downstream turbine hinders the normal wake development of the upwind turbine, which is reflected as a performance detriment onto the upwind turbine in some

arrangements. Finally, an increase in TSR led to greater flow accelerations and thus greater synergy, which is hypothesized to be due to a higher virtual permeability at higher TSRs.

Shaheen & Abdallah [40] utilized a CFD approach using URANS and the transition SST turbulence model to study the interaction of 3 co-rotating Savonius VAWTs arranged in a triangular-shaped cluster. Synergy is confirmed with a cluster-level power enhancement of about 26% relative to isolated units. Importantly, this proves that the synergy effect is not limited to straight-bladed Darrieus turbines.

In 2017, Lam & Peng [41] measured the wake characteristics of a pair of 5-bladed VAWTs and determined the presence of synergy. Also, it is clear from the results that U/U_0 exceeded 1.0 around and between the wakes of adjacent turbines, concurring with the results of Zanforlin & Nishino [39]. These two studies are also in agreement that the cross-stream or y-component of velocity induced in turbine-to-turbine interaction can improve the C_p. Finally, this study proposes that 2-/3-turbine clusters be used as fundamental building blocks in the micro-siting of a larger wind farm due to the smaller wake spreading rate and faster wake recovery observed.

In 2018, Peng [13] adopted a dynamic torque-driven approach to modelling the rotational velocity of VAWT pairs, which departs from and is more sophisticated than the constant, prescribed rotational velocity approach used in the majority of numerical studies. In this study, a pair of VAWTs composed of 5 J-shaped blades are found to have synergy ratios of 1.05 to 1.7 depending on turbine spacing. This crucially demonstrates that while a constant rotation simplification may detract from the simulation fidelity, it is not an artificially conflating factor for predicting synergy.

Shaaban et al. [42] explored optimal arrangements of 3-6 turbines simulated using URANS and a realizable k-ϵ model with enhanced wall treatment. The 3-turbine configurations are arranged based on an equilateral triangle shape, which was extended for the 6-turbine case. While synergy can be observed for some turbines, the arrays are found to have reduced power performance compared to isolated turbines. The authors thought this was due to downstream turbines being too close in proximity in the cross-stream direction to upwind turbines, leading to blockage and thus C_p deterioration of those turbines. This is in agreement with [39]. This indicates a problem with closely packing turbines despite synergy and challenges Dabiri's initial estimate of "order-of-magnitude" improvements in power density [6]. Nevertheless, parameterization of array shapes / spacings was not explored in this study, meaning it does not preclude the possibility of more optimal arrangements capable of exploiting synergy while mitigating interference, which is an important objective of the current thesis.

In 2019, Barnes & Hughes [43] conducted a 2-D study on VAWT farm performance using URANS with the Transition SST turbulence model. Several scenarios with 5, 6, 9, 15, and 16 turbines were evaluated, with turbines arranged according to the common triangular or "V-

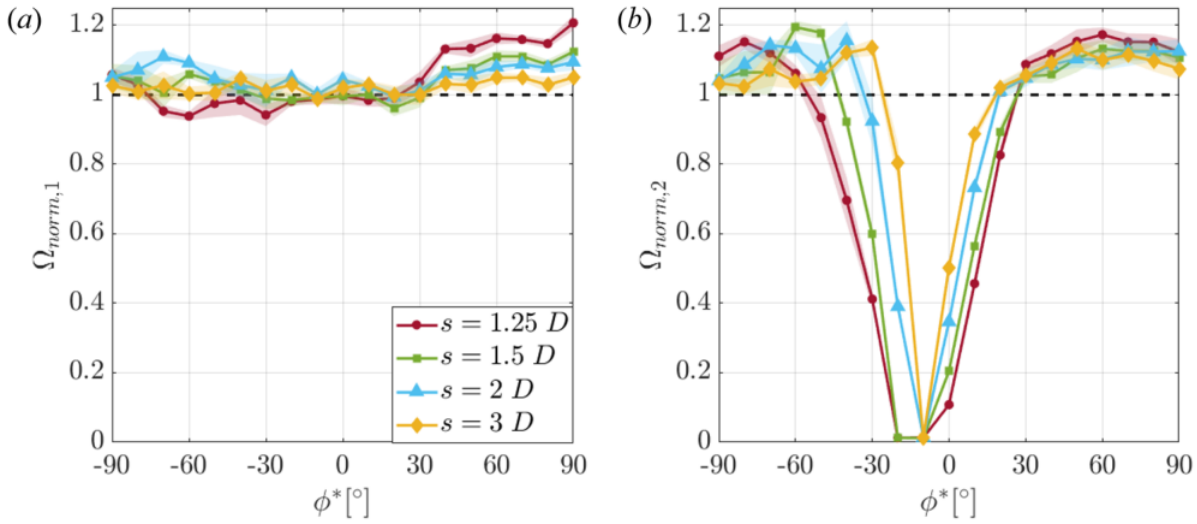


Figure 2.6: Normalized performance of (a) Turbine 1 and (b) Turbine 2 for various array spacings as a function of an adjusted array angle (ϕ^*), where the normalized rotational rate Ω_{norm} for a turbine is defined as $\Omega_{array}/\Omega_{isolated}$ (measured in an array configuration vs. in isolated operation), from the work of Brownstein et al. [7].

shaped" configurations. Across the various runs, it was found that as much as a 181% increase in power occurred. While this seems surprising due to its sheer magnitude, the authors posited that a Venturi effect has increased the flow velocity from the 10 ms^{-1} inflow to as high as 14.5 ms^{-1}. As evidence of this, the expected conversion of static pressure due to flow constriction into dynamic pressure is confirmed via contour plots. Thus, due to this increase in velocity, the maximum power increase possible is 205% since there is a cubic correlation with velocity. For the larger arrays, performance detriment is seen though synergy is still noticeable even as far as 3 rows deep downstream. The results are qualitatively comparable with prior literature [42], though the accuracy of the current study is likely tempered by the use of larger time steps (0.005 s vs. 0.001 s) and coarser mesh (160,000 cells for 4 turbines vs. 320,032 cells for 3 turbines) in comparison to Shaaban et al.'s work.

In 2019, Brownstein et al. [7] published a pivotal work detailing 3-D measurements of pairwise VAWT interactions over a large volume of space. Several arrangements were tested. The synergy effects were observed and quantitatively compared in Figure 2.6, where the indicator for performance uses a ratio of rotational speeds (Ω). Notably, for certain configurations the upwind turbine (T1) experiences a performance deterioration consistent with the findings of [39, 42]. Also, the performance of the downwind turbine (T2) degrades significantly when the array is

oriented such that T2 enters T1's wake, which features significantly reduced flow velocities. The authors argue that performance enhancements are due to bluff-body accelerations around the upstream turbine. The variations of synergy with array configurations and the momentum entrainment could be attributed to streamwise vortical structures, which scale proportionally with TSR and solidity (but otherwise are only weakly associated with turbine geometry).

Finally, in 2021, Hansen et al. [44] published a comprehensive work that utilized URANS and the k-ω SST model to evaluate the synergy between 2- and 3-turbine arrangements. This work is particularly important to reference in this thesis for two reasons. First, this work employs a highly refined grid composed of about 900,000 cells, verified using 2458 h of mesh convergence runs. The domain is also $60D$ by $90D$ (where D is turbine diameter), which is sufficiently large to prevent blockage effects from skewing the performance results. Second, this work devised a systematic way of parameterizing array configurations based on an array angle and inter-turbine spacing to attempt to provide a picture of how synergy varies as a function of these variables. This is in contrast to the studies of Shaaban et al. [42] and Barnes & Hughes [43], where the degree of freedom of array configuration was only briefly studied. This work also defines synergy rigorously using a *performance indicator* denoted Ω, which is a ratio of C_p in a synergistically operating cluster to the C_p of the same number of isolated VAWTs. This metric obtained across different configurations was compared and found highly similar to that measured from Brownstein et al.'s experiments [7] (also see Figure 2.6). Based on these reasons, this work is considered an important reference for developing the systematic approach to be used in studying the effect of array configuration on VAWT synergy herein.

2.3 Reduced-Order Modelling of VAWTs

The previous section has firmly established the presence of synergistic interactions between VAWTs as well as the feasibility of using CFD (with URANS in particular) to study and characterize synergy. While this is a tried-and-true approach, the computational cost of these "blade-resolved" CFD runs are prohibitively high for the evaluation of multiple turbines, since it is required to set highly refined mesh near the blades accompanied by proportionately small time steps. In other words, the blade-resolved CFD approach does not scale well for engineering application towards VAWT farm arrangement optimization based on the crucial element of VAWT synergy. Thus, it is of interest to develop and apply a *reduced-order* or *low-order model* for this purpose, as opposed to the *full-order* (blade-resolved) CFD method. Such a modelling approach would aim at simplifying calculations from the rigorous finite volume CFD while retaining a reasonable degree of accuracy in predicting VAWT performance, especially those contributions due to synergy. This thesis will use the terms *reduced-order model* and *low-order model* inter-

Method	Blade model	Wake model
Kinematic	Thrust coefficient	Self-similar solutions
BEM	Actuator disk + blade element	Quasi one-dimensional momentum theory
Vortex lattice, vortex particle	Lifting line/surface + blade element	Free/fixed vorticity sheet, particles
Panels	Surface mesh	Free/fixed vorticity sheet
Generalized actuator	Actuator disk/line/surface	Volume mesh, Euler/RANS/LES[a]
Direct	Volume mesh	Volume mesh, Euler/RANS/LES

[a]RANS, Reynolds-averaged Navier–Stokes; LES, large eddy simulation.

Figure 2.7: A classification of reduced-order / analytical models for studying VAWTs, from the work of Sanderse et al. [8].

changeably to convey that an approach is a lower-cost approximation of the *full-order model* in which VAWT blades are meshed and explicitly resolved in the CFD simulation.

The Actuator Line Model (ALM) to be used in the current thesis is indeed a reduced-order model. However, there are a plethora of reduced-order methods proposed in literature, some similar to ALM while others are completely distinct. A brief survey of these other models will be introduced here to provide context and identify limitations, while ALM will be discussed in the ensuing section. While some justification will be provided for the choice of ALM as the focus here, this work attempts to make no commentary on the superiority or exclusivity of ALM as an approach to study VAWT synergy. ALM is expressly chosen because of its promise and perceived robustness as one of the possible ways of approximating actual synergy characteristics.

Sanderse et al. [8] attempted to inventory and categorize the diverse landscape of reduced-order models of VAWTs. A summary of this survey is extracted and included in Figure 2.7. Du et al.'s [16] summary concurs with the findings of Sanderse et al. by identifying momentum and vortex models as the most applied aerodynamic models for studying H-Darrieus VAWTs. Within the overarching class of momentum models, there are the Single Streamtube model, the Multiple Streamtube model, and Double Multiple Streamtube model. These all utilize BE-M and aerodynamic "streamtubes" defined by simplified momentum and induced velocity formulations, and they tend to overpredict power [16]. Delafin et al. [45] performed a comparison of the accuracy of the Double Multiple Streamtube (DMST) model, a free vortex model, and CFD in predicting the experimental performance of a VAWT. The work concludes that power, torque, and thrust predictions of the low-order models, in particular DMST, resulted in significant errors away from the optimal TSR. Abhishek [14] used a DMST model coupled to airfoil performance tables to study a VAWT with variable amplitude dynamic blade pitching. In this work, it is emphasized that VAWT solidity is the most important scale-independent geometry factor for comparison and that not correcting for unsteady aerodynamics could lead to underprediction of power by as much as 35%.

Vortex (wake) models use potential flow theory to model the velocity field in the vicinity and wake of VAWTs based on vorticity shedding and interaction [16]. Tescione et al. [46] presented a free vortex wake/panel method for evaluating the 3-D dynamics of the near-wake of a VAWT. When compared to measurements, the vortical structures in the wake can be captured reasonably well qualitatively, though some key properties such as wake asymmetry could not be modelled. In spite of this, vortex models such as this one still offer insight into time-varying, vortex-driven flow behavior that cannot be modelled by the more simplistic momentum/streamtube models. Zanon et al. [47, 48] used the vortex panel method with a double wake concept to capture in greater fidelity the unsteady separated flow, blade-vortex interactions, and the formation / shedding of strong vortical structures. PIV velocity fields were used to establish validity. Dixon's [49] thesis also establishes the in-depth formulation of a 3-D unsteady free-wake vortex panel method and applies it towards the study of VAWTs. It is found that regular vortex patterns deriving from the epicycloidal paths of blades in the immediate vicinity of the rotor eventually evolves into irregular, aperiodic patterns, highlighting the complexity of flow through a VAWT.

There also exists analytical models for predicting the shape and extent of wake development for a single VAWT, though they do not possess the capability for modelling turbine performance. For example, Lam & Peng [50] attempted to calibrate a wake spread formulation based on two semi-ellipses (to represent wake asymmetry) using least-squares fitting / regression analysis of wind tunnel data. The extensibility or generalizability of such a model remains in question since all critical parameters have been derived for a specific turbine in operation. Abkar [51] also explored the characterization of mean wake patterns, using field data and LES to validate a top-hat-shaped distribution based on mass conservation and momentum theory as well as a 2-D Gaussian distribution. The latter was found to be more accurate. While these models hold some value for micro-siting with velocity deficit approximations, they are incapable of predicting synergy.

There have also been newly emerging techniques, such as the one proposed by Tingey & Ning [52], which is essentially a massive functional-fitting undertaking driven by 460 full-order CFD simulations. Polynomial surfaces are used to fit the parameters of an exponentially modified Gaussian distribution that describes vorticity strength at various locations of the wake. Such a function would be able to provide the wake vorticity as a function of TSR and solidity, which can then be "unpacked" into flow velocities for analysis. The model predicted maximum wake deficits with percent differences of 6.3% and 14.6% against two experimental studies, but it remains to be seen whether the model generalizes well to a greater variety of turbines, especially those with TSR / solidity outside the range tested in the seed CFD runs.

Finally, the generalized actuator models appear robust and promising. The rotor may be represented as an actuator disk [8], an actuator surface (e.g., cylinder) [9, 53], or an actuator line [8, 2, 3]. In all variants, the actuator exerts a force on the flow, which is incorporated into the

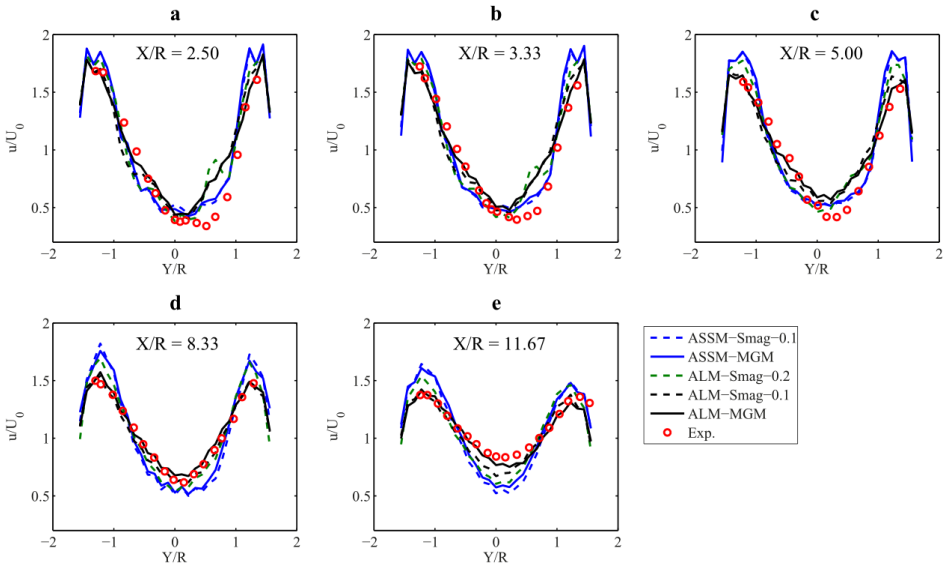


Figure 2.8: A comparison of normalized mean streamwise velocity profiles for water channel measurements, ALM + LES and ASSM + LES, from the work of Shamsoddin & Porté-Agel [9].

momentum equation as a source term. For actuator disk / cylinder methods, the formulation can be done in conjunction with CFD or without it, but is typically a time-averaged approximation that consists of no moving parts (in contrast to ALM). For the case without CFD, Ning [53] used actuator cylinders to model synergy between a pair of VAWTs across a full 360-degree suite of wind directions. At each point, an integral formulation of induced velocities accounting for blade self-influence and mutual interaction is used to compute body forces produced by the VAWT. This study confirms a 5-10% increase in power across wide ranges of wind directions for the VAWT pair. For techniques embedded into CFD, Shamsoddin & Porté-Agel [9] conducted the first study that uses LES with the actuator swept-surface model representing the time-averaged influence of the rotor and alternatively with the ALM that explicitly models the time-varying positions and aerodynamics of blades. The comparisons for normalized mean streamwise velocity is shown in Figure 2.8, which demonstrates the salient advantage of ALM in accuracy when compared to the Actuator Swept-Surface Model (ASSM). The fact that ALM performs the best and remains robust among the reduced-order models discussed is not controversial [8], though ALM is more expensive to compute. Also, note that in Figure 2.8, the normalized velocity $\bar{u}/U_0$ clearly exceeds 1.0 for y/R positions away from the wake deficit region, once again affirming the previously discussed flow speed-up which is a hallmark of synergy [38, 7, 43, 44].

2.4 The Actuator Line Model

This section will be dedicated to introducing important literature involving ALM with an emphasis on vertical-axis (wind) turbine applications, though HAWT ALM literature will be used to establish a solid background. While key works will be aggregated and discussed here, this thesis does not claim to have comprehensively surveyed all VAWT ALM literature.

The earliest known ALM implementation was proposed by Sørensen & Shen in 2002 [54] for the modelling of HAWTs. In this early formulation, BE-M is used to determine the body forces acting on spanwise-discretized elements of the blades. The local velocity field experienced by each aerodynamically independent section can be captured using a relative velocity U_{rel}, and an angle of attack α is formed. Based on these two key dynamic variables, the force acting on the blade per unit span at any time can be found using:

$$\boldsymbol{f}_{2D} = \frac{1}{2}\rho U_{rel}^2 c(C_L \boldsymbol{e}_L + C_D \boldsymbol{e}_D) \tag{2.3}$$

where ρ is the air density, and C_L / C_D are the 2-D lift / drag coefficients obtained from 2-D airfoil data corrected for 3-D effects, which is important because rotational effects at separation can produce increased lift and airfoil performance is not independent of blade aspect ratio [54]. This $\boldsymbol{f}_{2D}$ force vector can then be projected onto the desired coordinate system, such as θ-z in Sørensen & Shen's case. It is emphasized that this force cannot be readily supplied as the source term to the Navier-Stokes (N-S) equations, because a singularity of vorticity source would be produced at each blade element location. To remedy this, Sørensen & Shen introduced a *regularization kernel* to be *convolved* with the $\boldsymbol{f}_{2D}$ established above (or with its components separately). The following key equations are defined:

$$\boldsymbol{f}_\epsilon = \boldsymbol{f} \otimes \eta_\epsilon \tag{2.4}$$

$$\eta_\epsilon(r) = \frac{1}{\epsilon^3 \pi^{3/2}} \exp[-(r/\epsilon)^2] \tag{2.5}$$

$$\boldsymbol{f}_\epsilon(\boldsymbol{x}) = \sum_{i=1}^{3} \int_0^R \boldsymbol{f}_{2D}(r)\eta_\epsilon(|\boldsymbol{x} - r\boldsymbol{e}_i|)dr \tag{2.6}$$

where the $\otimes$ operator is used by Sørensen & Shen to denote the convolution procedure between a body force $\boldsymbol{f}$ and the regularization kernel η_ϵ, ϵ is described as a tuning factor to adjust the strength of regularization, r is the "distance between the measured point and the initial force

points on the rotor" [54]. Based on this definition, Equation 2.5 describes the exponential decrease of the kernel η_ϵ as a function of the distance r away from the initial point of force. Equation 2.6 requires more analysis since $\boldsymbol{x}$ and this formulation of force-kernel convolution is not explained by Sørensen & Shen nor is it justified using prior works.

The best interpretation for this process (which will also be assessed using later literature) is that $\boldsymbol{x}$ denotes an arbitrary vector indicating a mesh point with respect to some origin. It is desired to smoothly distribute the loading at this point [54]. Since $\boldsymbol{e}_i$ is the unit vector of the i-th blade's direction, r here is the integration variable used to radially traverse / span each blade from 0 (blade hub) to R (blade tip). Based on this interpretation, it logically follows that $\eta_\epsilon(|\boldsymbol{x} - r\boldsymbol{e}_i|)$ computes a kernel value for this arbitrary cell at $\boldsymbol{x}$ due to proximity to some point on the i-th blade. Then, η_ϵ is simply *multiplied* with the previously established blade body force at that radial point $\boldsymbol{f}_{2D}(r)$, and the integration over the range $r = [0, R]$ captures a *total* force (due to integration of a force per span over the span) contribution from the entire blade. This procedure is then repeated for all 3 blades for this particular turbine. In summary, Equation 2.6 merely states that the total force which should be *assigned* to a mesh cell / point at location $\boldsymbol{x}$ is the sum of all the differential influences from spanwise force elements of every blade. Using a "source-target" analogy to better intuit this relationship, the original force of the blade is a "source" and an arbitrarily distant cell is a "target" because a fraction of this force is to be allocated / distributed to the target cell. In the manner laid out by Sørensen & Shen, how much of a differential element's source force per span to distribute to a target cell is controlled via the magnitude of the regularization kernel, which can be thought of as a fractional modifier. The further away a cell is from a blade, the smaller this modifier, and hence the less significant the target force. Also, while it is not explicitly mentioned, the use of this technique in CFD requires the discretization of each blade into spanwise elements, each of which experiences a force of $\boldsymbol{f}_{2D} \cdot \Delta s$ where Δs is a spanwise length. At the smallest, Δs could be a cell dimension approximately in the blade direction. As such, the integral in Equation 2.6 would become an approximation using a sum of $\boldsymbol{f}_{2D} \cdot \Delta s$ (the source force) multiplied with η_ϵ.

For HAWTs, Sørensen & Shen's ALM was found to be able to predict the power vs. wind speed trend of a 500 kW Nordtank wind turbine with high accuracy compared to experimental measurements [54]. Over the past 2 decades, this reduced-order model has been employed to study the wake behavior and performance of HAWTs, such as in the theses of Troldborg [55], Johnson [56], and Claudio [57]. Troldborg used the ALM in unsteady 3-D CFD to comprehensively study vortex properties and turbulence characteristics of the wake of a HAWT operating at different TSRs and subject to sheared and turbulent inflow [55]. The formulation of ALM per Sørensen & Shen's work [54] concurs with the interpretation presented above. Johnson's thesis used the ALM for similar purposes, and the key equations documented therein are consistent with established literature [56]. Claudio's thesis is particularly useful as it documents the devel-

opment of the HAWT ALM algorithm using Fluent User-Defined Functions (UDFs) [57]. This work is particularly useful for the current Fluent implementation, and will be referenced more extensively in Chapter 3.

Since this work focuses on VAWTs, literature detailing the implementation of ALM for VAWTs or vertical axis / cross-flow turbines (e.g., not necessarily aimed at wind extraction but are formulated identically) are of particular interest.

One of the earliest studies on VAWTs using ALM was conducted by Shamsoddin & Porté-Agel [9] in 2014 (cited in previous sections), which features the first usage of ALM in conjunction with LES to study VAWT wakes. In this work, the formulation of body forces is identical to the HAWT case as per Sørensen & Shen [54], but there remains 2 key differences due to the unique dynamics of VAWTs [9]. First, the coordinate system and in particular the 2 axes along which the body forces can be decomposed changes. In the HAWT case, a polar coordinate system was formulated where forces are present in the θ (blade sweep / rotation) and z (streamwise, perpendicular to rotor disk) directions but not the r (radial) direction [54]. In the VAWT case, the forces can be decomposed into the n (radial) and s (tangential) directions (using the notation of [9]), which both lie on the plane perpendicular to blade span. Alternatively, when decomposed into fixed Cartesian coordinates, it is evident that forces exist along the streamwise and cross-stream direction, but not in the spanwise / normal direction. Due to the shift in the coordinate system and the underlying flow / rotor dynamics, the relative velocity V_{rel} (or, U_{rel}) and the angle of attack α should also be formulated differently. According to Shamsoddin & Porté-Agel [9], they are defined as:

$$\boldsymbol{V}_{rel} = V_{local,n}\boldsymbol{e}_n + (V_{local,s} - \Omega_b R)\boldsymbol{e}_s \tag{2.7}$$

$$\alpha = \arctan(\frac{V_{local,n}}{V_{local,s} - \Omega_b R}) \tag{2.8}$$

where $V_{local,n}$ and $V_{local,s}$ are the local wind velocity components in the inertial frame of reference, and the tangential component also consists of a contribution from the rotor angular velocity Ω_b multiplied with the rotor radius R. Furthermore, due to the relatively coarse grid size of a LES simulation compared to a full-order URANS CFD, this work does not consider force smoothing with the use of a regularization kernel. Instead, at each time step, the grid cell which encompasses the blade line is identified and assigned the total, singularity force [9] (most likely discretized in the spanwise direction). From the results, it is clear that this ALM-LES approach is useful in predicting mean velocity profiles, though turbulence intensity distributions are poorly predicted with respect to the experimental data.

In 2017, Hezaveh et al. [58] published a further study of VAWTs using ALM-LES, featuring additional validation with water channel and wind tunnel experiments and the use of URANS

simulations to produce higher-fidelity C_L and C_D values to feed into the ALM. These "joint but uncoupled" 2-D, blade-resolved simulations are used to provide an improved set of lift / drag coefficient tables for the NACA 4-digit symmetrical airfoils, which are more accurate than data from the commonly used Sheldahl & Klimas (1986) report [59]. The authors argue that such single-airfoil experiments do not capture interactions of blades with shed wakes and dynamic stall and show that instead full-order URANS provides a much closer approximation of experimental thrust coefficients (C_T) than a common Boeing-Vertol dynamic stall correction model [58]. The formulation of ALM equations is familiar, except with an explicit conversion of the streamwise (u) and cross-stream (v) local wind speeds into normal / tangential components through the azimuthal angle of the blade. It is noted that due to the nature of LES, the $\tilde{u}$ and $\tilde{v}$ used are time-averaged over 10 min for each time step. Furthermore, a completely different force distribution method is used wherein the force is not imposed at the center of the airfoil chord, but are initially imposed equally onto 5 grid points spanning the chord. Then, the force for each of these grid points is further divided across 8 surrounding cells at $1/10$ weight each, preserving a $2/10$ weight for the center point. This is in contrast to the absence of smoothing/distribution in [9] and the regularization kernel approach in [54]. The results demonstrate a similar degree of accuracy as Shamsoddin & Porté-Agel [9]: mean velocity profiles are adequately predicted but good turbulence intensity modelling via ALM-LES remains elusive. Some key results of this work are that increasing the TSR can reliably reduce the wake recovery distance and the aspect ratio (rotor diameter to blade span) strongly influences the wake behavior.

Creech et al. [60] used ALM-LES to study the wake effects of modelling support structures in a dual rotor, contra-rotating tidal turbine. While this study does not feature a vertical axis *wind* turbine, the effective rotor geometry and operating principles here are virtually identical to those of the VAWT. The ALM formulation is largely similar with existing works except that in contrast to [9, 58], a Gaussian distribution function (η) similar to the regularization kernel proposed by Sørensen & Shen [54] has been used:

$$\eta_i(\boldsymbol{x}) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}(\frac{d_i}{\sigma})^2} \tag{2.9}$$

where $\boldsymbol{x}$ is a point in space, the index i denotes a particular turbine blade, d_i is the "ring" distance between point $\boldsymbol{x}$ and the actuator line, and σ is a constant that acts similarly as the ϵ tuning factor from [54]. In this work, σ was tuned so that it is neither too large (smears the solution excessively) nor too small (demands very fine mesh and small time steps) at a value of $(1/12)R$ [60]. Importantly, this η differs with Sørensen & Shen's formulation [54] in that the leading normalization coefficient's denominator is $\pi^{1/2}\epsilon^{2/2}$ and not $\pi^{3/2}\epsilon^3$. A brief look at an online resource (https://pages.stat.wisc.edu/~mchung/teaching/MIA/reading/diffusion.gaussian.kernel.pdf.pdf) reveals that Sørensen & Shen's Gaussian kernel is a 3-D one while

Creech et al.'s is a 1-D one applied based on the scalar value d_i (resulting in the same η_i for all points with the same ring distance). Note that the exponent of ϵ (or σ) in the denominator of the "normalization" factor dictates the dimensionality of the Gaussian kernel. The 3-D version works as explained above, where a source force element is distributed to nearby cells in a spherical zone-of-influence. It is therefore assumed that with a 2-D Gaussian kernel, a source force element is only divided and assigned to cells on the same z-plane (spanwise cross-section) in a circular region. Additionally, the description that "u_{rel} is calculated for each cell point, using the local flow speed" [60] indicates there is a local, cell-wise force computation scheme, where each cell is treated as an independent airfoil. The implication of this will be described in detail in Chapter 3 in a comparison with alternative schemes.

One notable result of this work is that the inclusion of turbine support structures via ALM is integral to accurately predicting turbulence in the wake, as evidenced in non-negligible differences observed in the resolved turbulence spectra and power FFT. Also, it is found that modelling the full pair of turbines in LES as opposed to just a single turbine led to better agreement in power with the measured case, and the flow acceleration effect present in prior literature (e.g., [41, 7, 44]) are again clearly visible. This demonstrates preliminarily that ALM-LES is capable of predicting vertical axis turbine synergy to some extent.

In 2018, Abkar [61] published an ALM-LES study on the effect of subgrid-scale models (standard Smagorinsky model, Lagrangian scale-dependent dynamic model, and anisotropic minimum dissipation model) on VAWT wakes. While changing the subgrid-scale (SGS) model significantly altered the wake structure and turbulence statistics, the mean aerodynamic blade loads were invariant. Abkar's formulation of η is a 3-D Gaussian kernel, in contrast to Creech et al. [60]. The acquisition of C_L and C_D for the BE-M force per span calculation is not detailed, but the MIT dynamic stall model and Glauert tip-loss correction have been applied. No further details are documented for the specific implementation of the kernel-force convolution. However, this work notes that ϵ was set to $1.5\Delta x$, where Δx is the grid size, and a sensitivity study was performed on the differences between lateral profiles of streamwise velocity and velocity variance for $\epsilon = 1.5\Delta x$ and $2.5\Delta x$, with the former tested in two spatial resolutions. The results showed high conformance and weak sensitivity of all solutions to ϵ or the underlying mesh.

The above works all demonstrate existing and reasonably successful applications of ALM to the study of VAWTs, but they all feature the usage of LES [9, 58, 60, 61]. Despite the high degrees of accuracy being reported across various metrics for different turbines, it is important to recognize that LES employs relatively coarse grids and time-averaging. For example, a rough calculation of the streamwise cell dimension as a function of chord length c for these literature are: $41.5c$ [9], 0.15-$2.35c$ [58], and $2.17c$ [61]. With the exception of some geometries with larger blade chords in [58], these works have used $\Delta x > 1.0c$, meaning that one cell is larger than the entire blade chord. Furthermore, there is a lack of validation or post-processing pertaining to

aerodynamic blade loads, since LES is focused on the study of wake behavior and turbulence structures. This indicates a need to look at studies employing URANS with reasonably fine grids to further qualify ALM's ability to approximate VAWT performance at a granular level.

A frequently cited paper is that of Bachant et al. [62], which describes the development of an ALM library for simulating vertical axis turbines in OpenFOAM and applies it to the study of two vertical axis turbines (operating in water) using both URANS (with k-ϵ) and LES. The body force equations are consistent with prior conventions and the force projection is done with a Gaussian kernel similar to [54] and [61]. Each blade is divided into equal spanwise elements, with each represented by a control point named an "actuator point" where the local blade aerodynamics are aggregated into a point force to be distributed to the surrounding cells. Based on a survey of previous literature, Bachant et al. defines Gaussian width (ϵ) as the max of three values, based on chord length, mesh size, and momentum thickness due to drag force [63]:

$$\epsilon = \max\left[\frac{c}{4}, 4\sqrt[3]{V_{\text{cell}}}, \frac{cC_D}{2}\right] \tag{2.10}$$

though in practice, the center value due to mesh dominates during runs and can be manually tuned with a modifier at run-time. The model includes the flexibility of enabling dynamic stall, added mass, flow curvature corrections, end effects, and turbine strut / shaft options for a high fidelity simulation of the unsteady effects. For the UNH-RVAT turbine that was simulated, the C_p and C_D vs. TSR curves produced by ALM adhered to the experimental data closely but deviate significantly above the optimal TSR (corresponding to a peak C_p). On the other hand, there is a significant underprediction / misalignment of the C_p vs. TSR curve for the RM2 turbine, where the peak C_p was underpredicted by about 29%. In both cases, the mean streamwise velocity profiles are predicted at least adequately by the ALM but there are significant errors for mean turbulence kinetic energy. This is consistent with the analysis of prior ALM-LES simulations [9, 58].

The library developed by Bachant et al. is called `turbinesFoam` and has been made available as an open-source repository [64] — it will be used extensively in the investigations of the current thesis. At least two other works have used this library to carry out additional ALM simulations: Mendoza et al. [63] and Mendoza & Goude [65]. Mendoza et al. [63] carried out an extensive verification of the ALM-LES setup and established high degrees of consistency with experimental results for velocity profiles and angle of attack / normal force variations within a revolution. Interestingly, the results including and excluding the modelling of the tower / strut showed very little difference in the wake, a disagreement with the findings of Creech et al. [60]. Mendoza & Goude [65] then used the `turbinesFoam` library again in 2020 to compare ALM-LES results against that of two vortex models and experimental measurements. In this study, ALM was found to agree quite closely with experimental results in normal force prediction, though the

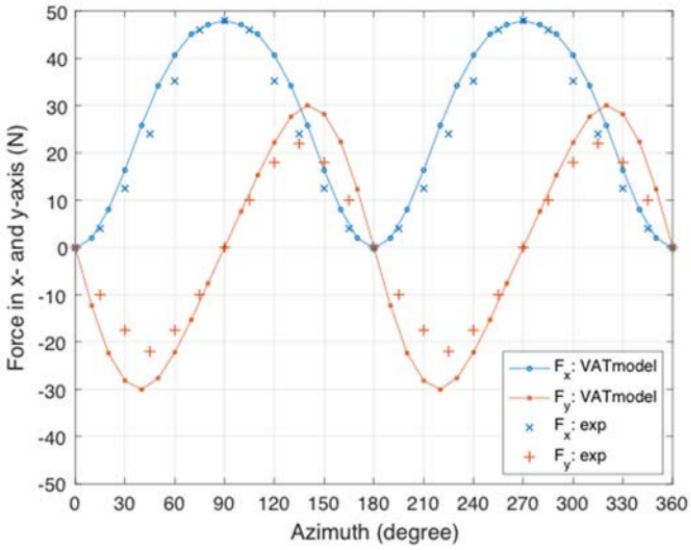


Figure 2.9: A comparison of the F_x and F_y turbine forces for a two-bladed H-type VAWT as predicted using ALM and measured from a wind tunnel experiment at TSR $= 3.7$, from the work of Zhao et al. [3].

ALM runs that used the Sheldahl & Klimas airfoil dataset [59] appreciably underpredicts C_p vs. TSR compared to the coefficients supplied by the XFOIL program.

Finally, Zhao et al. [3] published a study in 2020 detailing the use of ALM-URANS (with a k-ω SST turbulence model) to predict forces in a 2-bladed, H-type VAWT. This paper presents a comprehensive study to show the threshold cell numbers and time step size for accurate, grid-independent ALM solutions. A plot of the F_x and F_y turbine forces obtained from ALM and experimental measurements (extracted and shown in Figure 2.9) illustrates the consistent cyclic, smooth trends and a very high accuracy for the TSR $= 3.7$ case. Note that no support structures or shafts are modelled in the ALM. This plot will be particularly important for the verification / validation of the model developed in this thesis. Also, a comparison of the thrust coefficient C_T at various TSRs shows a good agreement at 3.5 and 3.7, but significant errors are present in the lower TSRs down to 2.5, which are caused by an absence of dynamic stall corrections in the ALM implementation. The ALM code employed in this study references Creech et al. [60], and thus uses a two-dimensional Gaussian regularization kernel. Zhao et al. indicates that $\vec{u_{rel}}$ is computed for each grid point within a volume of interest and thus adopts a multiple localized airfoil scheme to determine the kernel-adjusted forces at those points. These forces are then projected onto the flow field as momentum sources, the same way as all the ALM implementations surveyed.

2.5 Existing Studies on ALM & VAWT Synergy

Based on the current survey, very few published studies focuses on the application of ALM towards investigating patterns in VAWT synergy. The work of Creech et al. (2017) [60], as introduced above, identified some evidence of synergy in a pair of tidal turbines modelled using ALM-LES. However, the extent of synergy remained at the surface level, where aside from observed flow speed-ups (which has been found to correlate with increased power), only a crude comparison was made of the power generated. The actual measurements are for the pair together, and it was found that modelling the full pair overpredicted total power by 4.4% whereas multiplying an isolated turbine's power by 2 underpredicted power by 5.1%. This does not constitute a compelling evidence for synergy.

The only other work that merges the study of ALM and synergy together is by Hezaveh et al. (2018) [37], wherein ALM-LES is used to simulate synergy among 3-turbine clusters which is in turn used as building blocks for large-scale farms. This study demonstrated a power generation enhancement of a single turbine by about 10%, which confirms and solidifies the ability of ALM-LES to predict synergy — there is at least no fundamental barrier or incompatibility preventing this kind of study. Nevertheless, while this work sets out a groundbreaking approach to farm-level synergy simulation and solving the VAWT micro-siting problem using ALM, there are shortcomings. First, the entirety of validation for the pairwise interactions of 2 counter-rotating VAWTs is the comparison of a few mean velocity profiles with field experimental data. The aerodynamic loading / forces were not directly validated. Second, as mentioned previously on the subject of LES resolution, the grid size relative to the blade chord is $2.22c$ in this study, which means blade-scale dynamics are averaged within each cell. Based on these reasons, the true degree of accuracy of the ALM array-level synergy predictions is inconclusive, and so far the extent of investigations falls short of concretely establishing the capability of ALM to predict VAWT synergy.

At this point, sufficient context and background have been presented to justify and narrow the direction of the current thesis. VAWT synergy is a phenomenon that has been measured in experiments and predicted using full-order CFD techniques across many independent studies. Its presence signals an immense opportunity to increase the power density of VAWT farms / arrays by exploring the micro-siting of turbines such that synergy can be exploited. The previous sections have demonstrated that ALM, a reduced-order model, is a relatively nascent but promising approach for modelling the performance and wake characteristics of individual turbines. Such reduced-order models are key to developing any farm design or array optimization solutions in the future, simply due to costs constraints most likely precluding the adoption of multi-turbine, full-order URANS CFD. The specific application of ALM towards VAWT synergy remains a mostly open question, with only a few ALM-LES studies demonstrating the potential in this

area. Therefore, it is clear that there is a gap to be filled by investigating the ability of ALM-URANS simulations to predict the presence of synergy and the use of this lower-cost (relative to LES) technique towards multi-turbine cluster evaluations. As found by several research groups [9, 60, 37], ALM in LES is able to unambiguously produce flow accelerations around turbines, which is thought to be a key factor in causing the power augmentations. While URANS is perhaps unable to adequately capture turbulent fluctuations in a VAWT's wake, it is hypothesized that the mean flow velocities are a primary / sufficient attribute for manifesting synergistic interactions, and so this direction is worth pursuing. So far, no literature presenting an in-depth, program-level implementation of ALM-URANS for multi-turbine simulation has been found. Therefore, this thesis intends to undertake the pioneering work of reconciling the different approaches in ALM-URANS for single turbines, building a proof-of-concept code, and extending it to serve multi-turbine cases. Afterwards, a suitable code will be adopted to run ALM-URANS cases consisting of 2-/3-turbine clusters with the goal of understanding VAWT synergy in different arrangements.

Chapter 3

Building an ALM Implementation in Fluent

Two target CFD solvers / tools were identified for the development of an ALM implementation "from scratch" (but with the aid of details and methods documented in literature): ANSYS Fluent or OpenFOAM. This thesis chooses to develop ALM code for ANSYS Fluent for two reasons. First, Bachant et al. has already published the open-source `turbinesFoam` library for OpenFOAM [64], so there is not much value in reiterating this procedure. Meanwhile, there exists no extensively documented or open-source code for ANSYS Fluent made for the purpose of running the ALM for VAWTs. Second, the Fluent software provides greater ease-of-access when injecting add-on code into the solver when compared to OpenFOAM. A custom set of tools called User Defined Functions (UDFs) are provided by Fluent for writing programs using the C language to interact with the solver in ways that are necessary when implementing the ALM. In contrast to the reasonably robust UDFs, an ALM implementation in OpenFOAM entails the coding of a full-fledged library in C++, for which in-depth knowledge of object-oriented programming and the architectural underpinnings of OpenFOAM is required. While this framework comes with certain benefits (as revealed in later comparisons), it pulls the focus away from the theoretical aspects of ALM and tends to obscure a clear overview of the calculation steps. Therefore, for these reasons, this chapter will document the implementation of an ALM code written using Fluent UDFs for the purpose of clearly illustrating the required components of ALM calculations, the specific treatment thereof, and how this can be realized in C code.

In the following sections, this thesis intends to walk through the logic behind this UDF's architecture, how the airfoil performance tables are sourced and used, the core ALM computations, the results logging, and finally the complete program flow. The order of starting at a high level, then delving into component-level details, and finally returning to the high level is to offer

a sensible and digestible guide to creating one's own ALM implementation. These sections aim at being useful reference material for developing ALM UDFs in particular, but the organization of content is designed to present framework-agnostic information that will be useful for ALMs in general. To this end, frequent code snippets of the actual UDF will be used to concretely demonstrate the implementation for future researchers with the goal of presenting a working program. An additional note: "D" may be used as a shorthand for both "dimension" and "diameter". Here in this thesis, it is clarified that 2/3-D (with a hyphen and standard font-style) will always denote "dimension" and that $2D$ (without a hyphen and D is in mathematical typography) will denote "diameter", as in 2 *times* the diameter.

3.1 Program Structure Overview

The structure and rules regarding the use of UDFs warrant some explanation, since it is a rather unique system. It is noted here that the technical understanding presented in this thesis are mostly obtained from ANSYS Fluent's official documentation [66, 67] with supplemental information from Claudio's thesis [57] and the CFD Online forum (`https://www.cfd-online.com/Forums/fluent-udf/`), and the developed code works for ANSYS Fluent version 2020 R1/R2. Essentially, ALM is a set of logic / computations that are extraneous to CFD solvers like Fluent, so they must be somehow performed adjacently during run-time and "hooked" into the solver. The way this is achieved in Fluent is via hooking one or more UDFs written based on a built-in system, and for OpenFOAM it is through incorporating pre-compiled C++ libraries. A UDF is a `.c` extension program file, written by a user, which contains code that performs the core computations of ALM but also interfaces with the solver to exchange information. These exchanges are governed by preset APIs (application program interfaces, a common term in computer science that will be employed here for expedience) in the Fluent solver. These allow the UDF to obtain the velocity of a cell or to impose a momentum source term on that cell, for instance. Once the UDF file is written, it can be supplied to Fluent via GUI menus to be compiled (e.g., with a C compiler that comes with Fluent) and loaded, then "macros" are formally hooked to be executed by the solver automatically or manually. Any code in the loaded file not organized within a macro is run once (when the simulation starts), while macros are specially organized chunks of code that have their own, situational run-time rules.

These structures operate similar to functions and simply execute the instructions within, but different macros need to be hooked in their respective UDF sub-menus even after loading. For instance, the `DEFINE_ADJUST` and `DEFINE_SOURCE` macros are hooked in separate places. It is accurate to treat hooking as a critical step to enabling the operation of the macros that are written. Claudio's implementation for HAWTs [57] provides a brief but important idea of which macros

to use. Two important macros are identified as being useful for the current case: `DEFINE_ADJUST` and `DEFINE_SOURCE`. According to Claudio, the former is used to house the main logic of tracking the rotation of the actuator line, the instantaneous positions of actuator points, and looping over all cells to identify the cell which encloses these actuator points as well as storing other cell data (thus establishing a connection between the ALM discretized units and the actual cells engaged in the CFD procedure). The latter is the only way to impose any calculated forces as momentum source terms onto the solver-managed flow field. `DEFINE_ADJUST` executes once every iteration within a time step, then afterwards, all `DEFINE_SOURCE` macros (1 would be written for each direction and type of source term) are executed. To complicate the issue, with the recent versions of Fluent (e.g., 2020 and beyond), even single CPU core runs are essentially parallelized in that a `HOST` node is always spawned to oversee the calculation of multiple compute nodes (just `NODE_0` in a "serial" run). The implication is that the entire UDF and all the hooked macros will be run in *separate* instances on the `HOST` and all nodes, so statements to print to the console that one may include for debugging purposes show up at least twice (note that the `Message` API provided by Fluent should be used instead of the native C option of `printf`). In this framework, cells in the mesh are further divided based on user control and delegated to each node for CFD and UDF computation. There are no communication between the program state of nodes unless manually specified: each node is essentially running an independent version of the UDF with coverage for all cells in the mesh it is responsible for (there are some external cells which are overlapped between two nodes, which present problems for result logging). If some chunks of code should not be executed on `HOST` or nodes, this logic can be dictated by compiler directives as follows using `RP_HOST` or `RP_NODE`, respectively:

```
#if !RP_HOST
  /* code */
#endif
#if !RP_NODE
  /* code */
#endif
```

This is an advanced measure that will not be required in the current implementation, but it is included to demonstrate additional logic handling which might be useful in navigating more sophisticated UDF designs. Any potential issues of parallelization will be addressed in each of the following sections.

This content is discussed upfront, here, superseding any mention of low-level ALM computations because as is evident from these macro definitions, the technical framework directly constrains the shape of the solution. Here is a list of functionalities (roughly in sequential order) which the code must perform:

- Instantiate and populate / update data structures to store static parameters and results
- Load relevant airfoil performance data tables (C_L, C_D)
- Define and update the position of actuator lines / points / elements (discrete representations of blades in ALM) with time
- Calculate blade element aerodynamics based on current flow conditions in the CFD solver
- Distribute / project forces over adjacent cells according to a Gaussian regularization kernel
- Impose the forces as momentum source terms for the affected cells
- Log the desired results

These are derived from the literature on HAWT / VAWT ALM surveyed in Chapter 2 and must be done with awareness of the current time step or iteration. Based on the aforementioned restrictions, there are only a few ways of implementing this in a UDF. Claudio [57] performs several full node partition-wide cell loops (using `thread_loop_c`) inside nested loops for each actuator point in each blade. This is extremely costly, however: with 2 blades discretized into 26 actuator points each and a total cell count of say 10,000, the total number of cell-wise operations would be magnified to a number of $2 \cdot 26 \cdot 10,000 = 520,000$. This is staggeringly inefficient and can be circumvented, especially because the imposition of cell source terms can only occur inside `DEFINE_SOURCE`, which must loop through all cells indiscriminately. Even if the individual values are pre-computed, each cell must go through a non-negligible lookup operation to retrieve their appropriate terms when they are encountered sequentially in `DEFINE_SOURCE`. Thus, it would be more economical to push as much of the calculation as possible there instead of centralizing them and perhaps doing redundant work in `DEFINE_ADJUST`, though the specific approach depends upon the chosen actuator line discretization method and the Gaussian kernel.

Before moving on to the detailed implementation of each component, it is important to define the following at the global scope:

```
#include "udf.h"
#include <math.h>
#include <stdlib.h>
```

This essentially directs the C compiler to include the content corresponding to these header files in the UDF. This allows one to use mathematical, input/output, and other useful functions otherwise absent in the barebones C language from `math.h` and `stdlib.h`. Similarly, all the macros like `DEFINE_ADJUST` and the solver access APIs are only available with `"udf.h"`. In this

global scope, which is any code placed outside of a macro, variable and functional declaration should occur — these can then be referenced and updated by macros as the UDF executes. In programming, it is generally good practice to properly scope variables and avoid excessively defining global variables for expedience, since there could be unintentional collisions if they control the program state and are changed by several functions/macros. In the following code snippets, some liberty is taken in defining variables local to a macro or a global one if there is no practical difference.

3.2 Airfoil Performance Data Tables

Prior literature on ALM have clearly established the reliance of the technique on airfoil performance data tables, which are the tables of C_L and C_D as a function of the angle of attack α for various Re_c (chord-based Reynolds numbers). When α is determined for a blade element based on the local relative velocity, a data look-up must be performed to interpolate the corresponding C_L and C_D in order to approximate the actual blade aerodynamics. Even assuming a constant rotational speed of the blades, they experience fluctuations in relative velocity (u_{rel}) throughout their trajectories. This means the actual Re_c may conceivably fluctuate between fixed levels at which data is available. Thus, a look at the literature is necessary to guide this implementation.

Shamsoddin & Porté-Agel [9] used tabulated airfoil data to obtain C_L and C_D but did not document whether interpolations of coefficients across different Re are performed. Hezaveh et al. [58] employed a single table of $Re = 50,000$ for NACA 0015. Creech et al. [60] referenced the Airfoil Catalogue [68] to obtain data for a NACA 63-415 type airfoil, with no mention of the Re. Abkar [61] provided no information on how the coefficients are obtained. Bachant et al. [62] used the NACA 0021 airfoil data from the Sheldahl & Klimas report and interpolates across the $Re_c = 3.6 \times 10^5$ and 7×10^5 tables. Zhao et al. [3] used a single $Re_c = 1 \times 10^6$ table for NACA 0021 from the Sheldahl & Klimas report.

Based on this review, most of the literature considered only one Re_c table throughout the cycles of VAWT operation with the exception of Bachant et al. [62], which conducted interpolations. From a perusal of the source code [64], this is done by calculating the Re_c of the flow at each time step and determining the bounding Re_c with available data to perform a two-stage interpolation. For example, if an actual operating condition of $Re_c = 5 \times 10^5$ falls between the two tables for 3.6×10^5 and 7×10^5, then a linear interpolation is done to find the C_L and C_D for each table, which are then further interpolated to find the single pair of C_L / C_D values. Since a double interpolation is rarely used in the literature, it is assumed that it may not be required to obtain reasonably good results. Therefore, the current approach elects to use only one Re_c table for the entire ALM simulation. To fulfill this functionality, the following code declaration

is made in the global scope. Note that the numeric values illustrated here and in all code snippets are the actual ones used for the validation case of this thesis (to be described in Chapter 4). For the following code, the Sheldahl & Klimas report data for the NACA 0021 airfoil [59] will be loaded into the UDF.

```
const int N_TABLES = 10;
const int N_DATA = 97;
const int STR_LEN = 25;
float alpha_arr[N_TABLES][N_DATA];
float cl_arr[N_TABLES][N_DATA];
float cd_arr[N_TABLES][N_DATA];
float Re_arr[N_TABLES];
char fname_arr[N_TABLES][STR_LEN];
```

`N_TABLES` stores the number of Re_c tables that will be loaded and `N_DATA` represents the number of rows / entries in each table. These are declared as constants (which are guaranteed to be invariant throughout the UDF lifetime) because the C compiler requires this if they are to be used as size parameters for array declarations. If `N_DATA` varies, additional logic can be implemented to have a per-table row count. `STR_LEN` is the character count used for allocating memory to string variables (which are character arrays in C), where the value 25 can be changed to whatever value is required in the application. The declarations of `alpha_arr`, `cl_arr`, and `cd_arr` are storage arrays for α, C_L, and C_D, respectively. Memory is allocated for a 2-D array in which `N_DATA = 97` rows are reserved for each of the `N_TABLE = 10` Re_c tables. It is important that the above three data arrays are accessible from the global scope so that `DEFINE_ADJUST` and `DEFINE_SOURCE` can access them as necessary. Finally, `Re_arr` and `fname_arr` are arrays to hold a list of the Re_c values corresponding to each table and the associated file name to access that table.

Now, to actually import / load the data tables into the UDF, a one-time macro is required upon start-up of the ALM simulation. Fluent UDFs allow for several such macros. The one used here is `DEFINE_ON_DEMAND` which after hooking can be executed manually by the user in the GUI, though the more automatic `DEFINE_EXECUTE_ON_LOADING` (used by Claudio [57]) is also suitable. There would be no difference in the code apart from replacing the type of the macro in the declaration for the code block. The entire code for airfoil table loading is presented below, followed by detailed explanations.

```
DEFINE_ON_DEMAND(read_coefficients){
  int i = 0, j = 0;
  float a_data, cl_data, cd_data;
  float Re_data;
  char fname_str[STR_LEN];
```

```
6   FILE* fp;
7
8   fp = fopen("import_metadata.csv", "r");
9   for (i = 0; i < N_TABLES; i++){
10    fscanf(fp, "%f\t%s\n", &Re_data, fname_str);
11    Re_arr[i] = Re_data;
12    strcpy(fname_arr[i], fname_str);
13  }
14
15  fclose(fp);
16
17  for (i = 0; i < N_TABLES; i++) {
18    fp = fopen(fname_arr[i],"r");
19
20    for (j = 0; j < N_DATA; j++) {
21      fscanf(fp, "%f,%f,%f\n",
22             &a_data, &cl_data, &cd_data);
23      alpha_arr[i][j] = a_data;
24      cl_arr[i][j] = cl_data;
25      cd_arr[i][j] = cd_data;
26    }
27
28    fclose(fp);
29  }
30 }
```

First, it is important to note the structure of code encapsulation with macros: all code written within the curly brackets at the end of line 1 and at line 30 is part of the macro. The first string identifies the type of macro this should be, and can only be a valid name among the ones that Fluent supports [67]. The parentheses following this identifier should be populated with one or more arguments that are required by Fluent UDFs. For this kind of macro, only the user-chosen name for the macro is needed (e.g., you can define many `DEFINE_ON_DEMAND` macros within the same UDF with different names, with this one visible in the Fluent GUI as `read_coefficients`). Inside the macro, it is good practice to declare nearly all variables in one place ahead of their usage: `i` and `j` are simply loop indices; `a_data`, `cl_data`, `cd_data`, `Re_data`, and `fname_str` are temporary variables used in the loading process to contain the eponymous values (they can be omitted but used for clarity); and `fp` is a `FILE` type object used to handle files.

The `FILE` object `fp` is assigned a file to handle using the function `fopen` called with arguments of a file name string and `"r"` for read-only mode. This enables the reading of the

"import_metadata.csv" file, which contains entries of the Re_c value and the corresponding file name for that table. Thus, the current implementation will not interpolate across Re_c tables but will provide a full set of data to choose from during run-time. A for loop is used to read this file, iterating through the N_TABLES lines. For each line, fscanf is called to read the next line of the fp file, which should be parsed according to the pattern "\%f\t\%s\n". This means that first, a float value will be encountered ("%f"), followed by a tab separation ("\t"), then a string ("%s"), and finally the newline character at the end ("\n"). The next arguments are supplied to store the data scanned according to the placeholders "%f" and "%s" into the variables &Re_data and fname_str we have prepared. Note that the ampersand symbol & is required in this context because the fscanf function changes, in memory, the variable (e.g., Re_data) provided as an argument. Since one is working with the direct variable in memory and not a copy, one must pass that variable's memory address (viz., &Re_data). The next two lines in each iteration simply stores the Re_c value and the file name into their respective arrays at index i, where the function strcpy is needed to copy an entire string. Finally, it is important to "close" each file that is opened for reading / writing with fclose.

In the second chunk, another for loop is used to iterate through each of the tables, in which fp is successively assigned to the names corresponding to each table (which might look like "NACA0021_Re1E4.csv"). Then, for each of the files, another iterator j is used to loop through all the rows in the file. For each, fscanf is used again to read the row data, which for the current implementation was organized according to the comma-separated format of α, C_L, and C_D. After being assigned to intermediate variables, they are then formally stored in the alpha_arr, cl_arr, and cd_arr arrays at a position corresponding to the Re_c table index and the row index.

In addition to loading the airfoil data, it is important to write code here that looks up and interpolates the coefficients, since these procedures do not depend on the specific ALM formulation. First, a function is written in the global scope which uses the bisection algorithm to return a row index corresponding to a lower bound of the look-up value (the code is originally from p. 117 of *Numerical Recipes in C: The Art of Scientific Computing* by Press et al. [69]).

```
int locate(float xx[], int n, float x){
  int j = 0;
  int jm = 0;
  int jl = 0;
  int ju = n+1;
  int ascnd = 0;

  ascnd = (xx[n] >= xx[1]);
  while ((ju-jl) > 1)
  {
```

```
    jm = (ju+jl) >> 1;
    if ((x >= xx[jm]) == ascnd)
      jl = jm;
    else
      ju = jm;
  }

  if (x == xx[1])
    j = 1;
  else if (x == xx[n])
    j = n-1;
  else
    j = jl;

  return j;
}
```

This function is called `locate` and is defined with 3 parameters: `xx[]` being a vector (or ordered list) of values to search (which must be strictly increasing/decreasing), `n` is the size of this vector, and `x` is the particular value for which you want to find a lower bound index inside `xx[]`. For example, in each Re_c file, there are 97 data points indexed from `0` to `n - 1`, where say indices of (49, 50, 51) correspond to α values of (-1.0, 0, 1.0) degrees. Upon supplying a desired value `x = 0.5` to look up, the algorithm efficiently returns the index of 50, which corresponds to a value of α equal to or just below `x`. If `x` is out of range of the data vector, than the first or last indices are returned as applicable. Using this function, another one can be written (also in global scope) to perform the linear interpolations as follows.

```
float interp(int f_index, float x, float y_arr[]){
  float y, xL, xU, yL, yU;
  int row_index = locate(alpha_arr[f_index], N_DATA, x);
  if (x > alpha_arr[f_index][N_DATA]) {
    Message("Warning: alpha is above upper bound.\n");
    y = y_arr[N_DATA - 1];
  }
  else if (x < alpha_arr[f_index][0]) {
    Message("Warning: alpha is below lower bound.\n");
    y = y_arr[0];
  }
  else {
    xL = alpha_arr[f_index][row_index];
```

```
    xU = alpha_arr[f_index][row_index + 1];
    yL = y_arr[row_index];
    yU = y_arr[row_index + 1];

    y = yL + ((x - xL)/(xU - xL))*(yU - yL);
  }

  return y;
}
```

This function is named `interp` and takes arguments of `f_index` (which Re_c file to retrieve data from), `x` (an α value to interpolate with), and `y_arr[]` (a vector of the C_L or C_D data to be interpolated from, as desired). Note that the during the function call, the `f_index` would be pre-determined for the given ALM and be supplied with an external scope variable of say `f`, and either `cl_arr[f]` or `cd_arr[f]` would be supplied to `y_arr[]`. At the start of this function, variables are declared according to a generic linear interpolation scheme where an interpolated value of `y` corresponding to input `x` is calculated using the lower and upper bound values for both. The previously written function `locate` is used to obtain a `row_index` for α. Extreme cases with accompanying warning messages are handled conditionally where if the α is out of bounds, the bound value of C_L or C_D is returned without interpolation. If it is not a fringe case, the `else` block is reached, where the upper and lower bound values are extracted (recall that `f_index` accesses the correct table and `row_index` and it incremented by 1 would access the correct row values) to solve and return `y`. Note that the code written in the current UDF is simply for a proof-of-concept in building an ALM — some effort has been made to improve program clarity and efficiency, but improvements are still very much possible.

3.3 Core ALM Computations

In this section, the entirety of the ALM calculation logic will be described and implemented. First, it is necessary to agree upon a specific approach. From the literature analysis, there are two predominant approaches based on a 3-D [61, 62, 63, 65] or 2-D [60, 3] Gaussian regularization kernel, with literature also forgoing the use thereof [9, 58]. This is the most important distinction to make in an implementation because the choice entirely dictates the manner of discretization to be done on the blades. Because of the interconnectedness of the literature for each approach, Bachant et al. [62] can be chosen as the representative for the 3-D approach while Zhao et al. [3] can represent the 2-D approach. Bachant et al.'s implementation can be understood in detail by reading the source code [64], whereas Zhao et al. did not publish their code, so the

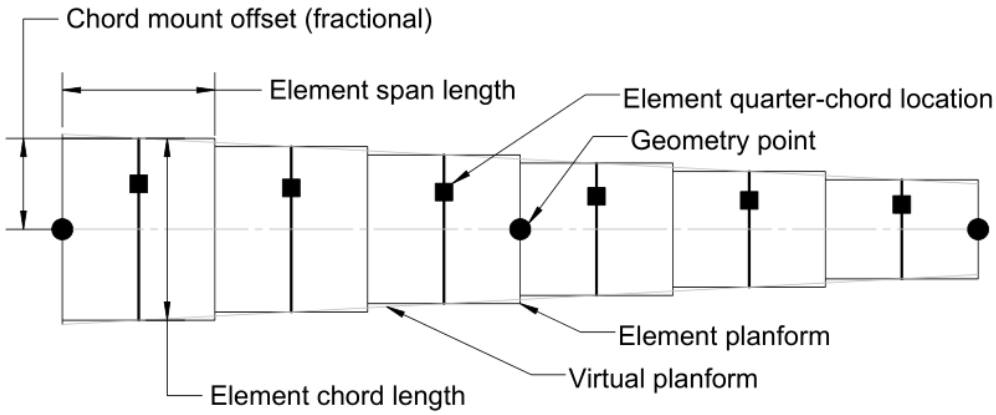


Figure 3.1: A diagram of the actuator line discretization performed within `turbinesFoam`, where circles denote points at which the geometry (e.g., chord) is defined and squares indicate the quarter-chord locations of an actuator line element, from the work of Bachant et al. [10].

most sensible interpretation will be used based on the paper. Since these two approaches are drastically different, the current implementation must decide which one to work with. It was decided that Zhao et al.'s approach be implemented as a proof-of-concept because Bachant et al.'s code is already published and accessible while Zhao et al.'s paper cites an experimental VAWT study by LeBlanc & Ferreira [2] for validation, enabling a direct comparison using the developed UDF code.

Due to the significant differences in approach, it is important to explain the framework-independent logic for the 3-D and 2-D Gaussian kernel formulations. Bachant's 3-D implementation is detailed in a thesis [10], and a diagram is extracted for display in Figure 3.1. In this approach, each blade's span is vertically discretized into *actuator line elements* of equal spanwise lengths. The aerodynamics of each element is encapsulated entirely by control points (*actuator points*) situated at the quarter-chord location, meaning it is at these locations that the point force for the element is computed based on the formulations of u_{rel} and α. After a point force in units of N has been computed, it is "convolved" with a 3-D Gaussian regularization kernel for cells in a spherical region of influence around the actuator point. This means that at each cell where force / momentum source imposition is required, a separate kernel value η must be calculated depending on the distance to the actuator line (the greater the distance, the lower the "weight"). The Gaussian regularization kernel itself does not consist of a cut-off distance beyond which no force imposition is done. This is particularly important to implement, otherwise every cell in the domain will be included in the calculation, with the vast majority of non-local cells having

a very small η. Bachant addresses this using a cut-off radius defined as $c + \epsilon(\ln(1.0/0.001))^{1/2}$ (where c is the blade chord length) [64]. For the isotropic 3-D Gaussian kernel, the choice of this value as the threshold would account for 92.5% of the total (probability) mass encompassed by the Gaussian kernel. According to Bachant, this radius is a good approximation for capturing almost the entire kernel, which is to say nearly 100% of the initial point force can be distributed and projected. The actual convolution is essentially implemented as a multiplication of the single point force with the cell-wise kernel η [62, 10]:

$$\eta = \frac{1}{\epsilon^3 \pi^{3/2}} \exp\left[-\left(\frac{|\vec{r}|}{\epsilon} \right)^2 \right] \tag{3.1}$$

which is a 3-D form where $\vec{r}$ is the distance from the actuator point to the cell at which η is being evaluated. When a force is convoluted with this 3-D Gaussian kernel, the output units is force per volume or Nm^{-3}. In summary, the 3-D approach of discretizing and evaluating blade span elements according to BE-M aerodynamics is very close to Sørensen & Shen's [54] original formulation.

In contrast, Zhao et al.'s approach [3] makes use of a 2-D Gaussian kernel described by:

$$\eta_i(d_i) = \frac{1}{2\pi\sigma^2} e^{-\frac{d_i^2}{2\sigma^2}} \tag{3.2}$$

where d_i is the notation for cell distance to the blade and σ is the Gaussian width (same as ϵ). While d_i is calculated based on a local blade coordinate (x_i, y_i), which is similar to Bachant's actuator point, the Gaussian kernel convolution works differently. η is still evaluated at every relevant cell within some threshold of $d_i \leq 2\sigma$, except the cell-wise convolution is not with respect to a single force at the actuator point, but rather with a local cell force. Furthermore, the spanwise discretization is done to the limit supported by the mesh fineness, which means the spanwise length of an element ds is equal to the cell size in this direction dz. Therefore, each 2-D slice of the blade (lying on the x-y plane) is projected onto neighboring cells within a circular region originating at the quarter-chord location of the blade. At each cell instead of only at the actuator points, the BE-M formulations for an airfoil section is computed based on u_{rel} and α. Essentially, this means that each cell in a 2-D slice is modelled as the blade airfoil with the force per unit span Nm^{-1} (instead of a point force) adjusted / scaled by the weight embedded with the corresponding 2-D kernel value to produce an output of Nm^{-3}. Notice the way the 2-D kernel transforms the units is different than the 3-D one by a length unit.

For the current implementation, Zhao et al.'s approach is chosen. Now, to write the actual code, the ALM logic to be executed at every time step should be decomposed into `DEFINE_ADJUST` and `DEFINE_SOURCE` logic, which execute sequentially. As mentioned above, it is not desirable to

handle a significant amount of cell computations in a centralized manner within DEFINE_ADJUST since momentum source terms cannot be imposed outside of DEFINE_SOURCE. Thus, through working backwards it is possible to find that all the core logic can be handled entirely inside DEFINE_SOURCE on a cell-by-cell basis, although adjusting the blade (actuator line) locations based on rotation can certainly be centralized in DEFINE_ADJUST for clarity. Thus, the following framework should be laid out in the global scope, wherein a separate macro must be defined for the x and y momentum sources and arguments such as d, c, t, dS, and eqn are standard boiler-plate objects for handling aspects of the solver / mesh:

```
DEFINE_ADJUST(actuator_line_model,d){
  /* code */
}
DEFINE_SOURCE(xmom_source, c, t, dS, eqn){
  /* code */
}
DEFINE_SOURCE(ymom_source, c, t, dS, eqn){
  /* code */
}
```

The following section will describe the lines of code to be inserted into each respective macro. The entire source code is presented one block at a time to facilitate understanding. First, in the global scope, it would be important to include the following declarations associated with the core ALM logic (recall that specific values can be tuned according to the particular case and the current values are sample ones for validation with Zhao et al.'s results).

```
const double PI = 3.14159265;
const int N_TURBINE = 2;
const int N_BLADE = 2;
const double z_lb = 0.675; // [m]
const double z_ub = 2.175; // [m]

const double u_fs = 4.01; // [m/s]
const double tsr = 3.7; // [-]
const double radius = 1.48 / 2.0; // [m]
const double rot_speed = (tsr * u_fs) / R; // [rad/s]
const double chord = 0.075; // [m]
const double dyn_viscosity = 0.0000181; // [kg/(ms)]

const double epsilon = 0.1;
const double threshold = 3.5*epsilon;

```

```
double act_line_pos_x[N_TURBINE][N_BLADE];
double act_line_pos_y[N_TURBINE][N_BLADE];
double theta[N_TURBINE][N_BLADE];

double Fx_total[N_TURBINE][N_BLADE];
double Fy_total[N_TURBINE][N_BLADE];
```

In the above code, `N_TURBINE` is the number of turbines to be modelled, `N_BLADE` is the number of blades per turbine, and `z_lb` / `z_ub` are the upper and lower bounds of the blade span in the z (spanwise) coordinate. Next, `u_fs` is the freestream (inlet) velocity, `tsr` is the tip-speed ratio, `radius` is the rotor radius, `rot_speed` is the constant rotor rotational speed, `chord` is the blade chord, and `dyn_viscosity` is the dynamic viscosity of air. The Gaussian kernel is defined by the parameters `epsilon` for the Gaussian width and `threshold` for the cut-off radius, currently defined as some function of `epsilon`. `act_line_pos_x` and `act_line_pos_y` are the actuator line location arrays, which store coordinates according to turbine and blade indices. `theta` is the time-varying azimuthal position of blades. Finally, `Fx_total` and `Fy_total` are the storage arrays for the total F_x and F_y rotor forces at a time step.

With the above established, the following code for updating the actuator line positions in each time step can be added inside the `DEFINE_ADJUST` macro.

```
const double rotor_x_init[N_TURBINE] = {1.5698,  0.00};
const double rotor_y_init[N_TURBINE] = {0.4598, -1.11};
const double theta_i[N_TURBINE][N_BLADE] =
    { {90*(PI/180), 270*(PI/180)},
      {90*(PI/180), 270*(PI/180)} };

// i is turbine index, j is blade index
for (int i = 0; i < N_TURBINE; i++) {
  for (int j = 0; j < N_BLADE; j++) {
    theta[i][j] = theta_i[i][j]
                  + rot_speed*CURRENT_TIMESTEP*N_TIME;
    act_line_pos_x[i][j] = radius*cos(theta[i][j])
                           + rotor_x_init[i];
    act_line_pos_y[i][j] = radius*sin(theta[i][j])
                           + rotor_y_init[i];
  }
}
```

In the first chunk of code, the initial x and y positions of the turbine centers are defined along with the initial azimuthal positions of the blades (`theta_i`). These are placed in the local

macro scope since no other macro would require direct access. In the second chunk, a nested loop iterates through all blades of all turbines to update the positions. First, the new azimuthal position of the blade is computed and stored in `theta`. The position as a function of time is the initial position added to a total change due to rotation. Note that here, `CURRENT_TIMESTEP` and `N_TIME` are Fluent-provided APIs for accessing the current time step size and number of time steps, respectively. With the azimuths calculated, the actuator line positions are simply a decomposition into the x and y Cartesian coordinates. After this set of calculations, which occurs in each iteration, the actuator line position arrays will hold the updated values in the global scope, to be subsequently referenced by `DEFINE_SOURCE`. As mentioned above, two macros would have to be written for x and y momentum source terms, separately. This is because each `DEFINE_SOURCE` macro is called successively in a cell loop, and at the end returns a `source` variable to the solver. The component (x, y, z) or type of source that is returned completely depends on *where* the macro is hooked in the Fluent GUI (i.e., the macro itself does not indicate what component / type it is). Therefore, code duplication between the x- and y-momentum macros becomes inevitable. The following code snippets will walk through the implementation of the x-momentum macro, which is identical to the y version in calculating a cell-wise force imposition except that it returns and logs the x component at the end. To begin, declarations are made within macro scope as usual:

```
real centroid_arr[ND_ND];
real source = 0;
C_CENTROID(centroid_arr, c, t);

double gauss_kernel;
double dist[N_TURBINE][N_BLADE];
```

The first part declares variables necessary for the Fluent solver API and source term injection. `centroid_arr` is an array for storing the (x, y) or (x, y, z) coordinates of the centroid of a cell which is being probed, depending on whether it is a 2-D or 3-D CFD case, respectively (this is handled automatically). The array indices corresponding to the coordinates are ordered 0, 1(, 2). When the `C_CENTROID` built-in function is called with such a provided array, with a cell reference object `c` (must be the same one specified in macro arguments), and with a cell thread reference object `t` (again specified in macro arguments; its meaning is not relevant to implementation), it will store the coordinates of `c`'s centroid into the array. The `source` is a variable created for the purpose of storing the source term value before being returned to the solver. Note that the type `real` is not defined in C, but is rather a type that only Fluent UDFs will recognize, and will translate into single or double precision types during compilation automatically. These are used here for the purpose of ensuring compatibility and good precision with Fluent APIs, but it is not recommended to declare all `double` / `float` as `real`. The `gauss_kernel` variable will hold the value of η and the `dist` array will store the distance of the current cell centroid to the

actuator points of all blades (across all turbines) for a threshold check. The following conditional structure with nested loops constitutes the essential logic of ALM:

```
if (centroid_arr[2] >= z_lb && centroid_arr[2] <= z_ub) {
  for (int i = 0; i < N_TURBINE; i++) {
    for (int j = 0; j < N_BLADE; j++) {
      dist[i][j] = sqrt(  pow(centroid_arr[0]
                          - act_line_pos_x[i][j], 2)
                        + pow(centroid_arr[1]
                          - act_line_pos_y[i][j], 2));
      if (dist[i][j] <= threshold) {
        double cell_volume = C_VOLUME(c, t);
        double u = C_U(c, t);
        double v = C_V(c, t);

        double u_az = (-1 / radius)
                      * (v * radius * cos(theta[i][j])
                        - u * radius * sin(theta[i][j]));
        double u_rel_mag = sqrt(  pow(u, 2) + pow(v, 2)
                                + pow(radius
                                  * rot_speed, 2)
                                + 2.0 * radius
                                  * rot_speed * u_az);
        // coordinate system correction
        if (theta_rel > 0) {
          theta_rel = PI - theta_rel;
        }
        else if (theta_rel < 0) {
          theta_rel = -1 * theta_rel - PI;
        }

        double beta = 0.0;
        double alpha = theta_rel - beta;

        double cell_density = C_R(c, t);
        double Re_c = (cell_density / dyn_viscosity)
                      * u_rel_mag * chord;

        double C_lift, C_drag;
        // corresponds to Re_c = 1,000,000 table
        int file_inex = 8;
```

```
        C_lift = interp(file_index, alpha * (180/PI),
                        cl_arr[file_index]);
        C_drag = interp(file_index, alpha * (180/PI),
                        cd_arr[file_index]);

        double F_lift_ps = 0.5 * cell_density
                           * C_lift * pow(u_rel_mag, 2)
                           * chord;
        double F_drag_ps = 0.5 * cell_density
                           * C_drag * pow(u_rel_mag, 2)
                           * chord;

        F_lift_ps = -1 * F_lift_ps;
        F_drag_ps = -1 * F_drag_ps;

        double F_tang_ps = F_lift_ps * sin(theta_rel)
                           - F_drag_ps * cos(theta_rel);
        double F_norm_ps = F_lift_ps * cos(theta_rel)
                           + F_drag_ps * sin(theta_rel);
        double F_x_ps = -1 * F_tang_ps * sin(theta[i][j])
                        + F_norm_ps * cos(theta[i][j]);
        double F_y_ps = -1 * F_tang_ps * cos(theta[i][j])
                        + F_norm_ps * sin(theta[i][j]);

        gauss_kernel = (1.0 /
                          (2.0 * PI * epsilon * epsilon))
                       * exp(
                          (-1.0 * dist[i][j] * dist[i][j])
                          / (2.0 * epsilon * epsilon));
        // force per volume [N/m^3]
        source = source + F_x_ps*gauss_kernel;
      }
    }
  }
}
```

The general logic of this code relies on the 2-D implementation where for any cell in the domain, it can only have force projections due to influences from actuator points lying on the same z-plane. Therefore, for efficiency, the first conditional checks whether the current cell is within the z bounds of the blade span, since otherwise there is no reason to perform any

calculations. Next, two nested loops iterate over each blade of each turbine (thus providing support for multi-turbine ALM) with `i` being the turbine index and `j` being the blade index. Thus, for each blade, the `dist` relative to the current cell's centroid is computed and the subsequent `if` statement only permits further evaluation if the current cell is within the range of influence of a particular blade. Next, a series of calculations are performed in a straightforward manner according to Zhao et al.'s ALM formulation for compatibility during validation [3]:

$$u_{az} = -\frac{1}{r}(xv - yu) \tag{3.3}$$

$$|\vec{u}_{rel}| = \sqrt{u^2 + v^2 + u_{bl}^2 + 2u_{az}u_{bl}} \tag{3.4}$$

$$\theta_{rel} = \tan^{-1}\left(\frac{u\cos\theta + v\cos\theta}{-u\sin\theta + v\cos\theta - \omega_{bl}r}\right) \tag{3.5}$$

where u_{az} is the azimuthal component of the fluid velocity, (x, y) are the coordinates of the actuator line, (u, v) are the local fluid velocity components, and u_{bl} is the relative velocity encountered by the blade due to rotation ($u_{bl} = r\omega_{bl}$, where ω_{bl} is the angular velocity of the blade). θ_{rel} is effectively the angle of attack α in the absence of blade pitching. The conditional logic involving `theta_rel` is to align the sign convention of α with that of the current frame of reference for the airfoil look-up. `C_U`, `C_V`, and `C_R` are Fluent-defined macros for probing the local u, v, and ρ of a cell, respectively.

For the airfoil coefficient look-up, a `file_index` can be specified in the current scheme, which is an index corresponding to the correct Re_c file (e.g., 8 corresponds to $Re_c = 10^6$). Subsequently, the previously defined `interp` function can be called with the index, α of the cell in degrees, and the desired coefficient data array. Forces are calculated as lift per unit span length (as denoted by the `ps` part of the variable names) and are converted from the lift / drag components to tangential / normal and finally x / y components. The direction of the forces are reversed by multiplying -1 in order to obtain the equal and opposite reaction of the blade on the flow field, which is key to ALM. The `gauss_kernel` for the cell is obtained, multiplied with the x / y force component at the same cell (depending on which macro), and then added to the `source` variable. This allows for influences for all turbines on the current cell to be considered.

```
return source;
```

Finally, the line above should be included at the end of the macro, outside of all the nested loops and conditional blocks, to actually impose the source term onto the solver flow field in the current iteration. If there is no source term (e.g., the current cell is outside the impact range of ALM), the initialization value of 0 will be returned, indicating to the solver that no modifications need to be done for the equation of this cell.

3.4 Results Logging

For this basic but complete implementation of the ALM in the UDF, it is desirable to have a way of logging and outputting results, such as the ALM forces of the blades imposed on the flow field. Due to the parallel nature of UDF execution in Fluent, a logging solution is tricky. Recall the early introduction to parallelized UDF execution: all cells in the domain are "assigned" to a particular compute node (a minimum of one), but a `HOST` node is still present in the serial case. For domains split into multiple partitions (to be executed in parallel), certain "external cells" are shared between two adjacent partitions at the boundary, which means that a UDF with `DEFINE_SOURCE` runs on the same cell twice on 2 partitions. The instances are separate / parallel and the source term interaction with the solver is intrinsically handled by the Fluent UDF framework, so the physics are sound without the need for additional compensation. However, if logging was to be done on the F_x and F_y calculated for this cell which will then be aggregated across all partitions, the contribution of external cells would be accounted for twice, leading to an incorrect total force across the turbine. In summary, there are three problems to be addressed for logging:

1. Avoid duplication of external cell force contributions across partitions
2. Communicate partition forces between compute nodes and aggregate forces for a per-time-step log
3. Write the result log into a file for post-processing

The first two problems are related and can be handled using parallel control APIs provided by Fluent. It should be emphasized that compute nodes do not communicate any UDF-related information by default and instead work within the scope of their assigned cells. For the sake of illustration, assume there are two nodes, `n0` and `n1`. All cells in a domain are uniquely traceable by some id, and say cells 0 to 9 are delegated uniquely to `n0`, 10 to 14 are external cells shared between `n0`/`n1`, and 15 to 24 are delegated to `n1`. The same UDF will be run on `n0` and `n1` as separate instances, being updated according to the subset of cells covered, and so far the UDF implementation is completely agnostic of the specific node it is running on. Since the sum of the forces for an entire blade is desirable regardless of mesh partitions, it is conceivable that each node can somehow write its force sum into a log file after every time step. However, this would mean additional post-processing overhead in aggregating the data afterwards. Instead, a better option would be to compute the sums across parallelized node instances and then output it from only one node. The following code snippets will illustrate one method for doing this, but it is not the most elegant one achievable.

```
double global_Fx_total[N_TURBINE][N_BLADE];
double global_Fy_total[N_TURBINE][N_BLADE];
```

The above code should be added to the global scope, with the force arrays serving as storage variables for a "global" sum across all compute nodes. To address the third problem listed above, it is desired to use the UDF to output a single line of results (that presumably have already been combined across partitions) at the end of a time step. However, DEFINE_ADJUST executes before and not after DEFINE_SOURCE, where the results are actually produced on a cell-by-cell cumulative basis. Thus, a crude solution has been developed to maintain the ALM force logs until the start of the next iteration, at which time they will be output into files within DEFINE_ADJUST. This staggered / delayed logging process works, but is less elegant than a later discovered solution that will also be presented. Nevertheless, it is presented here as a demonstration of the workings of UDFs. The following block of code can be inserted into DEFINE_ADJUST.

```
for (int i = 0; i < N_TURBINE; i++) {
  for (int j = 0; j < N_BLADE; j++) {
    global_Fx_total[i][j] = PRF_GRSUM1(Fx_total[i][j]);
    global_Fy_total[i][j] = PRF_GRSUM1(Fy_total[i][j]);
  }
}

if (first_iteration) {
  if (myid == 0) {
    FILE *f_data_out_1;
    f_data_out_1 = fopen("t1_force_log.txt", "a");
    fprintf(f_data_out_1, "%f,%f,%f,%f,%f,%f\n",
            PREVIOUS_TIME,
            theta[0][1] - rot_speed * CURRENT_TIMESTEP,
            global_Fx_total[0][0],
            global_Fx_total[0][1],
            global_Fy_total[0][0],
            global_Fy_total[0][1]);
    fclose(f_data_out_1);

    FILE *f_data_out_2;
    f_data_out_2 = fopen("t2_force_log.txt", "a");
    fprintf(f_data_out_2, "%f,%f,%f,%f,%f,%f\n",
            PREVIOUS_TIME,
            theta[1][1] - rot_speed * CURRENT_TIMESTEP,
            global_Fx_total[1][0],
```

```
            global_Fx_total[1][1],
            global_Fy_total[1][0],
            global_Fy_total[1][1]);
    fclose(f_data_out_2);
  }
}

for (int i = 0; i < N_TURBINE; i++) {
  for (int j = 0; j < N_BLADE; j++) {
    Fx_total[i][j] = 0.0;
    Fy_total[i][j] = 0.0;
  }
}
```

The first chunk of code will be evaluated at the start of each iteration. It consists of a call to the Fluent-defined `PRF_GRSUM1` function which aggregates (globally *reduces*) the instance / node-specific `Fx_total` and `Fy_total` array values into a sum to be stored in the global variables. This occurs indiscriminately in all nodes and combines information from all nodes. Afterwards, the logging can commence after checking the boolean API `first_iteration`, which is only true if the current macro run is the first iteration in a new time step. Also, it is checked to ensure that only a node with `myid` of 0 (is `n0`) performs the logging (since this node will exist even in a serial run). When these checks are fulfilled, separate file objects for the turbine 1 and 2 logs are used to write a line consisting of the flow time, azimuthal position of the second blade, and each blade's F_x and F_y. Note that the built-in variable of `PREVIOUS_TIME` is used to collect the actual flow time at which these results are logged and the azimuth is also reset to that of the last time step by subtracting the rotational speed times the time step size. Finally, at the end of each `DEFINE_ADJUST` execution, the partition's force logging arrays are zeroed to prepare afresh for the current iteration (this is necessary since all cells in `DEFINE_SOURCE` only accumulate their contributions, as is shown below). A simplification could be achieved using the `DEFINE_EXECUTE_AT_END` macro, which as its name entails, can contain code for the global reduction and file output of ALM forces for the converged time step.

```
if (C_PART(c, t) == myid) {
  Fx_total[i][j] += F_x_ps * gauss_kernel * cell_volume;
}
```

The above code when included in each `DEFINE_SOURCE`, right after updating `source`, can ensure duplicate logging is avoided. The conditional check compares `C_PART` to `myid`. The former determines which partition is defined to be the main / principal partition that a cell belongs to. Carrying on the simple example from above, this would return 0 (for `n0`) when run in cells 0

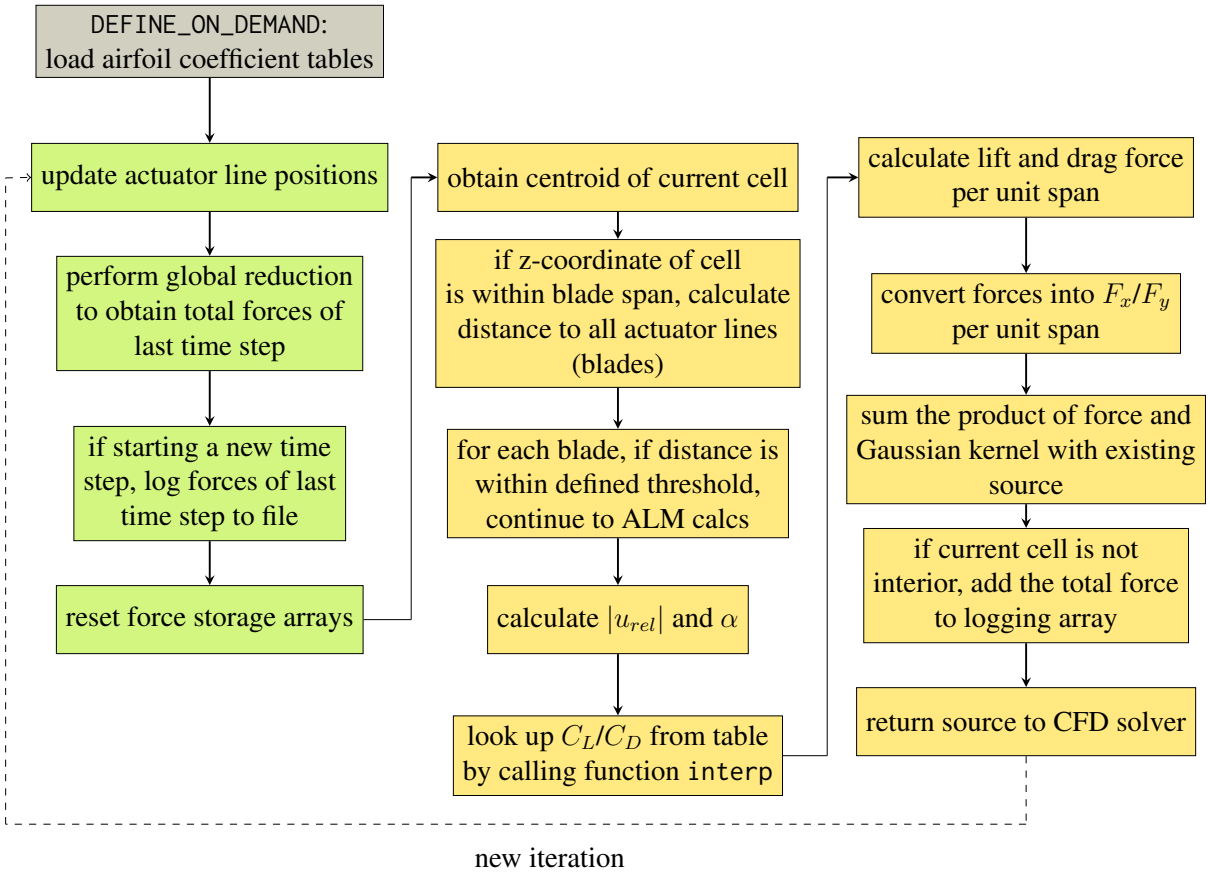


Figure 3.2: UDF program flowchart of the ALM implementation. Green blocks represent logic to be performed in DEFINE_ADJUST and yellow blocks represent DEFINE_SOURCE (for x / y-momentum)

to 9 and 1 for 15 to 24. Most importantly, even the shared external cells 10 to 14 have a unique, unambiguous value. With this check, it is guaranteed that logging only happens once for the main partition of an external cell, where a cell's force contributions in N would be added to the total per rotor blade so far (within the partition). This total will then be globally reduced and synchronized into a sum across all nodes.

3.5 Complete Program Flow

Now that the implementation is complete, a high-level overview can be presented. A demonstrative UDF program flow chart has been illustrated in Figure 3.2. At the beginning of the UDF, a one-time airfoil table loading procedure can be carried out using the `DEFINE_ON_DEMAND` macro. Afterwards, for each iteration of each time step, the `DEFINE_ADJUST` macro executes before the x / y-momentum `DEFINE_SOURCE` macros. In `DEFINE_ADJUST` for each partition / compute node, the actuator line position is updated according to a prescribed rotation and a global reduction is done to synchronize and aggregate total forces across partitions. In particular, for a specific turbine, its forces may be imposed on cells handled by different partitions, which by definition are isolated without parallel communication operations. If the current iteration is at the start of a new time step, that means the forces stored from the previous time step can be logged to files as the converged ALM results. Finally, at the end of every execution of this macro, the local partition / node's force arrays are reset to 0 to allow for a fresh accumulation in the ensuing `DEFINE_SOURCE` step. Therefore, the end of `DEFINE_ADJUST` of the next iteration effectively marks the end of the previous iteration's ALM logging logic. However, this can be simplified by simply delegating logging logic to a `DEFINE_EXECUTE_AT_END` macro at the end of the same time step.

Afterwards, `DEFINE_SOURCE` for both components of momentum are executed — the specific order and whether they are parallelized does not affect the functionality, since it is guaranteed for all source terms to be produced before proceeding to the CFD solver. For each component, much of the same logic must be implemented, which includes a linear flow of calculating u_{rel}, α, C_L / C_D, the point force components, and finally the kernel η after ensuring a particular cell is within the blade span in z and is in close proximity to an influencing blade. The reason for this duplication is the lack of an API for returning multiple types / components of source terms in one place. Thus, to avoid repeating the potentially expensive airfoil look-up functions, it would be necessary to create globally scoped storage arrays (or perhaps User Defined Memory) to store the source values for y at the time they are readily solved for in the x macro. However, this introduces issues with look-up costs (cells looped through for y must use a cell identifier to access the corresponding source value) and depends on a completely sequential and not parallel execution of `DEFINE_SOURCE` macros. Therefore, in this manner the UDF framework can be quite limiting for an ALM implementation.

Finally, the limitations of the current proof-of-concept implementation with respect to literature should be acknowledged. First, the constant rotational speed of turbines imposed here is not an accurate representation of the dynamics, but it is a standard assumption to simplify simulation in both full-order URANS studies for synergy and ALM studies. Furthermore, it is also standard that ALM produces no spanwise forces (in the z direction), generally lacks the ability to predict 3-D unsteady effects such as tip vortices [58, 63, 65], and performs somewhat poorly in pre-

dicting detailed wake characteristics [9, 60, 61]. The current thesis makes no attempt to remedy these deeply rooted issues in the above implementation. Similarly, all literature rely heavily on airfoil performance look-up tables (predominantly the ones reported by Sheldahl & Klimas [59]), though Hezaveh et al. [58] employed decoupled URANS runs to produce the force coefficients. This means that the accuracy of the dataset is a crucial baseline upon which additional correction models can be introduced to fine-tune the aerodynamic predictions. For instance, Bachant et al. [62] and Abkar [61] introduced dynamic stall models and tip-loss corrections in an attempt to modify the effective angle of attack α or the lift / drag coefficients to more robustly capture the physical minutia. This implementation opted to forego these corrections partially because Bachant et al. found that the dynamic stall modelling produced accurate results in one turbine but not the other (likely attributed to the improper tuning of a time constant parameter T_α [62]) and also because it will be validated with Zhao et al.'s case [3], which did not employ any such corrections (e.g., dynamic stall is not necessary since the TSR of interest is unaffected). The absence of turbine struts/shafts (potentially simulated as drag-producing elements), consistent with Zhao et al.'s setup, was also decided upon for the sake of a simpler proof-of-concept of the "bare-bones" ALM involving only the blades. Lastly, it is emphasized that the Gaussian kernel width (denoted as ϵ here) is a tuning parameter that can affect the physics of the entire ALM-embedded CFD. Because of its possible significance in influencing the converged turbine performance and wake development and the general lack of consensus or precision in its determination in literature (see Section 2.4), the following chapter will dedicate some efforts to exploring different ϵ values.

Chapter 4

Comparison of ALM Codes and Verification of the Isolated Turbine Case

Before proceeding with multi-turbine ALM in order to investigate synergy, which is the main focus of this thesis, it is important to verify that an isolated turbine case is implemented correctly and also validate the results to some extent. The main case to be focused on is the 2-bladed, H-type VAWT employed in Zhao et al.'s ALM study [3], which has been validated with good agreement to a wind tunnel experiment of the same turbine [2]. This is chosen due to the availability of data for the turbine's F_x and F_y as a function of rotor azimuth from both the converged ALM results and experimental measurements at TSR $= 3.7$. Because the UDF implementation heavily references Zhao et al.'s specific approach (despite the lack of publicly available source code), it is expected that Zhao et al.'s results can be reproduced with an identical case setup. At the same time, Bachant et al.'s OpenFOAM code [64] remains a candidate for running a large number of multi-turbine cases. Therefore, its results for the same isolated turbine will also be compared. Therefore, the structure of this chapter will be as follows. First, the setup of the TSR $= 3.7$ case will be detailed, followed by a study of how the mesh, the Gaussian width parameter, and the threshold radius for the distribution affect the ALM forces. Next, comparisons of the current Fluent UDF implementation, Bachant's code, and the full-order / blade-resolved CFD with the experimental forces will be presented in sequence, with each newly introduced result added to the multi-way comparison. Finally, a discussion of the validation will be presented with commentary on possible reasons for the discrepancy and justification for proceeding with Bachant et al.'s code to study multi-turbine ALM synergy in Chapters 5 and 6.

Property	Value
Turbine diameter (D)	1.48 m
Number of blades	2
Airfoil type	NACA 0021
Blade chord (c)	0.075 m
Blade span (L)	1.5 m
Blade pitch (β)	0°
Tip-speed ratio (λ)	3.7
Rotational speed (ω)	20.05 rad / s

Table 4.1: List of ALM and turbine properties and parameters

4.1 Case Setup

The case setup involving the ALM-related / turbine parameters are shown in Table 4.1 (obtained from [3, 2]) and the CFD parameters are shown in Table 4.2 (obtained from [3]). As indicated, the operating TSR will be set to 3.7 for the remainder of this study, with the reason being that dynamic stall can be avoided and detailed force trends are available for validation at this operating point. A turbine rotational speed of 20.05 rad / s is determined based on the TSR, the freestream velocity is 4.01 m / s, and the rotation direction is counter-clockwise (CCW) for all isolated turbines. The airfoil type is NACA 0021 and the $Re_c = 10^6$ dataset (from [59]) is used despite the actual operating Re_c being given by Zhao et al. is a single value of $19,838$ [3], which is consistent in order of magnitude with the *range* of Re_c within a revolution calculated by the current UDF implementation. Zhao's explanation (pers. comm., Apr. 23, 2021) for this is that the NACA 0021 table from Sheldahl & Klimas [59] is synthesized and assumes static airfoils, which differs from the actual dynamic, multi-bladed application, so the best-fitting table is used — this point will be further examined and discussed in a later section. For the ALM, the Gaussian kernel width η and the cut-off radial threshold (beyond which force distribution is not applied) are important parameters that will be explored in the next section. Zhao et al. does not specify the Gaussian width that is used, but states that it is calculated similarly as Bachant et al. [62], which is $\eta = 4\Delta x$ where $\Delta x \approx \sqrt[3]{V_{cell}}$. The cut-off is at a radius of 2ϵ [3].

For the Fluent case setup, time step size is set to 0.003 s for consistency with Zhao et al.'s case. Zhao finds this to be an acceptable size for accuracy, and a current test run using 0.002 s confirms that very minimal changes can be observed in F_x and F_y. Each revolution will complete in 0.313 s and require about 105 time steps. The domain size also reproduces that of Zhao et al., where the cross-section is a square and the flow domain resembles that of a rectangular channel. The lack of significant space in the cross-stream direction for flow development means a blockage

Parameter	Value
Inlet (freestream) velocity	4.01 m / s
Turbulence model	SST k-ω
Inlet turbulence kinetic energy	0.24 m^2 / s^2
Inlet specific dissipation rate	1.78 s^{-1}
Density (ρ)	1.207 kg / m^3
Time step size	0.003 s
Domain size: $L_x \times L_y \times L_z$	13 m $\times$ 2.85 m $\times$ 2.85 m

Table 4.2: List of CFD case properties

effect would be simulated, whereby the forces / power would be artificially inflated from the actual magnitudes, which is recognized by Zhao et al. [3]. The lateral wall boundary conditions are all set to symmetry, meaning ground effects are not simulated. The turbine is situated at 4.5 m downstream of the inlet in the center, and thus the blade span is between $z = 0.675$ m and 2.175 m. It appears that Zhao employs a roughly uniform cell size of $\sim$0.1 m in each direction for the base mesh, which is then refined by a factor of 2 into a size of $\sim$0.05 m in a refined region spanning from 2.5 m upstream of the turbine to 3.5 m downstream. This setup is shown in Figure 4.1. Note that since the dynamic viscosity is not provided by Zhao et al. nor LeBlanc & Ferreira, so a standard value of 1.81×10^{-5} kg m^{-1} s^{-1} has been used when it was of interest to determine the Re_c (not part of ALM logic) and in the CFD solver. It is expected that implications arising from any discrepancy would be negligible.

4.2 Study of Mesh and Gaussian Width Parameter

As the start of the ALM run, the virtual blades immediately begin rotating according to prescribed trajectories, though the initial flow field is one unperturbed by their presence. Therefore, by coupling the ALM and URANS CFD, a feedback loop is formed whereby flow conditions are modulated by momentum source term impositions supplied by ALM. Then, after an iteration completes, the new set of flow velocities are used to calculate updated source terms. As the wake emerges, the local velocity deficits become prominent in impacting the forces on the downwind half of each blade's trajectory, leading to lower F_x and F_y compared to the initial revolution. Gradually, because of a constant rate of rotation, the transient simulation *periodically converges* onto a time-invariant trend of F_x and F_y (the forces experienced by the turbine according to ALM) down to some small error threshold. At this point, further simulated revolutions would develop the wake downstream but would not predict significantly different force

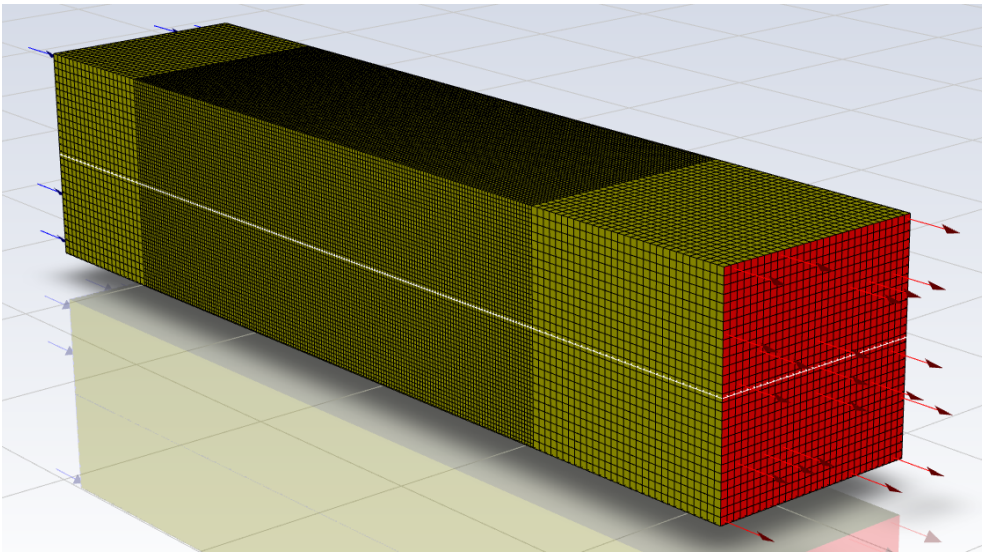


Figure 4.1: The standard mesh used for Fluent ALM with a dimension of $L_x \times L_y \times L_z$: 13 m $\times$ 2.85 m $\times$ 2.85 m and a refined section between $x = 2$ m and 8 m.

trends, as characterized by the magnitude of peaks and troughs. Typically, it is observed that a visually convergent force time series can be reached after just 3-4 revolutions, but without exception the only development from the first revolution is a small attenuation of the F_x peaks and a lesser reduction of the F_y positive/negative peaks. However, to adhere to Zhao et al., 10 revolutions / cycles are run for cases that seek to determine a periodically converged result, which will be maintained for the later multi-turbine cases. For rapid iteration through different parameters in the current section, fewer revolutions or only the first revolution are examined and compared to save time, since the trend characteristics of interest are immediately observable.

Figure 4.2 displays the ALM-produced forces for an isolated turbine as a function of rotor azimuth. This format will be used for all force vs. azimuth plots in this chapter. These forces are aggregated from cells to which each blade element's forces have been distributed in the ALM. The convention of the blade azimuth is as shown in Figure 2.3a, with 0° aligned with the $+x$ axis and the angle increases CCW. Based on this system, all ALM cases are initialized where one blade is situated at an azimuth of 90° (top) and another is situated at 270° (bottom). For ease of visualization, this configuration is considered to be a "rotor azimuth" of 0°, which represents a steady progression of rotation of the entire rotor. F_x and F_y vs. rotor azimuth are deemed the most important result to compare and validate for amongst ALM runs because they enable a direct observation of the predicted turbine aerodynamics, thus allowing for straightforward quantitative comparison between cases. There may be some value in comparing mean flow profiles

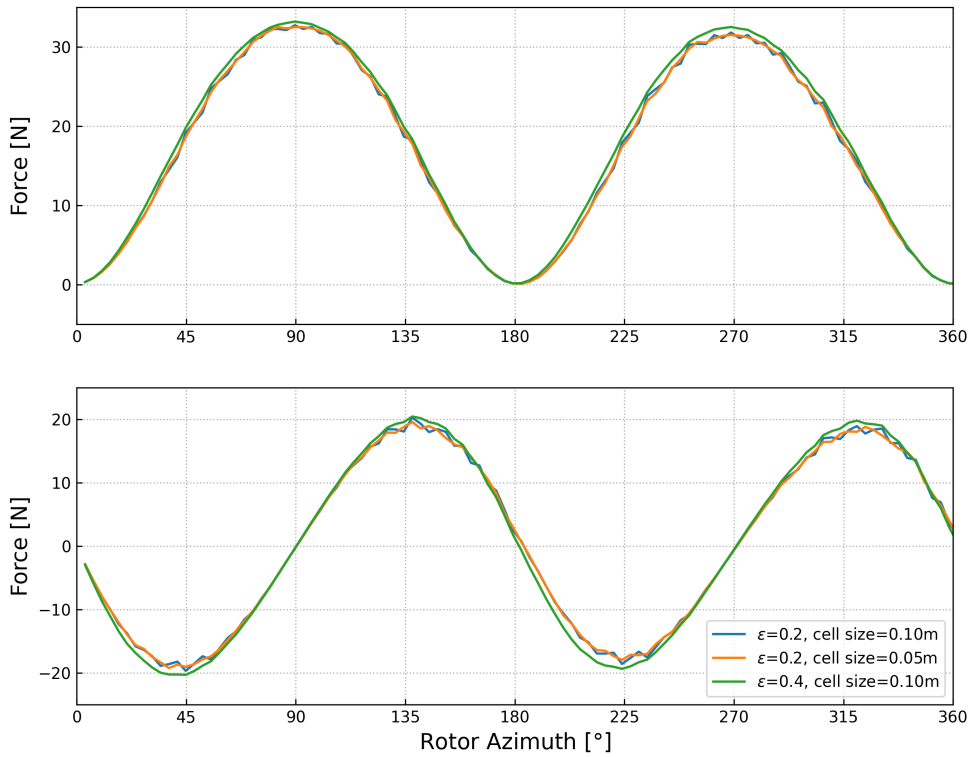


Figure 4.2: Turbine F_x (top) and F_y (bottom) as a function of rotor azimuth of a UDF implementation using the 3-D Gaussian kernel. Runs have varying Gaussian width (ϵ) and cell size, with results taken at revolution 1.

and characteristics, such as velocity / turbulence kinetic energy, though it is the forces that most directly drive power generation, the most important metric for quantifying multi-turbine synergy. As indicated in the figure caption, the data displayed in Figure 4.2 are of the first revolution of the turbine. Although this does not provide insight on the periodically converged forces, it is found that the initial revolutions are immensely useful for assessing relative differences in the forces while saving on computational time.

The results in this figure have been produced using an ALM approach different than the one established in Chapter 3. Instead of a 2-D Gaussian kernel based on Zhao et al. [3], this approach was initially developed as a prototype utilizing a 3-D Gaussian kernel and actuator point discretization logic similar to Bachant et al. [62]. It is interesting to present the results here for comparison with the main 2-D Gaussian approach to observe the implications of the chosen Gaussian distribution on ALM forces.

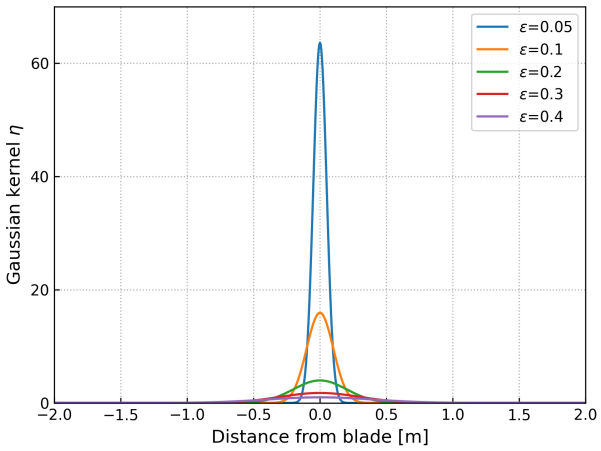


Figure 4.3: A demonstration of the distribution produced by the 2-D Gaussian kernel used in the current ALM implementation, where the kernel value η is plotted as a function of cell distance from the actuator line (blade quarter chord) for different ϵ.

It is apparent from the literature review that there is no consensus on a standard tuning procedure or general starting point for ϵ: Creech et al. [60] asserts the value should be roughly $(1/12)R$, Abkar [61] prescribes a range of 1 to 3 grid sizes, while Zhao et al. [3] and Bachant et al. [62] effectively uses about 4 times the grid size. Therefore, some exposition is clearly required on the choice of ϵ for simulating multi-turbine ALM and a range centered around 0.2 as per Zhao et al. is a sensible choice given that it is the main validation target. In Figure 4.2, a Gaussian width (ϵ) of 0.2 was simulated with a cell size of both 0.1 m and 0.05 m in the domain, and $\epsilon = 0.4$ was also tested for a cell size of 0.1 m. The cut-off threshold used was about 2.63ϵ following Bachant et al., and although the threshold should be scaled proportional to ϵ as is done here, the relationship is not linear due to the exponential nature of the Gaussian kernel. Thus, additional tuning was required in subsequent cases to establish a threshold commensurate with an ϵ value's effective range of influence while also avoiding an excessively large threshold to save time. The $\epsilon = 0.2$ case features some jaggedness in the curves, while the $\epsilon = 0.4$ case is much smoother, but with subtle variations near the peaks. For the $\epsilon = 0.2$ case, a smaller cell size also seems to reduce the jaggedness in the curve. However, across all three cases, it is evident that the forces are not significantly sensitive to a refinement of the mesh or a change of ϵ.

To explain the choice of ϵ, Figure 4.3 is generated for visualization. In Zhao et al.'s case, $\epsilon = 0.2$ was used. The current study attempts to vary this parameter and observe the scale / nature of its impacts on the ALM forces, and hence the very physics of the reduced-order model. Based on Figure 4.3 which uses Zhao et al.'s [3] 2-D formulation (different from the 3-D kernel used to

produce Figure 4.2, but it suffices for illustration), it is clear that a smaller ϵ produces a higher peak at the center, which is to say the distribution is more centrally concentrated. Recall that η is analogous to a weight multiplied onto a blade force to scale it down to a proper fractional contribution at some cell influenced by a nearby blade. Thus, at a value of $\epsilon = 0.05$, forces are highly concentrated and could begin to pose singularity issues for the solver. This is why Zhao et al. [3], Bachant et al. [62], and many other researchers assert that a smaller ϵ demands a smaller grid size. Because of the steep gradient, a smaller cell size (equivalent to some distance along the x-axis in the plot) from the discretization scheme means greater errors in approximating the integral of the curve. As an exaggeration, a 1 m cell centered on the blade would over-estimate η and thus the total force to be imposed on this cell. At the same time, while a larger ϵ relaxes demands on the grid size, the central tendency of the distribution becomes significantly weaker. At $\epsilon = 0.4$, the spatial presence of the virtual blade becomes barely perceivable, which may pose problems for the realistic emulation of blade-flow interactions. Thus, $\epsilon = 0.4$ is taken as a "soft" upper bound of the parameter under the current considerations. Given this context, it is significant that ALM forces remain almost invariant across the meaningful range of ϵ values.

Figure 4.4 illustrates a general comparison with results based on a 3-D Gaussian kernel with $\epsilon = 0.2$ (presented in 4.2) and 2-D Gaussian kernel results with $\epsilon = 0.1$ or 0.2, calculated using various threshold / mesh combinations. Immediately, it can be seen that the force trends are remarkably coherent and similar to each other, especially the 3-D Gaussian kernel results (marked by circles) relative to the other 2-D kernel results. This reasonably indicates that both methods have been implemented correctly and are consistent in capturing the trend and magnitude of ALM forces in an isolated VAWT. Between the coarse and regular mesh settings for an ϵ of 0.2, there are almost no differences in forces. However, further refining the mesh beyond that deemed valid by Zhao et al. for $\epsilon = 0.1$ led to a small increase in the F_x peaks. This suggests the extent of mesh refinement in the VAWT region required for grid independence varies depending on the Gaussian width parameter. Such an observation supports the assumption stated above: that discretization errors may arise from insufficiently small cell sizes for resolving increasingly centrally concentrated Gaussian distributions. Furthermore, the physical implications may also be significant, since each combination of ϵ and cell size produces a unique physical presence and feedback response from the solver in terms of the flow field. Thus, a future study seeking to meticulously perform a grid independence study must do so by iterating through many combinations with ϵ. These nuances notwithstanding, it should be noted that although there is a perceivable difference in the F_x peaks, the actual effect on mean C_p for a revolution (which is the most crucial result for studying synergy) remains insignificant. At the same time, a refinement factor of 2 in the cell dimensions leads to 8 times the number of cells. Thus, in the interest of maintaining a fairly feasible run duration for simulating numerous multi-turbine cases, a near-VAWT cell size of 0.05 m is sustained. Finally, with the increasing threshold values tested for $\epsilon =$

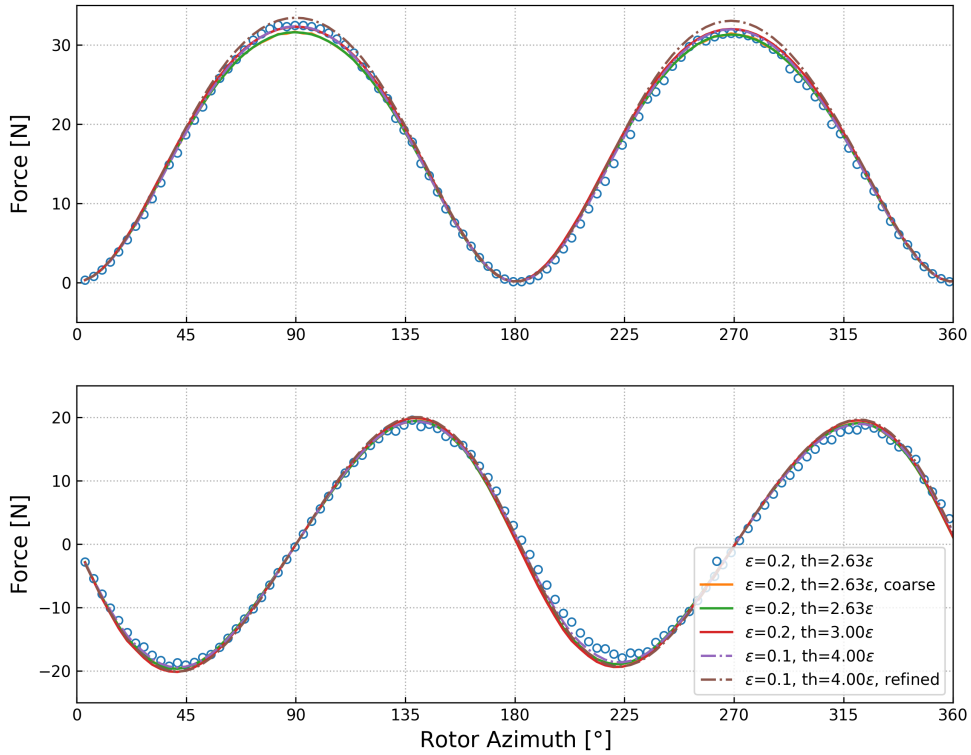


Figure 4.4: Turbine F_x (top) and F_y (bottom) as a function of rotor azimuth, where the circle marker denotes results from the 3-D Gaussian kernel implementation above and all lines are from the 2-D implementation that has been demonstrated. Cell size in the mesh are kept at 0.05 m, except where "coarse" and "refined" are noted, which indicate that a cell size of 0.1 m and 0.025 m have been used, respectively. Results are taken at revolution 1.

0.2, it can be seen that a threshold of 2.63ϵ is insufficient for capturing all the forces because 3ϵ produces slightly greater F_x near the peaks. In fact, this higher threshold value compares better with the 3-D implementation, which used 2.63ϵ. Note that thresholds of 2.63ϵ and 3ϵ encompass 96.8% and 98.9% of the total (probability) mass under an isotropic 2-D Gaussian distribution. Hence, small differences in F_x near the peaks must result from contributions in the extreme tails of the distribution that are neglected. As a point of reference, a threshold of 2.63ϵ encloses 92.5% of the total (probability) mass for an isotropic 3-D Gaussian kernel.

Figure 4.5 displays a more rigorous review of ALM forces resulting from a variation of only the Gaussian kernel cut-off threshold, for a constant ϵ of 0.1. These results are simulated for and presented at the 3rd revolution to allow for development that is much closer to periodic

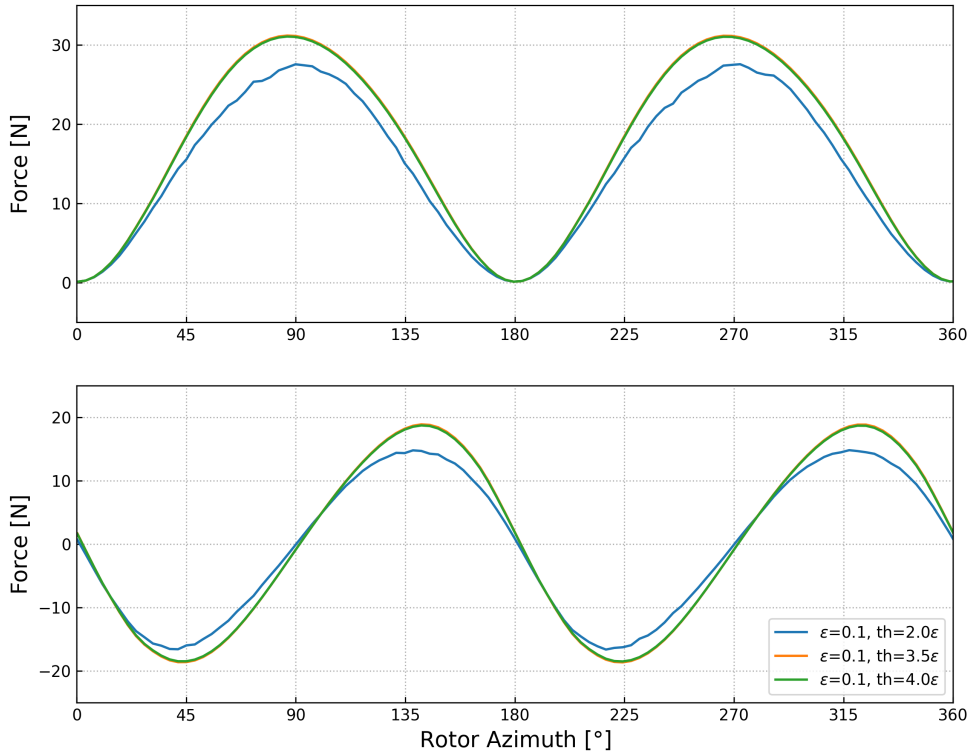


Figure 4.5: Turbine F_x (top) and F_y (bottom) as a function of rotor azimuth, where various threshold values are presented for $\epsilon = 0.1$. Results are taken at revolution 3.

convergence, thus eliminating any numerical artifacts that may hinder interpretive legibility. The cell size is refined to 0.05 m near the VAWT. From this figure, the impact of threshold is saliently demonstrated to be consistent with the hypothesis derived from its implementation. If the value is too small, forces will be noticeably underpredicted due to cells with the missing contributions eliminated outside the threshold; conversely, if the value is sufficient or greater, no difference in the forces will be observed. In Figure 4.5, a threshold of 2ϵ in this case is insufficient, so both F_x and F_y are significantly under-predicted near the peaks. In contrast, a threshold of 3.5ϵ is sufficient since increasing to 4ϵ shows virtually no difference, as expected. This exercise shows that threshold need not be considered as a third variable in addition to ϵ and mesh when verifying a case. Instead, for an arbitrary case, the threshold can be increased until forces remain invariant.

In Figure 4.6, a greater range of Gaussian width parameters are used, with threshold set to be some appropriate value (specific to each ϵ) to not restrict force projection. Local cell sizes remain at 0.05 m, which is the standard used by most ALM runs in this section. There are two

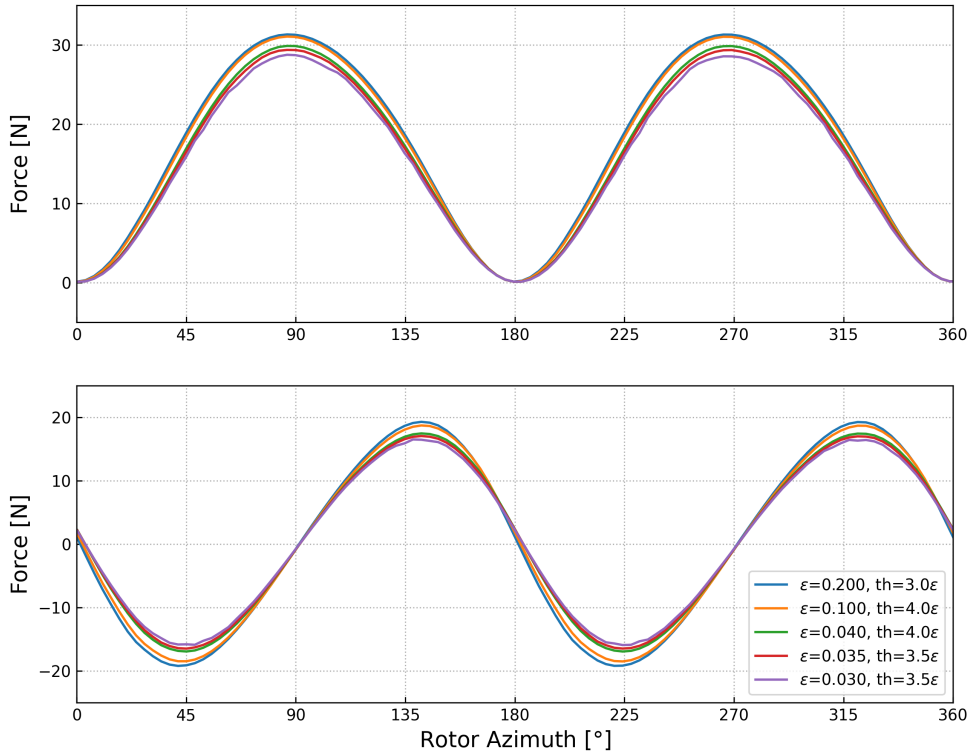


Figure 4.6: Turbine F_x (top) and F_y (bottom) as a function of rotor azimuth, where various Gaussian width ϵ values are compared. Results are taken at revolution 3.

major observations to make: the forces only vary near the peaks to a fairly small extent and there is a distinctive trend of force magnitude peak decreasing with decreasing ϵ. Thus, it is once again clear that ϵ may play a significant role, in conjunction with the mesh fineness, in determining the physics of the system. It is not immediately obvious here whether $\epsilon = 0.2$ or 0.03 better emulates the actual physical presence of the rotating blades immersed in the flow. Assuming that an expensive and comprehensive range of cell sizes can be tested for each ϵ, the results may converge when cell sizes reach an abundantly small dimension, but this *converged* set of forces and thus physics would most likely still vary for different ϵ. Hence, tuning / calibration with respect to experimental results, like Zhao et al. appears to have done, seems to be the only feasible method. Additionally, there could be an insensitivity band around $\epsilon = 0.1$ to 0.2 where the current results begin to align, which agrees with Zhao et al.'s choice of $\epsilon = 0.2$. These results taken at revolution 3 allow for significant development and periodic convergence of forces, thus mitigating unequal initial start-up behaviors due to ϵ. Based on the relatively minute differences in force demonstrated here, it is not significantly problematic to proceed with $\epsilon = 0.1$ or 0.2, as

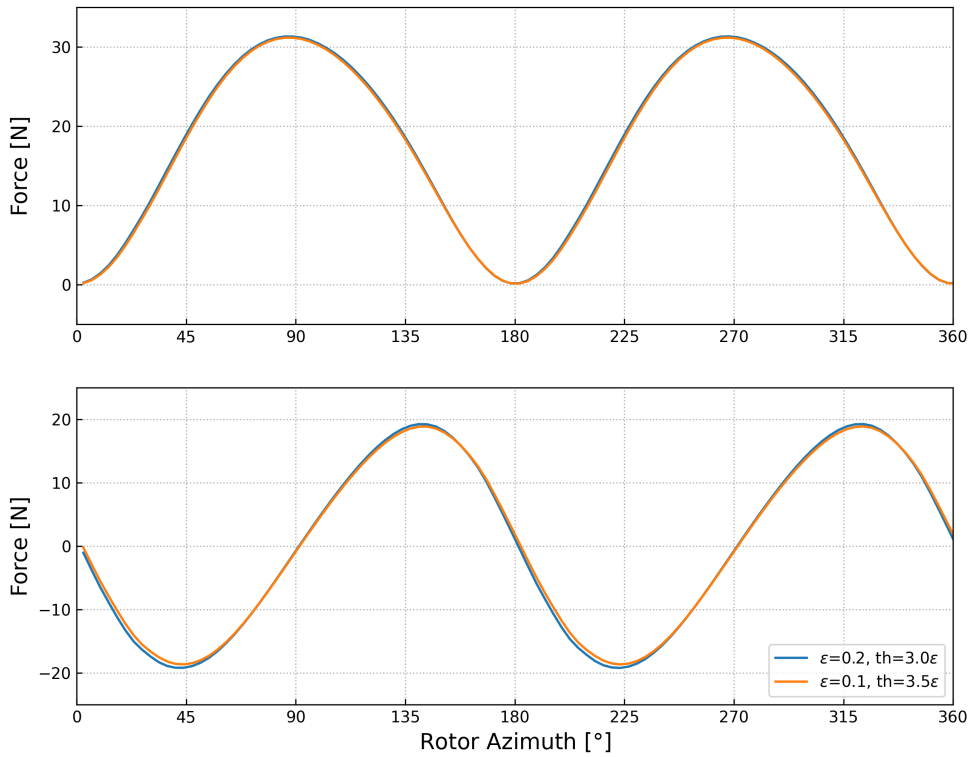


Figure 4.7: Turbine F_x (top) and F_y (bottom) as a function of rotor azimuth for $\epsilon = 0.1$ and 0.2, at a periodically converged revolution.

any possible errors will be very minor.

Finally, Figure 4.7 demonstrates the ALM forces produced by $\epsilon = 0.1$ and 0.2 at a periodically converged revolution of 8. This one-to-one comparison confirms that few comparative insights are lost in the previous comparisons taken at an earlier revolution: the forces are remarkably similar with only minor differences in F_y. Therefore, this is a satisfying result that points to some degree of validity in using these ϵ values for the current case. Overall, while this thesis had not undertaken a comprehensive verification of ϵ-mesh combinations (which would be a highly expensive endeavor), a meaningful portion of the search space has been explored. The results broadly indicate that when centered around the suggested values of Zhao et al. for the current case, which is $\epsilon = 0.2$ and $\Delta x = 0.05$ m, the ALM is insensitive to changes in these parameters.

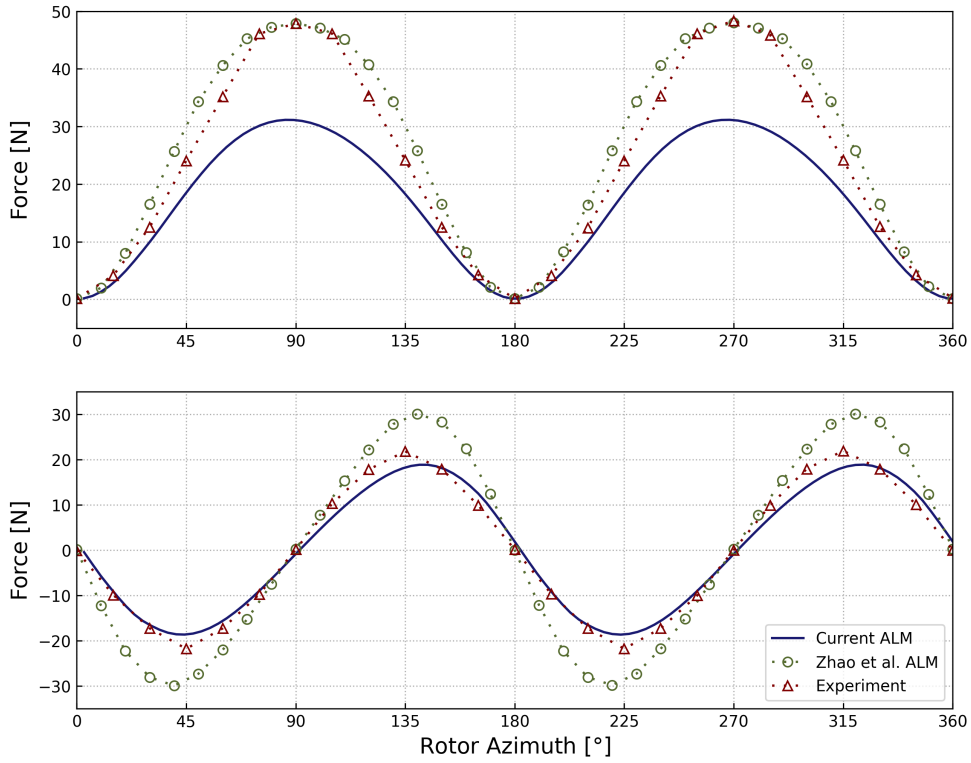


Figure 4.8: Turbine F_x (top) and F_y (bottom) as a function of rotor azimuth for the current ALM implementation, Zhao et al.'s implementation, and experimental measurements for TSR $= 3.7$. Results are taken at a periodically converged revolution for the ALM runs.

4.3 Comparison of Current ALM Implementation with Experiment

Next, it is crucial to compare the current implementation's results with that of Zhao et al. [3] and LeBlanc & Ferreira's experimental measurements [2] in order to perform validation. Since the current implementation and case have been developed for this validation target, the results are directly comparable. All set-up and operating parameters including freestream velocity, inlet TKE / dissipation rate, turbine geometry, rotational velocity, and data table for C_L / C_D look-up have been rigorously verified to be identical to those of the validation targets. Therefore, it is expected that the current results should align very closely with those of Zhao et al.

Figure 4.8 shows this validation comparison using only the $\epsilon = 0.1$ case from the current

Fluent UDF implementation (since $\epsilon = 0.2$ showed nearly identical results). For F_x, the current implementation captures the trend well but the magnitudes throughout the azimuthal positions are scaled down by about 35% compared to both Zhao et al. and the experimental data. For F_y, the current results match quite closely in trend and magnitude with the experimental measurements, but again features scaled-down magnitudes compared to Zhao et al. Because F_x is the more significant force component for generating power and the high degree of similarity with Zhao et al.'s detailed ALM logic by design, the misprediction of the current model should be addressed. The following two sections will each add a set of results to the cross-validation of ALM, and an overall discussion regarding the validity of these ALM implementations will be presented at the end of this chapter.

4.4 Comparison with `turbinesFoam` Code

In Figure 4.9, the results obtained using Bachant et al.'s `turbinesFoam` library [62, 64] are presented for additional cross-comparison. As with the current UDF implementation, this library allows extensive customization of the number of turbines, each turbine's airfoil data, and blade characteristics. The supporting OpenFOAM case is set up with equivalent parameters to Zhao et al. and the current Fluent ALM case to ensure a direct comparison. It is noted that there are discrepancies arising from the use of a different (3-D) Gaussian kernel approach, which was initially prototyped in the UDF code developed here (this was described in an earlier section). With a 3-D projection, there would be overlap in the spanwise direction, where certain cells may have force imposed by multiple actuator points. Also, because of the "centralized" calculation of force for a discrete spanwise element done at the actuator point, the Gaussian kernel is guaranteed to distribute a total force that conserves the correct blade aerodynamics. This is in contrast to Zhao et al. and the current implementation, where airfoil dynamics are evaluated at each cell individually before being multiplied by a 2-D Gaussian kernel value scaled based on proximity to an actuator element on the same z-plane. In this sense, the physics of the virtual blade is decomposed into several small cell-scale airfoil sections, with the difference mainly emerging from *where* the flow velocity is extracted from the solver to determine u_{rel}. Zhao et al.'s approach essentially probes a broader range of neighboring cells in which the blade resides to capture the granular impacts of varying local velocities on the ALM forces. Note that as previously demonstrated with the current ALM code, Zhao et al. and Bachant et al.'s approaches did not lead to any significant differences in ALM forces.

Based on the figure, it is apparent that the `turbinesFoam` ALM run agrees very well with the current ALM implementation and not the Zhao et al. or experimental results for F_x. For F_y, the current ALM implementation, Bachant's `turbinesFoam`, and the experimental results

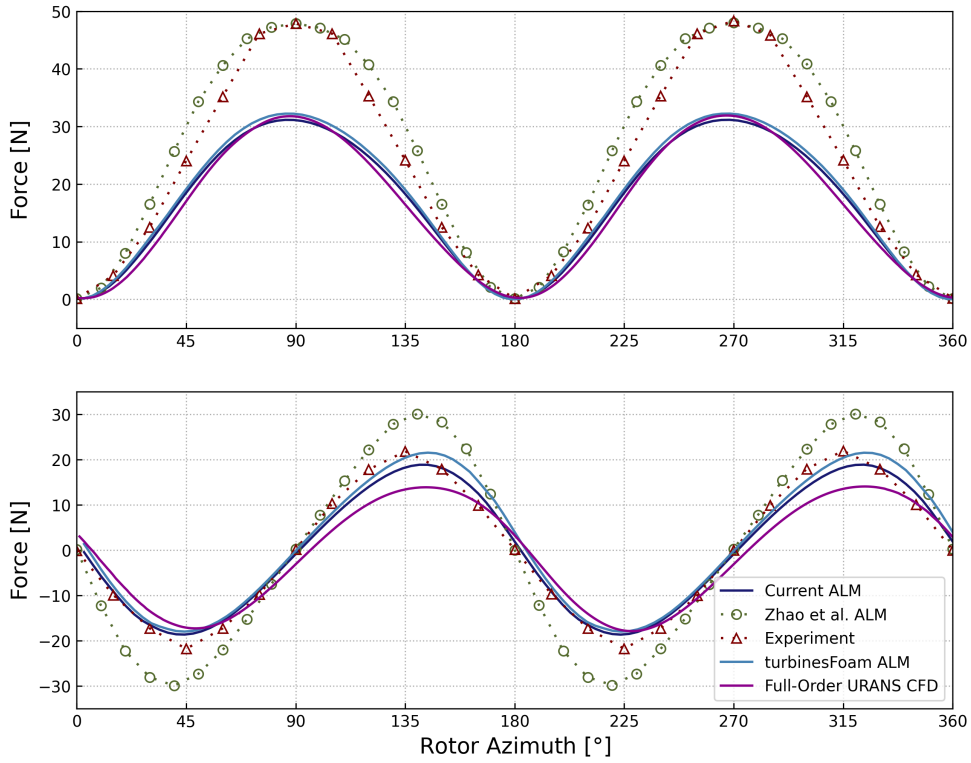


Figure 4.9: Turbine F_x (top) and F_y (bottom) as a function of rotor azimuth for the current ALM implementation, Zhao et al.'s implementation, experimental measurements, Bachant et al.'s `turbinesFoam` library, and full-order (blade-resolved) CFD for TSR $= 3.7$. Results are taken at a periodically converged revolution for the ALM runs.

are well-aligned. This comparison thus yields an interesting perspective on the current validation problem. The approach and code established in the current thesis is highly independent from `turbinesFoam` in that the the modelling approach is divergent and the programmatic logic are implemented in relation to completely different frameworks / APIs, as has been extensively emphasized above. Hence, this level of agreement between the current implementation and the `turbinesFoam` library strikingly demonstrates the correctness of both models and the consistency of the underlying logic. No fine-tuning on ALM parameters has been done to achieve this alignment beyond the use of similar Gaussian widths and the same airfoil look-up table of $Re_c = 10^6$ for NACA 0021.

4.5 Comparison with full-order URANS CFD

Finally, a set of forces is produced using full-order / blade-resolved CFD for validation. This is particularly important since it is otherwise unclear whether ALM as a method consists of any conspicuous flaws or shortcomings relative to the highly accurate and robust method of full-order URANS CFD in predicting VAWT performance. The current run is a 2-D URANS simulation using SST k-ω, same as the above ALM runs, consisting of only two blades (with the NACA 0021 profile) and no shafts or supporting structures. The simulation was done in ANSYS Fluent using a steady-state precursor run for initialization followed by a transient run for 10 revolutions. The domain consists of a rotating, refined mesh region that encapsulates the turbine, for which the motion has been modelled using the Moving Reference Frame (MRF) and Sliding Mesh (SM) built-in Fluent techniques for steady and transient simulation, respectively. The VAWT CFD case developed by Gebreel Abdalrahman has been used as reference for the current case [31]. Each blade is meshed with 600 cell divisions around the profile, 50 layers of inflation cells starting at a first layer thickness of 1.08×10^{-5} m, guaranteeing a non-dimensional wall-normal distance of less than 5 throughout the simulation.

From the results in Figure 4.9, once again the forces are highly similar to that of the current ALM implementation and the forces obtained using Bachant's `turbinesFoam` library. Some underprediction for the positive F_y peaks relative to the other two sources could be due to inadequate mesh fineness or a coarser time step of 0.001 s. Nevertheless, this result lends further credence to the mutual validity of all 3 sources currently explored: the original ALM implementation constructed in Chapter 3, Bachant et al.'s `turbinesFoam` ALM library, and a full-order URANS CFD.

4.6 Discussion of Validation

To assess whether the current models (UDF implementation and `turbinesFoam`) are valid, it is first crucial to disentangle Zhao et al.'s results [3] from the experimental measurements on the same turbine by LeBlanc & Ferreira [2]. This means that it is important to extricate one from the other and treat the datasets as unrelated. First, looking at only LeBlanc & Ferreira's experimental measurement of the 2-bladed VAWT in a wind tunnel, there are clear underpredictions by the currently employed models / cases for F_x or the thrust force (see Figure 4.9). Assuming the reported experimental configuration, parameters, and results are accurate, errors in the ALM force predictions can be attributed to various sources. First, to eliminate some possibilities: it was found that the usage of a particular solver (e.g., PISO or Coupled), choice of the discretization schemes, and size of the time step did not contribute any numerical errors to the

solution. The general correctness of the UDF code had also been comprehensively tested such that the Gaussian kernel distribution calculations, cut-off, and parallel logging are done as intended, error-free. Similarly, Bachant's `turbinesFoam` library had been validated in the initial publication [62] as well as later studies using it [65]. Therefore, the only factors within the control of this investigation that could influence the forces are the Gaussian kernel width and choice of airfoil performance data table. The forces have been shown to be generally insensitive to the choice of ϵ. The airfoil data table used is the $Re_c = 10^6$ table reported by Sheldahl & Klimas [59] according to Zhao et al.'s procedure, despite the actual operating Re_c of the blades being two orders of magnitude lower. This seemingly arbitrary choice is justified by Zhao et al. as necessary because of the inherent inaccuracy of the source data, which was synthesized and not experimentally measured [59, 62]. Furthermore, static airfoil data would not fully represent the loading on dynamically pitching airfoils. The work of Mendoza & Goude [65] also demonstrated that the report's data tables resulted in force underpredictions compared to lift / drag coefficients supplied by the XFOIL program. In Scheurich & Brown's study [29] on a VAWT, using only static airfoil data lead to significant underpredictions of the sectional tangential force coefficient near the extreme degrees of angle of attack encountered (viz., $|\alpha| \leq 15°$), whereas a dynamic stall model was able to correctly develop higher force coefficients. Abhishek's work [14] also found that symmetrical airfoils operating in curvilinear flow during rotation could become affected by virtual camber, which if uncorrected could lead to an underprediction of C_L by 0.4 to 0.65 depending on azimuthal position. Finally, in Bachant et al.'s validation, the significant underprediction of power for one of the two turbines studied shows the extent of misalignment observed in the current work is plausible.

Therefore, it is conceivable that some or all of these issues are impacting the magnitude of forces (F_x in particular) in the current ALM studies. Though unlikely, it is possible that the full-order URANS CFD performed was too crude in terms of mesh refinement and time step size. Based on experimentally determined mean thrust coefficient as a function of TSR, a linear increase can be observed between TSR = 3.1 and 3.7. However, observing the full thrust coefficient as a function of a converged revolution's rotor azimuth, the increase in the peak value with TSR is superlinear: a peak of 0.9, 1.1, 1.6, 2.25 for TSR = 3.1, 3.3, 3.5, 3.7, respectively [2]. Thus, it appears that the actual physical forces on a VAWT increase due to a steady, linear contribution from the increased TSR / rotational speed (and thus increased u_{rel}) as well as some unknown component that becomes more impactful at higher TSR (this will be discussed shortly in the next part).

Having established possible causes for discrepancies of current results with experimental measurements, it is important to validate against Zhao et al.'s results separately. The inability to replicate those results indicates certain inconsistencies in Zhao et al.'s procedure or documentation. As described during the UDF development process, Zhao et al.'s procedure has been

extensively scrutinized and applied, so it should not be possible to use the same methodology and obtain 35% lower F_x forces. This claim can be substantiated as follows. First, it is conceivable that perhaps Zhao et al. used a different Gaussian width and mesh combination than what was documented. This is very unlikely to be the cause of underprediction, since the sensitivity test above showed that any variations were insignificant, with results converging as ϵ is increased to 0.2 and a smaller ϵ led to smaller forces instead. Also, the 0.003 s time step and 6.72×10^5 cell count in the 3-D domain [3] indicates that an especially low ϵ value accompanied by proportionately small time steps and cell sizes could not have been used. Next, it can be examined whether Zhao et al. may have used a different Re_c dataset. First, Figure 8 in Zhao et al.'s paper plotted the C_L curve as a function of α, which aligns with the $Re_c = 10^6$ dataset for the NACA 0021 airfoil in Sheldahl & Klimas. All datasets near the Re_c magnitude of 10^6 exhibit similar trends for C_L vs. α, which is a roughly linear increase in $|C_L|$ with $|\alpha|$ until a plateau, which dictates some upper bound for $|C_L|$. The maximum achievable value ranges from 1.1 for $Re_c = 10^6$ to 1.3 for $Re_c = 5 \times 10^6$, which is the highest Re_c dataset available in the report. A simple analytical calculation can be done for the first revolution at a known azimuthal position where F_x of a blade will be maximum. The conditions are assumed to be: no cross-stream velocity component v, $u = 4.01$ m s^{-1}, and TSR $= 3.7$. The angle of attack calculation can be skipped and the largest achievable C_L of 1.3 will be used. At this point, the single blade F_x is found to be 19.7 N, which can be multiplied by two to find a rotor F_x of 39.4 N (although the actual max F_x will be smaller since the downwind blade is operating in the wake). Thus, it has been demonstrated that even if the highest value possible is used for C_L, 39.4 N is still significantly lower than Zhao et al.'s purported max F_x of 48 N. Finally, according to Figure 6 of Zhao et al.'s publication, the ALM-produced mean thrust coefficient C_T as a function of TSR follows a completely linear trend but exhibits a significant jump in value only at TSR $= 3.7$ (the current validation case) [3]. The 5 data points spanning a TSR of 2.7 through 3.5 follow an approximately fitted slope of 0.3375. This pattern is directly due to a linear increment in TSR, thus by extension, rotational speed and u_{rel}. Furthermore, the data point at TSR $= 3.3$ is verified to be consistent with the current ALM results. Extrapolating to TSR $= 3.7$ results in a value of about 0.8375, which is again consistent with the result of the current ALM implementation, whereas Zhao et al.'s reported mean C_T is as high as 1.21. Regardless of whether such a disproportionate jump at TSR $= 3.7$ can be justified via experimental results and the underpinning physics, it is fully demonstrated that an implementation following Zhao et al.'s documented procedures cannot possibly attain this trend-breaking F_x peak.

Overall, the verdict on the validity of the current two ALM approaches (the Fluent UDF and Bachant et al.'s `turbinesFoam`) is that they are both reasonably valid for predicting VAWT forces. First, a high degree of confidence regarding the correctness of the ALM approach has been established: despite errors arising from using static airfoil data without any dynamic cor-

rections, the two ALM approaches and the full-order URANS CFD agree very well with each other. While the discrepancy with experimental results for the F_x peak is somewhat significant at about 35%, single point values are less important than mean forces and C_p for evaluating VAWT synergy, and the practical mean difference is indeed smaller. The discrepancy with Zhao et al.'s ALM results is not considered problematic due to the above justifications. Therefore, the uncertainties and errors in the current ALM approaches do not pose a significant challenge to the key assumption underlying this study, which is that synergy is a measure relative to the isolated turbine performance. Thus, because of this essential normalization process, the errors would be significantly mitigated if not rendered negligible.

Given the availability and equal validity of both the current Fluent UDF implementation and the `turbinesFoam` library, one framework can be chosen to continue onto the multi-VAWT synergy investigation. It is decided that `turbinesFoam` and OpenFOAM be adopted for use mainly due to performance reasons. Based on a rough estimate, for a full 10-revolution run, `turbinesFoam` is 5 times as fast. This performance advantage could be attributed to inherent limitations of the UDF architecture and/or the difference in Gaussian kernel formulation. First, all cells in the domain are mandatorily looped through in `DEFINE_SOURCE`, regardless of whether any meaningful logic will be performed. This fundamental and unavoidable rigidity of the system could be adding significant computational overhead that the OpenFOAM tool is not affected by. OpenFOAM instead allows for solver access via a broad selection of APIs and programmatic objects, giving any library creator appreciable room for optimizing the computational logic. Furthermore, due to the UDF approach using a 2-D Gaussian kernel with local cell-wise force computations, airfoil coefficient look-up operations (which are likely the most expensive among all operations) must be repeated twice for all cells. In `turbinesFoam`, calculations need only be done on a handful of actuator points spanning each blade, and force projection on surrounding cells are handled by highly performant cell access logic. Therefore, the open-source and highly customizable `turbinesFoam` is deemed the most efficient tool to continue with.

Chapter 5

Study of 2-Turbine Configurations

In this chapter, a baseline of an isolated VAWT (with the same geometry as above) will be established by simulating and determining the power coefficient C_p according to ALM. Then, a set of configurations / relative positions of 2 turbines will be designed to investigate how individual and cluster turbine performance will vary. Both the co-rotating cases and the counter-rotating cases will be tested to determine whether the relative rotational directions affect the presence and magnitude of synergy.

5.1 Isolated Turbine Baseline

Based on the testing of Chapter 4, the OpenFOAM and `turbinesFoam` setup for the 2-bladed VAWT operating in a freestream velocity of 4.01 m s^{-1} and TSR of 3.7 has been found to be valid. The current mesh consisting of cell sizes of approximately 0.05 m near the VAWT (based on Zhao et al. [3]) gradually coarsening to larger cells away from the critical regions of the domain is also deemed adequate for a reasonably accurate ALM simulation. According to Mendoza et al. [63], who performed a mesh sensitivity test using `turbinesFoam` code, mesh refinement did not significantly affect the predicted angle of attack and normal force. However, based on testing, it is clear that the previously used domain size of 13 m $\times$ 2.85 m $\times$ 2.85 m ($L_x \times L_y \times L_z$) is inappropriate for the simulation of a domain-independent single or multi-turbine case. This is because the lateral / cross-stream domain of 2.85 m is insufficiently large and causes blockage of wake development, ultimately leading to increased forces and C_p. Since this prohibits a direct and accurate comparison of C_p and thus the determination of synergy, it is important to find a sufficiently large domain size. Based on Hansen et al.'s study [44], a domain size of $90D$

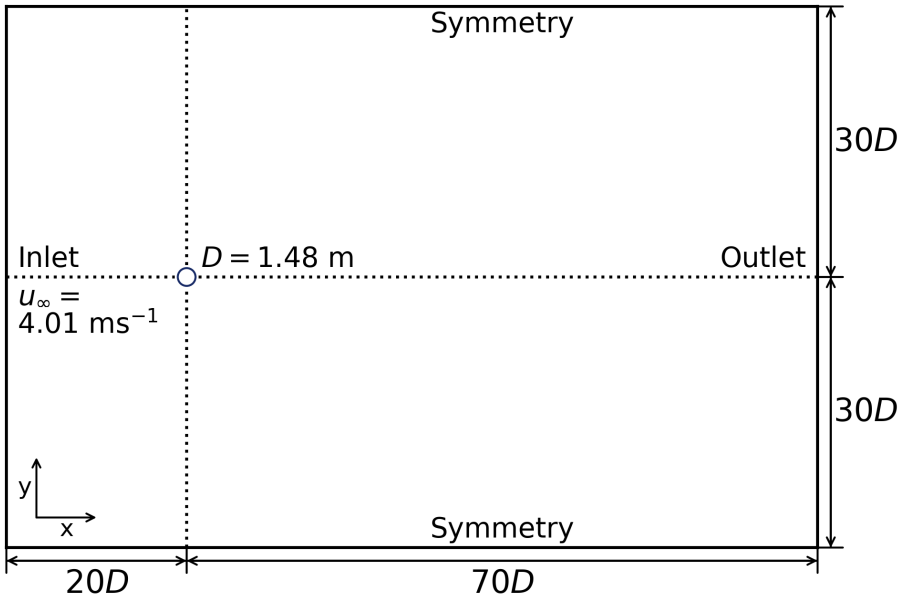


Figure 5.1: Diagram of the domain setup for isolated and 2-/3-turbine ALM cases. The lateral boundaries are set to be symmetry, as shown. The top and bottom boundaries (parallel to the x-y plane) are also set to be symmetry.

(length) $\times$ $60D$ (width) is sufficient for ensuring the domain does not restrict the development of 2-/3-turbine configurations. Thus, to ensure accuracy, this is adopted in the current ALM runs as shown in Figure 5.1. Further testing shows that a near-VAWT mesh refinement region of roughly $4D$ (length) $\times$ $2.7D$ (width) is sufficient, since expanding this region of 0.05 m cells does not lead to changes in the C_p. A sample illustration for a 2-turbine ALM mesh is presented in Figure 5.2.

With these setup conditions, the isolated turbine C_p is calculated to be **0.5458**. This important value is hereby established as the baseline for measuring the presence and magnitude of synergy in multi-turbine clusters: if an individual turbine's C_p in those arrangements exceeds 0.5458, then synergy due to array influence is present.

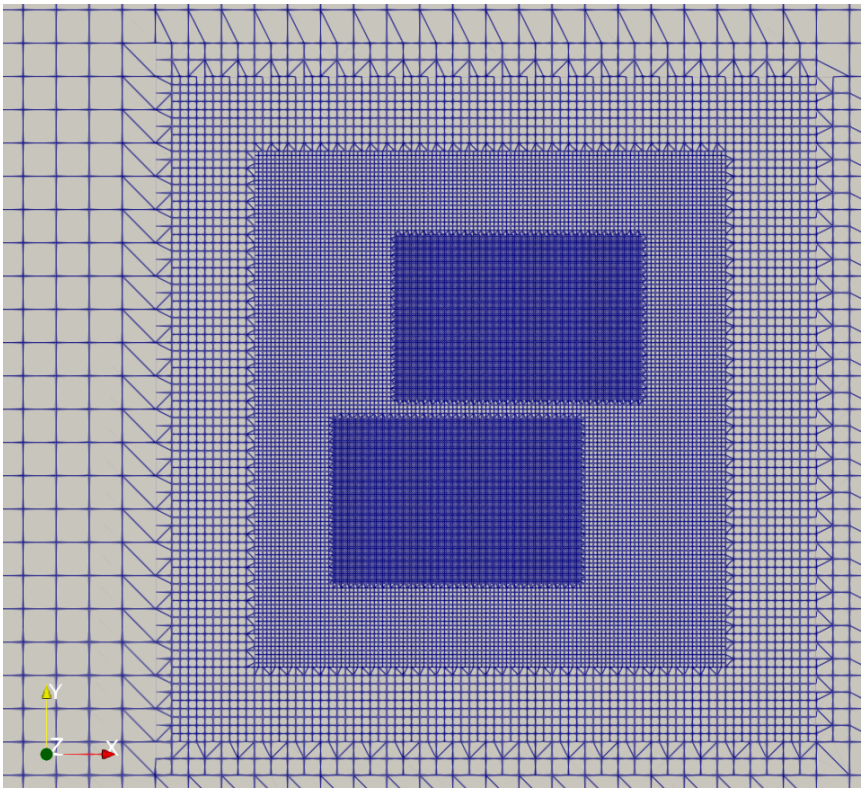


Figure 5.2: Diagram of the mesh close to the virtual VAWTs in a 2-turbine ALM case. The smallest cells are approximately 0.05 m in size.

5.2 2-Turbine Configurations

It is important to design a suite of configurations to systematically test for patterns in synergy. Previous literature in this area, such as [38, 41, 40, 42], have used mostly *ad hoc* and limited arrangements that did not rigorously explore the configuration space. Hansen et al. [44] appears to be the first work in this area which sought to develop a systematic approach for evaluating the extent of synergistic effects across multiple 2-turbine configurations. This was done by labelling the two turbines as rotor 1 (R1) and rotor 2 (R2), where R1 is defined as the leading turbine in all cases except when R1 and R2 are adjacently arranged with respect to incoming wind. Thus, the position of R2 is parameterized to vary relative to R1's position based on two variables: a turbine spacing $dist$ and an array angle β. $dist$ is used to control the straight-line spacing between the centers of R1 and R2, while β is an angle increasing CCW from a reference axis aligned with the wind direction. For example, for a β of 0°, R2 is directly downwind behind R1, and for a

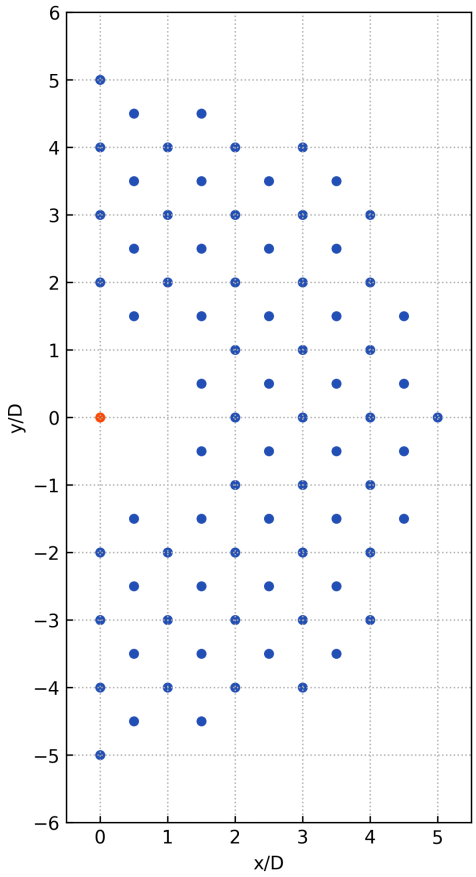


Figure 5.3: Turbine locations for 2-turbine arrangements, where T1 is fixed at $(0, 0)$ (orange point) and T2 is situated relative to T1 in a staggered grid (blue points).

β of 90° / -90°, R2 is adjacent to R1 on the top / bottom sides, respectively. In this manner, the meaningful or orthogonal test cases can be easily selected for testing — e.g., it is only necessary to test $|\beta| \leq 90°$ due to symmetry. This formulation naturally leads to the exploration of 2-turbine configurations constrained to "rings" centered on R1 and at some fixed $dist$ of $1D$, $2D$, etc. While this is certainly useful for studying synergy, it presents only a limited picture.

This thesis aims at providing a higher-resolution picture of the synergy landscape for a 2-turbine cluster. The VAWTs will also be labelled turbine 1 (T1) and turbine 2 (T2), and the relative positions will follow Hansen et al.'s convention in that T1 is leading in non-adjacent arrangements. However, instead of using a polar coordinate system to parameterize T2's location, a simple Cartesian grid will be used in which T1 is set as the origin. Using this method, a synergy

map can be formed on some range of relative positions spaced at regular intervals. A set of positions to be simulated for the 2-turbine co-rotating case is illustrated in Figure 5.3, where T2 is placed in a regular, staggered grid with a radial distance from T1 constrained between $1.5D$ and $5D$. Turbines are separated horizontally and vertically with a $1D$ spacing. The wind direction is fixed in the positive x direction. The reason for the $1.5D$ lower limit is to avoid simulating cases where rotor interference would not be captured by ALM (viz., at a distance of around $1D$, the blades of T1 and T2 would collide in reality, but these collisions or cyclic proximity effects cannot be captured using ALM due to fundamental limitations of the technique). A $5D$ upper limit is set since this was tested to be the range at which synergy effects are significantly weakened and approach zero, indicating that this a sensible range to explore all interesting interactions arising from different arrangements.

Note that it is only important to preserve the relative turbine positions in simulation. In practice, each set of positions as displayed in Figure 5.3 is converted into actual domain coordinates such that the cluster is centered in the domain, thus allowing for good space utilization which minimizes any blockage due to proximity to the lateral boundaries. For instance, a configuration where T1 is at $(x/D, y/D) = (0, 0)$ and T2 is at $(2, 2)$ would be converted into the actual positions of T1: $(0, -1)$ and T2: $(2, 1)$. To measure synergy, a *power ratio* (similar to Hansen et al.'s performance indicator) is defined as:

$$\text{power ratio} = \frac{C_p}{C_{p,iso}} \tag{5.1}$$

where $C_{p,iso}$ is the power coefficient of the isolated turbine (found to be 0.5458 for current purposes) and C_p is the power coefficient of any turbine within a cluster or the cluster mean C_p. Here, C_p is taken as a 2-revolution mean at the periodically converged revolutions. If this ratio is greater than 1, it means that synergy is present and is benefiting the performance of a particular turbine or the entire cluster (depending on context of comparison).

5.3 Co-Rotating 2-Turbine Results

Based on the suite of simulations conducted for configurations like those found in Figure 5.3, it is determined that **synergy is present** in the results of many configurations when simulating VAWT pair interactions using ALM because both turbine and cluster mean power ratios exceeded 1. This means that the first major objective of this thesis has been fulfilled: it has been ascertained that ALM is at least capable of predicting the presence of synergy effects between multiple VAWTs. The x-velocity contours for these 2-turbine cases (as shown in Figure 5.4) reveal similar patterns

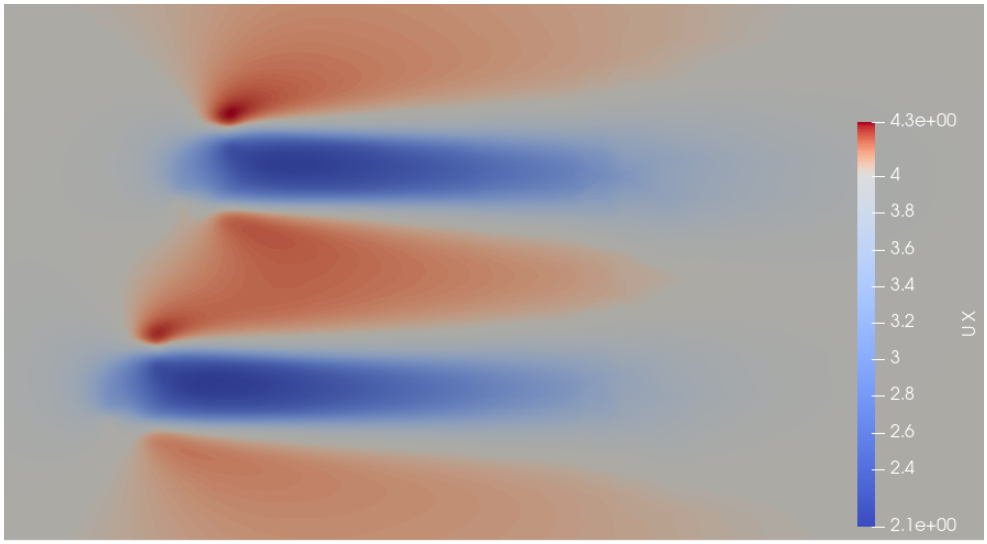
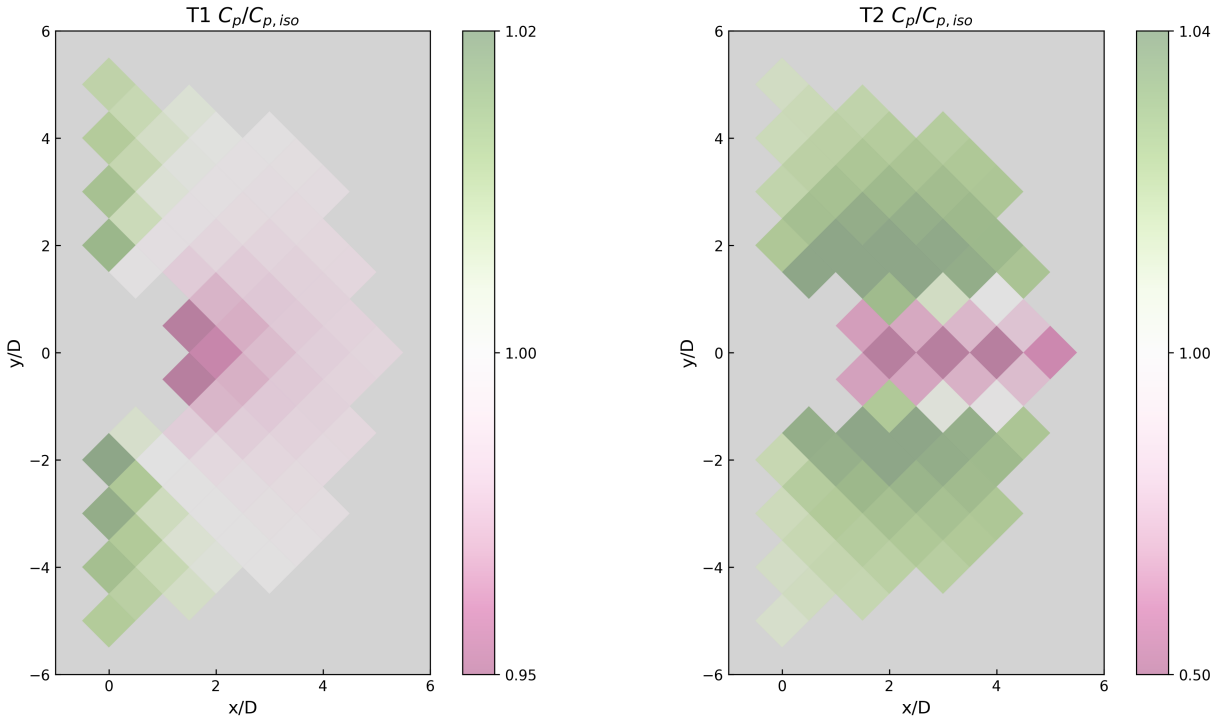


Figure 5.4: Streamwise velocity (U_x) contour of a 2-turbine ALM case at 3.15 seconds of flow time, exhibiting high velocity regions featuring $U_x > u_\infty$ around the turbines and wakes.

as those observed in blade-resolved URANS CFD [44] and in experimental measurements [41, 7], namely that high velocity regions featuring flows faster than the inlet velocity are developed adjacent to / outside of the turbine vicinity and wake regions. Based on the literature review, this appears to be one of the necessary characteristics for the presence of synergistic inter-turbine interactions. In the current study, the ALM has demonstrated its ability to capture these mean flow velocity enhancements despite the lack of blade-level and turbulence details, thus producing synergy. Next, it is important to visualize and understand the patterns surrounding synergy for the 78 co-rotating pair cases in order to investigate how cluster arrangement affects VAWT synergy.

Each of the 78 co-rotating VAWT cases (in which both VAWTs are rotating in the CCW direction) outputs the C_p of turbine 1 (T1) and turbine 2 (T2). These values are then divided by the isolated turbine's C_p found above, which is $C_{p,iso} = 0.5458$ to obtain the T1 and T2 power ratios. For either turbine, if the power ratio exceeds 1, then that turbine has experienced synergy in that it generated more power than it would have alone, operating under the same conditions. Additionally, it is also of interest to determine whether the entire cluster (composed of 2 turbines here) experiences a net positive effect from inter-turbine interactions, since these are not always positive / synergistic for all turbines involved (to be elaborated in the following discussion). Thus, a straightforward metric to evaluate this would be the use of a *cluster mean* power ratio where the turbine C_p in the equation can be replaced with an arithmetic mean C_p of all the turbines in the cluster. With this formulation, it is easy to verify that if the cluster mean power ratio exceeds 1,



(a) (b)

Figure 5.5: Heatmap of power ratios ($C_p/C_{p,iso}$) of (a) T1 and (b) T2 in a co-rotating pair of VAWTs, where the value of each cell corresponds to the configuration which has T2 located at that cell relative to T1 at the origin.

then the cluster will generate more power than the same number of isolated VAWTs. Therefore, for the current chapter, the ALM results to organize and present are the T1, T2, and cluster mean power ratios.

Figure 5.5 presents power ratio heatmaps that will be used to analyze patterns in VAWT synergy in a compact and relatively intuitive fashion. The grid of various T2 positions relative to a T1 situated at the origin (according to Figure 5.3) are extended to diamond-shaped cells with the centroid located at the same x-y coordinates. For a heatmap, each cell then represents the result for one of the 78 VAWT pair runs, which approximates the true synergy map down to a resolution of about $0.5D$. However, as mentioned above, there are three power ratios (T1, T2, and cluster mean) to present for each configuration, so three separate heatmaps are made with the corresponding power ratios encoded in a color map. At a $C_p/C_{p,iso}$ ratio of around 1, the color is a neutral white (which is distinct from a background color of gray for non-data cells); a ratio

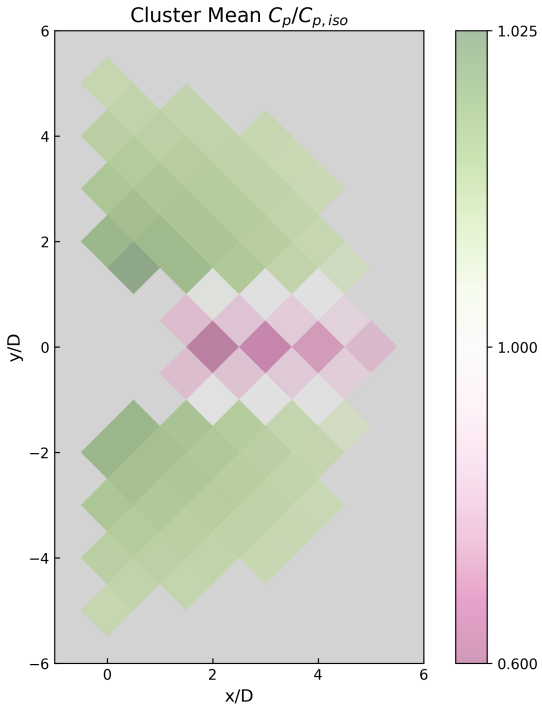


Figure 5.6: Heatmap of cluster mean power ratios ($C_p/C_{p,iso}$) for a co-rotating pair of VAWTs.

greater than 1 is colored green while a ratio less than 1 is colored pink/purple. Therefore, using each heatmap it is possible to visualize the presence and extent of synergy in all the simulated configurations.

For the co-rotating cases, T1 (upwind turbine in non-adjacent arrangements) experiences limited or no synergy in many cases when compared to T2. T1 performed the best, with an enhancement of about 2% over an isolated turbine, in cases where T1 and T2 are arranged adjacently with respect to the incoming flow (in the first "column" of cells in the T1 heatmap). There is some asymmetry where T1's synergy improved slightly if T2 is situated under T1, which may be attributed to the inherent asymmetry of VAWTs. For the middle 3 rows of cells (T2's position is near $y/D = 0$) corresponding to a location in the wake of T1, the power performance of T1 and T2 both suffer significantly. While T2 is directly disadvantaged by operating in a velocity deficit (thus lowering u_{rel} and power-driving ALM forces), this indicates the blockage negatively reflects upstream onto T1 as well. However, it can be seen that for many configurations that do not position T2 in the wake (i.e., T2 experiences unambiguous and substantial synergy), T1's performance is still deteriorated or only benefited a miniscule amount by synergy. In fact, based on T1's heatmap, the nature of T1's behavior can be divided into roughly three sectors (based on

presence of synergy) through which the power ratios transition gradually. When T2 is situated in a roughly 90° sector centered on $y/D = 0$, there is no synergy but rather power decline relative to the isolated turbine. When T2 is situated in the 45° sectors on either ends, synergy is present and intensifies as T2 approaches the adjacent arrangement. This pattern clearly reveals that T1 becomes more negatively impacted by T2 downstream as T2 approaches T1's wake.

The situation with T2 is much clearer. Based on Figure 5.5b, T2 only performs worse than an isolated turbine in the narrow streamwise band of 3 rows about $y/D = 0$. In this region, T2 could perform about 50% worse than the isolated case, which is a massive performance detriment. However, outside of these unfavorable configurations, T2 essentially universally experiences synergy, possibly due to exploiting the mean flow speed-up outside of T1's wake. The synergy magnitude is generally higher than T1, ranging up to about 4-5% enhancement relative to the isolated case. In these green areas, it can be seen that the maximum synergy is not observed in the adjacent pairs (which was true for T1), but rather when T2 is just outside of the wake across several columns in the downstream direction. An interpretation of this result is that the magnitude of T2's synergy may be correlated with the streamwise velocity increase generated by T1, which is strongest near these locations according to Figure 5.4.

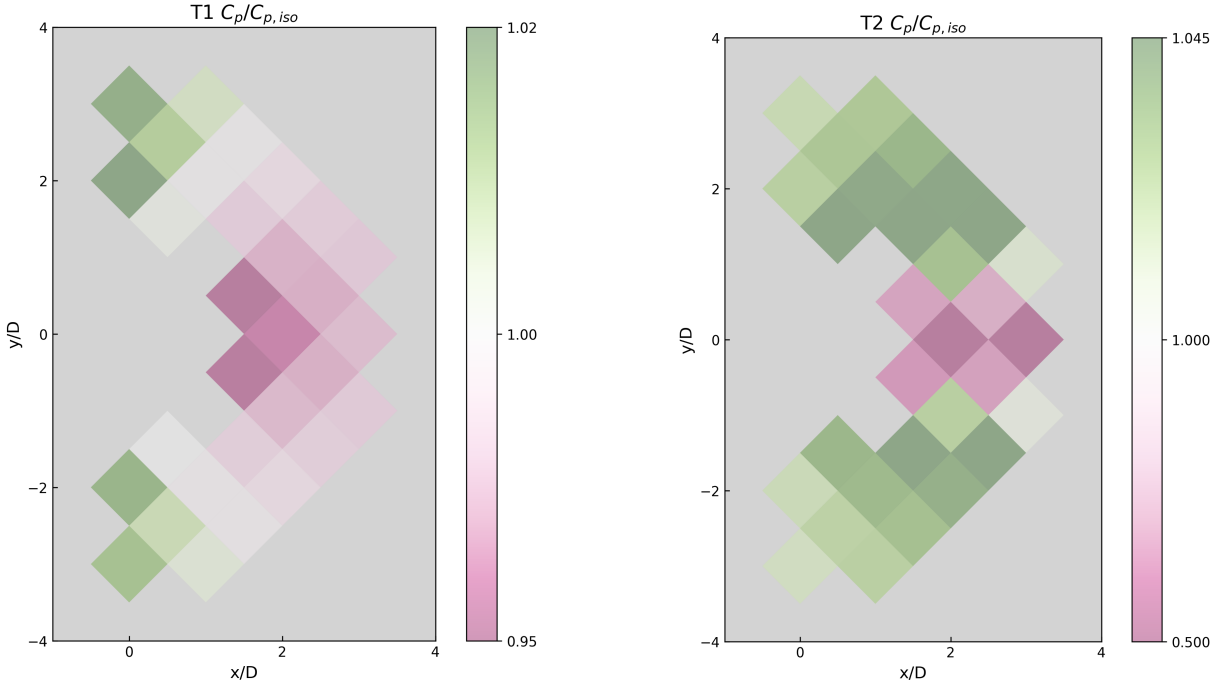
Finally, the cluster mean power ratios are also visualized for the co-rotating cases in Figure 5.6. Due to the competition between T1 and T2's synergy maps, where the maximum synergy for each occur at different configurations, it is particularly interesting to see what the optimal configuration is considering these trade-offs. First, the cluster mean heatmap's categorization of whether synergy is present or not aligns closely with T2. This is due to the omnipresence of T2 synergy outside of T1's wake as well as the generally higher synergy magnitudes, which leads to T2 almost solely determining whether the cluster also experiences synergy. Nevertheless, the competition effect determines the optimal configuration to be a compromise between the adjacent arrangements (favored by T1) and the nearly directly downstream arrangements (favored by T2). This configuration is for a trailing T2 oriented at 71.6° and spaced at $1.58D$ with respect to T1, producing a cluster-level synergy of 1.0247, or a 2.47% increase in power compared to 2 individually operating turbines. On average, a power increase of about 1.5% can be expected as long as T2 is positioned outside of T1's wake.

While it is difficult to draw direct comparisons with Hansen et al.'s results [44] due to a different configuration parameterization, general comparisons show broad agreement with only minor differences. First, the variation of T1's power ratio as a rough function of T2's relative orientation concurs between current results and Hansen et al., which also observed a maximum T1 performance / synergy in the adjacent configurations. In addition, the asymmetric bias of synergy towards the adjacent arrangements with T2 below T1 (co-rotating) is confirmed in Hansen et al.'s plots. T1's performance would decrease as T2 moved more directly downstream of it. Interestingly, a brief rebound in the power ratio of T1 when T2 is exactly downstream of T1

(with an angle of 0°) is found in both the current results and that of Hansen et al.'s full-order simulation. This can be found in the heatmap in Figure 5.5a, in which the center row for $y/D = 0$ performed slightly better than the immediately upper and lower rows of $y/D = 0.5$ and -0.5. Hansen et al. also demonstrates that it is reasonable and expected for T1's power ratio to drop below 1 as T2 moves more directly downstream. Finally, the current findings are consistent with patterns presented by Hansen et al. in that T2 experiences larger synergy magnitudes which peak for T2 positions just outside of T1's wake. Therefore, as evidenced by the highly coherent comparisons, more confidence can be allotted to the validity of the current ALM approach and the patterns deduced thereof. These aforementioned patterns can be even be aggregated into helpful intuitions that are generally reliable in the presently explored configurations. For example, there exists a trade-off relationship whereby a downstream turbine's synergy can be improved by moving it more directly behind the leading turbine, but the leading turbine's performance will be increasingly diminished.

5.4 Counter-Rotating 2-Turbine Results

Having explored a wide range of configurations in which the two VAWTs are co-rotating at CCW, it is important to also explore a set of cases where the turbines are counter-rotating. Thus, an additional set of 30 configurations are simulated using ALM where by convention T1 rotates CCW while T2 rotates CW. These configurations lie on the same positional grid as shown in Figure 5.3, except for a reduced spacing limit of $3.5D$ instead of $5D$. The reason for this reduction is to further reduce the configuration space based on the interesting range uncovered from the co-rotating cases. Figure 5.7 shows the T1 and T2 heatmaps and Figure 5.8 shows the cluster mean power ratio heatmap (all generated in a similar fashion as the co-rotating ones) for these 30 configurations. In terms of the macro trends, there are very little differences between these counter-rotating pair results and the previously analyzed co-rotating results. T2 experiences synergy consistently as long as it is positioned outside of T1's wake deficit region, with synergy tending to be maximized at certain $|y/D|$ spacings (viz., $|y/D| = 1.5$–2) for a broad range of downstream x/D spacings. The trade-off relationship intuited above holds in that these benefits to T2 power performance come at the cost of T1 performance, which could lead to a worse performance than isolated (albeit the trade-off is not proportional, so on the cluster scale it is not zero-sum). The cluster mean power ratio heatmap demonstrates that T2's performance dominates that of the overall cluster, with the optimal synergy of 1.0266 occurring at the same configuration as the co-rotating set of cases — T2 oriented at 71.6° and spaced at $1.58D$ relative to T1. However, the power synergy is about 0.19% higher in the counter-rotating case.



(a) (b)

Figure 5.7: Heatmap of power ratios ($C_p/C_{p,iso}$) of (a) T1 and (b) T2 in a counter-rotating pair of VAWTs, where the value of each cell corresponds to the configuration which has T2 located at that cell relative to T1 at the origin.

5.5 Comparison of Co- and Counter-Rotating Turbine Pairs

It is of interest to perform additional in-depth comparisons between the performance and synergy of the co- and counter-rotating VAWT pairs even though they are generally consistent. First, Figure 5.9 shows the cluster mean $C_p/C_{p,iso}$ of the co-rotating and counter-rotating cases as a function of the y/D (cross-stream) location of T2, with different lines representing x/D (downstream) locations. It is noted that for the comparable ranges (since the counter-rotating plot features a spatially reduced set of arrangements), the trends are virtually identical. In the vast majority of lines, the cluster as a whole experiences some optimal amount of synergy around $|y/D| = 1.5$ to 2 regardless of relative rotational directions, which is consistent with the observation based on the heatmaps above. A small exception to the consistency arises at the adjacent case where the co-rotating cluster exhibits similar synergy at $y/D = \pm 2$, but the counter-rotating cluster instead exhibits a noticeably higher synergy for $y/D = 2$ than -2. This difference

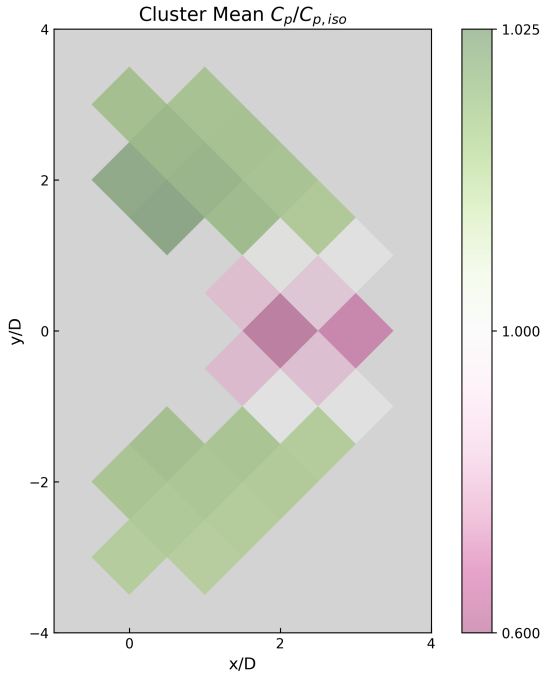


Figure 5.8: Heatmap of cluster mean power ratios ($C_p/C_{p,iso}$) for a counter-rotating pair of VAWTs.

can likely be attributed to the counter-rotating pair creating more favorable interactions for a T2 placement at $y/D = 2$, where both turbines are rotating into / against the flow in the inner channel. However, such biases seem to vanish rapidly as soon as T2 moves slightly downstream, and differences become unnoticeable after $x/D = 1$.

Figure 5.10 and Figure 5.11 show the heatmaps for a computed difference in $C_p/C_{p,iso}$ between the 30 overlapping and thus directly comparable co- and counter-rotating cases, illustrated separately for T1, T2, and cluster mean power ratios. The difference is defined as the power ratio of the counter-rotating case minus the co-rotating case, so a positive difference indicates that setting the turbines to counter-rotate for the same configuration is more beneficial, and vice versa. Using these figures, it is possible to see at which configurations and to what extent the rotational directions affect turbine and cluster synergy.

For T1, relatively significant differences can be found in the adjacent or nearly adjacent configurations. The sign of the differences are also roughly split into several zones. For x/D between 0 and 1.5, the upper region favors counter-rotation while the lower region favors co-rotation. In other words, it is beneficial for T1 to have T2 placed above it if they are counter-

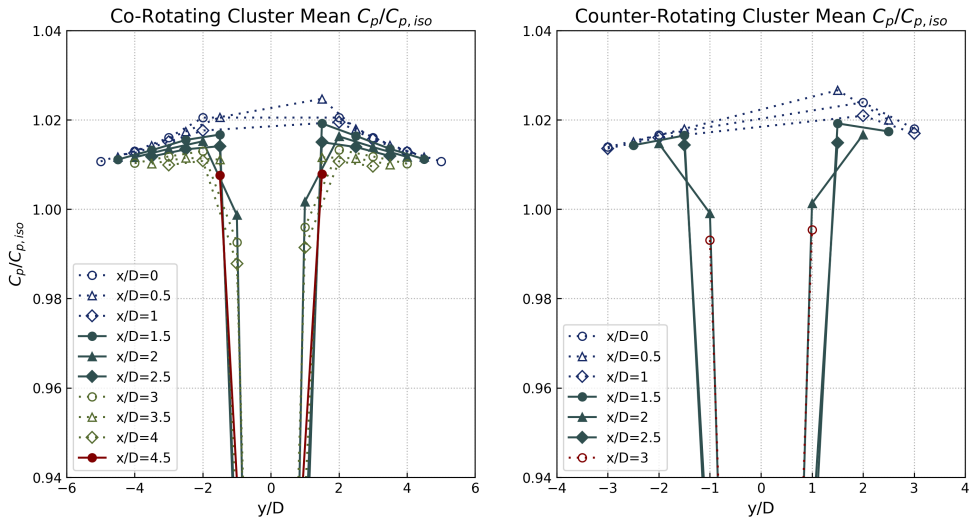
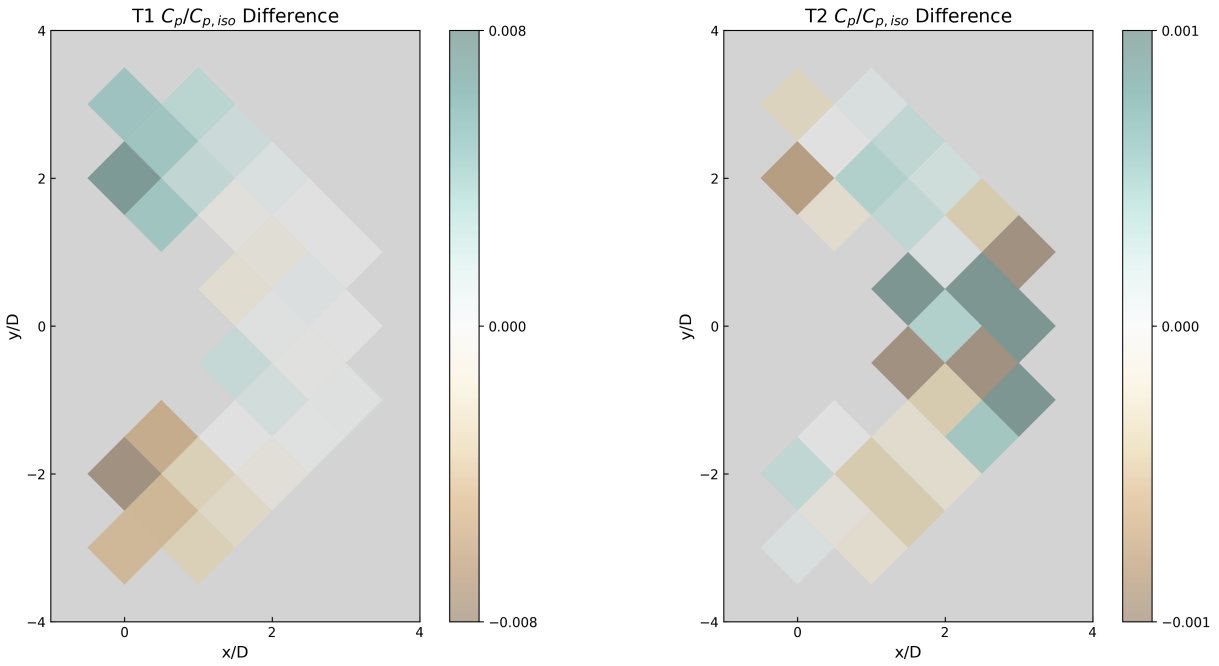


Figure 5.9: Plot of power ratio ($C_p/C_{p,iso}$) as a function of normalized y/D location of T2 relative to T1 for different T2 x/D locations.

rotating and below it if they are co-rotating. However, apart from one cell in each zone, the difference is essentially capped at a magnitude of 0.008. This indicates that any comparative advantage is fairly insignificant. It is interesting to observe that the subset of cells or T2 positions with consistent signs tends to flip about the x-axis several times throughout the remainder of the configurations that were simulated. For instance, a group of about 3 cells on either side near $x/D = 1.5$ exhibit a favored relative direction of rotation that has been interchanged from the intial group (near-adjacent configurations). Thus, for these groups, it is instead beneficial for T1 to have T2 placed below for counter-rotating cases and vice versa. The pattern seems to reverse once more at the outer rim of T2 positions, indicating that trends are possibly sensitive to the relative turbine angle as well as the inter-turbine spacing. It is important to note that these small differences and complex patterns may not be statistically significant and could instead have been caused by small numerical errors in the simulation.

For T2, the oscillation of this pattern can also be observed, though cells belonging to each grouping (same difference sign) have shifted. For both T1 and T2, the groupings seem to be separated predominantly across the x-axis, which makes sense since this is the dividing line across which the relative effects of T1 / T2's rotation reverses. The overall magnitude of differences are smaller for T2, which has a color-map bounded at 0.001. Furthermore, all configurations which exhibit differences above this limit are such that T2 operates in the wake of T1, which are undesirable arrangements for the cluster since synergy would not be present. For the overall



(a) (b)

Figure 5.10: Heatmap of power ratio ($C_p/C_{p,iso}$) difference (counter-rotating case minus co-rotating case) of (a) T1 and (b) T2. Note that a positive value indicates the counter-rotating case results in better performance.

cluster mean differences, the patterns are heavily dominated by T1, which is in contrast to the observation for only the power ratios. The more significant differences of 0.008 or higher are only found in the wake-limited configurations, while a co-rotating or counter-rotating pair makes very little difference otherwise. This means that although technically the rotational direction of the VAWTs in a pair configuration can be optimized for a known wind direction, the advantage would be so marginal that it is not worthwhile to consider this nuance for the simulated turbine geometry. Therefore, counter-rotation is not absolutely necessary for the generation of synergy and also does not provide a universal enhancement to synergy, which is an insight that agrees with prior literature on VAWT synergy.

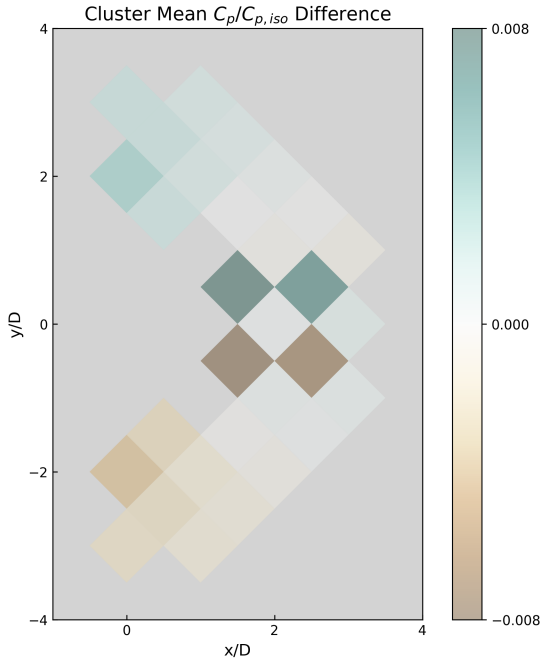


Figure 5.11: Heatmap of cluster mean power ratio ($C_p/C_{p,iso}$) difference (counter-rotating case minus co-rotating case) for different pair configurations.

Chapter 6

Study of 3-Turbine Configurations

6.1 3-Turbine Configurations

In this chapter, we explore the effect of different 3-turbine configurations on the synergy of each involved turbine as well as the whole cluster. This is an important exercise that lays the foundation for future work aiming to more rigorously tackle the micro-siting problem of VAWTs and perform layout optimization for larger turbine numbers. In contrast to the 2-turbine arrangement parameterization, it is significantly more difficult to systematically arrange 3-turbine clusters due to an increased degree of freedom. To solve this problem, the intuitive guidelines discovered above exhibit great utility in at least eliminating cases that are not promising for developing cluster-wide synergy. For example, it is clearly undesirable to have one or more turbines operating in another's wake, due to the assured performance degradation. Thus, the 3 turbines should be arranged in patterns that allow for some cross-stream separation. This line of thinking led to the creation of 3 cluster shapes, which are named as the V, Reverse V, and Line shapes for ease of reference. These are illustrated in Figure 6.1.

Based on the 2-turbine investigation above, all 3-turbine cases will be co-rotating in that every turbine will rotate in the CCW direction. Also for simplicity and for the sake of establishing a starting point before delving into more complex arrangements, each shape is parameterized in a manner that retains symmetry. For example, according to Figure 6.1, T1 and T3 of the V / Reverse V shapes are separated an equal distance from T2 in both x and y (via the parameters x_{sep} and y_{sep}). The two spacing parameters are manipulated within all cluster shapes to control the specific configuration to be tested. Note that while V and reverse V are similar, the Line configuration involves a slightly more complex parameterization where the y_{sep} undergoes a similar range of variations as the former two shapes but the x separation of the entire cluster is

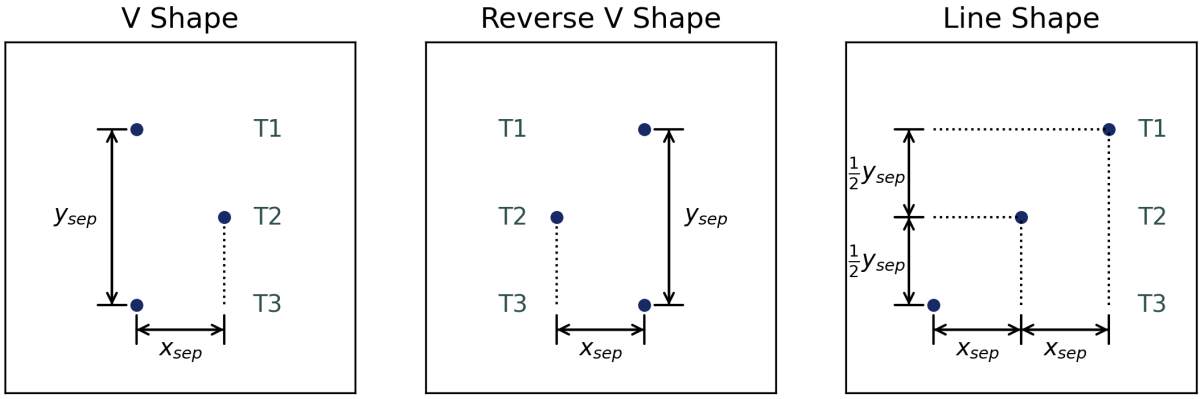


Figure 6.1: Diagrams of the V, Reverse V, and Line shapes for 3-turbine cluster arrangements illustrating conventions for defining x_{sep}, y_{sep}, and turbine labels in each case.

twice as large and starts at 0. For an x_{sep} of 0, turbines in the Line shape are aligned side-by-side with the resulting line oriented perpendicular to the wind direction. Increasing x_{sep} has the effect of rotating the line such that it is oriented diagonally with respect to the wind, causing downstream separation. The general design behind all the shapes is intended to allow the leading turbine(s) to interact synergistically with the trailing turbine(s) while mitigating any negative upstream effects. In accordance with the schematic, the V / Reverse V shapes will be simulated with x_{sep}/D ranging from 0.34 to 0.68 and y_{sep}/D ranging from 3 to 6. The Line shape will have the same y_{sep}/D range with an additional x_{sep}/D value of 0 (for the adjacent arrangement).

6.2 3-Turbine Results

The results for the V shape are presented in Figure 6.2, with subplots showing the effect of spacing parameters on the T1, T2, T3, and cluster mean power ratios. It is immediately apparent that the leading turbines T1 and T3 exhibit different trends than the trailing turbine T2. For T1 and T3, as x_{sep} is increased, the synergy declines. This is strong evidence that despite negative upstream effects originating from a downstream turbine (T2) exploiting synergistic flow development, there remains strong inter-turbine synergy. T1 / T3 power ratio variations across y_{sep} is altered by the downstream separation and is even different between these two leading turbines. For T1, the smallest $x_{sep} = 0.34D$ case is different from the greater separation cases in that there is a straightforward decline in power ratio with y_{sep}. This appears logical since an increase in y spacing contributes to a greater separation, thus diminishing proximity-driven inter-turbine

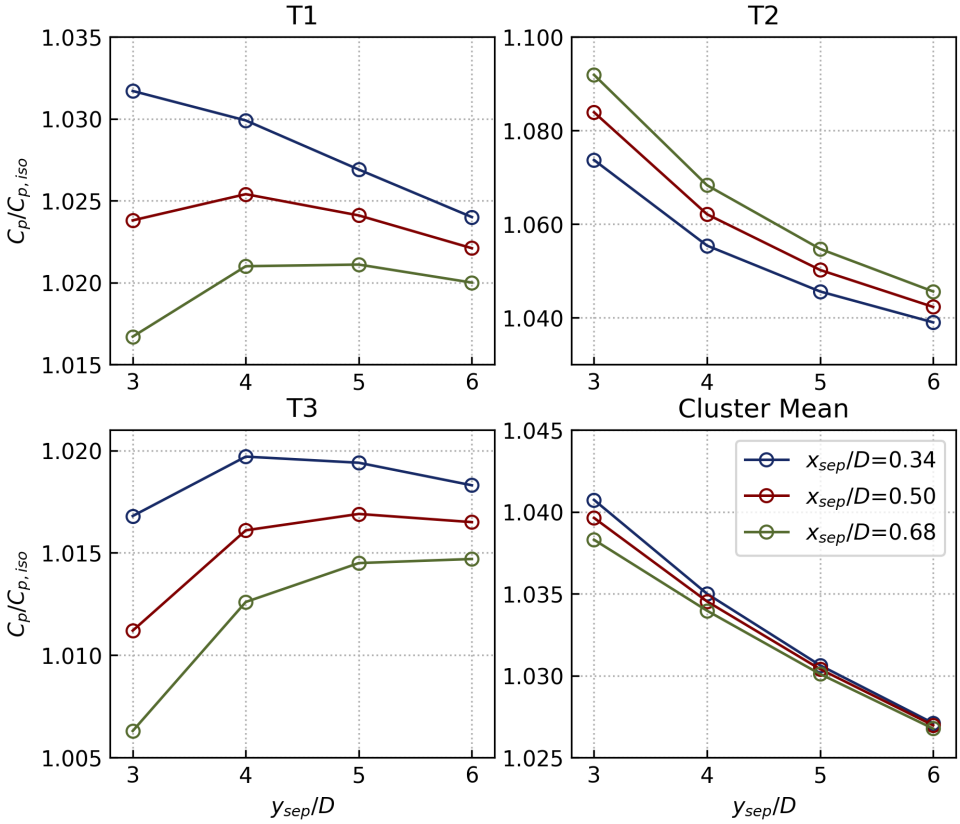


Figure 6.2: Power ratio ($C_p/C_{p,iso}$) as a function of inter-turbine x and y separations for the V shape cases. Subplots present the T1, T2, T3, and the cluster mean power ratios.

interactions. However, for $x_{sep} = 0.50D$ and $0.68D$, T1's power ratio actually first climbs to an optimum at $y_{sep}/D = 4$ before declining with additional y separation. For T3, all of the x_{sep} cases tested follow this more complex trend. An interpretation may be that the competition between synergy and detrimental effects (from a downstream turbine) both as a function of spatial proximity allows for such a peak to emerge. As to the difference between T1 and T3, it is likely due to the effect of co-rotation. Since T2 is placed in the middle, between T1 and T3, the rotationally induced flow changes are opposite between the two pairwise interactions. Thus, the negative impact due to obstruction may be larger on T3 than on T1.

The trailing turbine T2 experiences a consistent decrease in power ratio with increased y separation. However, the trend is yet again different from the leading turbines such that an increased x_{sep} leads to higher, not lower synergy. This unique behavior is explainable according

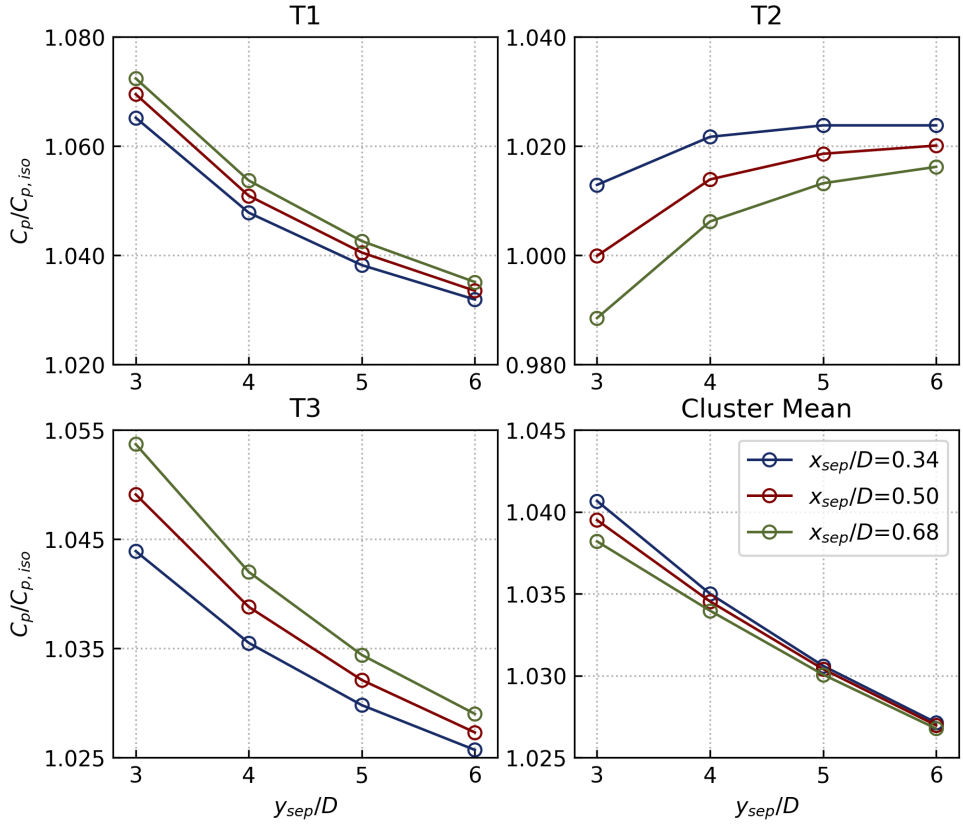


Figure 6.3: Power ratio ($C_p/C_{p,iso}$) as a function of inter-turbine x and y separations for the Reverse V shape cases. Subplots present the T1, T2, T3, and the cluster mean power ratios.

to the heatmaps in the previous chapter, in which downstream turbine can exploit greater synergy if it moves further downstream. However, this increase with downstream distance may plateau as the high synergy band of x_{sep} is reached at about 1 to 1.5 x/D. Thus, it would be expected that a set of cases at $x_{sep}/D = 1$ would overlap or be barely above the data points of $x_{sep}/D = 0.68$. Finally, for the cluster mean power ratio, the consistent decrease with y separation is dominated by T2 whereas a decrease with x separation is dominated by T1 / T3. The magnitude of differences in power ratios due to varying x_{sep} also diminishes with increased y_{sep}, and at $y_{sep} = 5D$–$6D$, the competing dynamics of the upwind / downwind turbines become roughly balanced.

For the Reverse V shape, the results are presented in Figure 6.3. The general trends are consistent with that of the V shape, with the only difference being that there are now one leading

turbine and two trailing turbines. As such, T1 and T3 now reflect the trailing turbine trends while T2 reflects the leading turbine trend. The functional dependence of (T1, T3) and T2 on y_{sep} and x_{sep} in the V array appears to be simply swapped (approximately) with T2 and (T1, T3), respectively, in the Reverse V configuration. This is not very surprising as the V and Reverse V cases represent "swapped" / mirrored configurations (cf. Figures 6.2 and 6.3). Correspondingly, it is observed that the cluster mean power ratios of the V and Reversed V cases are almost identical for the same x_{sep} and y_{sep}. This seems to suggest there may be some upper bound to the exploitation of mean flow properties induced by an upwind turbine, since simply adding another downstream turbine also incurred a cost on the upwind turbine. It remains to be determined using a greater range of configurations and possibly other turbine geometries / operating conditions whether there is an underlying conservation law restricting synergy, or if these current results are merely coincidental.

Finally, the Line shape results are presented in Figure 6.4. This configuration shape is distinct from the previous two in that there is an additional intermediate turbine that is not totally upstream or downstream relative to the other turbines. T2 is downstream of T3 but upstream of T1, and the plots corroborate a heuristical prediction of the respective turbine power performances. To elaborate, it is expected that T3 as the upstream turbine should exhibit trends similar to the upstream turbines in the V and Reverse V cases, featuring a decline in power ratio with increased x_{sep} but a circumstantially parabolic trend with increasing y_{sep} due to the competition of synergy and negative blockage effects. T1 would likely exhibit the characteristics expected of a totally downstream turbine, while T2 being an intermediate might show patterns on some spectrum between the two extrema. Also, the behavior of T2 being the intermediate turbine is very similar to the cluster mean in terms of synergy. All this is exactly what is portrayed in Figure 6.4, where T1 and T3 showed trends in agreement with the previously established patterns and T2 visibly shows the competing effects. While T1 shows synergy increasing with x_{sep} and T3 shows synergy decreasing with x_{sep}, T2 is situated near an inflection point of this trend. For smaller y_{sep} values, there is no longer a strictly increasing / decreasing trend, and the more favorable separations from greatest to least power is ranked $x_{sep} = 0.34D, 0.50D, 0.68D, 0$. Also, the magnitude of the variations due to x separation is also much smaller. Overall, this evidence supports the interpretation of competing dynamics. The cluster mean power ratio is also comparable with that of the V and Reverse V shapes.

To conclude this section, a brief discussion will be offered with regards to the *power density* of these different configurations. Power density can be estimated using cluster C_p (as simulated) divided by the occupying area of the turbine cluster. A simplistic measure of area occupied can be calculated by taking the area as a rectangular bounding box that just encapsulates the rotors of every turbine in the cluster. In this manner, even the side-by-side Line configurations can have a non-zero area and thus a finite power density. It should also be noted that the range of

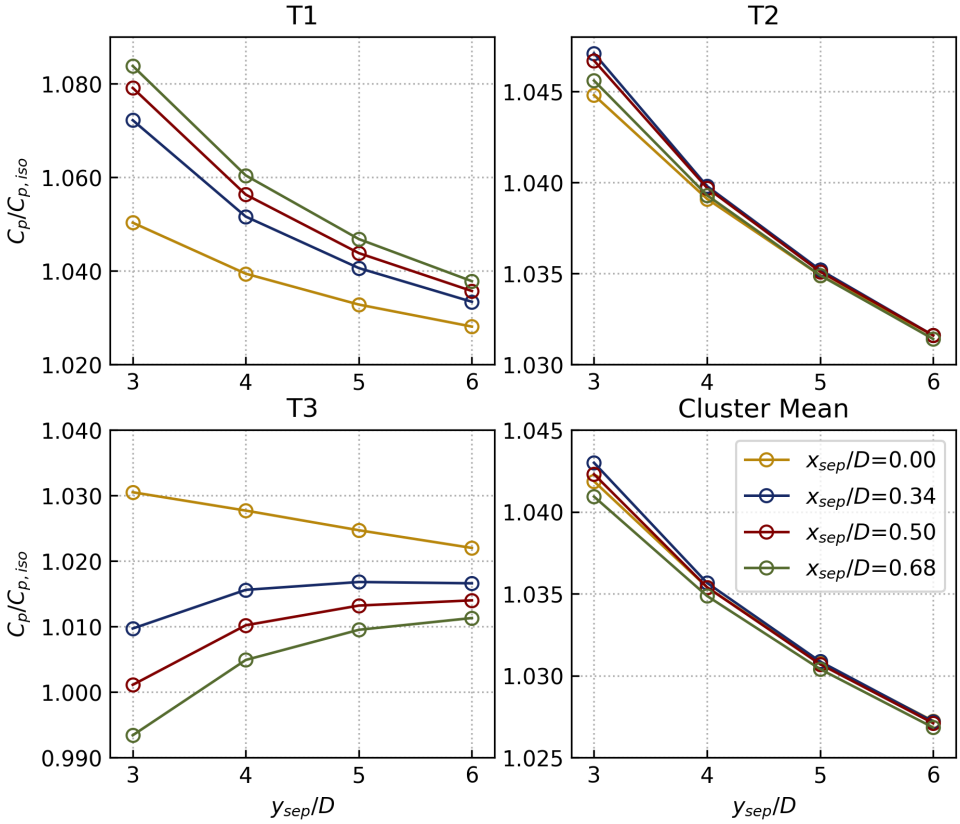


Figure 6.4: Power ratio ($C_p/C_{p,iso}$) as a function of inter-turbine x and y separations for the Line shape cases. Subplots present the T1, T2, T3, and the cluster mean power ratios.

areas explored by Line shapes is greater than the other shapes, since cases span from the side-by-side arrangements that minimize area to the larger turbine-wide x separations than V / Reverse V shapes. Also, the V and Reverse V shapes produced nearly identical power densities due to similar cluster C_p, as explained above. For the same area, it appears that the Line shape offers a marginally higher power density for larger areas while V / Reverse V had higher power densities for smaller areas. As an example, for an area of 13.14 m^2, power density is 0.1295 (V / Reverse V) $>$ 0.1284 (Line); and for an area of 14.68 m^2, power density is 0.1158 (V / Reverse V) $<$ 0.1163 (Line). Thus, while there exist subtle differences between shapes for the same area, they are far from being significant enough to warrant the formation of any universal design guidelines. However, power density is found to decrease exponentially with increased area for all shapes, which likely parallels the decline of synergistic interactions as inter-turbine spacing increases.

Furthermore, it appears that for 3-turbine clusters, the Line shape which features turbines side-by-side perpendicular to the incoming flow maximizes power density. Hence, compactness is essential for achieving intense synergy and thus optimal power density.

6.3 Synergy Superposition Scheme

In the preceding discussions within this chapter, intuitions surrounding pairwise interactions are often used to supplement the analysis of 3-turbine interactions. This alludes to a rough concept that the 3-turbine cluster synergy (or other negative effects) might be evaluated as some linear superposition of all the pairwise VAWT interactions. In this section, this hypothesis is posed and tested using the current ALM framework. The specific procedure is as follows. For each 3-turbine configuration, the relative pairwise sub-arrangements are identified and simulated, with existing 2-turbine cases and effectively identical cases avoided (which typically arose from the Line shapes). After the corresponding 2-turbine cases have been run, the superimposed estimate of some turbine's power ratio is formulated according to:

$$\left(\frac{C_p}{C_{p,iso}}\right)_{si} = 1 + \sum_{i}^{n} \left[\left(\frac{C_p}{C_{p,iso}}\right)_i - 1\right] \tag{6.1}$$

where the subscript si denotes the superimposed (approximate) power ratio of a turbine and the summation is done for n pairs indexed as i ($n = 2$ pairs for 3-turbine clusters), wherein the pairwise power ratio is extracted for a turbine with the same relative positioning as the turbine in the 3-turbine cluster. Essentially, this assumes that difference between the power ratio and unity (whether positive or negative) is the quantity that can be linearly superimposed via addition. There is no special motivation for this formulation other than to establish a simple relationship to test.

Based on the results for every case among the V, Reverse V, and Line shapes, the synergy superposition hypothesis holds with surprising accuracy. By defining a percent error using the actual simulated power ratio and the pairwise superimposed power ratio, it is possible to see the error magnitude bounded at 0.38%, 0.32%, and 0.35% for V, Reverse V, and Line shapes, respectively. The specific case-by-case errors are plotted in Figures 6.5, 6.6, and 6.7. These plots reveal complex variations of error with array spacings. For instance, while some turbines exhibit a consistent decrease in error with the y_{sep}, the error magnitude does not converge onto 0 but rather crosses over to the negative side. Thus, it is not guaranteed that a greater separation leads to lower errors since the magnitude of synergy to predict could also be smaller. At the same time, this may also indicate the presence of some systematic / calibration error. Also,

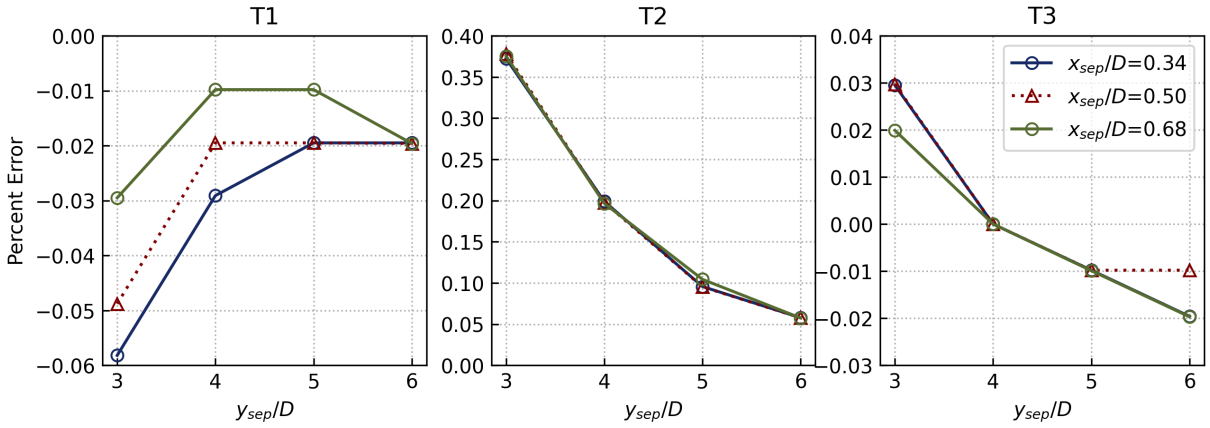


Figure 6.5: Plot of percent errors when using pairwise power ratio superposition to approximate 3-turbine interactions for the V shape cases. A positive error indicates the superposition overpredicted the power ratio and a negative error indicates underprediction.

for certain turbines, especially in the Line-shaped cases, the error trend varies more drastically across different x_{sep}. The center turbines (the ones enveloped by a turbine on either side in the y direction) exhibited errors an order of magnitude higher than edge turbines, regardless of configuration (whether it is leading, trailing, or intermediate). This seems to suggest that the proposed scheme may lead to greater over-estimation for shorter-range interactions, which is a plausible explanation since the synergistic and detrimental effects intensify with proximity. Overall, no cohesive pattern can be deduced from these plots, and it should be noted that given the remarkably small error magnitudes (all $< 1\%$), simulation numerical errors may account for a substantial portion of these differences.

Overall, it is unexpected yet valuable to find that the superposition hypothesis works to such great accuracy for 3-turbine cases. The presence of interjecting turbines (i.e., the center one) and corresponding wakes did not seem to hinder the prediction accuracy in a significant manner. Although the currently proposed superposition scheme appears to hold quite well for 3-turbine arrays, it is anticipated that this accuracy may drop as the quantity of closely placed turbines increase, which may be of interest for larger-scale wind farm micro-siting. It is hypothesized that "higher-order effects" between several turbines may cause the current simple superposition results to deviate from the actual case in the larger turbine groups. Thus, further work is advised to determine the limitations of this new approach.

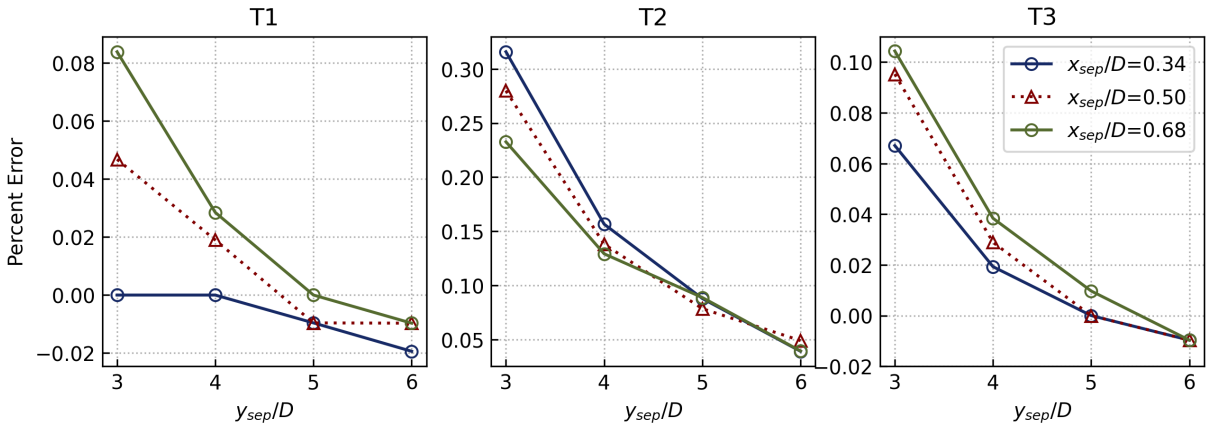


Figure 6.6: Plot of percent errors when using pairwise power ratio superposition to approximate 3-turbine interactions for the Reverse V shape cases. A positive error indicates the superposition overpredicted the power ratio and a negative error indicates underprediction.

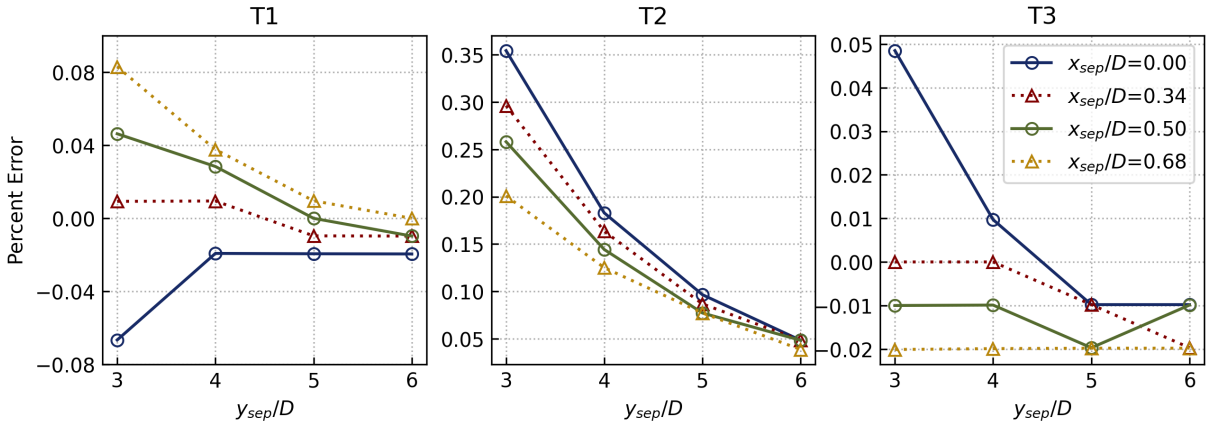


Figure 6.7: Plot of percent errors when using pairwise power ratio superposition to approximate 3-turbine interactions for the Line shape cases. A positive error indicates the superposition overpredicted the power ratio and a negative error indicates underprediction.

Chapter 7

Conclusions and Recommendation for Further Studies

In this thesis, the synergistic interactions between multiple vertical axis wind turbines was investigated using the Actuator Line Model, a kind of fast-simulation technique that forgoes the meshing of the physical structure. Past literature in this area have confirmed the presence of a type of power-enhancing phenomenon that manifests between multiple VAWTs operating in close proximity, which is referred to as "synergy" in the current work. The nature of these synergistic effects opens new opportunities for configuring VAWT arrays such as in wind farm applications, specifically making it permissible and even optimal to group several turbines together in tight clusters. However, there exists a major contemporary barrier to solving the VAWT farm micro-siting problem with consideration of synergistic effects — the computational cost of full-order CFD. As an attempt to explore reduced-order modelling options, the Actuator Line Model originally developed for HAWTs has been adapted for use towards single and multiple VAWT simulation. The current thesis sought to extend in this direction and analyze the synergy of 2- and 3-turbine ALM simulations. In Chapter 3, a from-scratch implementation of the ALM logic using Fluent UDFs has been detailed. This code as well as an available OpenFOAM ALM library called `turbinesFoam` (by Bachant et al.) has been validated in Chapter 4 using prior literature, experimental measurements, and a URANS CFD simulation. Chapters 5 and 6 present the methodology, results, and analysis of 2- and 3-turbine array performance and synergy using ALM, respectively. To conclude, the following key findings are hereby presented (in the order that they were documented in this thesis):

1. Turbine ALM forces are relatively insensitive to changes in the Gaussian kernel width parameter but are strongly dependent on the set of airfoil lift and drag coefficients used for

look-up

2. For the currently explored 2-bladed VAWT operating at a TSR of 3.7, the ALM technique in conjunction with URANS + a k-ω SST model is capable of predicting synergy in 2- and 3-turbine clusters

3. For 2-turbine cases, the leading turbine T1 (if configuration is non-adjacent) can experience performance detriment if the trailing turbine T2 moves close to T1's wake, revealing a trade-off relationship in the synergistic interactions

4. For mutually beneficial configurations, T2 still experiences over 2 times the synergy of T1

5. Co-rotating and counter-rotating VAWT pairs exhibited small differences ($< 0.01\%$) in the power ratio and synergy, and any comparative advantages are not universal across the configuration space

6. For all 2-turbine cases, the cluster experiences net synergy as long as T2 is not positioned directly in the wake deficit region downstream of T1

7. Among the 3-turbine cluster shapes investigated (V, Reverse V, Line), it is found that: a downstream turbine's power ratio decreases with increasing y_{sep} but increases with increasing x_{sep}; an upstream turbine's power ratio clearly decreases with increasing x_{sep} but generally increases to a maximum before decreasing for increasing y_{sep}; an intermediate turbine that is both upstream and downstream to another turbine exhibits a balance of both patterns

8. The aforementioned 3-turbine cluster synergy variations as a function of array spacing are explainable based on the 2-turbine trade-off where a significantly beneficial position for the downstream turbine may disadvantage the upstream one

9. The superposition hypothesis (wherein the individual power ratios of turbines in a 3-turbine cluster can be approximated by a summation of pairwise interactions) is found to work very well with percent errors of $< 1\%$ in magnitude

Overall, the key results summarized above are positive and convincing. They point to a promising application of ALM, which can be an order of magnitude faster than full-order CFD, towards array layout optimization and design. It is demonstrated that ALM does not fundamentally prohibit the manifestation of synergy, predicts patterns of synergy as a function of the configuration which are in good agreement with previous studies, and can even be used to construct a simple superposition scheme for power ratios. However, further work is recommended

before certifying ALM as a valid engineering tool for micro-siting purposes. Essentially, the four aspects that require improvement or further investigation are:

1. More rigorously validating isolated turbine performance (procuring a high-fidelity C_L / C_D look-up table for the chord-based Reynolds number of blade operation, testing different Gaussian kernel formulations, applying correction models) and the power ratios and synergy observed in the 2-/3-turbine ALM cases (using blade-resolved URANS CFD)
2. Simulating and validating similar configurations for different turbine geometries (number of blades, airfoil profile, solidity) and operating conditions (TSR)
3. Extending study to 4+ turbines, with emphasis on evaluating the generalizability of the proposed superposition scheme
4. Incorporating disturbances due to dynamic effects to supplement the the ALM

ALM is inherently limited and encumbered by several fundamental simplifications / assumptions, such as the representation of blade aerodynamics entirely using a BE-M formulation driven by a C_L / C_D look-up table and the use of Gaussian kernels to distribute the force. Future work should seek to increase the accuracy by systematically studying the effects of each on the isolated VAWT power. Additionally, an expensive suite of high-fidelity, full-order CFD should be undertaken to fully validate the current power ratios (similar to what Hansen et al. has done). This is the most reliable method for confirming whether ALM can predict these interactions with great accuracy using relative power metrics, or whether there are embedded shortcomings due to isolated C_p mispredictions. Having achieved a confident degree of validation, it is recommended that additional turbine geometries and operating conditions be tested to understand how synergy or the lack thereof depends on these factors. The accuracy of ALM and in particular the pairwise superimposition scheme found thus far to be promising should be rigorously tested at higher turbine counts. Lastly, it is suggested that the dynamic disturbances underpinning VAWT interactions be studied in-depth and incorporated explicitly to improve the ALM's ability to resolve and predict wind farm performance. This may include but is not limited to the incorporation of correction models (e.g., dynamic stall), considerations for turbulence phenomena, and other supplemental dynamically driven components. These directions for future work would fortify the technical footing of using ALM to evaluate synergy and refine the technique towards the important application of VAWT micro-siting. For example, one important extension of the research contributions presented herein would be towards the study of *heterogeneous* wind farms, which are composed of VAWTs with different configurations or sizes. This practical application aids the efficient determination of the feasibility of such wind farms and even their optimal design once the robustness of the ALM approach can be further improved based on the above points.

References

[1] E. Möllerström, P. Gipe, J. Beurskens, and F. Ottermo, "A historical review of vertical axis wind turbines rated 100 kW and above," *Renewable and Sustainable Energy Reviews*, vol. 105, pp. 1–13, 2019.

[2] B. P. LeBlanc and C. S. Ferreira, "Experimental determination of thrust loading of a 2-bladed vertical axis wind turbine," *J. Phys.: Conf. Ser.*, vol. 1037, p. 022043, 2018.

[3] R. Zhao, A. C. W. Creech, A. G. L. Borthwick, V. Venugoal, and T. Nishino, "Aerodynamic analysis of a two-bladed vertical-axis wind turbine using a coupled unsteady RANS and Actuator Line Model," *Energies*, vol. 13, no. 4, pp. 776–801, 2020.

[4] A. Korobenko, M. C. Hsu, I. Akkerman, and Y. Bazilevs, "Aerodynamic simulation of vertical-axis wind turbines," *Journal of Applied Mechanics*, vol. 81, 2014.

[5] J. McNaughton, F. Billard, and A. Revell, "Turbulence modelling of low Reynolds number flow effects around a vertical axis turbine at a range of tip-speed ratios," *Journal of Fluids and Structures*, vol. 47, pp. 124–138, 2014.

[6] J. O. Dabiri, "Potential order-of-magnitude enhancement of wind farm power density via counter-rotating vertical-axis wind turbine arrays," *Journal of Renewable and Sustainable Energy*, vol. 3, p. 043104, 2011.

[7] I. D. Brownstein, N. J. Wei, and J. O. Dabiri, "Aerodynamically interacting vertical axis wind turbines: performance enhancement and three-dimensional flow," *Energies*, vol. 12, no. 14, p. 2724, 2019.

[8] B. Sanderse, S. P. van der Pijl, and B. Koren, "Review of computational fluid dynamics for wind turbine wake aerodynamics," *Wind Energy*, vol. 14, pp. 799–819, 2011.

[9] S. Shamsoddin and F. Porté-Agel, "Large Eddy Simulation of vertical axis wind turbine wakes," *Energies*, vol. 7, pp. 890–912, 2014.

[10] P. Bachant, "Physical and numerical modeling of cross-flow turbines," Ph.D. dissertation, University of New Hampshire, 2016.

[11] M. D'Ambrosio and M. Medaglia, "Vertical axis wind turbines: History, technology, and applications," Master's thesis, Halmstad University, 2010.

[12] K. Liu, M. Yu, and W. Zhu, "Enhancing wind energy harvesting performance of vertical axis wind turbines with a new hybrid design: A fluid-structure interaction study," *Renewable Energy*, vol. 140, pp. 912–927, 2019.

[13] J. Peng, "Effects of aerodynamic interactions of closely-placed vertical axis wind turbine pairs," *Energies*, vol. 11, p. 2842, 2018.

[14] P. J. A. Abhishek, "Performance prediction and fundamental understanding of small scale vertical axis wind turbine with variable amplitude blade pitching," *Renewable Energy*, vol. 97, pp. 97–113, 2016.

[15] M. K. Johari, M. A. A. Jalil, and M. F. M. Shariff, "Comparison of horizontal axis wind turbine (HAWT) and vertical axis wind turbine (VAWT)," *International Journal of Engineering & Technology*, vol. 7, no. 4.13, pp. 74–80, 2018.

[16] L. Du, G. Ingram, and R. G. Dominy, "A review of H-Darrieus wind turbine aerodynamic research," *Journal of mechanical engineering science*, vol. 233, pp. 23–24, 2019.

[17] M. Ghasemian, Z. N. Ashrafi, and A. Sedaghat, "A review on computational fluid dynamic simulation techniques for Darrieus vertical axis wind turbines," *Energy Conversion and Management*, vol. 149, pp. 87–100, 2017.

[18] I. Paraschivoiu, *Wind turbine design: With emphasis on Darreius concept*. Presses internationales Polytechniques, 2009.

[19] A. Meana-Fernández, J. M. F. Oro, K. M. A. Díaz, and S. Velarde-Suárez, "Turbulence-model comparison for aerodynamic-performance prediction of a typical vertical-axis wind-turbine airfoil," *Energies*, vol. 12, p. 488, 2019.

[20] M. R. Castelli, A. Englaro, and E. Benini, "The Darrieus wind turbine: Proposal for a new performance prediction model based on cfd," *Energy*, vol. 36, pp. 4919–4934, 2011.

[21] A. Posa and E. Balaras, "Large Eddy Simulation of an isolated vertical axis wind turbine," *Journal of Wind Engineering & Industrial Aerodynamics*, vol. 172, pp. 139–151, 2018.

[22] K. Hamada, T. Smith, N. Durrani, N. Qin, and R. Howell, "Unsteady flow simulation and dynamic stall around vertical axis wind turbine blades," *46th AIAA Aerospace Sciences Meeting and Exhibit*, 2008.

[23] R. Howell, N. Qin, J. Edwards, and N. Durrani, "Wind tunnel and numerical study of a small vertical axis wind turbine," *Renewable Energy*, vol. 35, no. 2, pp. 412–422, 2010.

[24] A.-J. Buchner, M. W. Lohry, L. Martinelli, J. Soria, and A. J. Smits, "Dynamic stall in vertical axis wind turbines: Comparing experiments and computations," *Journal of Wind Engineering and Industrial Aerodynamics*, vol. 146, pp. 163–171, 2015.

[25] A. Orlandi, M. Collu, S. Zanforlin, and A. Shires, "3D URANS analysis of a vertical axis wind turbine in skewed flows," *Journal of Wind Engineering and Industrial Aerodynamics*, vol. 147, pp. 77–84, 2015.

[26] W. Zuo, X. Wang, and S. Kang, "Numerical simulations on the wake effect of H-type vertical axis wind turbines," *Energy*, vol. 106, pp. 691–700, 2016.

[27] C. Li, S. Zhu, Y. Xu, and Y. Xiao, "2.5D large eddy simulation of vertical axis wind turbine in consideration of high angle of attack flow," *Renewable Energy*, vol. 51, pp. 317–330, 2013.

[28] M. Elkhoury, T. Kiwata, and E. Aoun, "Experimental and numerical investigation of a three-dimensional vertical-axis wind turbine with variable-pitch," *Journal of Wind Engineering and Industrial Aerodynamics*, vol. 139, pp. 111–123, 2015.

[29] F. Scheurich and R. E. Brown, "Effect of dynamic stall on the aerodynamics of vertical-axis wind turbines," *AIAA Journal*, vol. 49, no. 11, pp. 2511–2521, 2011.

[30] R. Nobile, M. Vahdati, J. F. Barlow, and A. Mewburn-Crook, "Unsteady flow simulation of a vertical axis augmented wind turbine: A two-dimensional study," *Journal of Wind Engineering and Industrial Aerodynamics*, vol. 125, pp. 168–179, 2014.

[31] G. Abdalrahman, "Pitch angle control for a small-scale Darrieus vertical axis wind turbine with straight blades (H-type VAWT)," Ph.D. dissertation, University of Waterloo, 2019.

[32] D. D. Tavernier, C. Ferreira, and G. van Bussel, "Airfoil optimisation for vertical-axis wind turbines with variable pitch," *Wind Energy*, vol. 22, pp. 547–562, 2019.

[33] B. Strom, S. L. Brunton, and B. Polagye, "Intracycle angular velocity control of cross-flow turbines," *Nature Energy*, vol. 2, p. 17103, 2017.

[34] R. J. Barthelmie, G. C. Larsen, S. T. Frandsen, L. Folkerts, K. Rados, S. C. Pryor, B. Lange, and G. Schepers, "Comparison of wake model simulations with offshore wind turbine wake profiles measured by Sodar," *Journal of Atmospheric and Oceanic Technology*, vol. 23, no. 7, pp. 888–901, 2006.

[35] X. Yang and F. Sotiropoulos, "Analytical model for predicting the performance of arbitrary size and layout wind farms," *Wind Energy*, vol. 19, no. 7, pp. 1239–1248, 2015.

[36] M. Kinzel, Q. Mulligan, and J. O. Dabiri, "Energy exchange in an array of vertical-axis wind turbines," *Journal of Turbulence*, vol. 13, no. 38, pp. 1–13, 2012.

[37] S. H. Hezaveh, E. Bou-Zeid, J. O. Dabiri, M. Kinzel, G. Cortina, and L. Martinelli, "Increasing the power production of vertical-axis wind-turbine farms using synergistic clustering," *Boundary-Layer Meteorology*, vol. 169, pp. 275–296, 2018.

[38] M. Ahmadi-Baloutaki, R. Carriveau, and D. S. Ting, "A wind tunnel study on the aerodynamic interaction of vertical axis wind turbines in array configurations," *Renewable Energy*, vol. 96, pp. 904–913, 2016.

[39] S. Zanforlin and T. Nishino, "Fluid dynamic mechanisms of enhanced power generation by closely spaced vertical axis wind turbines," *Renewable Energy*, vol. 99, pp. 1213–1226, 2016.

[40] M. Shaheen and S. Abdallah, "Development of efficient vertical axis wind turbine clustered farms," *Renewable and Sustainable Energy Reviews*, vol. 63, pp. 237–244, 2016.

[41] H. F. Lam and H. Y. Peng, "Measurements of the wake characteristics of co- and counter-rotating twin h-rotor vertical axis wind turbines," *Energy*, vol. 131, pp. 13–26, 2017.

[42] S. Shaaban, A. Albatal, and M. H. Mohamed, "Optimization of H-Rotor Darrieus turbines' mutual interaction in staggered arrangements," *Renewable Energy*, vol. 125, pp. 87–99, 2018.

[43] A. Barnes and B. Hughes, "Determining the impact of VAWT farm configurations on power output," *Renewable Energy*, vol. 143, pp. 1111–1120, 2019.

[44] J. T. Hansen, M. Mahak, and I. Tzanakis, "Numerical modelling and optimization of vertical axis wind turbine pairs: A scale up approach," *Renewable Energy*, vol. 171, pp. 1371–1381, 2021.

[45] P.-L. Delafin, T. Nishino, A. Kolios, and L. Wang, "Comparison of low-order aerodynamic models and RANS CFD for full scale 3D vertical axis wind turbines," *Renewable Energy*, vol. 109, pp. 564–575, 2017.

[46] G. Tescione, C. J. S. Ferreira, and G. J. W. van Bussel, "Analysis of a free vortex wake model for the study of the rotor and near wake flow of a vertical axis wind turbine," *Renewable Energy*, vol. 87, pp. 552–563, 2016.

[47] A. Zanon, P. Giannattasio, and C. J. S. Ferreira, "A vortex panel model for the simulation of the wake flow past a vertical axis wind turbine in dynamic stall," *Wind Energy*, vol. 16, pp. 661–680, 2013.

[48] ——, "Wake modelling of a VAWT in dynamic stall: impact on the prediction of flow and induction fields," *Wind Energy*, vol. 18, pp. 1855–1874, 2015.

[49] K. R. Dixon, "The near wake structure of a vertical axis wind turbine," Master's thesis, Delft University of Technology, 2008.

[50] H. F. Lam and H. Y. Peng, "Development of a wake model for Darrieus-type straight-bladed vertical axis wind turbines and its application to micro-siting problems," *Renewable Energy*, vol. 114, pp. 830–842, 2017.

[51] M. Abkar, "Theoretical modeling of vertical-axis wind turbine wakes," *Energies*, vol. 12, no. 1, p. 10, 2019.

[52] E. B. Tingey and A. Ning, "Development of a parameterized reduced-order vertical-axis wind turbine wake model," *Wind Engineering*, vol. 44, no. 5, pp. 494–508, 2020.

[53] A. Ning, "Actuator cylinder theory for multiple vertical axis wind turbines," *Wind Energy Science*, vol. 1, no. 2, pp. 327–340, 2016.

[54] J. N. Sørensen and W. Z. Shen, "Numerical modeling of wind turbine wakes," *Journal of Fluids Engineering*, vol. 124, pp. 393–399, 2002.

[55] N. Troldborg, "Actuator line modeling of wind turbine wakes," Ph.D. dissertation, Technical University of Denmark, 2008.

[56] B. M. C. Johnson, "Computational Fluid Dynamics (CFD) modelling of renewable energy turbine wake interactions," Ph.D. dissertation, University of Central Lancashire, 2015.

[57] P. Claudio, "Implementation of an actuator line method for aerodynamic analysis of horizontal axis wind turbines," Ph.D. dissertation, University of Padua, 2019.

[58] S. H. Hezaveh, E. Bou-Zeid, M. W. Lohry, and L. Martinelli, "Simulation and wake analysis of a single vertical axis wind turbine," *Wind Energy*, vol. 20, pp. 713–730, 2017.

[59] R. E. Sheldahl and P. C. Klimas, "Aerodynamic characteristics of seven symmetrical airfoil sections through 180-degree angle of attack for use in aerodynamic analysis of vertical axis wind turbines," Sandia National Laboratories, Tech. Rep. SAND8O-2114, 1981.

[60] A. C. W. Creech, A. G. L. Borthwick, and D. Ingram, "Effects of support structures in an les actuator line model of a tidal turbine with contra-rotating rotors," *Energies*, vol. 10, pp. 726–750, 2017.

[61] M. Abkar, "Impact of subgrid-scale modeling in actuator-line based large-eddy simulation of vertical-axis wind turbine wakes," *Atmosphere*, vol. 9, pp. 257–270, 2018.

[62] P. Bachant, A. Goude, and M. Wosnik, "Actuator line modeling of vertical-axis turbines," arXiv, 2018, preprint arXiv:1605.01449v4.

[63] V. Mendoza, P. Bachant, C. Ferreira, and A. Goude, "Near-wake flow simulation of a vertical axis turbine using an actuator line model," *Wind Energy*, vol. 22, pp. 171–188, 2019.

[64] P. Bachant, A. Goude, and M. Wosnik, "turbinesFoam: v0.0.8," Zenodo, 2018, https://doi.org/10.5281/zenodo.1210366.

[65] V. Mendoza and A. Goude, "Validation of actuator line and vortex models using normal forces measurements of a straight-bladed vertical axis wind turbine," *Energies*, vol. 13, pp. 511–526, 2020.

[66] "ANSYS® Fluent User Guide, Release 15.0," SAS IP, Inc, 2013.

[67] "Fluent 2020 R2: Fluent Customization Manual," ANSYS, Inc, 2020.

[68] F. Bertagnolio, N. Sørensen, J. Johansen, and P. Fuglsang, "Wind turbine airfoil catalogue," Risø National Laboratory, Tech. Rep. Risø-R-1980(EN), 2001.

[69] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, 1992.

[70] C. R. Vogel and R. H. J. Willden, "Investigation of wind turbine wake superposition models using Reynolds-averaged Navier-Stokes simulations," *Wind Energy*, vol. 23, pp. 593–607, 2020.