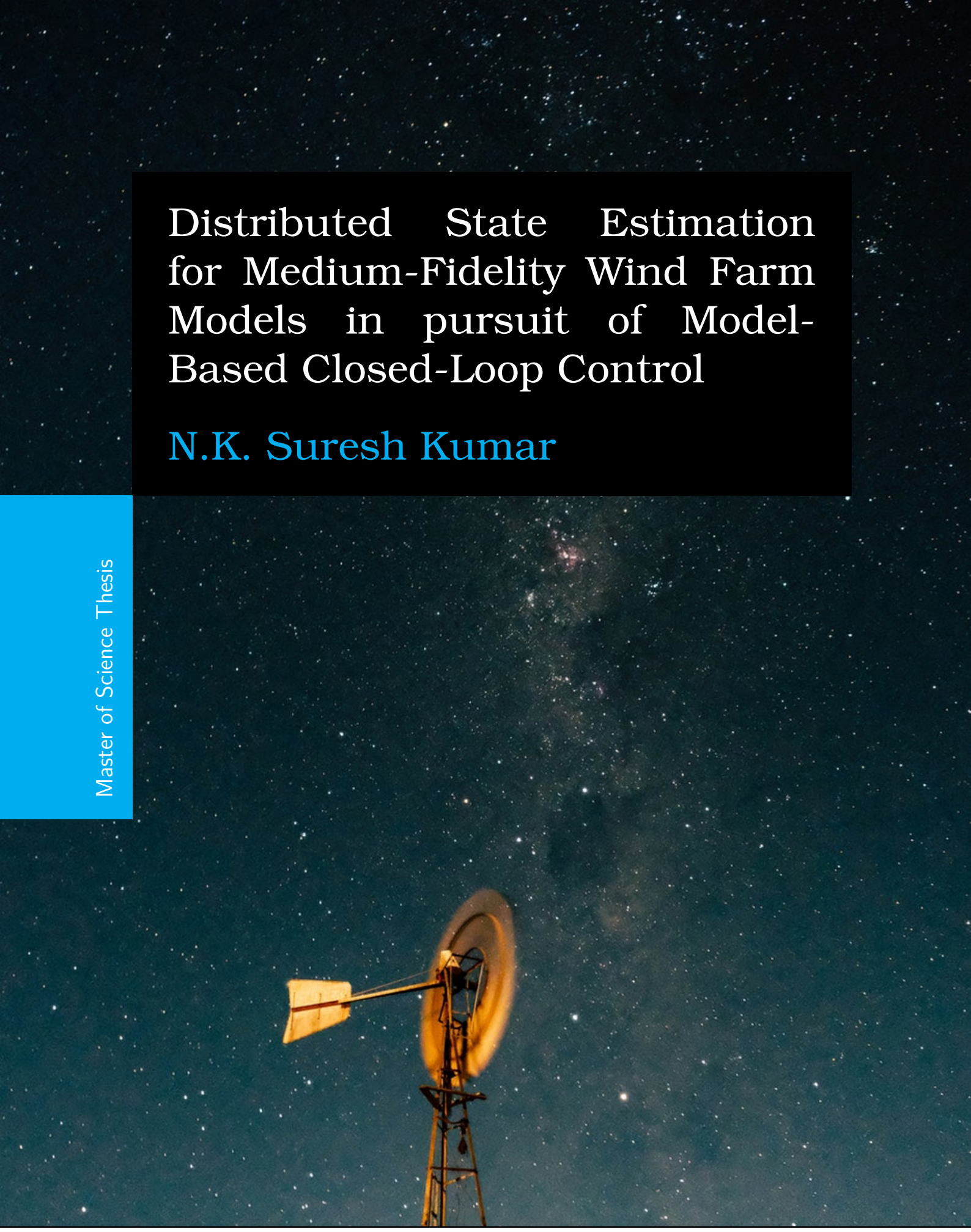
Distributed State Estimation for Medium-Fidelity Wind Farm Models in pursuit of Model-Based Closed-Loop Control

N.K. Suresh Kumar

Master of Science Thesis



TU Delft Delft University of Technology

Delft Center for Systems and Control

Distributed State Estimation for Medium-Fidelity Wind Farm Models in pursuit of Model-Based Closed-Loop Control

Master of Science Thesis

For the degree of Master of Science in Systems and Control at Delft University of Technology

N.K. Suresh Kumar

September 12, 2018

Faculty of Mechanical, Maritime and Materials Engineering (3mE) · Delft University of Technology

The work in this thesis was supported by Karlsruhe Institute of Technology (KIT). Their cooperation is hereby gratefully acknowledged.

Abstract

Wind turbines are often placed closed to each other as a wind farm, for the advantages it offers. The advantages, i.a., are reduced land- or sea-usage, installation, and maintenance costs. However, like a two-sided coin, wind farms do come with disadvantages. That is, when turbines are placed closed to each other, the amount of power left for the downstream turbines to capture is less than what an equal number of individually-placed turbines can capture, as the upstream turbines try to capture most of the incoming flow for maximizing their own power production. Consequently, the overall power production in a wind farm decreases, with the increase in the structural loading.

From the control perspective, this issue can be tackled by employing wind farm control. Typically, a model-based wind farm controller, in an open-loop setting, operates based on the velocity of the wind flow, predicted by a wind farm model. For such a controller to achieve the desired level of power production, a wind farm model has to be accurate and computationally tractable.

Generally, high-fidelity wind farm models are accurate, but are computationally complex. Hence, real-time control is not feasible. This issue can be addressed by employing closed-loop control. In this approach, low- or medium-fidelity wind farm models, which are computationally tractable, are used, and their accuracy is improved by employing an estimator.

In the current research, a centralized estimation approach is employed. In this framework, a single estimator is employed for estimating all the states (second-to-second wind field at turbine hub height) in a wind farm. Simulation results show that the accuracy of an open-loop model can be improved. However, the problem is the state size of the wind farm models. Each iteration, here, takes at least 10 times more time for computation than what is available.[1] This leads to the objective of the thesis, which is "***Can the accuracy of a wind farm model be improved while maintaining computational tractability?***".

In this thesis, distributed estimation is proposed as a solution to this problem, and four distributed architectures are devised, using the medium-fidelity model WindFarmSimulator

[1] In this thesis, simulations are carried out with a sampling time of 1 Hz. Hence, each iteration should take less than a second for generating the control signals. However, depending upon the configuration, centralized estimation approach can approximately take somewhere between 10 to 16 seconds for each iteration.

(WFSim). Distributed architecture 1 follows centralized architecture, for modelling a wind farm, i.e., a single wind farm model is employed to predict the wind flow throughout a wind farm. However, unlike the centralized architecture where a single estimator is employed to estimate all the states in a wind farm, a number of estimators are employed here to collectively estimate all the states in the wind farm, with each estimator around a turbine. For each estimator, a spatial domain is selected around their respective turbine, and each estimator estimates only the states that are within their domain, in parallel. Hence, each estimator is computational tractable, and since the estimators operate in parallel, the overall complexity reduces. Simulations show negligible loss in performance, with gains in computation time by a factor of 10.

In architecture 2, a wind farm model is defined for each turbine, with each model predicting the flow field throughout the wind farm. In other words, the spatial domain of each model is selected to be the entire wind farm. This can be imagined as a combination of a number of centralized wind farm models, with each model describing only one turbine. Further, an estimator is employed to independently estimate the states of each subsystem. Since the size of each model is the same and each estimator estimates all the states in their respective wind farm model, the time taken by each iteration is actually more than the centralized case. Nevertheless, architecture 2 gives useful insights on the effects of considering only one turbine in a wind farm. Simulation shows that each subsystem in architecture 2 is capable of improving the accuracy of the estimates only around the turbine it models. This is because the interactions between the turbines are not described in architecture 2, as each model considers the effects of only one turbine. To include the interactions, an ad hoc correction for wake effects is implemented using sensor measurements and fusion algorithms. Simulation shows that architecture 2 with data fusion is capable of incorporating the interactions, and the accuracy of the estimate is similar to the centralized case.

Architecture 3 is a computationally efficient version of architecture 2. This architecture is devised by combining architectures 1 and 2. Similar to architecture 2, a wind farm model is defined for each turbine, with the spatial domain of each model being the entire wind farm. For estimation, like architecture 1, a small spatial domain is selected around the turbine in each subsystem, and an estimator is employed to estimate only the states that are within that domain. Hence, the time taken for each iteration reduces, as the number of states estimated by each estimator have reduced, and the accuracy of the estimate in architecture 3 with data fusion is similar to the centralized case.

In architecture 4, a wind farm model is defined for each turbine, but this time the spatial domain of the models are reduced. Hence, each model is restricted to predict the velocity of the wind flow only around their respective turbine, as opposed to predicting throughout the wind farm in the centralized and rest of the distributed architectures. Further, the size of the spatial domain is selected such that there is an overlap between the subsystems. This helps to maintain the coupling between the subsystems, and model the interactions between the turbines. Then, similar to architecture 2, an estimator is employed for independently estimating the states of each subsystem. Simulation shows a drastic decrease in the time taken for each iteration, and the accuracy of the estimate is reasonably good.

Apart from the aforementioned simulations, tests are also performed to study the effects of localization[2] on the centralized filters, frequency of linearization of the WFSim model on the

[2]Localization is employed to restrict the number of states that are estimated.

centralized Extended Kalman Filter (ExKF)[3], and size of the subsystems on the distributed architectures. Simulations show that the localization length affects the performance of the centralized filters only to a certain size, the frequency of linearization of the WFSim has no effect on the ExKF for a constant atmospheric condition, and the size of the subsystems in the distributed architectures increases the accuracy of the estimates only to a certain size.

In conclusion, distributed architectures are capable of improving the accuracy of the open-loop wind farm models to the same level of accuracy offered by the centralized architecture. More importantly, distributed architectures offer computational tractability, irrespective of the estimation algorithms being employed. Furthermore, distributed architecture 4 is preferred over other distributed architectures, as architecture 4 reduces the size of the wind farm model. Consequently, this has the prospect for reducing the computational complexity of the controllers. However, the performance of such a controller is highly questionable. The work presented in this thesis is first-of-a-kind in the field of distributed estimation for closed-loop control of wind farms.

[3]Conventional Kalman filters like the ExKF demand a linear system model. However, WFSim is a non-linear system. Hence, the model has to be linearized.

Table of Contents

List of Figures

List of Tables

Acknowledgements

The last two years in my life has been nothing short of a crazy ride on the violent *Whomping Willow* from Harry Potter, with so many ups and downs. At the start of the course, it seemed like an easy two-year journey, with the reward on the other side of the two-year time-line being "*M.Sc. in Systems and Control*". In reality, this journey, through the M.Sc. course, needed a lot of black magic, filtering, and a one heck of a course named 'Control Theory' to reach the other end. Yet, here, I am on the verge of graduation, with the degree being just a few days away from my reach. Over the course of last two years, I have a lot of people to thank for. However, here, I will be thanking those people who helped me in traversing my master thesis, a chapter of my life which seemed unassailable.

Firstly, I would like to thank my supervisor prof. dr. ir. J.W. van Wingerden for providing me this valuable opportunity to perform my thesis. At the same time, this work would not have been made possible without the support of ir. B. Doekemeijer, who was more than just a daily supervisor. I would also like to extend my thanks to dr. ir. B. Noack (post-doctoral researcher at Karlsruhe Institute of Technology, Germany), for hosting me at KIT, and helping me in critically analyzing the topic.

Secondly, I would like to thank my parents, Santhi and Suresh, for raising me to become the man that I'm today. I would not have reached this feat without their endless support and the hope that they had in me.

Thirdly, a special thanks to my friend Sathish in helping me to proof-read my report. Apart from this, whether or not he liked listening to my work, he always lend his ears. In addition, I would like to extend my thanks to Vidhya Rohini and her friends for not only hosting me in Germany but also for being a great support.

Finally, I would like to thank my other friends, Naveen, Srinath, and Navdeep, for our movie nights and the support they provided me during my hard times in the last two years.

Delft, University of Technology — N.K. Suresh Kumar
September 12, 2018

This is for you, Amma and Appa.

Chapter 1

Introduction

The chapter introduces the thesis by presenting its significance in the present-day world, background information on the topic at hand, and what this research adds to the current research in the field. Firstly, the reasons for carrying out the thesis are laid out in Section 1-1. Secondly, in Section 1-2, background information on wind turbine wakes and wind farm control are presented. Then, the objectives of the thesis is outlined in Section 1-3. Finally, the chapter concludes with the report structure in Section 1-4.

1-1 Motivation

Energy has been a majorly debated topic since the dawn of industrial revolution. Without a doubt, energy plays a vital role in building the economy of a nation and its orderly functioning. This is because energy is indispensable for the functioning of industries, transportation, and other national income yielding sectors [11]. According to a statistics from the International Energy Agency (IEA), energy is not only needed for the aforementioned nation-building sectors but also for the building sector[1] [1]. From the data shown in fig. 1-1, it can be seen that the building sector alone accounts for about 35% of the world's energy consumption. Consequently, energy is truly an indispensable component of the present-day world.

Due to the indispensable nature of energy and the ever-expanding world population, the world's energy consumption is alarmingly increasing with no prospect of slowing down in the future. In fact, since 1850, it has been increasing with a stable annual growth rate of 2.4% [12]. To match this never-ending energy demand, history has seen the development of numerous energy resources. It all started with wood (biomass). As the years passed, with the invention of the steam engine by Watt in 1769 and with the prices of wood-fuel hiking up [13], coal replaced wood as the primary energy source in the 1780s. This period of the history between 1760 to 1840 is referred to as the first industrial revolution and the machinery of this period was mainly fueled by coal, like steam-powered railway and ships [13]. Furthermore,

[1]Generally, end-use energy consumption of the building sector constitutes of space heating, water heating, space cooling, lighting, cooking, appliances and other equipment [1].

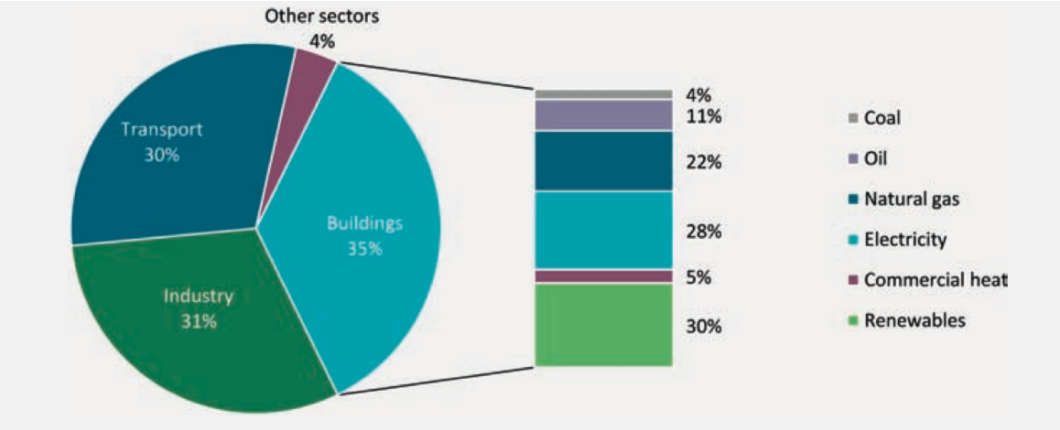


Figure 1-1: Energy consumption by various sectors in 2010. Source: IEA [1]

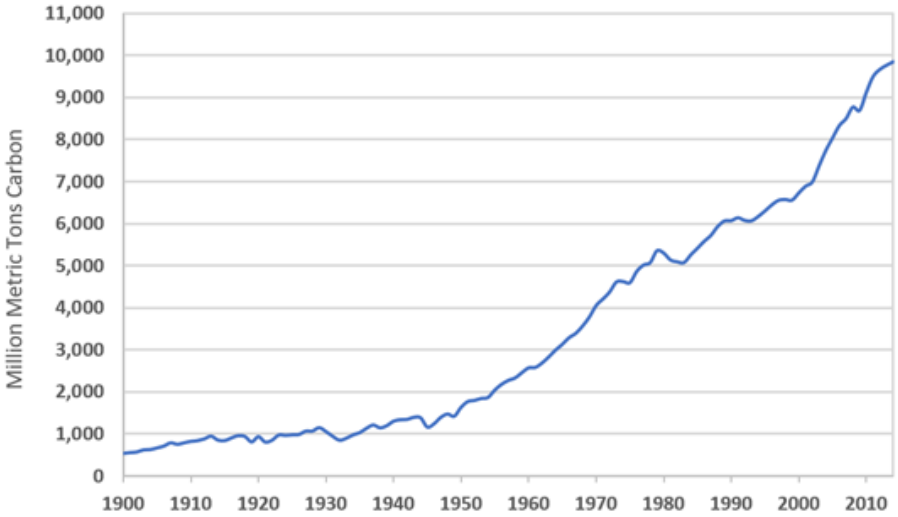


Figure 1-2: World carbon emission from fossil fuels. Source: Carbon Dioxide Information Analysis Center [2]

the first ever coal-based power plant was built in 1875 at France [14]. Later, during the early 19^{th} century, the demand for coal was so high that it raised concerns about coal scarcity [13]. However, advancements in refining and other new technological developments maintained a constant supply of coal with the help of new energy resources, such as oil and gas. With the invention of internal combustion engine [13], consumption of oil and gas started to increase substantially. By the year 1965, oil and gas accounted for more than 50% of the primary energy mix [14]. In a recent study by the United States Energy Information Administration (EIA) [15], coal, oil and gas are predicted to still account for 77% of the world's energy consumption in 2040. Hence, coal, oil and gas, collectively called fossil fuels, had been the primary energy source in the past and will continue to exist even in the near future.

Though fossil fuels are easily accessible and efficient (low cost per unit energy), they are problematic because of their side effects. Firstly, fossil fuels are not renewable within our lifetime. In other words, all the present-day fossil fuels were formed in the prehistoric period.

Thus, at the current rate of consumption, fossil fuels will eventually run out sometime in the future. However, there have been significant differences in the predicted period of decline. Lately, less attention is being given to the decline of fossil fuels [16]. The second and the most significant side effect of fossil fuels is the emission of carbon dioxide and other greenhouse gas (GHG) when fossil fuels are burned for energy. Naturally, the percentage of carbon dioxide in the atmosphere is controlled by the carbon cycles. In other words, natural processes like photosynthesis absorb the carbon dioxide and regulate the flux between the land and atmosphere [17]. However, as shown by fig. 1-2, since the start of the 19^{th} century, anthropogenic (human-induced) emissions of carbon dioxide have increased substantially, producing more carbon dioxide than what can be absorbed by the natural processes. Eventually, this imbalance has resulted in the increase in atmospheric concentration of GHG, thereby affecting Earth's temperature. Naturally, Earth's temperature is maintained at a habitable level by trapping radiant energy using atmospheric GHG [17]. Owing to the increase in atmospheric concentration of GHG, more radiant energy is getting trapped than the usual amount, resulting in global warming. An assessment by Intergovernmental Panel on Climate Change (IPCC) [18] indicates anthropogenic emissions of GHG as the likely reason for global warming. Some of the adverse effects of global warming include melting of polar ice capes [19], erratic weather patterns, floods [20],[21],[22], droughts [23], and disruption of ecological patterns [24].

A solution to this human-induced climate change is to shift to renewable energy. Like fossil fuels, renewable energy can be used for energy generation in the form of electricity, vehicle fuels and other forms of energy [16]. In general, renewable energy sources include wind, solar, hydroelectricity, biomass, geothermal and ocean thermal [25]. However, this report restricts itself to wind energy. In 2008, shares of wind energy in renewable energy (excluding large hydro and biomass) was 53% [25]. This demonstrates the enormous potential of wind energy in supplying clean energy. Also, wind energy will have a huge role to play to comply with the Paris climate accord [26], where the world leaders from 195 countries pledged to maintain the global warming below 2°C. In order to achieve this feat, wind energy has to overcome its challenges.

The main challenge faced by wind energy is its levelized cost of energy (LCOE).[2] For wind energy to compete with the already-existing conventional energy sources like fossil fuels, its LCOE has to decrease. To achieve this, wind turbines are typically placed together as wind farms. This helps in reducing the installation and maintenance costs, visual pollution and land usage. However, stationing the wind turbines in close proximity introduces aerodynamic interactions between the turbines [28] as can be seen in fig. 1-3. This decreases the power production and increases the structural loading on the downstream turbines due to the increased turbulence in the downstream flow. Hence, a wind farm will under-perform when compared to an equal number of individually deployed turbines [29], thereby taking its toll on the LCOE of wind energy.

From the control system perspective, the depreciating effects of aerodynamic interactions on a wind farm can be taken care by designing a wind farm controller. Wind farm control aims to operate the turbines in a wind farm at their collective optimal settings, instead of optimizing them individually. This increases the overall power production of a wind farm, thereby

[2]The LCOE is a quantitative measure which is used for assessing the cost of power generating sources. Usually, it includes all the investment needed by a power generating source since its inception: initial investment, operating and maintenance, and the cost of total power produced over its lifetime [27].

Figure 1-3: Horns Rev 3 is an offshore wind farm situated near Denmark. Turbulence introduced as a result of aerodynamic interactions between the wind turbines can be clearly seen. Source: Vattenfall [3]

reducing the LCOE. Furthermore, wind farm control can be materialized by modelling aerodynamic interactions using control-oriented wind farm models. Unfortunately, low-fidelity models can only be employed for real-time control, as the Navier-Stokes (NS) based high-fidelity models are computationally costly.[3] Hence, a balance has to be struck between the accuracy and real-time applicability of the control-oriented wind farm models.

1-2 Background and State-of-the-Art

In the previous section, the interactions between the turbines were introduced as a reason for the decrease in the LCOE of wind energy, and wind farm control was provided as a solution. While in this section, the form in which the interactions occur (wake) and its properties are outlined. Furthermore, a top-level introduction on wind farm control is presented here.

1-2-1 Wind Turbine Wakes

In the process of energy extraction, the properties of the free-stream flow alter on interacting with the turbines. These altered wind flow is termed as the wake, and its characteristics are [4]:

- Reduced mean flow velocity.
- Widening of the cross-sectional area of the wake in the downstream flow: Generally, according to the principle of conservation of mass, the mass flow rate has to be constant before and after interacting with the turbines. On assuming the flow to be incompressible, the density of the air does not vary. Owing to this, the mass of the flow also remains constant. Hence, to balance the decrease in the downstream velocity, the cross-sectional area of the downstream flow increases.

[3] For real-time control, the computation time taken by the controller, based on a wind farm model, must be less than the simulation time, i.e., the collective computation time taken by the model for predicting the flow and the controller must be less than the simulation time.

- Increase in the intensity of the turbulent flow. This is caused as a result of flow interacting with the rotor blades, tower and nacelle.
- Horizontal and vertical oscillating movements of the wake. This effect is called as wake meandering, and it is possibly caused by the perturbation created by the large-scale turbulent structures in the atmosphere and the usual formation and shedding of vortices at the rotor blades [4].

The magnitude of wake interaction depends on the area that is being overlapped by the upstream turbines' wake and the downstream turbines' rotor, the distance between the turbines [4], and the farm's topology [30].

1-2-2 Wind Farm Control

Typically, the objectives of wind farm control are

- to maximize the overall power production in a wind farm, and
- to minimize the structural loading experienced by the turbines in a wind farm.

Currently, wind turbines are controlled individually, following a greedy-control approach. In this approach, each turbine is operated to maximize their own power production. This might help the upstream turbines in maximizing their power capture, but will affect the power production of the downstream turbines as they will be left with less power (velocity of the wind flow reduces in the downstream) to tap on. Consequently, the overall power captured in a wind farm will be less than what an equal number of individually-placed turbines can produce [29].

This issue can, however, be solved by controlling the turbines as a whole, rather than individually. This is what is practiced in wind farm control, and the two important control strategies are

1. **Axial-induction control (AIC)**
   In induction control [31], the power captured by the upstream turbines are deliberately scaled-down so that more power will be left for the downstream turbines to capture. Thus, the overall power production in a wind farm can be increased. However, for the control to be worthwhile, the increase in power capture in the downstream have to be more than the power lost in the upstream.
2. **Wake redirection control (WRC)**
   In redirection control [32], the upstream turbines are voluntarily misaligned with the direction of the incoming wind flow. Thus, the overlapping area between the wake and the downstream turbines will reduce. As a result, the overall power capture in a wind farm increases. Figure 1-4 illustrates the working of WRC.

Typically in research, model-based wind farm control is employed in an open-loop setting to achieve the control objectives of increasing the overall power production and decreasing the structural loading. Figure 1-5 shows the control scheme of open-loop model-based wind farm

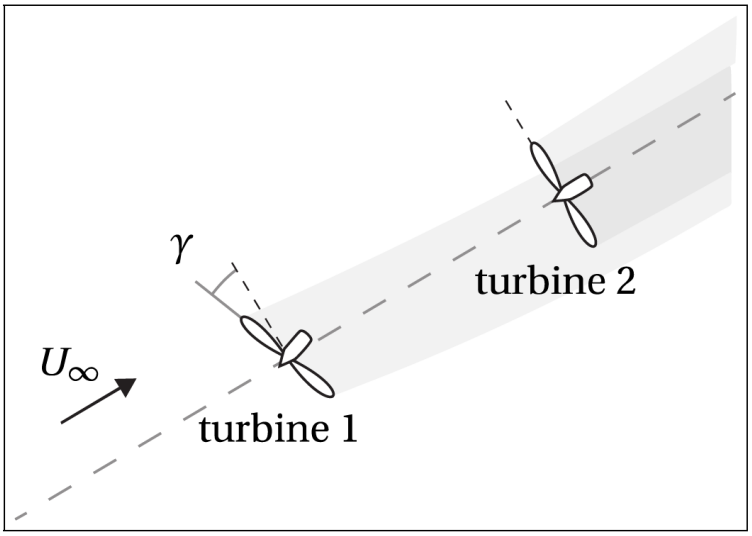


Figure 1-4: Wake redirection control [4], where the upstream turbine, turbine 1, is deliberately mis-aligned to reduce the overlapping area between the upstream wake and the downstream turbine, turbine 2. Wake is shown in gray.

control. As it can be seen, first, an open-loop wind farm model predicts the velocity of the wind flow. Then, based on the velocity of the wind flow, the control signals are generated by a controller. Now, for a controller to achieve the desired level of power production, a wind farm model has to fulfill the following two requirements:

- Accuracy: For instance, in the case of induction control, the amount of reduction in the power capture in the upstream turbines are selected based on the predicted velocity, and the control is worthwhile if and only if the decrease in the power capture in the upstream turbines are less than the increase in the power capture in the downstream turbines [33]. If a wind farm model is not accurate enough, i.e, if the predicted velocity deviates too much from the actual velocity, the increase in the downstream power production will be less than the decrease in the power captured by the upstream, derated turbines. This will result in the reduction of the overall power production in a wind farm. Hence, a wind farm model has to accurately predict the wind flow in a wind farm.
- Computational tractability: Generally, real-time control is desired, and it is achievable if and only if the time taken for computing the control signals, at each iteration, is less than the sampling time. In this thesis, the WindFarmSimulator (WFSim) model (explained in the next chapter) is sampled at 1 Hz. Hence, at each iteration, the control signals have to be generated within a second. As a result, a wind farm model, used for model-based control, has to be computational tractability.

Basically, wind flow is a three dimensional time-varying phenomenon. To accurately capture it, a wind farm model has to predict the dynamic wind flow in all three spatial dimensions over time. High-fidelity models, like Simulator fOr Wind Farm Applications (SOWFA) and Parallelized Large Eddy Simulation Model (PALM), predict the dynamic wind flow to a high precision even under varying atmospheric conditions. However, these models can not be used for the purpose of real-time control, as a few minutes of simulation in these models can take

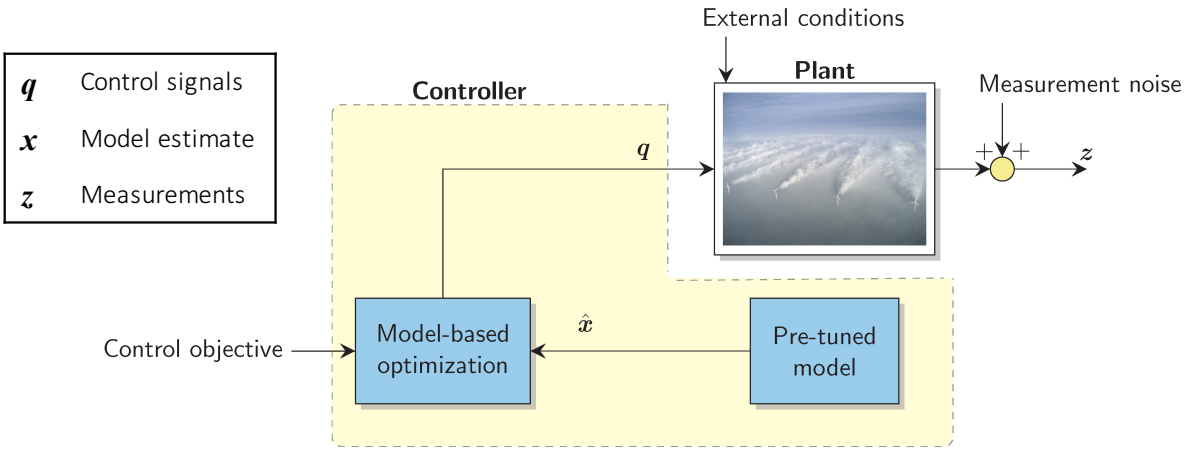


Figure 1-5: Open-Loop Model-Based Wind Farm Control [5]

days even on a supercomputer [34]. Meanwhile, medium-fidelity models, like WFSim, are comparatively very fast. However, these models fail to accurately predict the dynamic wind flow under varying atmospheric conditions. Thus, a closed-loop approach is needed to address this problem.

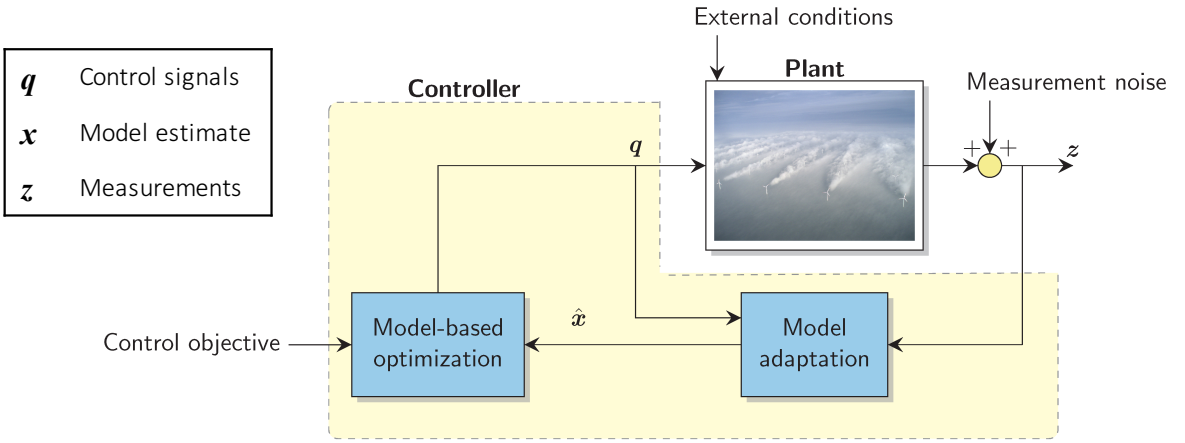


Figure 1-6: Closed-Loop Model-Based Wind Farm Control [5]

Figure 1-6 shows the control scheme of closed-loop model-based wind farm control. As it can be seen, in a closed-loop approach, real-time sensor observations are used to update the estimates of an open-loop model. This would help the open-loop model in adapting to the changing environmental conditions, thereby reducing the deviation between the actual and predicted velocity of the wind flow. Hence, a control algorithm will be able to work to its maximum potential, and the desired control objectives can be reached.

1-3 Thesis Objectives

Model adaptation plays an integral role in the closed-loop approach, and an estimator is used for this purpose. An estimator works in close co-operation with an open-loop wind farm model to embed the sensor observations with the estimates of the open-loop model.

In the current research [35, 36], centralized estimation approach is followed. In the centralized framework, a single estimator is used for estimating all the states (velocity of the wind flow) in a wind farm. The problem, here, is the size of the wind farm models, and the computational complexity associated with it. In this thesis, distributed architecture is proposed as a solution to this problem, and four distributed architectures are devised, using the medium-fidelity wind farm model WFSim.

Distributed architecture 1, like the centralized architecture, defines a single wind farm model for predicting the wind flow throughout a wind farm. For model adaptation, a number of estimators are employed to collectively estimate all the states in a wind farm, as opposed to employing a single estimator in the centralized architecture.

Distributed architecture 2 defines a wind farm model for each turbine, with each model predicting the wind flow throughout a wind farm. In addition, each wind farm model includes only one turbine.

Distributed architecture 3 is a blend of architectures 1 and 2. For predicting the wind flow, architecture 2 is employed to define the wind farm models. For model adaptation, architecture 1 is employed to design the estimators.

Distributed architecture 4 distributes a wind farm into a number of small spatial domains (subsystems), and defines a wind farm model for each of these subsystems to independently predict the wind flow in their respective spatial domain. For model adaptation, each subsystem employs an estimator to independently estimate their respective states, in parallel.

This gives rise to the objective of the thesis, which is
"With the proposed distributed architectures, can the accuracy of a wind farm model be improved while maintaining computational tractability?"

This research objective can be formulated into the following sub-objectives:

1. How often should the WFSim model be linearized for employing the Extended Kalman Filter (ExKF) in both the centralized and distributed frameworks? Can the frequency of linearization be reduced?
2. What are the effects of applying localization on the centralized filters?[4] How does the localization size affect the performance?
3. How small or big should the size of the subsystems in the distributed architectures be?
4. What happens when no measurements or noisy measurements are available? In this case, how can the interactions be taken into account?

The reason for choosing each of the aforementioned sub-objectives will be motivated, along with their solutions, in the report.

[4] Localization is employed to restrict the number of states that are estimated.

1-4 Structure of the Report

The structure of the report is as follows: In Chapter 2, the wind farm model used in this thesis, WFSim, is discussed elaborately. This chapter also gives a brief overview on the high-fidelity wind farm model used in this thesis, SOWFA. Then, in Chapter 3, a brief overview on already-existing centralized estimation algorithms, and an in-depth explanation on the newly-devised distributed architectures are presented. In Chapter 4, simulation results and the answers to the above framed research questions are outlined. Conclusions for this research work are discussed in Chapter 5, along with future recommendations.

In the appendix, a brief overview on wind farm modelling is outlined in Appendix A. In Appendix B, the basics of the Kalman Filter (KF) and its non-linear variants are explained. Then, an alternate method to measurement-based data fusion for distributed architecture 4 is presented in Appendix C. Finally, the report concludes with a list of references, and list of acronyms.

Chapter 2

Wind Farm Model

The chapter focuses on WindFarmSimulator (WFSim), a medium-fidelity wind farm model. This model is used in this thesis for designing estimators for the purpose of closed-loop wind farm control, and will be explained in Chapter 3. This chapter starts by explaining the components of WFSim in Section 2-1. In Section 2-2, the meshing and boundary conditions used to both spatially and temporally discretize the WFSim equations are outlined. In Section 2-3, the WFSim equations are linearized. Finally, a brief overview on Simulator fOr Wind Farm Applications (SOWFA), a high-fidelity wind farm model which is used in this thesis for testing the performances of the estimators, is presented in Section 2-4. For more information on the basics of wind farm modelling, the readers are referred to Appendix A.

2-1 WFSim

WFSim is a Large Eddy Simulation (LES)-based wind farm model. It is developed for control purposes at Delft University of Technology (TU Delft) [7, 6]. In this thesis, it is employed for the following reasons:

1. WFSim predicts the flow velocity in the entire wind farm, using the Navier-Stokes (NS) equations. Hence, it is a medium-fidelity model.
2. WFSim simplifies the three-dimensional (3D) flow, by neglecting flow in the vertical direction z, thereby predicting the two-dimensional (2D) flow at the hub height. Hence, the computational complexity is largely reduced.

Generally, a wind farm model, like WFSim, consists of three components:

1. Flow model: It is used to predict the flow velocity throughout a wind farm. In WFSim, 2D NS equations are used for this purpose.

2. Turbulence model: In WFSim, flow model will only resolve large-scale eddies, as WFSim follows the LES approach (refer Appendix A). Hence, a turbulence model is employed to account for wake recovery. In WFSim, Prandtl's mixing-length model is used.

3. Rotor model: It is used to describe the aerodynamic forces between the flow and rotors. In WFSim, the Actuator Disk Model (ADM) is used.

In the remainder of the section, all the relevant equations in the WFSim model will be explained.

2-1-1 Flow Model

In WFSim, the dynamic behaviour of the wind flow is predicted in two dimensions, at the height of the hub. More importantly, WFSim assumes the flow to be incompressible, and thus with constant density [6]. The 2D NS equations used in WFSim is given by

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u} + \nabla \cdot \tau + \nabla p - \mathbf{f} = 0, \tag{2-1}$$

$$\nabla \cdot \mathbf{u} = -\frac{\partial v}{\partial y}, \tag{2-2}$$

where Eq. (2-1) and Eq. (2-2) represent the momentum and continuity equations in both longitudinal and lateral directions, respectively. In these equations, x is the longitudinal direction in m, y the lateral direction in m, $\mathbf{u} = \begin{pmatrix} u & v \end{pmatrix}^T$ the velocities in longitudinal and lateral directions in m/s, p the pressure term in Nm^{-2}, $\nabla = \begin{pmatrix} \partial/\partial x & \partial/\partial y \end{pmatrix}^T$ the vector operator that represents the variation of a variable with respect to a 2D space, τ the sub-grid stress, and $\mathbf{f} = \begin{pmatrix} f_x & f_y \end{pmatrix}^T$ represents the aerodynamic forces exerted on the flow by the rotors.

2-1-2 Turbulence Model

There are numerous methods available for modelling the sub-grid stress. Each method will represent stress to a different level of accuracy. However, the idea behind WFSim is computational tractability, for the purpose of real-time control. Moreover, the accuracy of the open-loop WFSim model can be improved by employing feedback (discussed in the next chapter). Hence, Prandtl's mixing length model is employed in WFSim, as it is a zero-order model and is relatively simple [6].

Mixing length model follows the Boussinesq approximation, and represents the sub-grid stress in terms of the mean velocity gradients. It is given as

$$\tau = -\frac{1}{2}\nu_t\Big(\nabla \mathbf{u} + (\nabla \mathbf{u})^T\Big), \tag{2-3}$$

where ν is the eddy viscosity, and it is modelled as

$$\nu_t = \ell_u(x,y)^2\left|\frac{\partial u}{\partial y}\right|, \tag{2-4}$$

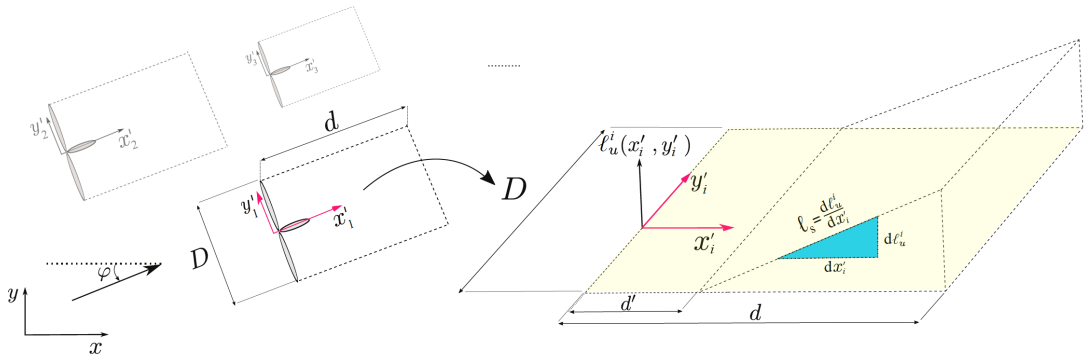


Figure 2-1: Spatially varying mixing length parameter is employed in WFSim to model the turbulence. [6]

where $\ell_u(x, y)$ is the mixing length. Mixing length can be defined for each position in the wind farm model. However, this will increase the computational complexity, as there will be number of tuning variables [6]. Hence, spatial parameterization is applied for the mixing length, and it is given as

$$\ell_u(x,y) = \begin{cases} G(x_i', y_i') * \ell_u^i(x_i', y_i'), & if x \in \mathcal{X} \text{ and } y \in \mathcal{Y}, \\ 0, & \text{otherwise}, \end{cases}, \tag{2-5}$$

where $G(x_i', y_i')$ is a spatial domain filter, and $*$ the 2D spatial convolution operator. See Figure 2-1 for the schematic diagram. Then, $\mathcal{X}$ and $\mathcal{Y}$ represent the spatial domain behind the rotor for which the turbulence model is applied, and is given by

$$\begin{aligned} \mathcal{X} &= \{x : x_i' \leq x \leq x_i' + \cos(\phi) \cdot d\}, \\ \mathcal{Y} &= \{y : y_i' - \frac{D}{2} + \sin(\phi) \cdot x_i' \leq y \leq y_i' + \frac{D}{2} + \sin(\phi) \cdot x_i'\}, \end{aligned} \tag{2-6}$$

where (x_i', y_i') is a new co-ordinate system, with its center at the turbine rotor, D the rotor diameter, ϕ the direction of the mean wind flow in the original (x, y) co-ordinates, and d a length parameter. Then, $\ell_u^i(x_i', y_i')$ is given by

$$\ell_u^i(x_i', y_i') = \begin{cases} (x_i' - d_i')\ell_s, & d' \leq x_i' \leq d \text{ and } -\frac{D}{2} \leq y_i' \leq \frac{D}{2}, \\ 0, & \text{otherwise}, \end{cases}, \tag{2-7}$$

where ℓ_s is the slope of the mixing length, and d' an additional length parameter. From Eq. (2-5) to Eq. (2-7), it can concluded that the spatial parameterization reduces the number of tuning parameters drastically, resulting in only three parameters, namely the length parameters d and d', and the slope of the mixing length ℓ_s.

2-1-3 Rotor Model

In WFSim, the ADM is employed to describe the aerodynamic forces exerted by the rotors on the flow in a wind farm. Figure 2-2 depicts the schematic of a wind turbine interacting

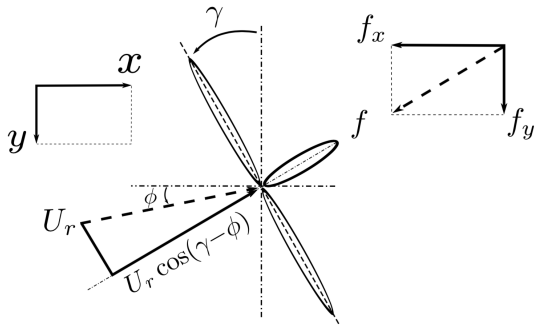


Figure 2-2: Schematic representation of a turbine, yawed at an angle of γ, interacting with the wind flowing at an angle of ϕ with the x-axis. U_r is the velocity of the rotor, and it is same as the velocity of the disk, U_D, in the ADM. [7]

with the wind flow. According to the ADM (see Appendix A), the force term is given as

$$\mathbf{f} = \frac{1}{2}\rho A_r C_T U_\infty^2, \tag{2-8}$$

where ρ is the density of the air, U_∞ the velocity of the upstream wind, and A_r the area swept by the rotor (same as A_D in Appendix A-3-1). C_T is the thrust co-efficient, and it is defined as $C_T = 4a(1-a)$ with a the axial induction factor.

Unfortunately, U_∞ is not well defined. Hence, it is replaced with the velocity of the rotor, U_r. This relationship is given as

$$U_\infty = \frac{U_r \cos(\gamma - \phi)}{1-a}, \tag{2-9}$$

$$U_r = \sqrt{u^2 + v^2}, \tag{2-10}$$

where γ is the wind direction, and ϕ the turbine yaw angle.

Substituting Eq. (2-9) in Eq. (2-8) yields the following relation for the force:

$$\mathbf{f} = \frac{1}{2}\rho A_r C_T' [U_r \cos(\gamma - \phi)]^2, \tag{2-11}$$

where $C_T' = C_T/(1-a)^2$.

Resolving the force, $\mathbf{f}$, into $x-$ and $y-$direction yields

$$f_x = \mathbf{f}\cos(\gamma), \tag{2-12}$$

$$f_y = \mathbf{f}\sin(\gamma). \tag{2-13}$$

2-2 Discretization of WFSim

WFSim becomes complete on substituting the turbulence, given by Eq. (2-3), and rotor models, given by Eq. (2-11), into the flow model, given by Eq. (2-1) and Eq. (2-2). The next step is solving these WFSim equations, to predict the velocity of the wind flow. Before solving

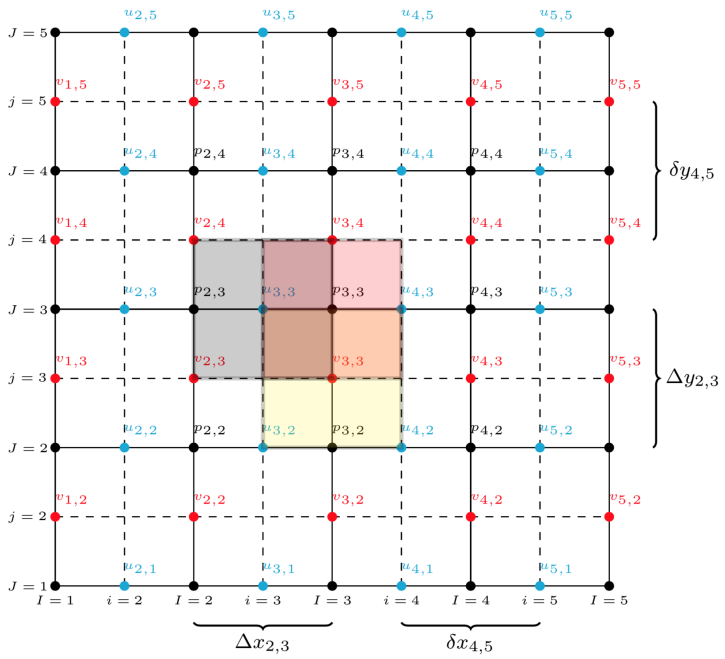


Figure 2-3: Example of a mesh with linear spacing. Solid black lines represent the first mesh, while the dashed black lines represent the second mesh. Longitudinal velocity components are computed at the blue dots, lateral velocity components are computed at the red dots, and pressure terms are computed at the black dots. [6]

the WFSim equations, an important property of the WFSim model is that it is spatial-cum-time dependent system. Hence, the WFSim equations have to solved not only in time but also in space. Furthermore, an analytical solution is not possible. Hence, the WFSim equations have to be discretized in both time and space.

The first step in discretization is defining a mesh. Then, the boundary conditions must be defined, as the NS equations can not be computed at the boundary layers. Finally, the WFSim equations are discretized. These steps are elaborated, next.

2-2-1 Meshing

For discretization, WFSim employs linear-spaced meshing. In linear spacing, grids are spaced at equidistant from each other, resulting in cells of equal size. An example of linear spacing is shown in Figure 2-3. As it can be seen in the example meshing, the WFSim equations will be evaluated only at the vertices of the cells. In other words, the longitudinal and lateral velocity components, and pressure terms will be predicted only at the two meshes, instead of computing them at all the points in a 2D wind farm. Hence, a solution is feasible for the NS equations.

Now, evaluating Eq. (2-1) and Eq. (2-2) on the vertices of each cell yields a non-linear descriptor model, and it is given as

$$E(x_{k-1})x_k = Ax_{k-1} + b(x_{k-1}, w_k), \tag{2-14}$$

where w_k is the input vector, and x_k the state vector. The state vector is obtained by stacking the velocity and pressure vectors together. This is given by

$$x_k = \begin{bmatrix} u_k \\ v_k \\ p_k \end{bmatrix},$$

where u_k, v_k and p_k are the vectors of longitudinal and lateral velocity components, and pressure terms, respectively. These vectors are given by

$$u_k = \begin{bmatrix} u_{3,2} \\ \vdots \\ u_{3,N_y-1} \\ u_{4,2} \\ \vdots \\ u_{4,N_y-1} \\ \vdots \\ u_{N_x-1,2} \\ \vdots \\ u_{N_x-1,N_y-1} \end{bmatrix}, v_k = \begin{bmatrix} v_{2,3} \\ \vdots \\ v_{2,N_y-1} \\ v_{3,3} \\ \vdots \\ v_{3,N_y-1} \\ \vdots \\ v_{N_x-1,3} \\ \vdots \\ v_{N_x-1,N_y-1} \end{bmatrix}, p_k = \begin{bmatrix} p_{2,2} \\ \vdots \\ p_{2,N_y-1} \\ p_{3,2} \\ \vdots \\ p_{3,N_y-1} \\ \vdots \\ p_{N_x-1,3} \\ \vdots \\ p_{N_x-1,N_y-2} \end{bmatrix}.$$

On closely observing the terms in these vectors, it can be inferred that not all the velocity and pressure terms are included. To be exact, the velocity and pressure terms on the boundary grids are absent because the NS equations can not be evaluated at these boundary grids. Hence, boundary conditions are specified for these terms.

Furthermore, in Eq. (2-14), E is the descriptor matrix, and it is a function of the state vector x_k. While A is the system matrix, and it is a constant. b is the input matrix, and it is a function of state x_k and input w_k.

Here, these system matrices are not explained in detail, as the goal of the thesis and this document is estimator design. However, for more information on Eq. (2-14), the reader is referred to the article by Boersma [6].

2-2-2 Boundary Conditions

As explained earlier, boundary conditions have to be defined to ensure the existence of a solution for the NS equations. For defining the boundary conditions, the direction of the wind flow is important. A zero flux boundary condition is assumed for the longitudinal and lateral velocity components on all the boundary grids except the ones on the side of the inflow. While, Dirichlet boundary conditions are assumed for the velocity components on the side of the inflow. This results in the following conditions:

$$\begin{aligned} u_{2,J} &= u_\infty \quad \text{for } J = 1, 2, \ldots, N_y, & v_{i,j} &= v_\infty \quad \text{for } j = 2, 3, \ldots, N_y, \\ u_{i,N_y} &= u_{i,N_y-1} \quad \text{for } i = 3, 4, \ldots, N_x, & v_{I,N_y} &= v_{I,N_y-1} \quad \text{for } i = 2, 3, \ldots, N_x, \\ u_{i,1} &= u_{i,2} \quad \text{for } i = 3, 4, \ldots, N_x, & v_{I,2} &= v_{I,3} \quad \text{for } I = 2, 3, \ldots, N_x, \\ u_{N_x,J} &= u_{N_x-1,J} \quad \text{for } i = 2, 3, \ldots, N_x - 1, & v_{N_x,j} &= v_{N_x-1,j} \quad \text{for } i = 3, 4, \ldots, N_y - 1, \end{aligned}$$

where N_x and N_y are the number of grid points in $x-$ and $y-$direction, respectively.

2-3 Linearization of WFSim

As explained in the previous section, the goal of the thesis is estimator design. The Extended Kalman Filter (ExKF) is the estimator that is predominately used in this thesis, and a linear model is required for applying the ExKF. Hence, the non-linear descriptor WFSim model, given by Eq. (2-14), is linearized in this section.

A non-linear system is linearized at a linearization point, which is usually the states and control inputs of a system. Thus, linearizing Eq. (2-14) around a certain linearization point (x_k^0, w_k^0) yields a linear descriptor model, and it is given as

$$\begin{aligned} \underbrace{E(x_k^0)}_{\bar{E}} \delta x_k &= \underbrace{\left(A + \left.\frac{\partial b(x_{k-1}, w_k)}{\partial x_{k-1}}\right|_{x_k^0, w_k^0} - \left.\frac{\partial E(x_{k-1})}{\partial x_{k-1}}\right|_{x_k^0, w_k^0} x_k \right)}_{\bar{A}} \delta x_{k-1} \\ &\quad + \underbrace{\left.\frac{\partial b(q_{k-1}, w_k)}{\partial w_k}\right|_{x_k^0, w_k^0}}_{\bar{B}} \delta w_k, \end{aligned} \tag{2-15}$$

where $\delta x_k = \begin{bmatrix} \delta {u_k}^T & \delta {v_k}^T & \delta {p_k}^T \end{bmatrix}^T$ is the change of state x_k, with $\delta u_k = u_k - u_k^0$, $\delta v_k = v_k - v_k^0$ and $\delta p_k = p_k - p_k^0$.

The linearized equation given by Eq. (2-15) is in the descriptor form. A linear model in the standard state-space form for WFSim is obtained by re-arranging the equation, and it is given by

$$\delta x_k = \underbrace{\bar{E}^{-1}\bar{A}}_{F_k} \delta x_{k-1} + \underbrace{\bar{E}^{-1}\bar{B}}_{G_k} \delta w_k. \tag{2-16}$$

2-4 SOWFA

SOWFA is a high-fidelity wind farm model from National Renewable Energy Laboratory (NREL). In this thesis, SOWFA is used for testing the performance of the estimation algorithms. Here, a brief overview of the model is presented. For a more detailed information on the model, the reader is referred to the article by Churchfield *et al.* [30].

SOWFA predicts the velocity of the wind flow in all three dimensions using the 3D incompressible NS equations. For flow modeling, SOWFA follows a LES approach. In the LES approach, large-scale flow dynamics are only resolved, while small-scale dynamics are modeled using a sub-grid scale model (see Appendix A).

For rotor modeling, SOWFA employs the Acutator Line Model (ALM). The ALM is more sophisticated than the ADM, which is used in WFSim. Furthermore, unlike WFSim, SOWFA describes the effect of structural loading, on the turbine structure, using a turbine model. The Fatigue, Aerodynamics, Structures and Turbulence (FAST) model from NREL is used for turbine modelling [37].

Finally, the reason for using SOWFA data over a real plant data is that SOWFA provides highly accurate flow data at a fraction of the cost of field tests.

Chapter 3

Estimator Design

The need for a closed-loop approach for the purpose of real-time control is re-iterated here, to stress its importance. High-fidelity models, like Simulator fOr Wind Farm Applications (SOWFA) and Parallelized Large Eddy Simulation Model (PALM), predict the dynamic wind flow to a high precision even under varying atmospheric conditions. However, these models can not be used for the purpose of real-time control, as a few minutes of simulation in these models can take days even on a supercomputer [34]. Meanwhile, medium-fidelity models, like WindFarmSimulator (WFSim), are comparatively very fast. However, these models fail to accurately predict the dynamic wind flow under varying atmospheric conditions. Thus, a closed-loop approach is followed to address this problem.

The process of designing a closed-loop controller can be summarized as a three-stage process, namely system modelling (open-loop), estimator design and controller design. However, the third stage (controller design) is beyond the scope of the thesis, and even though the thesis is concerned with the first two stages, it focuses exclusively on the estimator design part.

Nevertheless, depending upon the information flow between these three stages, there can be two architectures, namely the centralized and distributed architectures. In the centralized architecture, a single model, estimator and controller is designed for the entire system. On the other hand, in the distributed architecture, more than one system model, estimator and controller are employed to respectively model, estimate and control the entire system.

Generally, medium- and high-fidelity wind farm models are spatial-cum-time dependent systems. Hence, a spatial domain, over which a wind farm is defined, is very important for explaining as well as differentiating the various closed-loop architectures. Thus, the wind farm architectures will be explained with the help of a 2-turbine wind farm shown in fig. 3-1.

In this chapter, various methods of designing the estimators, by following centralized and distributed architectures, are explained in detail. The chapter starts by introducing the centralized estimation architecture in Section 3-1. Then, the various distributed architectures for estimator design are discussed in Section 3-2. Finally, the chapter concludes with a comparison study between the centralized and distributed architectures. For more information on the estimation algorithms as such, the reader is referred to Appendix B.

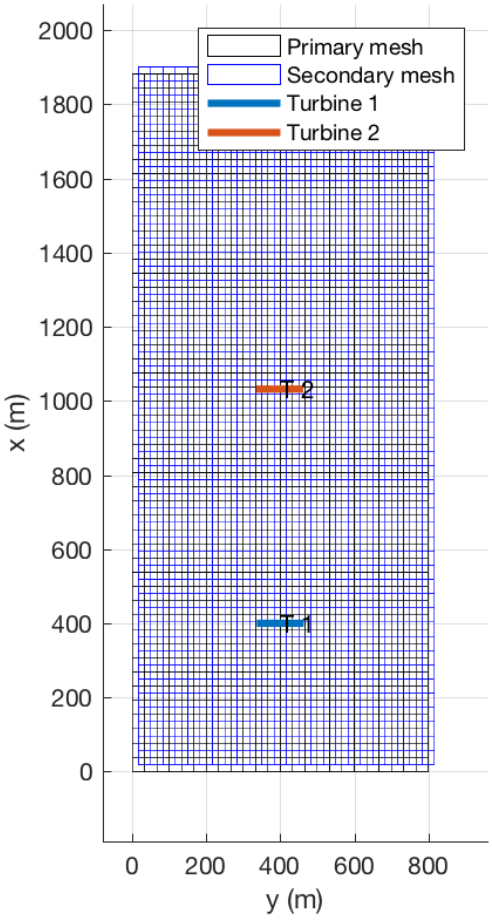


Figure 3-1: Spatial domain of a 2-turbine wind farm following the centralized architecture.

3-1 Centralized Architecture

Typically, a wind farm is described (modelled) as a whole using a single wind farm model, irrespective of the number of turbines in a wind farm and the size of the wind farm [34, 35, 36]. To enhance the accuracy of the wind farm models, a closed-loop approach is followed by designing an estimator for the entire wind farm. This method of describing and estimating a system as a whole is referred as the centralized architecture. For example, a wind farm, of size 1880m×800m with 2 turbines, following the centralized architecture is shown in fig. 3-1. This example uses primary and secondary meshes of 25×50 grids to describe the entire wind farm as a single model.

Some of the advantages of following the centralized architecture are that the models and estimators are easier to design and implement, as only one model and estimator needs to be designed for the entire wind farm. However, this causes the size of the model to increase with the increase in the size of the wind farm, especially for medium- and high-fidelity models. As it can be seen in fig. 3-1, even for a small wind farm, there are 2500 states, which are velocities in two-dimensions. Hence, the centralized architecture makes the wind farm model a large-scale system.

The problem with large-scale systems is their size. The large size of the system makes the computation costly. On employing a regular Kalman Filter (KF), the computational cost increases exponentially, with the increase in the system size. For example, in a large-scale system like fig. 3-1, the Extended Kalman Filter (ExKF) takes approximately 15 seconds for each iteration on a modern ultrabook PC, making it highly unsuitable for real-time applications.[1] Furthermore, on analyzing the steps of the ExKF algorithm, to minimize the computational resources, the most time-consuming operation is the prediction of the state auto co-variance matrix $P_{k\,|\,k-1}$, and is given by

$$P_{k\,|\,k-1} \quad = \quad F_k P_{k-1\,|\,k-1} F_k^T + Q_k, \tag{3-1}$$

where $F_k \in \mathbb{R}^{n\times n}$ is the linearized system matrix at time k, $P_{k-1\,|\,k-1} \in \mathbb{R}^{n\times n}$ the updated state auto co-variance at the previous time step $k-1$, $Q_k \in \mathbb{R}^{n\times n}$ the process noise and n the size of the system.

The computational complexity of the aforementioned operation is $\mathcal{O}(n^3)$. In the case of a small-scale system where the number of states are less, this operation is not time-consuming. However, for the large-scale systems, this operation is extremely time-consuming. This issue related to the computational complexity can be addressed by employing the Ensemble Kalman Filter (EnKF) [34, 35, 36]. The EnKF computes the sample predicted auto co-variance of the state vector, instead of the actual predicted auto co-variance of the state vector given by Eq. (3-1). Hence, the computational cost of the estimation algorithm reduces drastically from $\mathcal{O}(n^3)$ to $\mathcal{O}(nmn)$, where m is the number of ensemble members (samples) and $m \lll n$. In terms of numbers, the EnKF takes approximately half a second for each iteration on a modern ultrabook PC for the 2-turbine case given in Figure 3-1. Nevertheless, control will still be computationally complex, as the size of the system is still large. This issue is taken care in this thesis by employing a distributed architecture. The main idea behind the distributed architectures is to distribute a large-scale system into numerous small-scale systems, which is explained in the upcoming section. Hence, the wind farm models are no longer large, and their computational costs reduce irrespective of the algorithms being used or the processes being carried out.

A top-level introduction on the aforementioned filters is presented in Appendix B. For a more detailed explanation on filtering, please refer the literature study preceding this work [38].

3-2 Distributed Architecture

As mentioned in the previous section, the objective of the distributed architectures is to make the closed-loop computation cheap, irrespective of the estimation and control algorithms being used. On following the distributed architecture, either the estimation or control or both the algorithms can be distributed. Based on the extent to which the states are updated (measurement-update), model distribution, and size of the subsystems, four types of distributed architectures are devised in this thesis:

[1] Generally, estimation is performed every iteration, and the frequency of real-time control depends on how fast the dynamics of a system change. If the state vector of the system does not vary too much for every iteration, the frequency of estimation could be reduced. However, in this thesis, to push the limits of the computational efficiency of the estimators, the real-time control and estimation are employed every iteration, with a sampling frequency of 1Hz. Hence, the combined computation time of an estimator and controller must take less than 1 second.

1. **Distributed architecture 1**
   It is a direct alternative for the centralized EnKF. In this architecture, a wind farm model is not distributed, instead an estimator is distributed. This architecture is implemented by employing a number of estimators to collectively estimate all the states in a wind farm, as opposed to using a single estimator in the centralized architecture. Each estimator in this architecture estimates only part of the state vector. Hence, the computational complexity associated with the state estimation decreases. However, similar to the centralized architecture, control will still be computationally complex, as the model is still large-scale.

2. **Distributed architecture 2**
   In this architecture, a wind farm model is distributed. In other words, a number of wind farm models are employed. Each wind farm model considers only one turbine, and predicts the wind flow throughout a wind farm. Consequently, the complexity of this architecture is larger than the centralized architecture. However, this architecture is still considered here, as it gives useful insights on the effects of considering only one turbine in the wind farm.

3. **Distributed architecture 3**
   This architecture is a computationally efficient version of architecture 2. In architecture 2, an estimator is employed for each wind farm model, to independently estimate all the states in their respective model. Consequently, the computational complexity of each estimator will be equivalent to that of a centralized estimator, making the overall computation very costly. In architecture 3, all the estimators follow architecture 1, i.e., each estimator estimates only part of the state vector that are near their respective turbine. Hence, the computation complexity reduces.

4. **Distributed architecture 4**
   The general idea behind distributed architecture 4 is to create a wind farm model around each turbine, to accurately predict the wind flow in a small spatial domain around their respective turbine. In short, a wind farm model is defined for each turbine, with the spatial domain of the models reduced. Hence, each model is restricted to predict the velocity of the wind flow only around their respective turbine, as opposed to predicting throughout the wind farm in the centralized and rest of the distributed architectures.

More importantly, the difference between the wind farm models employed for the subsystems in distributed architecture 1 and the remaining distributed architectures is: In architecture 1, a centralized wind farm model is employed, i.e., a single wind farm model is defined for the entire wind farm area like the centralized architecture. Then, based on the spatial domain of each subsystem, a model for each subsystem is generated, by considering only the system matrices that belong to the states of their respective subsystem. On the other hand, in the case of the remaining distributed architectures, a spatial domain is, first, defined for each subsystem. Then, based on the spatial domain, a wind farm model is independently defined for each subsystem. In short, a model for each subsystem in architecture 1 is not modelled distributively, but is generated by splitting a centralized wind farm model. Hence, from now on, a subsystem model in architecture 1 will be referred as a sub-model.

A top-level introduction on the distributed architectures was presented in this section. From the upcoming section, a more in-depth explanation on these architectures is given.

3-3 Distributed Architecture 1

Distributed architecture 1, like the centralized architecture, models the the entire wind farm area using a single wind farm model, i.e., it does not distribute a wind farm model. However, unlike the centralized architecture where a single estimator is used to estimate all the states, this architecture estimates only a selected number of states around each turbine. In other words, the states that are far from the turbines are not estimated. Thus, a wind farm model with n states are not required for estimation, instead parts of the system that belong to the estimated states alone can be employed. Hence, the computational cost of the ExKF, especially the most resourceful step given by Eq. (3-1), will reduce exponentially, as the number of estimated states is lower than the actual number of states n.

Wind farm estimation following distributed architecture 1 is implemented by creating a subsystem around each turbine. Thus, instead of using a single estimator for the entire wind farm, this architecture employs an estimator around each turbine (for each subsystem). Once the states of the subsystems are estimated independently, a global state estimate for the entire system is obtained using data fusion algorithms. In short, wind farm estimation following distributed architecture 1 can be considered as a four-stage process, and the stages, in order, are state decomposition, estimator model decomposition, filter design, and state fusion. Next, each of these stages are elaborated.

3-3-1 State Decomposition

As mentioned earlier, a subsystem is created around each turbine. The first step in creating a subsystem is to define a spatial domain for each subsystem. A spatial domain can be a circle or a square, with a turbine at the centre. The dimension of the domain can be specified in terms of rotor diameter D. Figure 3-2 shows the circular spatial domain, of radius 2 rotor diameter ($2D$), created around each turbine for the example wind farm from the centralized architecture, shown in fig. 3-1.

Once a spatial domain for each subsystem is defined, the next step is to determine the states of each subsystem. A state of a wind farm model is considered as a state of a subsystem if and only if that state falls under the spatial domain of that particular subsystem. For example, in fig. 3-2, the yellow-colored stars are considered as the states of the first subsystem, as they fall under the spatial domain of the first subsystem, located around the first turbine. Similarly, the purple-colored stars are considered as the states of the second subsystem, as they fall under the spatial domain of the second subsystem. Furthermore, from fig. 3-2, it is apparent that there are states that are not part of any of the subsystems as they are really far from the turbines. Hence, measurement-update is performed only for the states of the wind farm model that are part of the subsystems, while the remaining states of the wind farm model are only time-propagated.

In short, the states of a wind farm model are decomposed into the estimated (X_{est}) and unestimated states (X_{unest}). Then, the estimated states are further decomposed into the

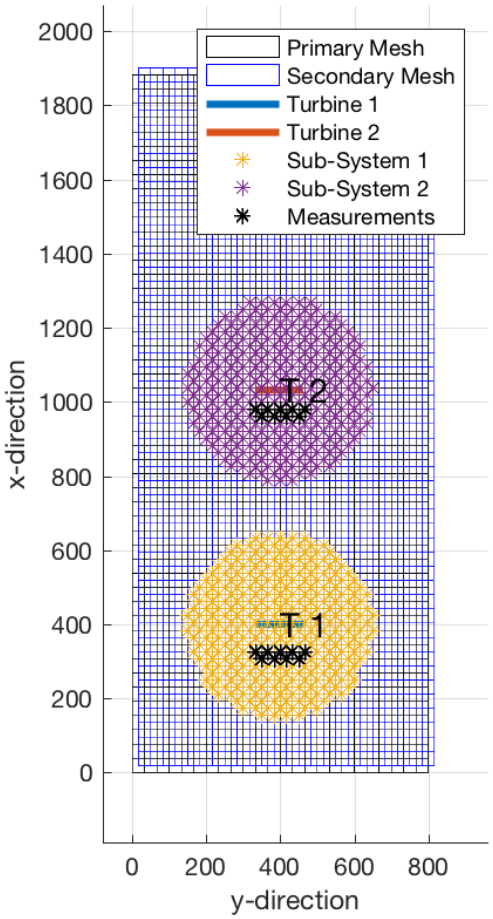


Figure 3-2: States of a 2-turbine wind farm following distributed architecture 1. Black- and blue-colored meshes represent the entire states of the wind farm, and these states are state-propagated. Yellow- and purple-colored stars are the states of the first and second subsystem, respectively, and these states are measurement-updated.

states of the subsystems (X_i). In mathematical notations, this can be represented as

$$x_k = \begin{bmatrix} X_{est,k} \\ X_{unest,k} \end{bmatrix}, \tag{3-2}$$

$$X_{est,k} = \bigcup_{i=1}^{N_{turb}} X_{i,k}, \tag{3-3}$$

$$= \begin{bmatrix} X_{i,k} \\ d_{i,k} \end{bmatrix}, \text{ where } i \in \{1, \ldots, N_{turb}\}, \tag{3-4}$$

where x_k is the state vector of the entire wind farm model at k, $X_{est,k}$ the vector of states that are collectively estimated by all the subsystems at k, and $X_{unest,k}$ the vector of states, at k, that are part of the entire wind farm model but do not belong to any subsystem. $d_{i,k}$, the disturbance state vector, is the vector of states that are estimated by all the subsystems except subsystem i. N_{turb} is the number of subsystems which is equal to the number of turbines, and X_i the vector of states that are included in subsystem i, with $i = 1 \ldots N_{turb}$.

3-3-2 Estimator Model Decomposition

Having decomposed the states of the wind farm model, the next step is to define sub-models for each subsystem so that an estimator can be defined around each turbine. The following model decomposition method is performed assuming that a linearized wind farm model is available at every time step. However, the following decomposition method can be extrapolated to a non-linear model as well.

Consider a linearized wind farm model in the standard state-space form:

$$x_k = Ax_{k-1} + Bu_k, \tag{3-5}$$
$$y_k = Cx_k + Du_k, \tag{3-6}$$

where A, B, C, D are the linearized system matrices at k, x_k and x_{k-1} the state vectors of the wind farm model at k and $k-1$ respectively, u_k the input vector at k, and y_k the output vector.

Following the state decomposition specified by Eq. (3-2), the linearized wind farm model, described by Eq. (3-5) and Eq. (3-6), can be decomposed into linearized sub-models for the estimated and unestimated states, and the decomposed sub-models are given by

$$x_k = \begin{bmatrix} X_{est,k} \\ X_{unest,k} \end{bmatrix}, \tag{3-7}$$
$$\begin{bmatrix} X_{est,k} \\ X_{unest,k} \end{bmatrix} = \begin{bmatrix} A_{est} & A_{est,unest} \\ A_{unest,est} & A_{unest,unest} \end{bmatrix} \begin{bmatrix} X_{est,k-1} \\ X_{unest,k-1} \end{bmatrix} + \begin{bmatrix} B_{est} \\ B_{unest} \end{bmatrix} u_k, \tag{3-8}$$
$$y_k = C \begin{bmatrix} X_{est,k} \\ X_{unest,k} \end{bmatrix} + Du_k, \tag{3-9}$$

where $A = \begin{bmatrix} A_{est} & A_{est,unest} \\ A_{unest,est} & A_{unest,unest} \end{bmatrix}$, and $B = \begin{bmatrix} B_{est} \\ B_{unest} \end{bmatrix}$ are the decomposed system matrices.

Eliminating the state-propagation of the unestimated states from Eq. (3-8) and Eq. (3-9), the linearized model of the estimated states is

$$X_{est,k} = A_{est}X_{est,k-1} + A_{est,unest}X_{unest,k-1} + B_{est}u_k, \tag{3-10}$$
$$y_k = C \begin{bmatrix} X_{est,k} \\ X_{unest,k} \end{bmatrix} + Du_k, \tag{3-11}$$

Further, following Eq. (3-4), the linearized model of the estimated states, Eq. (3-10) and Eq. (3-11), can be decomposed into a linearized model of each subsystem. The resulting linearized model of subsystem i is given by

$$X_{est,k} = \begin{bmatrix} X_{i,k} \\ d_{i,k} \end{bmatrix}, \text{ where } i \in \{1, \ldots, N_{turb}\}, \tag{3-12}$$
$$\begin{bmatrix} X_{i,k} \\ d_{i,k} \end{bmatrix} = \begin{bmatrix} F_i & D_i \\ A^* & B^* \end{bmatrix} \begin{bmatrix} X_{i,k-1} \\ d_{i,k-1} \end{bmatrix} + \begin{bmatrix} E_i \\ C^* \end{bmatrix} X_{unest,k-1} + \begin{bmatrix} G_i \\ D^* \end{bmatrix} u_k, \tag{3-13}$$
$$y_k = \begin{bmatrix} H_i & E^* & F^* \end{bmatrix} \begin{bmatrix} X_{i,k} \\ d_{i,k} \\ X_{unest,k} \end{bmatrix} + Du_k, \tag{3-14}$$

where $d_{i,k}$ is the disturbance state vector, and $A_{est} = \begin{bmatrix} F_i & D_i \\ A^* & B^* \end{bmatrix}$, $A_{est,unest} = \begin{bmatrix} E_i \\ C^* \end{bmatrix}$, $B_{est} = \begin{bmatrix} G_i \\ D^* \end{bmatrix}$ and $C = \begin{bmatrix} H_i & E^* & F^* \end{bmatrix}$ the decomposed system matrices. A^*, B^*, C^*, D^*, E^* and F^* are part of the decomposed system matrices; however, these matrices will not be used from now on.

Finally, on eliminating the time-propagation of the disturbance state vector from Eq. (3-13) and Eq. (3-14), the linearized sub-model of each subsystem is given by

$$X_{i,k} = F_i X_{i,k-1} + D_i d_{i,k-1} + E_i X_{unest,k-1} + G_i u_k, \forall i \in \{1, \dots, N_{turb}\}, \tag{3-15}$$
$$y_{i,k} = H_i X_{i,k} + D_i u_k, \tag{3-16}$$

3-3-3 Filter Design

Once a sub-model for each estimator is obtained, the next step is to independently design an estimator for each subsystem. The ExKF for the linearized model of each subsystem, Eq. (3-15) and Eq. (3-16), is given as

Extended Kalman Filter

1. Time-Propagation

$$\overline{\hat{x}}_{k \mid k-1} = f_k(\overline{\hat{x}}_{k-1 \mid k-1}, u_k, w_k), \tag{3-17}$$
$$\begin{aligned} P_{i,k \mid k-1} = {} & F_i P_{ff,k-1 \mid k-1} F_i^T + F_i P_{fd,k-1 \mid k-1} D_i^T \\ & + \left(F_i P_{fd,k-1 \mid k-1} D_i^T\right)^T + F_i P_{fe,k-1 \mid k-1} E_i^T \\ & + \left(F_i P_{fe,k-1 \mid k-1} E_i^T\right)^T + D_i P_{dd,k-1 \mid k-1} D_i^T \\ & + D_i P_{de,k-1 \mid k-1} E_i^T + \left(D_i P_{de,k-1 \mid k-1} E_i^T\right)^T \\ & + E_i P_{ee,k-1 \mid k-1} E_i^T + Q_{ff,k}, \forall i \in \{1, \dots, N_{turb}\}, \end{aligned} \tag{3-18}$$

2. Measurement-Update (repeated for $\forall i \in \{1, \dots, N_{turb}\}$)

$$\tilde{y}_{i,k} = y_{i,k} - H_{i,k} \overline{\hat{X}}_{i,k \mid k-1}, \tag{3-19}$$
$$P_{i,yy} = R_{i,k} + H_{i,k} P_{i,k|k-1} H_{i,k}^T, \tag{3-20}$$
$$P_{i,xy} = P_{i,k \mid k-1} H_{i,k}^T, \tag{3-21}$$
$$K_{i,k} = P_{i,xy} P_{i,yy}^{-1}, \tag{3-22}$$
$$\overline{\hat{X}}_{i,k \mid k} = \overline{\hat{X}}_{i,k \mid k-1} + K_{i,k} \tilde{y}_{i,k}, \tag{3-23}$$
$$P_{i,k \mid k} = (I - K_{i,k} H_{i,k}) P_{i,k \mid k-1}, \tag{3-24}$$

where $\overline{\hat{x}}_{k-1 \mid k-1}$ is the expected value of the updated state estimate at $k-1$, $\overline{\hat{x}}_{k \mid k-1}$ the expected value of the predicted state estimate at k, $\overline{\hat{X}}_{i,k \mid k}$ is the expected value of the updated state estimate of subsystem i at k, and w_k the process noise at k. $Q_{ff,k}$ is the

part of the process noise co-variance matrix that belongs to the states of the subsystem at k. $P_{k-1\,|\,k-1}$ is the updated co-variance at $k-1$, and is given as

$$P_{k-1\,|\,k-1} = \begin{bmatrix} P_{ff,k-1\,|\,k-1} & P_{fd,k-1\,|\,k-1} & P_{fe,k-1\,|\,k-1} \\ P_{df,k-1\,|\,k-1} & P_{dd,k-1\,|\,k-1} & P_{de,k-1\,|\,k-1} \\ P_{ef,k-1\,|\,k-1} & P_{ed,k-1\,|\,k-1} & P_{ee,k-1\,|\,k-1} \end{bmatrix},$$

with

$$\begin{aligned}
P_{ff,k-1\,|\,k-1} &= E[(\hat{X}_{i,k-1\,|\,k-1} - \overline{\hat{X}}_{i,k-1\,|\,k-1})(\hat{X}_{i,k-1\,|\,k-1} - \overline{\hat{X}}_{i,k-1\,|\,k-1})^T], \\
P_{fd,k-1\,|\,k-1} &= E[(\hat{X}_{i,k-1\,|\,k-1} - \overline{\hat{X}}_{i,k-1\,|\,k-1})(\hat{d}_{i,k-1\,|\,k-1} - \overline{\hat{d}}_{i,k-1\,|\,k-1})^T], \\
P_{fe,k-1\,|\,k-1} &= E[(\hat{X}_{i,k-1\,|\,k-1} - \overline{\hat{X}}_{i,k-1\,|\,k-1})(\hat{X}_{unest,k-1} - \overline{\hat{X}}_{unest,k-1})^T], \\
P_{df,k-1\,|\,k-1} &= E[(\hat{d}_{i,k-1\,|\,k-1} - \overline{\hat{d}}_{i,k-1\,|\,k-1})(\hat{X}_{i,k-1\,|\,k-1} - \overline{\hat{X}}_{i,k-1\,|\,k-1})^T], \\
P_{dd,k-1\,|\,k-1} &= E[(\hat{d}_{i,k-1\,|\,k-1} - \overline{\hat{d}}_{i,k-1\,|\,k-1})(\hat{d}_{i,k-1\,|\,k-1} - \overline{\hat{d}}_{i,k-1\,|\,k-1})^T], \\
P_{de,k-1\,|\,k-1} &= E[(\hat{d}_{i,k-1\,|\,k-1} - \overline{\hat{d}}_{i,k-1\,|\,k-1})(\hat{X}_{unest,k-1} - \overline{\hat{X}}_{unest,k-1})^T], \\
P_{ef,k-1\,|\,k-1} &= E[(\hat{X}_{unest,k-1} - \overline{\hat{X}}_{i,k-1})(\hat{X}_{i,k-1\,|\,k-1} - \overline{\hat{X}}_{i,k-1\,|\,k-1})^T], \\
P_{ed,k-1\,|\,k-1} &= E[(\hat{X}_{unest,k-1} - \overline{\hat{X}}_{unest,k-1})(\hat{d}_{i,k-1\,|\,k-1} - \overline{\hat{d}}_{i,k-1\,|\,k-1})^T], \\
P_{ee,k-1\,|\,k-1} &= E[(\hat{X}_{unest,k-1} - \overline{\hat{X}}_{unest,k-1})(\hat{X}_{unest,k-1} - \overline{\hat{X}}_{unest,k-1})^T],
\end{aligned}$$

where $P_{ff,k-1\,|\,k-1}$, $P_{dd,k-1\,|\,k-1}$ and $P_{ee,k-1\,|\,k-1}$ are the auto co-variances of the states of the subsystems, the disturbance state vector and the unestimated states respectively, and $P_{fd,k-1\,|\,k-1}$, $P_{fe,k-1\,|\,k-1}$, $P_{df,k-1\,|\,k-1}$, $P_{de,k-1\,|\,k-1}$, $P_{ef,k-1\,|\,k-1}$ and $P_{ed,k-1\,|\,k-1}$ the cross co-variance matrices.

Now, on inspecting Eq. (3-18) and Eq. (3-24) closely, it can be inferred that the prediction of the auto co-variance of the state vector (the most resourceful step of the ExKF) and the measurement-update steps of the estimation process are computed distributively by employing a separate estimator, in parallel, for each subsystem. In addition, the computational complexity of the most resourceful step of the ExKF will reduce drastically because the co-variance of the states that fall under the selected spatial domain around the turbines are only computed, while the computation of the co-variance for the states that are far from the turbines are bypassed. In short, the states of the subsystems are independently measurement-updated, while the remaining states are only time-propagated.

Furthermore, in distributed architecture 1, a wind farm is described centrally using a single wind farm model. In other words, distributed architecture 1, like the centralized architecture, solves the Navier-Stokes (NS) equations centrally for the entire wind farm area, resulting in a single wind farm model for the entire wind farm, as given by Eq. (3-17). Consequently, the time-propagation step, at the starting of every iteration, requires a single state vector for the entire wind farm. However, on inspecting Eq. (3-17) and Eq. (3-24) closely, each estimator, at the end of an iteration, yields an estimate of the states of their corresponding subsystem $X_{i,k\,|\,k}$. Hence, a data fusion algorithm is employed to fuse the local state estimates of all the subsystems. This process will yield a single global estimate for the entire state vector if

and only if all the states of the system were estimated. However, in most of the cases like the 2-turbine example, given in Figure 3-2, not all the states are estimated. As a result, data fusion would yield a single vector just for the estimated states. Now, to obtain a single estimate for the entire state vector $x_{k\,|\,k}$, the time-propagated estimate of the unestimated states, $X_{unest,k\,|\,k-1}$, is appended with the fused estimate of the estimated states, obtained from data fusion. On the other hand, the co-variance for the entire state vector, $P_{k\,|\,k}$, is obtained by appending the fused co-variance of the estimated states, the time-propagated co-variance of the unestimated states, and their cross co-variance terms. In mathematical notations, this can be represented as

$$x_{k\,|\,k} = \begin{bmatrix} X_{fus,est,k\,|\,k} \\ X_{unest,k\,|\,k-1} \end{bmatrix}, \tag{3-25}$$

$$P_{k\,|\,k} = \begin{bmatrix} P_{fus,est,k\,|\,k} & P_{est,unest} \\ P_{est,unest}^T & P_{unest,k\,|\,k-1} \end{bmatrix}, \tag{3-26}$$

where $X_{fus,est,k\,|\,k}$ is the fused estimate of the local estimates of all the subsystems, $X_{unest,k\,|\,k-1}$ the time-propagated estimate of the unestimated states, $P_{fus,est,k\,|\,k}$ the fused co-variance of all the subsystems, $P_{unest,k\,|\,k-1}$ the time-propagated co-variance of the unestimated states, and $P_{est,unest}$ the cross co-variance between the estimated and unestimated states.

Unfortunately, Eq. (3-26) cannot be implemented, as the cross co-variance is not unknown. This issue can be addressed by employing a conservative bound of the co-variance, instead of the actual co-variance $P_{k\,|\,k}$. This is given by

$$\underbrace{\begin{bmatrix} \frac{1}{\omega_1} P_{fus,est,k\,|\,k} & 0_{N_{est},N_{unest}} \\ 0_{N_{unest},N_{est}} & \frac{1}{\omega_2} P_{unest,k\,|\,k-1} \end{bmatrix}}_{P^*_{k\,|\,k}} \geq \underbrace{\begin{bmatrix} P_{fus,est,k\,|\,k} & P_{est,unest} \\ P_{est,unest}^T & P_{unest,k\,|\,k-1} \end{bmatrix}}_{P_{k\,|\,k}}, \tag{3-27}$$

where $P^*_{k\,|\,k}$ is the conservative bound of $P_{k\,|\,k}$, ω_1, and ω_2 positive weights, with $\omega_1 + \omega_2 = 1$.

However, in distributed architecture 1, the time-propagated co-variance of the unestimated states is not computed, to reduce the computational cost. Generally, the co-variance is high if the states are not measurement-updated. Hence, a randomly-selected diagonal matrix is used as the co-variance for the unestimated states, with the magnitude of the diagonal entries to be high,[2] and $P_{fus,est,k\,|\,k}$ itself is used as the co-variance of the estimated states. This can be considered as assuming a very low value for ω_2, and a high value for ω_1. In mathematical terms, this is represented as

$$P_{k\,|\,k} \approx \begin{bmatrix} P_{fus,est,k\,|\,k} & 0_{N_{est},N_{unest}} \\ 0_{N_{unest},N_{est}} & rI_{N_{unest}} \end{bmatrix}, \tag{3-28}$$

where r is randomly-selected high value, I identity matrix, 0 zero matrix, and N_{est} and N_{unest} the length of the estimated and unestimated states, respectively.

Hence, on following Eq. (3-25) and Eq. (3-28) at the end of every estimation iteration, a single global estimate for the entire state vector is obtained. It can, then, be used for the time-propagation at the next iteration.

[2]The magnitude is taken such that the estimates do not diverge. In this thesis, r was taken to be 20.

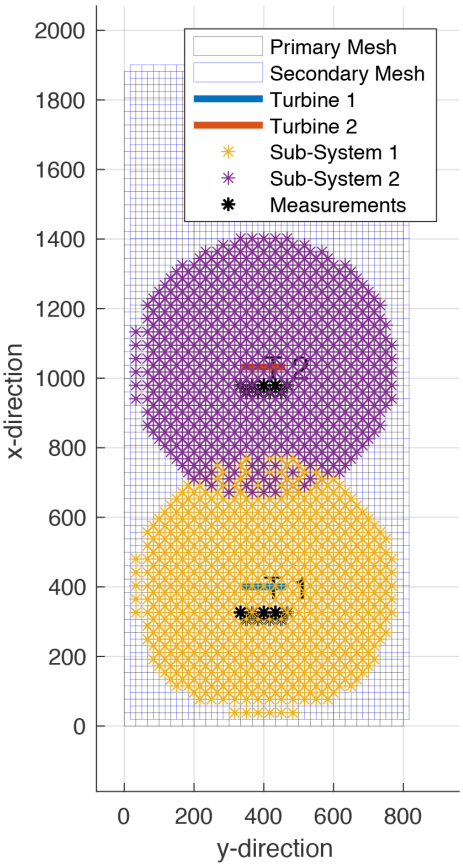


Figure 3-3: States of a 2-turbine wind farm following distributed architecture 1 with a spatial domain of $3D$.

3-3-4 Data Fusion

Data fusion is a method to fuse the exclusive information of the local estimates from all the subsystems into a single estimate. Based on the size of the spatial domain of the subsystems, the local estimates of the subsystems may or may not be overlapping. In the case of the 2-turbine example from Figure 3-2, the spatial domains of the subsystems are not overlapping. Consequently, there will be no overlapping states between the local estimates of the subsystems. On the other hand, in the same 2-turbine example from Figure 3-3, the size of the spatial domains is increased from $2D$ to $3D$. Now, the spatial domains are overlapping, with partially overlapping states. However, irrespective of the overlapping between the subsystems, the subsystems are correlated to each other through the downstream wakes. Hence, there exists a non-zero cross-correlation matrix between the estimates of the subsystems. Thus, a data fusion algorithm is employed to obtain a consistent estimate for the estimated states.

When the cross co-variance between the estimates to be fused are known, the fusion method by Bar-Shalom and Campo [39] and the linear minimum Mean Squared Error fusion method [40] can be employed. In case the cross co-variance is unknown, the conservative fusion algorithms

like the näive approach [41], Co-Variance Intersection (CI) [41], Ellipsoidal Intersection (EI) [42] and Inverse Co-Variance Intersection (ICI) [43, 44] can be employed. Generally, the cross-correlation between the sources to be fused is either unknown [41] or assumed to be unknown [45] to make it practically feasible and computationally efficient. In this case, the cross co-variance between the subsystems is assumed to be unknown, and the conservative fusion algorithms are used.

In short, distributed architecture 1 reduces the computational complexity of the estimation process, by subdividing the system's state-space and developing individual estimators for each of these small-scale subsystem. However, like the centralized architecture, the computation complexity of the control algorithm will still be high, as the model is not distributed. This issue will be addressed in the upcoming distributed architectures.

3-4 Distributed Architecture 2

Distributed architecture 2, unlike the centralized architecture and distributed architecture 1, distributes a wind farm model by defining a separate model for each turbine. Hence, depending upon the number of turbines, there will be an equal number of wind farm models. For example, applying distributed architecture 2 to the aforementioned 2-turbine example, in Figure 3-1, yields two wind farm models, with each model around a turbine. Now, as the models are distributed, not only the estimation algorithms but also the control algorithms can be applied independently for each subsystem. However, each subsystem describes the entire wind farm area. Thus, the size of each subsystem is same as that of the centralized architecture. As a result, the computation complexity of the estimation and control algorithms do not reduce.

Closed-loop wind farm control following distributed architecture 2 can be considered as a five-stage process, and the stages, in order, are spatial decomposition, model decomposition, estimator design, data fusion and control design. Since the controller design is beyond the scope of this research, only the first four stages are explained next.

3-4-1 Spatial Decomposition

As mentioned earlier, a wind farm model is described distributively, and a model is described for each turbine. The first step in creating a separate model for each turbine is to define a spatial domain over which each model operates (predicts the wind flow). In this architecture, the spatial domain is selected to be the same for all the subsystems, and the entire wind farm area is selected as the spatial domain. In other words, like the centralized architecture, the models in this architecture are defined over the entire wind farm area. However, unlike the centralized architecture where the spatial domain describes all the turbines, each model in this architecture will include only a single turbine as a model is defined for each turbine. For example, on applying distributed architecture 2 for the aforementioned 2-turbine example, in Figure 3-1, two subsystems are created, with each subsystem around a turbine, and Figure 3-4 shows the spatial domains of subsystems 1 and 2 respectively.

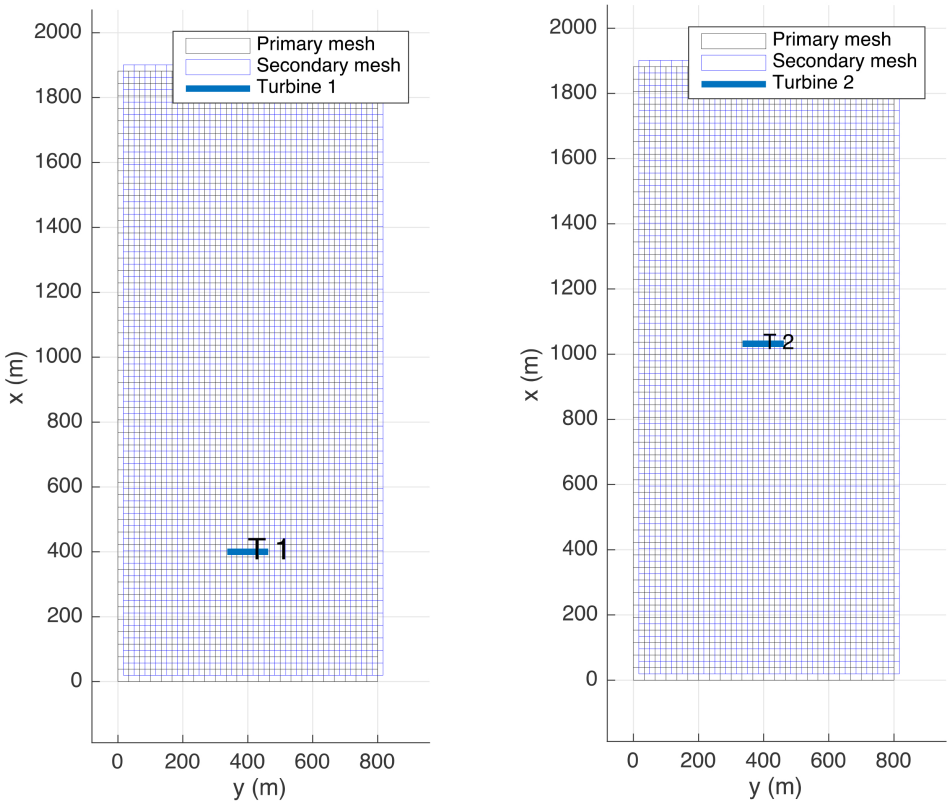


Figure 3-4: Spatial domain of the first and second subsystems of a 2-turbine wind farm following distributed architecture 2.

3-4-2 Model Decomposition

Once a spatial domain is defined for each subsystem, the next step is to define a wind farm model for each subsystem, over the selected spatial domain. A model for each subsystem is obtained by independently computing the WFSim equations, from Eq. (2-1) and Eq. (2-2), for each subsystem. In other words, this process of defining a wind farm model for each subsystem can be approximated as employing a centralized wind farm model for each subsystem but with only one turbine in each subsystem.

Consider a linearized wind farm model following the centralized architecture of the form,

$$x_k = Ax_{k-1} + Bu_k, \tag{3-29}$$
$$y_k = Cx_k + Du_k, \tag{3-30}$$

with $u_k = \begin{bmatrix} u_{1,k} & u_{2,k} & \dots & u_{N_{turb},k} \end{bmatrix}^T$. The variables $u_{i,k}$, $u_{j,k}$ and $u_{N_{turb},k}$ are the control signals of the turbines T_i, T_j and $T_{N_{turb}}$ respectively. On comparing Figure 3-1 with Eq. (3-29) and Eq. (3-30), it can be inferred that only one spatial domain is defined for the entire wind farm area, and all the turbines are included. In mathematical notations, this is represented by defining a single state-space equation containing a single global state x_k and input vector u_k.

Now, on applying distributed architecture 2 to Eq. (3-29) and Eq. (3-30), the linearized model of subsystem i is given by

$$x_{i,k} = Ax_{i,k-1} + B_i u_{i,k}, \forall i \in \{1, \dots, N_{turb}\}, \tag{3-31}$$
$$y_{i,k} = C_i x_{i,k} + D_i u_{i,k}, \tag{3-32}$$

where $B = \begin{bmatrix} \dots & B_i & B_j & \dots & B_{N_{turb}} \end{bmatrix}$ is the input system matrix. On comparing Figure 3-4 with Eq. (3-31) and Eq. (3-32), it can be inferred that the spatial domains of both the subsystems describe the entire wind farm area, and only one turbine is included in each subsystem. In mathematical notations, this is represented by including all the states in both the subsystems and considering the control inputs from only one turbine for each subsystem respectively.

3-4-3 Estimator Design

Once a separate model is obtained, the next step is to independently design an estimator for each subsystem. The ExKF for the linearized model of each subsystem, Eq. (3-31) and Eq. (3-32), is given as

Extended Kalman Filter

1. Time-Propagation (repeated for $\forall i \in \{1, \dots, N_{turb}\}$)

$$\overline{\hat{x}}_{i,k \mid k-1} = f_k(\overline{\hat{x}}_{i,k-1 \mid k-1}, u_{i,k}, w_k), \tag{3-33}$$
$$P_{i,k \mid k-1} = AP_{i,k-1 \mid k-1}A^T + Q_k, \tag{3-34}$$

2. Measurement-Update (repeated for $\forall i \in \{1, \dots, N_{turb}\}$)

$$\tilde{y}_{i,k} = y_{i,k} - H_{i,k}\overline{\hat{x}}_{i,k \mid k-1}, \tag{3-35}$$
$$P_{i,yy} = R_{i,k} + H_{i,k}P_{i,k|k-1}H_{i,k}^T, \tag{3-36}$$
$$P_{i,xy} = P_{i,k \mid k-1}H_{i,k}^T, \tag{3-37}$$
$$K_{i,k} = P_{i,xy}P_{i,yy}^{-1}, \tag{3-38}$$
$$\overline{\hat{x}}_{i,k \mid k} = \overline{\hat{x}}_{i,k \mid k-1} + K_{i,k}\tilde{y}_{i,k}, \tag{3-39}$$
$$P_{i,k \mid k} = (I - K_{i,k}H_{i,k})P_{i,k \mid k-1}, \tag{3-40}$$

On analyzing Eq. (3-33), a model is defined for each turbine, with each subsystem describing the entire wind farm area x_k, and the time-propagation is performed independently for each subsystem by considering only the control inputs from one turbine. Thus, the effect of considering only one turbine on an entire wind farm area can be studied. Meanwhile, in distributed architecture 1, since the model is not distributed, a single time-propagation is performed, considering the control inputs from all the turbines, as given by Eq. (3-17).

Furthermore, from Eq. (3-33) to Eq. (3-40), the entire estimation process is distributed. However, the computation complexity of this architecture does not reduce, as the state vector is not distributed in this architecture.

In addition, on analyzing distributed architecture 2, it can be inferred that the interaction between the turbines are not considered, as the control inputs from only one turbine are included in each subsystem. Consequently, the predicted flow field will be different from the actual flow. The estimate can, however, be improved by including the interactions between the turbines, and data fusion, with the help of sensor measurements, can be employed for this purpose.

3-4-4 Data Fusion

As mentioned earlier, data fusion is employed in architecture 2 to include the interactions between the turbines. In addition, unlike architecture 1 where data fusion is employed to fuse the local estimates back into a single global estimate for the entire state vector, data fusion is employed here to improve the local estimates. Thus, at the end of the data fusion step in architecture 1, there will be only one estimated state vector of the size of the actual system n. Meanwhile, at the end of the data fusion step in distributed architecture 2, there will be number of local estimates, equal to the number of turbines, with each estimate of the size of their respective models.

Furthermore, unlike architecture 1, the data fusion algorithms can be directly applied to the local state estimates of the subsystems, as all the states are estimated by the subsystems and also, the local state estimates are completely overlapping. Moreover, any conservative fusion algorithms, from architecture 1, can be used here, as the cross co-variance between the subsystems is assumed to be unknown.

In short, distributed architecture 2 plays an important role in providing useful insights on the effects of considering only one turbine in a wind farm, even though it neither offers computationally efficiency nor better accuracy. The issue with the computational complexity will be addressed in the upcoming distributed architectures.

3-5 Distributed Architecture 3

Distributed architecture 3 is a computationally efficient version of distributed architecture 2, which was discussed previously. In distributed architecture 2, only the control inputs of the turbines are distributed among the subsystems, while each subsystem still considers the entire state vector. In other words, the size of each subsystem is same as that of the centralized architecture. In addition, each estimator estimates the entire state vector. Hence, distributed architecture 2 is computationally heavy.

Generally, the main idea behind the distributed architectures is to design a subsystem to exclusively describe the states that are near their respective turbine. On analyzing Figure 3-4, it can be inferred that subsystem 1, which is designed to describe turbine 1, unnecessarily estimates states that are far from turbine 1. Similarly, in case of subsystem 2, it is sufficient to just estimate the states around turbine 2. This gives rise to architectures 3 and 4.

Distributed architecture 3 is a combination of architectures 1 and 2. It models a wind farm following architecture 2, and estimates the states following architecture 1. In other words, distributed architecture 3 defines a separate wind farm model for each turbine by distributing

the control inputs of the turbines but not the states, and estimates only a selected number of states around each turbine. For example, applying architecture 3 to the aforementioned 2-turbine example, in Figure 3-1, yields two wind farm models, like architecture 2, with each model around a turbine describing the entire wind farm area. In addition, in each subsystem, only states that fall under the spatial domain around each turbine are measurement-updated, like architecture 1. Thus, the computational complexity of the estimators become tractable. Figure 3-5 shows the effects of applying architecture 3 to the 2-turbine example given in Figure 3-1.

In short, distributed architecture 3 is implemented by following spatial and model decomposition steps from architecture 2, Eq. (3-31), to distribute a wind farm model, and spatial decomposition, estimator model decomposition and filter design steps from architecture 1, Eq. (3-18) to Eq. (3-24), to estimate only the states in the spatial domain around each turbine. Finally, data fusion from architecture 2 is employed for ad hoc correction of wake effects, as architecture 3 also does not implicitly model the interactions between the turbines. However, these steps are not repeated here, as architecture 3 follows equations directly from architectures 1 and 2 without any change.

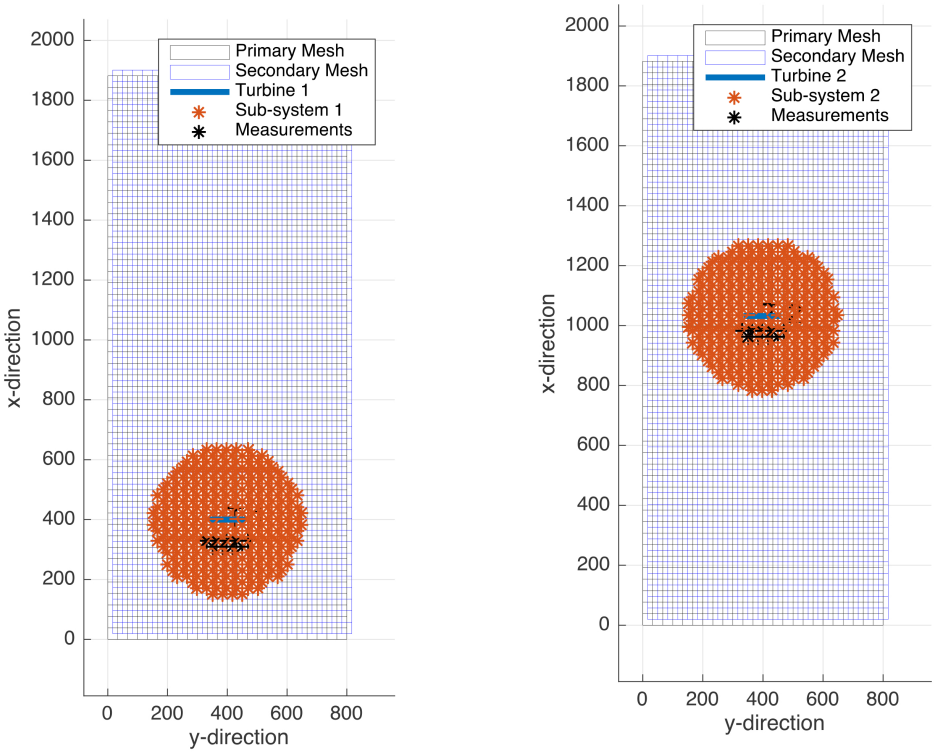


Figure 3-5: States of the first and second subsystems of a 2-turbine wind farm following distributed architecture 3. Orange-colored stars are the only states of the system that are measurement-updated by either of the subsystems.

3-6 Distributed Architecture 4

Though architecture 3 is computationally efficient when compared to architecture 2, control will still be computationally intense, as each subsystem in architecture 3 describes the entire wind farm area. In other words, each subsystem considers the entire state vector. This issue is addressed in architecture 4. Distributed architecture 4 not only circumvents estimating the states that are far from the turbines as in architecture 3 but also does not describe wind flow far from the turbines. Thus, the subsystems are no longer large-scale, making both estimation and control tractable.

Furthermore, similar to architecture 2, closed-loop wind farm control following distributed architecture 4 can be considered as a five-stage process, and the stages, in order, are spatial decomposition, model decomposition, estimator design, data fusion and control design. Since the controller design is beyond the scope of this research, only the first four stages are explained next.

3-6-1 Spatial Decomposition

First, a spatial domain is defined, around a turbine, for each subsystem, to distributively describe a wind farm. Unlike architectures 2 and 3 where the entire wind farm area is selected as the spatial domain for the subsystems, here, a smaller spatial domain around each turbine is considered. Meanwhile, to maintain coupling between the subsystems, the dimension of the spatial domains are selected such that the domains of the subsystems overlap with each other. More importantly, depending upon the dimension of the domains, each subsystem can have more than one turbine, and the meshing is selected in such a way that the overlapping states are defined at the same spatial locations. Furthermore, a spatial domain can be a circle or a square with a turbine at the centre, and the dimension of the domains can be specified in terms of rotor diameter D.

For example, in Figure 3-6, architecture 4 is applied to the aforementioned 2-turbine example, from Figure 3-1. Here, a square spatial domain of size $4D$ is selected for each subsystem. This yields two wind farm models, with each model around a turbine, and describes wind flow only in the spatial domain of size $4D$ for each subsystem instead of the entire wind farm area, as was the case in architectures 2 and 3.

In mathematical notations, this can be represented as

$$x_k = \begin{bmatrix} X_k \\ X_k^* \end{bmatrix}, \tag{3-41}$$

$$X_k = \bigcup_{i=1}^{N_{turb}} X_{i,k}, \tag{3-42}$$

$$X_{i,k} = \begin{bmatrix} X_{ii,k} \\ X_{il,k} \end{bmatrix}, \text{ where } i \in \{1, \ldots, N_{turb}\}, \tag{3-43}$$

where x_k is the state vector of the entire wind farm model (centralized architecture) at k, X_k the combined vector of states that are defined by all the subsystems in architecture 4, X_k^* the vector of states that are defined in the centralized architecture but not by any subsystems

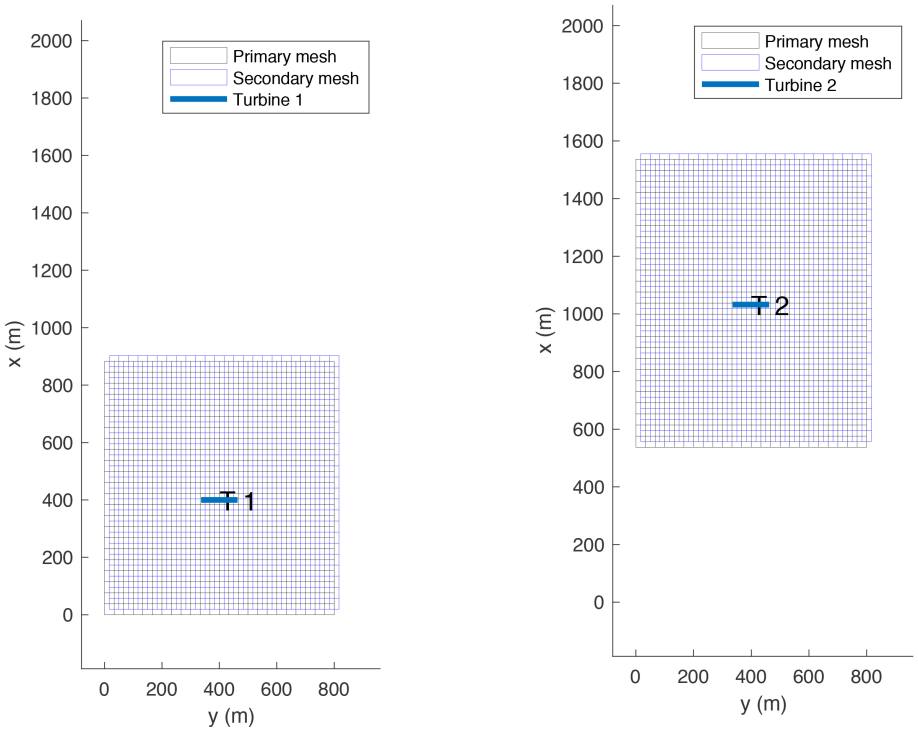


Figure 3-6: Spatial domain of the first and second subsystems of a 2-turbine wind farm following distributed architecture 4.

in architecture 4, $X_{i,k}$ the vector of states described by subsystem i at k, $X_{ii,k}$ the vector of states that are exclusively described by subsystem i at k, l group of neighbouring subsystems of subsystem i, $X_{il,k}$ the vector of states that are, independently, described by subsystems i and l at k, i.e., the vector of states that fall under the overlapping regions between subsystems i and l, and N_{turb} the number of subsystems, which is equal to the number of turbines.

For easier understanding of the aforementioned mathematics, a schematic diagram of a 9-turbine wind farm, following architecture 4, is given in Figure 3-7.

3-6-2 Model Decomposition

Once a spatial domain is defined for each subsystem, the next step is to define a wind farm model for each subsystem over the selected spatial domain. Similar to architecture 2, a model for each subsystem is obtained by independently computing the WFSim equations, from Eq. (2-1) to Eq. (2-2), for each subsystem.

On applying Eq. (3-42) to a linearized wind farm model considered in Eq. (3-5) and Eq. (3-6), and eliminating the undefined states X_k^*, the linearized model of subsystem i, which overlaps

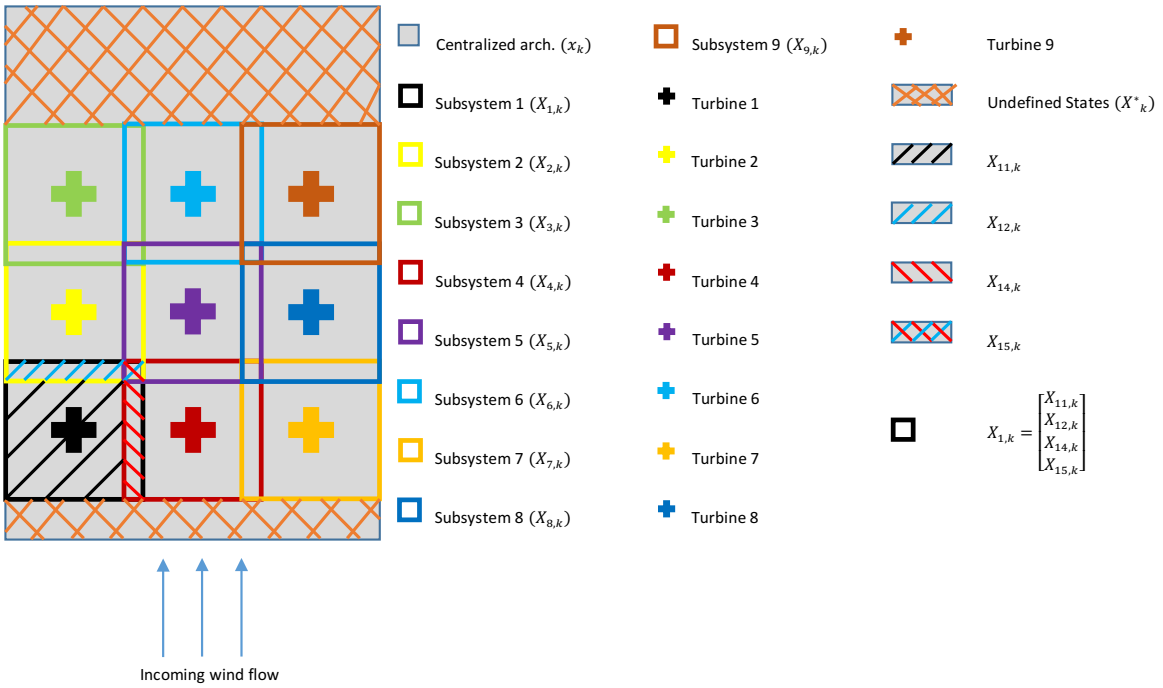


Figure 3-7: Effect of applying distributed architecture 4 to a 9-turbine wind farm.

with its neighbours l, is given by

$$X_{i,k} = \begin{bmatrix} X_{ii,k} \\ X_{il,k} \end{bmatrix}, \text{ where } i \in \{1, \ldots, N_{turb}\}, \tag{3-44}$$

$$\underbrace{\begin{bmatrix} X_{ii,k} \\ X_{il,k} \end{bmatrix}}_{X_{i,k}} = \underbrace{\begin{bmatrix} A_{ii} & A_{iil} \\ A_{ili} & A_{il} \end{bmatrix}}_{A_i} \underbrace{\begin{bmatrix} X_{ii,k-1} \\ X_{il,k-1} \end{bmatrix}}_{X_{i,k-1}} + \underbrace{\begin{bmatrix} B_{ii} \\ B_{il} \end{bmatrix}}_{B_i} u_{i,k}, \tag{3-45}$$

$$Y_{i,k} = C_i \begin{bmatrix} X_{i,k-1} \\ X_{il,k-1} \end{bmatrix} + D_i u_{i,k}, \tag{3-46}$$

where $A_i \in \mathbb{R}^{N_i \times N_i}$, B_i, C_i and D_i are the system matrices of subsystem i, and $u_{i,k}$ the control inputs of the turbines in subsystem i and not the control inputs of just turbine T_i, as was the case in architectures 2 and 3. In other words, unlike architectures 2 and 3 where only one turbine is considered in each subsystem, here, depending upon the dimension of the domains, there can be more than one turbine in each subsystem.

3-6-3 Estimator Design

Once a separate model is obtained, the next step is to independently design an estimator for each subsystem. The ExKF for the linearized model of each subsystem, Eq. (3-45) and Eq. (3-46), is given as

Extended Kalman Filter

1. Time-Propagation (repeated for $\forall i \in \{1, \ldots, N_{turb}\}$)

$$\overline{\hat{X}}_{i,k\,|\,k-1} = f_{i,k}(\overline{\hat{X}}_{i,k-1\,|\,k-1}, u_{i,k}, w_k), \tag{3-47}$$
$$P_{i,k\,|\,k-1} = A_i P_{i,k-1\,|\,k-1} A_i^T + Q_{i,k}, \tag{3-48}$$

2. Measurement-Update (repeated for $\forall i \in \{1, \ldots, N_{turb}\}$)

$$\tilde{Y}_{i,k} = Y_{i,k} - H_{i,k}\overline{\hat{X}}_{i,k\,|\,k-1}, \tag{3-49}$$
$$P_{i,yy} = R_{i,k} + H_{i,k} P_{i,k|k-1} H_{i,k}^T, \tag{3-50}$$
$$P_{i,xy} = P_{i,k\,|\,k-1} H_{i,k}^T, \tag{3-51}$$
$$K_{i,k} = P_{i,xy} P_{i,yy}^{-1}, \tag{3-52}$$
$$\overline{\hat{X}}_{i,k\,|\,k} = \overline{\hat{X}}_{i,k\,|\,k-1} + K_{i,k}\tilde{Y}_{i,k}, \tag{3-53}$$
$$P_{i,k\,|\,k} = (I - K_{i,k} H_{i,k}) P_{i,k\,|\,k-1}, \tag{3-54}$$

where $f_{i,k}$ is the WFSim equations independently evaluated for subsystem i, $\overline{\hat{X}}_{i,k-1\,|\,k-1}$ the expected value of the measurement-updated state estimate of subsystem i at $k-1$, $P_{i,k-1\,|\,k-1}$ the updated co-variance of subsystem i at $k-1$, $\overline{\hat{X}}_{i,k\,|\,k-1}$ the expected value of the predicted state estimate of subsystem i at $k-1$, $P_{i,k\,|\,k-1}$ the predicted co-variance of subsystem i at $k-1$, $\overline{\hat{X}}_{i,k\,|\,k}$ the expected value of the measurement-updated state estimate of subsystem i at k, and $P_{i,k\,|\,k}$ the updated co-variance of subsystem i at k.

3-6-4 Data Fusion

Similar to architectures 2 and 3, the local measurement-updated estimates, Eq. (3-53) and Eq. (3-54), can be used for time-propagation of their respective subsystems, Eq. (3-47), at the next iteration. However, this would completely ignore the interactions between the turbines, even though the subsystems are overlapping, as the subsystems operate independently. Hence, to include interactions between the subsystems, data fusion is employed. As a result, the local estimates of the subsystems will improve. Furthermore, one of the conservative fusion algorithms, like the näive approach, CI, EI and ICI, is employed here, as the cross co-variance between the subsystems is assumed to be unknown.

3-7 Conclusion

As mentioned at the starting of this chapter, a closed-loop approach is formulated to improve the accuracy of the medium-fidelity wind farm models. Typically, an estimator is designed for a wind farm model by following the centralized approach. However, conventional Kalman filter is computationally expensive for such a large-scale system, making it unsuitable for real-time applications. To overcome this issue, four distributed architectures are devised in this thesis. Table 3-1 lists the differences between the typical centralized and newly-devised

Table 3-1: Centralized versus Distributed Architectures

Parameters	**Centralized Architecture**	**Distributed Architectures**			
		Arch.:1	**Arch.:2**	**Arch.:3**	**Arch.:4**
Wind Farm Model	Centralized	Centralized	Distributed	Distributed	Distributed
Number of turbines in each subsystems	all the turbines	all the turbines	1	1	≥ 1
Size of each subsystem	Covers the entire wind farm	Same as centralized	Same as centralized	Same as centralized	Smaller than centralized
Time-propagated States	All the states	All the states	All the states	All the states	States within the selected spatial domains
Estimated States	All the states	States within the selected spatial domains	All the states	States within the selected spatial domains	All the states of the subsystems
Undefined States	None	None	None	None	States that are not in any subsystem
Computational Complexity (ExKF)	$\mathcal{O}(n^3)$ (in case of ExKF) or $\mathcal{O}(nmn)$ (in case of EnKF)	$\mathcal{O}({N_i}^3)$	$\mathcal{O}(n^3)$	$\mathcal{O}({N_i}^3)$	$\mathcal{O}({N_i}^3)$
Computational Complexity (Controller)	Intense	Intense	Intense	Intense	Tractable

distributed architectures. From the table, it is quite evident that control is computationally complex in architectures 1 and 3, though estimator is tractable, as the wind farm area is not distributed. This issue is addressed in architecture 4. In short, architecture 4 is the most valuable of the four distributed architectures, and a more detailed research can be performed on this architecture in the future.

Chapter 4

Simulation Results and Discussions

As explained in the previous chapter, numerous distributed architectures are devised in this thesis for improving the accuracy of the medium-fidelity wind farm model WindFarmSimulator (WFSim) while maintaining its computational tractability. In this chapter, these newly-devised distributed state estimation algorithms are tested against the already-existing centralized architectures.

The simulation scenario is outlined in Section 4-1. In Section 4-2, the various measures employed in this thesis for comparing and evaluating the performance of the estimation algorithms are explained. Then, the research questions formulated in Chapter 1 are motivated in Section 4-3. Later, the simulation results of the centralized and distributed estimation algorithms are presented in Section 4-4 to Section 4-9. In Section 4-10, the optimality of both the centralized and distributed estimation algorithms are analyzed. Finally, the chapter concludes by providing answers to the research questions in Section 4-11.

4-1 Simulation Scenario

The settings used in this thesis for testing the various estimation algorithms are almost similar to the ones used in Doekemeijer [36]. The simulation scenarios used in this thesis are listed below:

1. **Observations**
   Typically, the sensor observations from a real plant are employed for testing the observers. However, in this thesis, high-fidelity data obtained from Simulator fOr Wind Farm Applications (SOWFA) are used for this purpose.

   Furthermore, turbulent forces in the flow develop even in the absence of turbines, eventually resulting in a turbulent, quasi-steady flow field.[1] SOWFA model supports this

[1] A flow at a point is considered to be quasi-steady if the characteristic quantities of the flow, like pressure and velocity, varies gradually over time, making the flow appear steady over short time intervals [46, 47].

feature, and this turbulent flow can be used as an initial field for the actual simulation. This type of simulation is called as "precursor simulation" [34]. However, currently, WFSim supports no such feature. Hence, a 2-turbine case SOWFA simulation without a precursor simulation is used for the observations.

Moreover, the measurement locations for a 2-turbine case with equidistant quadrilateral grid is presented in Figure 4-1. In the literature [35, 36], wind farm observers employ the local wind conditions in front of the turbine, based on turbine power and generator torque, as the sensor observations for the measurement-update. Hence, in this thesis, for a one-to-one comparison with the centralized filters, measurements are assumed to be available only for the states that are in front of the turbines, and the states of the SOWFA model that are present in these locations are used as the sensor observations for the measurement-update.

2. **Atmospheric Conditions**
   For the simulations, the free-stream wind flow is assumed to be steady, and the turbines are the only source of disturbances for the wind flow in a wind farm. Hence, it can be safely assumed that the velocity will continue to remain the same at places where the influence of the turbines are minimal.

   Furthermore, the velocity of the longitudinal inflow component has a mean value of 8.0 m/s. However, to test the full potential of the observers, all the estimation algorithms will be simulated for an initial free-stream flow condition of 11 m/s. This is because the open-loop WFSim model itself is reasonably accurate, and it would be hard to judge the performance of the estimation algorithms under such an ideal scenario. Also, in reality, the actual initial flow velocities are not always accurately known. Hence, it makes sense to use a different initial velocity conditions for the WFSim model.

   With that said, a value smaller than the actual velocity, for example 5 m/s, could have been used as the initial free-stream velocity of the WFSim model too. Instead, a value larger than the actual velocity is selected because a smaller value for the initial value would make the estimation error misleading in the distributed architectures. This will be motivated in the upcoming section.

3. **Process and Measurement Noise**
   For a fair comparison between the different estimation algorithms, the same process and measurement noise co-variances are used for all the simulations. The co-variance of the process and measurement noises is selected to be 0.10 m^2/s^2. The co-variance matrix is selected heuristically, such that the results would not be biased towards any particular filtering algorithm. Furthermore, no additional noise is introduced into the system.

4. No artificial time delay is introduced between the actual plant, SOWFA model, and the estimator. In other words, the observer will be initialized from the first time step.

4-2 Performance Measures

Before introducing the performance measures employed in this thesis, the goal of the thesis is re-iterated here. The objective is to design an estimation algorithm to accurately estimate the states at a computationally tractable cost for the purpose of real-time control. Hence,

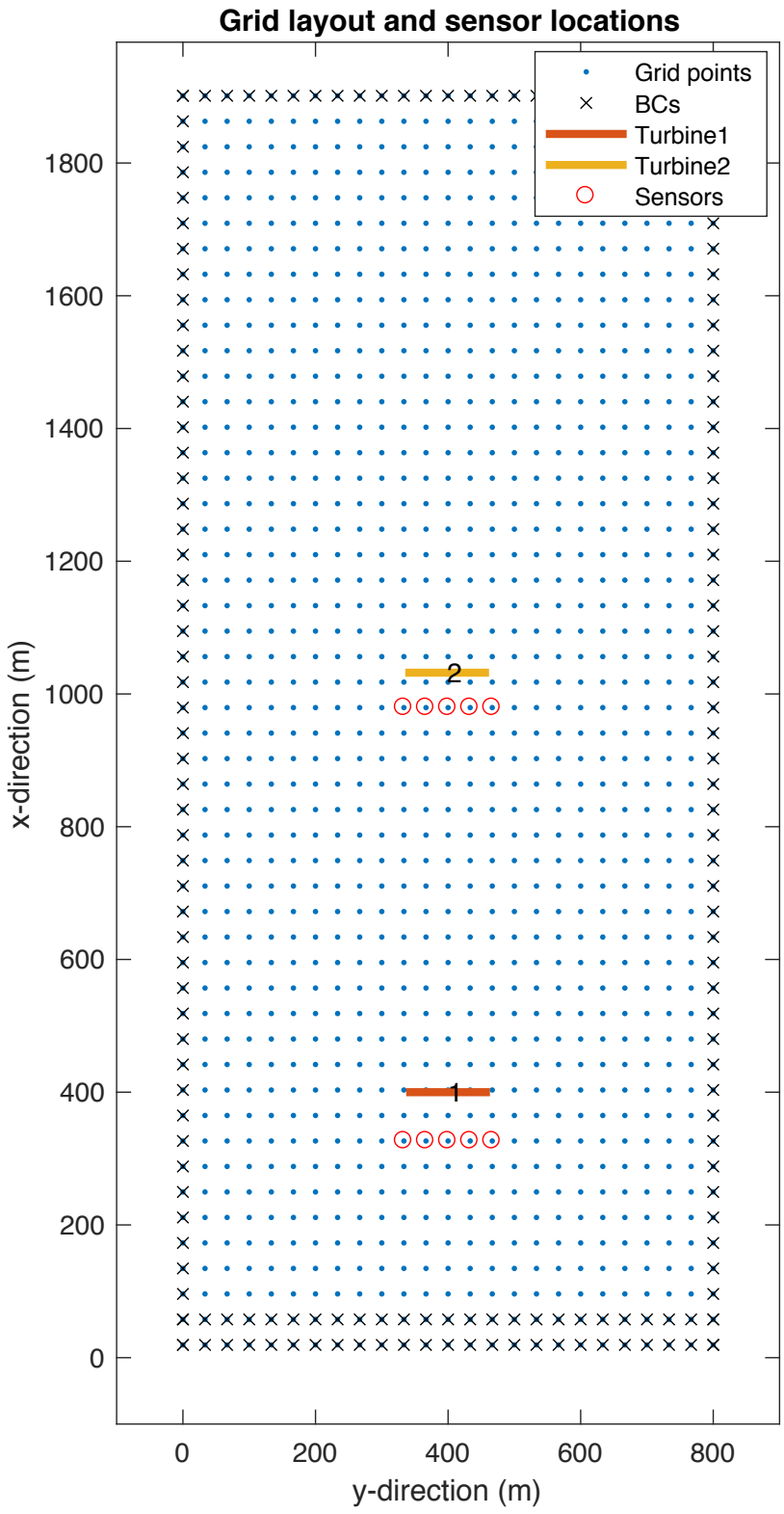


Figure 4-1: Location of longitudinal velocity measurements.

accuracy of the estimate and computational complexity of an estimator are the two criteria on which the estimation algorithms are assessed.

1. **Accuracy**
   The filter estimates have to accurately match the SOWFA data-set, and the root mean square (RMS) error is used for this. The RMS is the average error between two data-sets. The RMS between the estimated data $y \in \mathbb{R}^Z$ and true data $z \in \mathbb{R}^Z$ is calculated as

$$\text{RMS}(y,z) = \sqrt{\frac{1}{Z} \cdot [(y_1 - z_1)^2 + (y_2 - z_2)^2 + \cdots + (y_Z - z_Z)^2]}. \tag{4-1}$$

   Ideally, to improve the accuracy, the RMS must be minimized.

2. **Computational complexity**
   For real-time control, the closed-loop computations, involving both estimator and controller, have to take place within the sampling time. In this thesis, a sampling time of 1 Hz is used. Hence, the closed-loop computations have to, ideally, take less than a second. Thus, a computation time of less than a second will be aimed for in this thesis.

The aforementioned performance criteria were largely influenced by [34].

4-3 Research Questions

In Chapter 1, the objective of the thesis, designing an estimation algorithm to accurately estimate the states and maintain the computational tractability, was formulated into four sub-objectives. In this section, the reasoning behind this formulation is presented.

1. **Frequency of linearization of the WFSim model for the Extended Kalman Filter (ExKF)**
   The WFSim model is obtained by discretizing and re-arranging the Navier-Stokes (NS) equations, Eq. (2-1) to Eq. (2-2). The resulting WFSim model is a non-linear descriptor state-space model, given by Eq. (2-14). Now, for employing the ExKF in both the centralized and distributed architectures, a linear model in the standard state-space form is required. However, linearizing the WFSim model will only yield a linear descriptor state-space model, given by Eq. (2-15). Hence, a descriptor model has to be converted into a standard state-space model, given by Eq. (2-16). The issue, here, is the computational complexity associated with this step. It requires computing an inverse of a large-sized matrix $\bar{E}$, of size $n \times n$. This requires $2n^3$ number of computations, making it computationally undesirable. This leads to the first question: *How often should the WFSim model be linearized for employing the ExKF in both the centralized and distributed frameworks? Can the frequency of linearization be reduced?*

2. **Effects of localization on the centralized filters**
   Typically, a localization matrix is applied to an ensemble co-variance matrix in the Ensemble Kalman Filter (EnKF) (centralized), as the ensemble co-variance matrix is rank deficient because the ensemble members are usually less than the number of states.

However, there is no article to the best of author's knowledge that discusses about the effects of localization length on the estimates of a wind farm model.[2] Hence, the effects of localization length on the centralized EnKF are analyzed in this thesis. In addition, the effects of applying localization on the ExKF are also studied here.

3. **Effects of the size of the subsystems in the distributed architectures**
   In the distributed architectures, the size of the subsystems plays an important role. Depending upon the size of the subsystems, the number of measurement-updated states in architectures 1 and 3, and the number of undefined states in architecture 4 varies. As a result, if the size of the subsystems are too small, the accuracy of the system will be affected. On the other hand, if the size of the subsystems are too large, computation would become complex. This leads to the third question: *How small or big should the size of the subsystems in the distributed architectures be?*

4. **Effects of noisy measurements on wake interactions**
   As explained in the previous chapter, the distributed architectures, especially architectures 2 to 4, do not directly consider the interactions between the turbines, instead an ad hoc correction for wake effects is implemented using sensor measurements and fusion algorithms. This leads to the fourth question: *What happens when no measurements or noisy measurements are available? In this case, how can the interactions be taken into account?*

In the upcoming sections, simulations will be carried out to answer each of the aforementioned questions, and conclusions will be drawn from it.

4-4 Extended Kalman Filter

Firstly, the non-linear WFSim model will be linearized at different frequencies to study its effects on the ExKF. Followed by that, the simulation data will be analyzed, and the best linearization frequency, i.e., the linearization frequency which offers both good estimation accuracy and minimum computation complexity, will be selected. For this selected frequency, the ExKF will be studied for the effects of localization. All these simulations will be carried out according to the conditions specified in Section 4-1.

4-4-1 Effects of Linearization Frequency

The ExKF linearized at different frequencies are simulated for a total simulation time of 1997 seconds, and the plots of the flow fields at the last iteration, $1997s$, are shown in Figure 4-2 and Figure 4-3.[3] Before interpreting these plots, the need for linearization at every iteration (time-step) and the problem associated with it are discussed.

[2] Effects of the number of ensemble members on the estimates of a wind farm model can be found in [35].

[3] Due to the space constraints, only the plots at the last iteration are included in the report. For all the simulations, the plots of the flow fields at the earlier time-steps were converging and similar results were observed. Hence, the observations and conclusions made for the plots at the last iteration are applicable for the plots at the other time-steps as well.

A non-linear system is linearized at a linearization point, which is usually the states and control inputs of a system. If the current state vector and/ or input signal of a system varies too much from the linearization point, the linearized model will no longer be able to accurately represent the actual non-linear system. Hence, in the ExKF, a non-linear system is usually linearized at every time-step, and the updated state vector from the previous time-step will be used as the linearization point instead of the state vector from the first time-step. Consequently, the variation between the linearization point and the current state vector will be minimum, and the linearized model will be a good fit of the non-linear model at that time-step. However, in the case of the WFSim model, linearizing the non-linear WFSim model is a computationally heavy process, for the reasons stated in Section 4-3. As a result, simulations are performed to analyze if the WFSim model can be linearized less frequently and to observe the extent to which this affects the accuracy of the estimation.

On comparing the flow fields obtained for the ExKF linearized at different frequencies as displayed in Figure 4-2 and Figure 4-3, it can be inferred that the estimates of the flow fields are almost the same for all the linearization frequencies. In other words, linearizing the non-linear WFSim model at different frequencies does not have a prominent effect on the accuracy of the estimation of the flow fields. Also, a similar response is observed on comparing the RMS error for the ExKF at different linearization frequencies, displayed in Figure 4-4 and Figure 4-5.[4] A possible reason postulated for this response is the atmospheric conditions under which the simulations are performed. Since a precursor simulation is not employed and the wind direction does not change during the course of the simulations, the state vector (velocity of the wind) do not vary too much, in time, if not for the effects of the turbines. However, the effects of the turbines are experienced only in the neighbourhood of the turbines, while the states that are far from the turbines are least influenced by the turbines. Hence, the states that are near the turbines only vary. These small variations can be incorporated by marginally increasing the magnitude of the process noise. As a result, the frequency of linearization of the non-linear WFSim model affects the accuracy of the estimation to the very least. However, if the wind direction/ velocities were to change with respect to time and/ or if a precursor simulation is considered, the WFSim model should be linearized more frequently.

Now, on comparing the time taken for an iteration by the ExKF at different linearization frequencies given in Figure 4-6, it can be inferred that linearizing the non-linear WFSim model less frequently helps in reducing the computational complexity drastically. Typically, a non-linear model will be linearized at every iteration for maximizing the estimation accuracy. However, in the case of the WFSim model, since linearizing the model less frequently does not affect the accuracy of the estimation and it reduces the computational complexity drastically, the WFSim model will be linearized only at the first iteration for the remaining simulations.

[4]The centerline is a quadrilateral spatial domain in a wind farm, mostly rectangular in shape, that runs for the entire longitudinal length of the wind farm between the turbine's rotor length, from rotor end to rotor end.

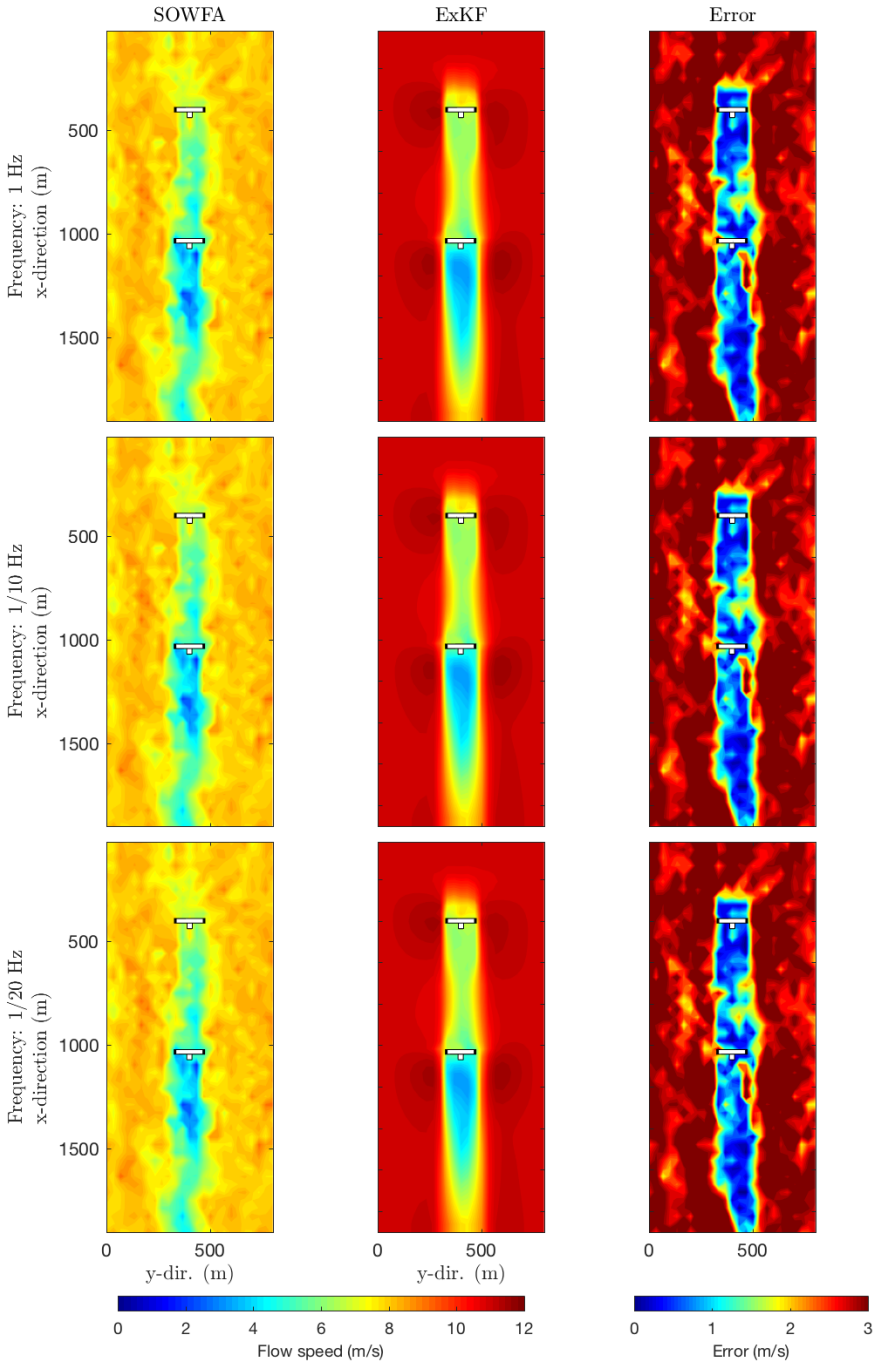


Figure 4-2: Snapshots of the longitudinal flow velocity (m/s) throughout the grid for various linearization frequencies, f = 1Hz, 1/10Hz, 1/20Hz, for the ExKF.

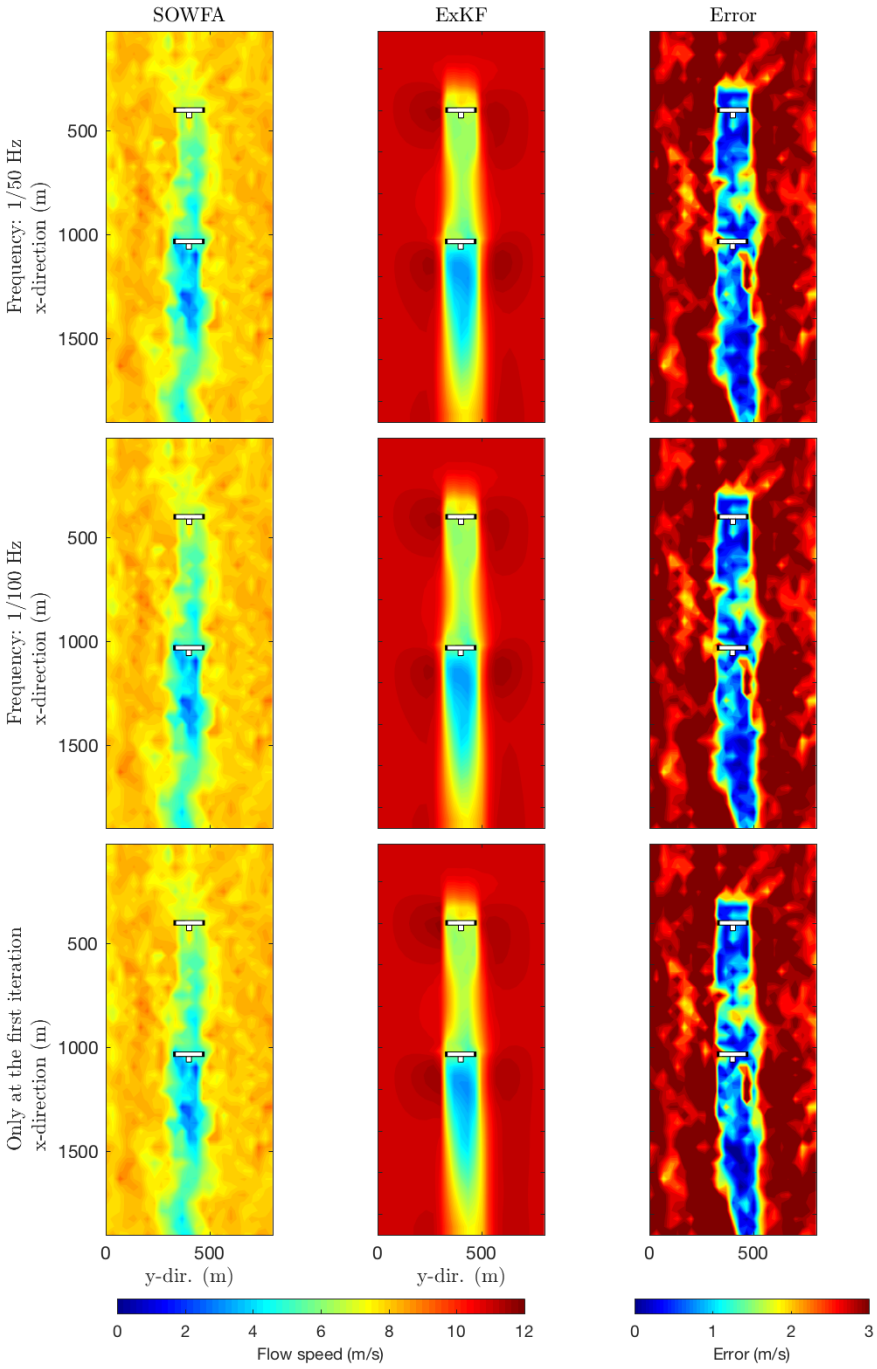


Figure 4-3: Snapshots of the longitudinal flow velocity (m/s) throughout the grid for various linearization frequencies, $f = 1/50$Hz, $1/100$Hz and only at the first time step, for the ExKF.

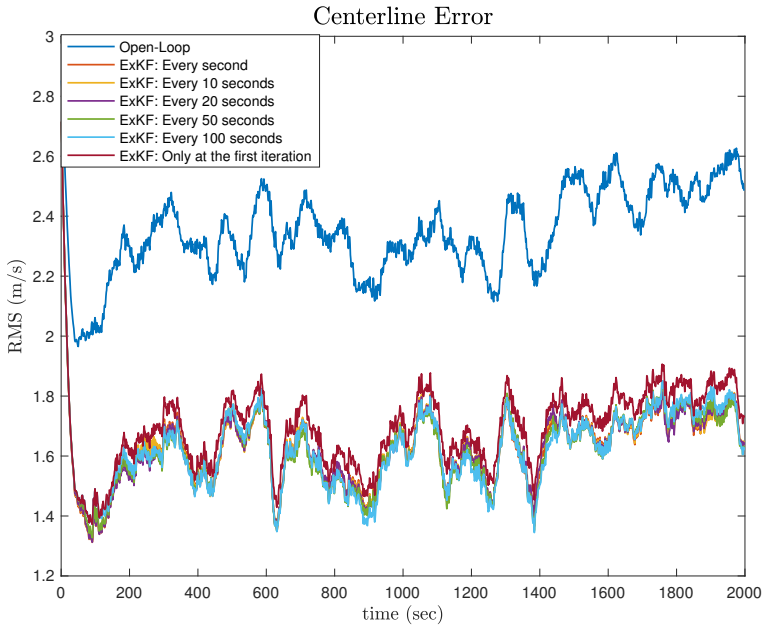


Figure 4-4: Centerline error between the SOWFA and the ExKF linearized at different frequencies over time (m/s).

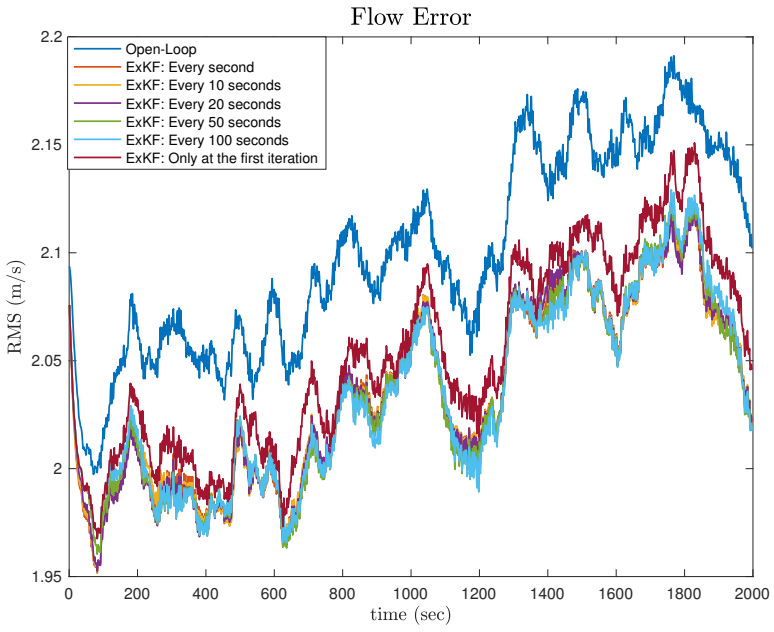


Figure 4-5: Flow error between the SOWFA and the ExKF linearized at different frequencies over time (m/s).

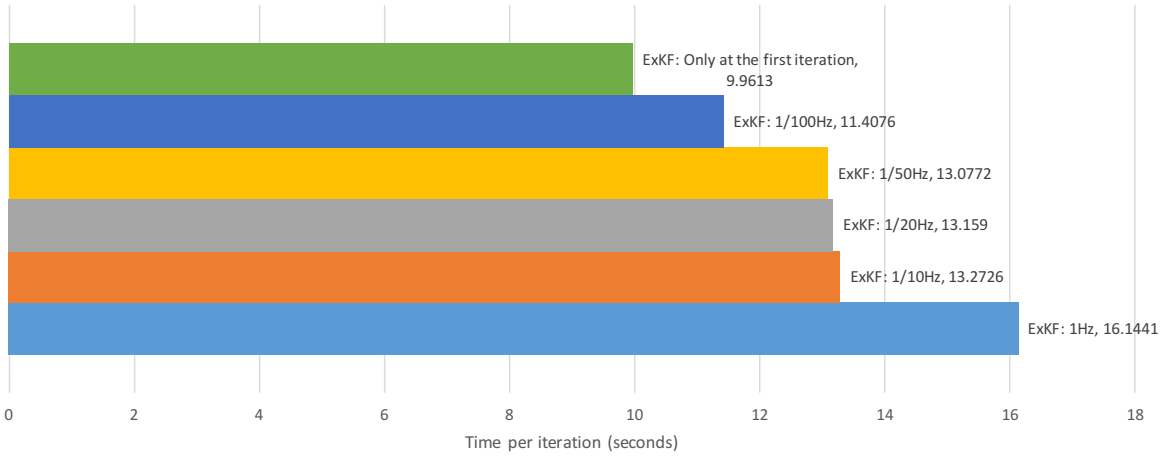


Figure 4-6: Computational complexity of the ExKF at different linearization frequencies.

4-4-2 Effects of Localization

The ExKF is localized by employing the localized co-variance matrix, $P^*_{k-1\,|\,k-1}$, in place of the actual updated co-variance matrix $P_{k-1\,|\,k-1}$. The localized co-variance matrix is obatined by multiplying a localization matrix with the updated co-variance at $k-1$, and it is given as

$$P^*_{k-1\,|\,k-1} \quad = \quad \Phi^x \circ P_{k-1\,|\,k-1}, \tag{4-2}$$

where $\circ$ is the element-wise product (Hadamard) of two matrices, $P_{k-1\,|\,k-1}$ the actual updated co-variance at $k-1$, and $P^*_{k-1\,|\,k-1}$ the localized updated co-variance at $k-1$. Φ^x is the localization matrix, and it is given as

$$\Phi^x \quad = \quad \begin{bmatrix} \Phi(d^x_{1,1}) & \cdots & \Phi(d^x_{1,n}) \\ \vdots & \ldots & \vdots \\ \Phi(d^x_{n,1}) & \cdots & \Phi(d^x_{n,n}) \end{bmatrix}, \tag{4-3}$$

where $d_{i,j}$ is the distance between states i and j, and Φ, the weighting function, is given as

$$\Phi \quad = \quad \begin{cases} 1, & d_{i,j} \leq L, \\ 0, & d_{i,j} \geq L, \end{cases} \tag{4-4}$$

with L the localization length. L can be specified in terms of the rotor diameter D.

Now, the ExKF localized for different lengths are simulated for a total simulation time of 1997 seconds, with the non-linear WFSim model linearized only at the first iteration, and the plots of the flow fields at the last iteration, 1997s, are shown in Figure 4-7 and Figure 4-8. On comparing the flow fields, it can be inferred that the estimation error decreases with the increase in the localization size L, and this trend continues until a certain localization size. Thereafter, the accuracy of the estimate of the flow fields saturates, and the estimate of the flow fields remains almost the same. Also, a similar response is observed on comparing the RMS error between the ExKF localized for different size, displayed in Figure 4-9 and

Figure 4-10. A possible reason postulated for this response is the number and location of the observations (sensor measurements). Since the observations are available only for the states that are near the turbines, measurement-updates are valuable only for these nearby states. As the distance of the states increases from the observations (turbines), the effect of filtering starts to diminish, i.e., the observability reduces with increasing distance from the sensor measurements, as less information is known about the states that are further away from the measurements [48]. Hence, the accuracy of the estimate of the flow fields saturates beyond a certain localization size and does not improve on increasing the size of the localization any further.

Furthermore, on comparing the RMS error between the different localized ExKF, it can be inferred that the ExKF localized for a certain smaller localization size performances slightly better than the ExKF localized for a larger size. This could be because of the non-linearity of the WFSim model and/ or the inherent differences between the WFSim and SOWFA models.

In addition, the computational complexity of the ExKF localized for different size remains the same. This is because varying the size of localization only affects the entries of the co-variance matrix and not the size.

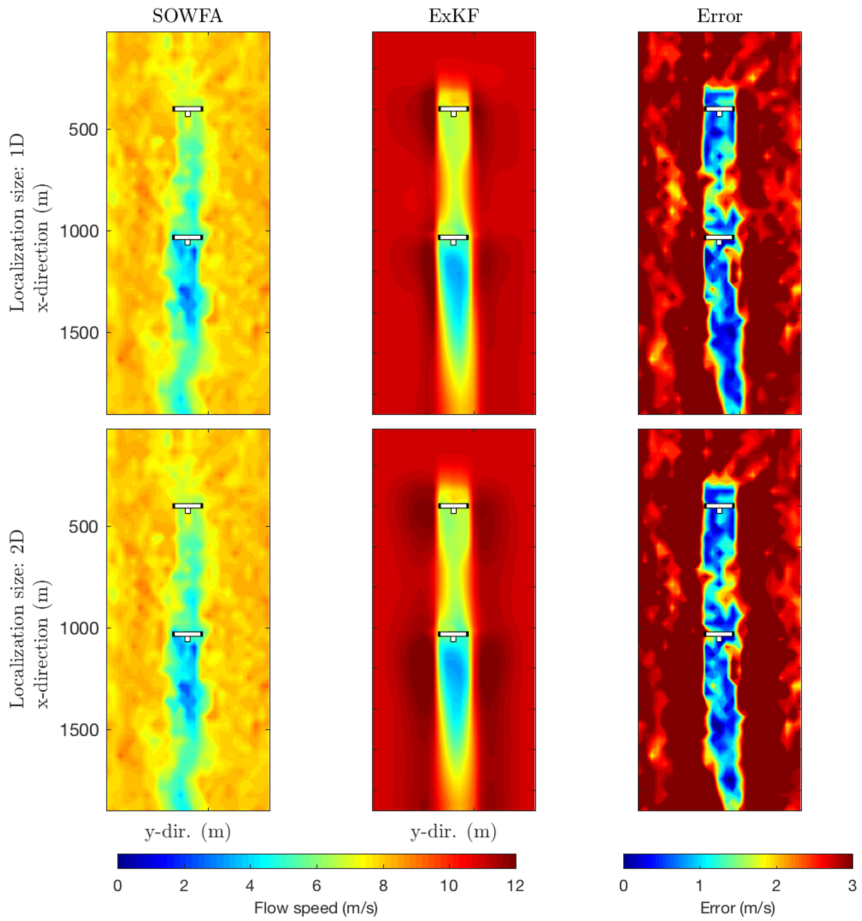


Figure 4-7: Snapshots of the longitudinal flow velocity (m/s) throughout the grid for various localization size, $L = 1D, 2D$, applied on the ExKF.

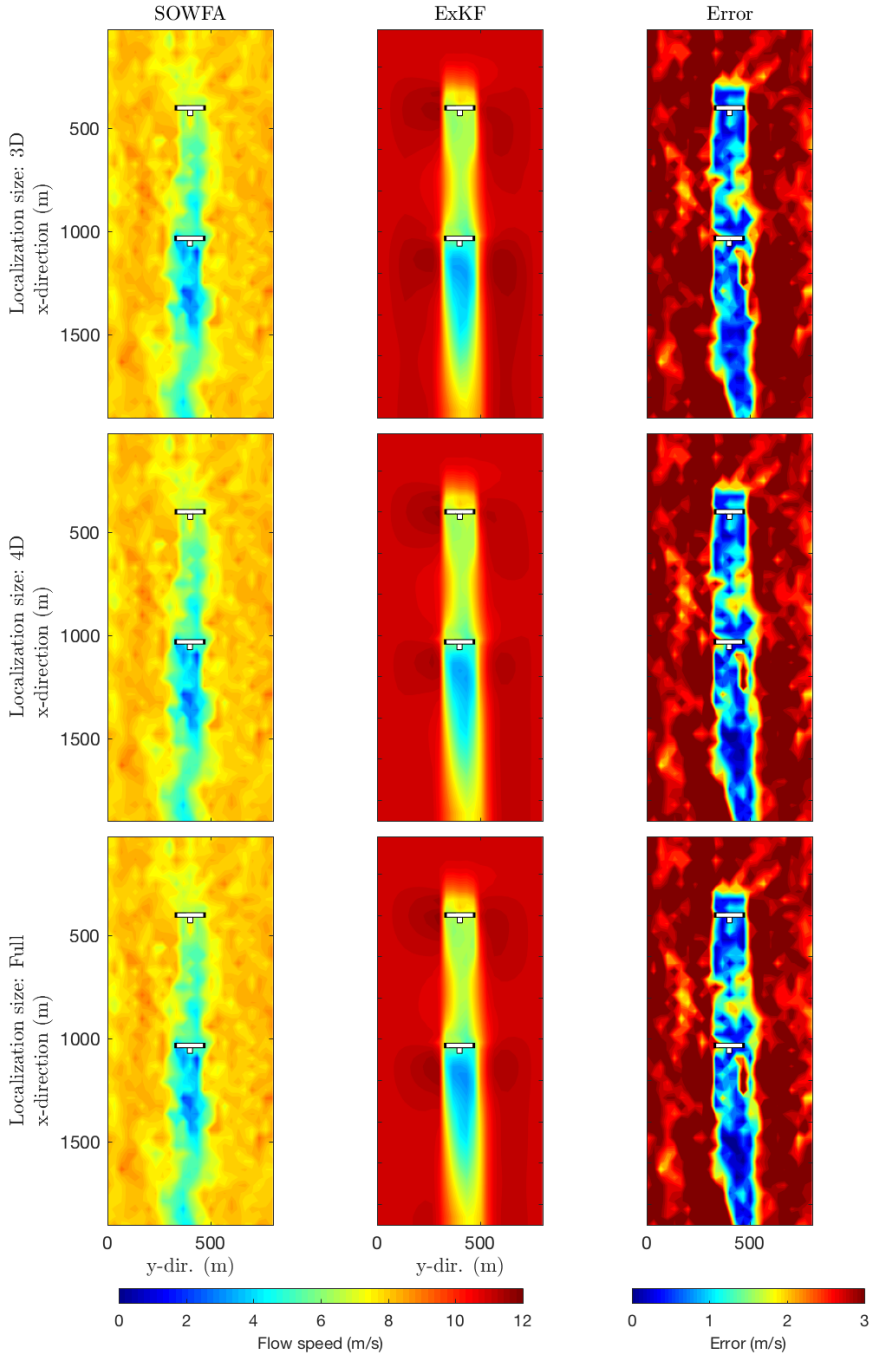


Figure 4-8: Snapshots of the longitudinal flow velocity (m/s) throughout the grid for various localization size, $L = 3D, 4D$ and no localization, applied on the ExKF.

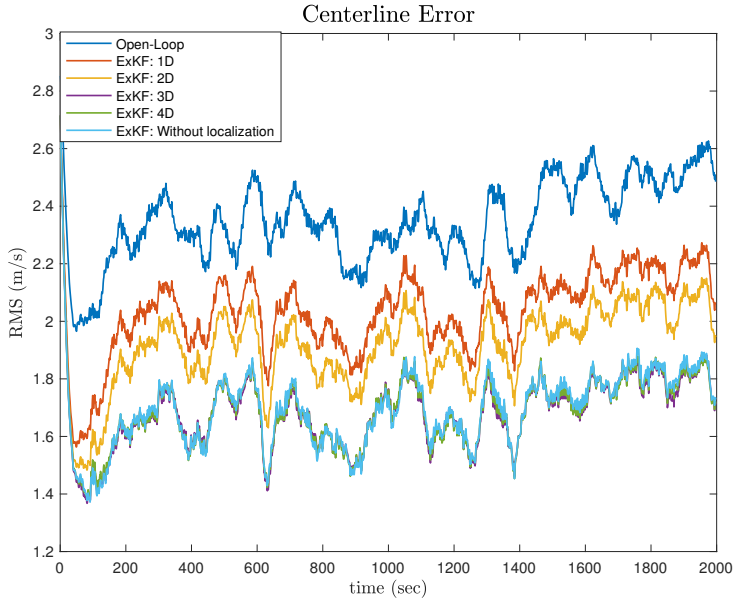


Figure 4-9: Centerline error between the SOWFA and various localized ExKF over time (m/s).

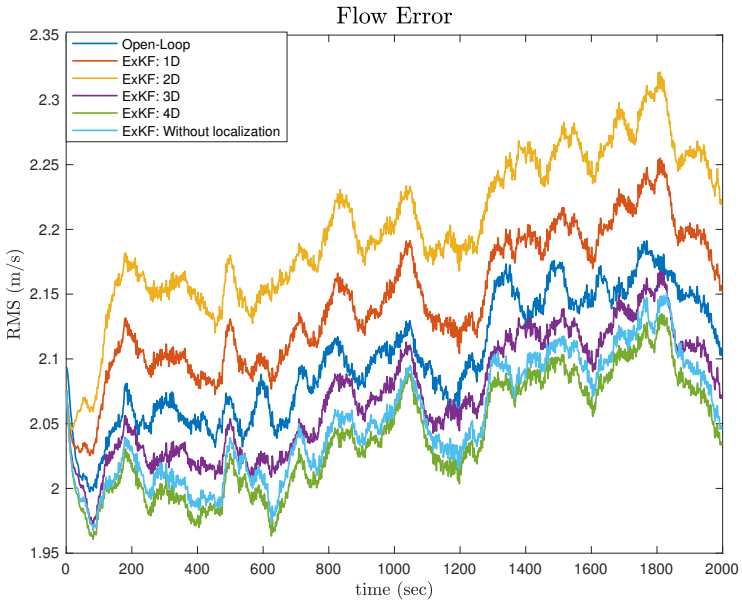


Figure 4-10: Flow error between the SOWFA and various localized ExKF over time (m/s).

4-5 Ensemble Kalman Filter

For the simulation scenario specified in Section 4-1, the EnKF is localized for various localization lengths, with an ensemble size of 150 members.[5] The ensemble size is heuristically selected, such that the estimates of the EnKF localized for various localization lengths do not diverge. In addition, all the simulations are performed for a total simulation time of 1997 seconds, and the plots of the flow fields at the last iteration, 1997s, are shown in Figure 4-11, and Figure 4-12.

On comparing the flow fields, it can be inferred that the estimation accuracy is high for a smaller localization length itself, and the accuracy almost saturates with the localization length. A possible reason postulated for this response is the use of the non-linear WFSim model, and unobservability of the distant states. Since the non-linear WFSim model is used directly without linearization, the EnKF localized for a smaller localization length, $1D$, itself performs accurately. Also, the observability reduces with increasing distance from the sensor measurements, as less information is known on the distant states [48]. Hence, with the EnKF, the increase in the accuracy with the localization length is not that prominent, as was the case with the ExKF.

Furthermore, from the mean of the RMS error plots[6] as displayed in Figure 4-13 and Figure 4-14, it can be inferred that the estimation error increases subtly with the localization length. A possible reason postulated for this response is the number of ensemble members employed. Typically, an ensemble co-variance matrix is rank deficient due to the usage of less number of ensemble members than the size of the actual state vector. Consequently, the sample co-variance matrix represents the correlations between the distant states with large terms, though they are not that much correlated in reality. To counter this, localization is applied. Here, since the number of the ensemble members is fixed and the localization length is increased, the spurious correlations between the uncorrelated states are considered. Hence, the accuracy of the estimate deteriorates subtly.

Now, on comparing the computational complexity between the EnKF localized for various localization length as displayed in Figure 4-15, the complexity of the filter is almost the same. This is because varying the size of localization varies only the entries of the ensemble co-variance matrix, and the size of the co-variance matrix remains the same as the number of the ensemble members is constant.

[5] The ensemble members define the number of realizations of the state vector taken around the initial state estimate for approximating the co-variance matrix. On the other hand, the localization length defines the number of states that are allowed for measurement-update, using the cross correlation terms.

[6] The mean of the RMS error is considered here, as the RMS error plot over time is not so insightful.

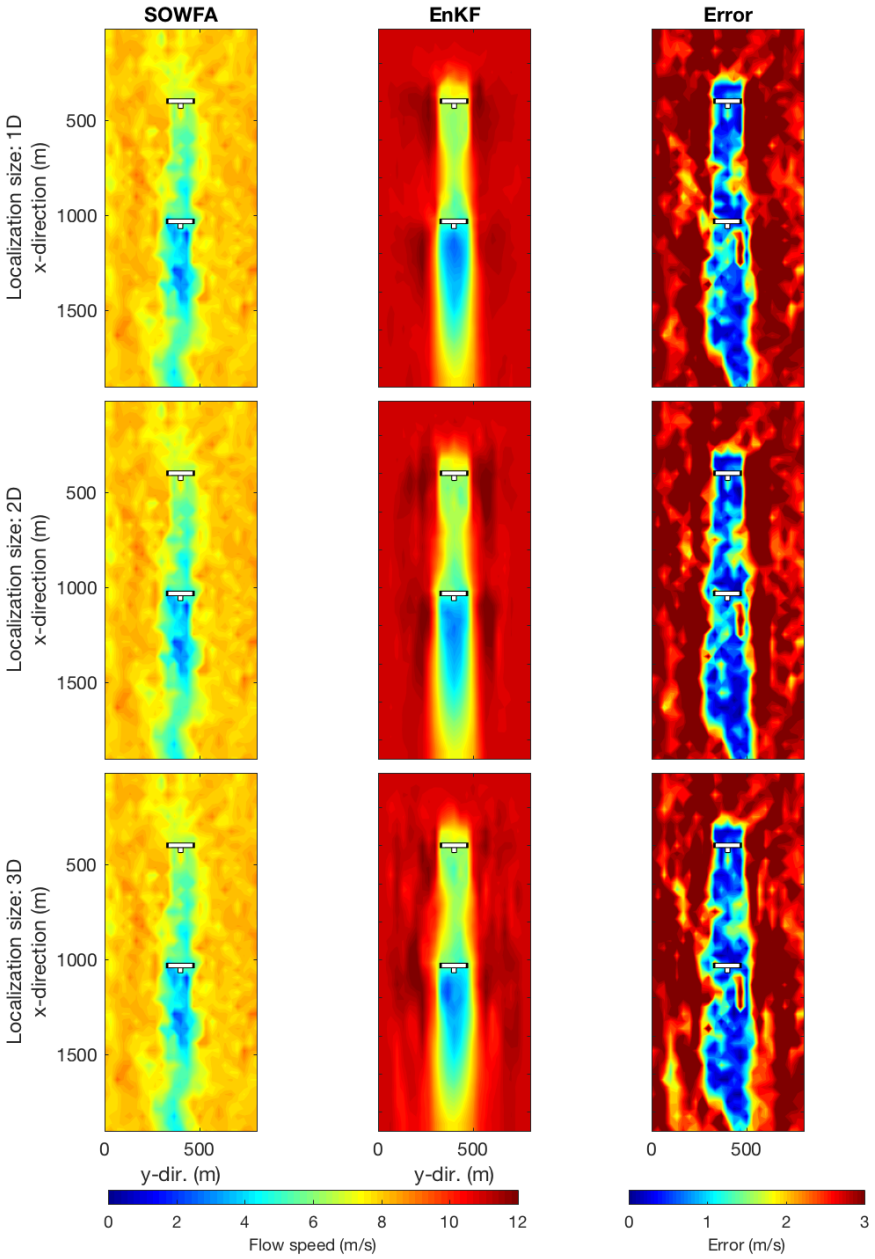


Figure 4-11: Snapshots of the longitudinal flow velocity (m/s) throughout the grid for the EnKF localized for a length of $1D, 2D, 3D$, applied on the EnKF.

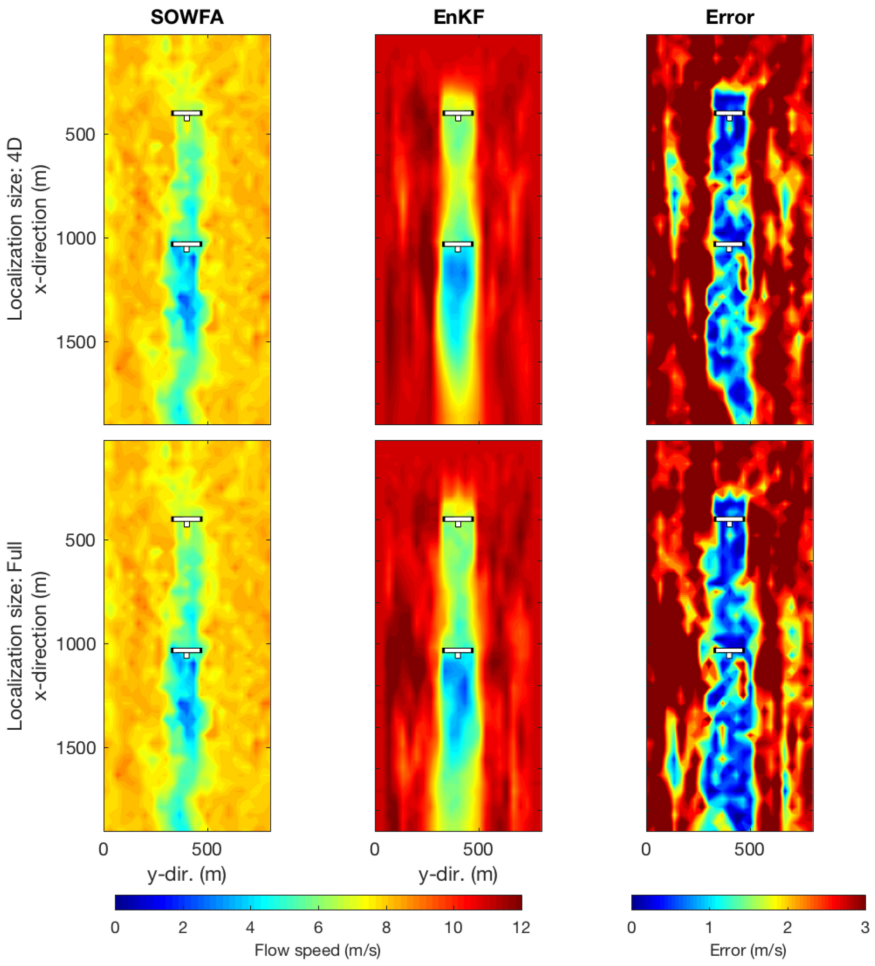


Figure 4-12: Snapshots of the longitudinal flow velocity (m/s) throughout the grid for various localization length, $4D$ and no localization, applied on the EnKF.

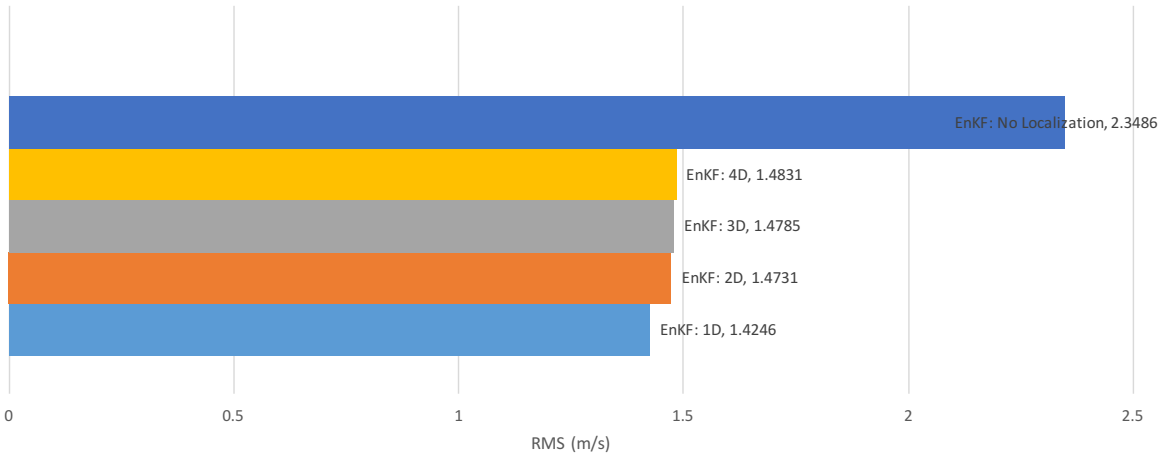


Figure 4-13: Mean centerline error between the SOWFA and various localized EnKF (m/s).

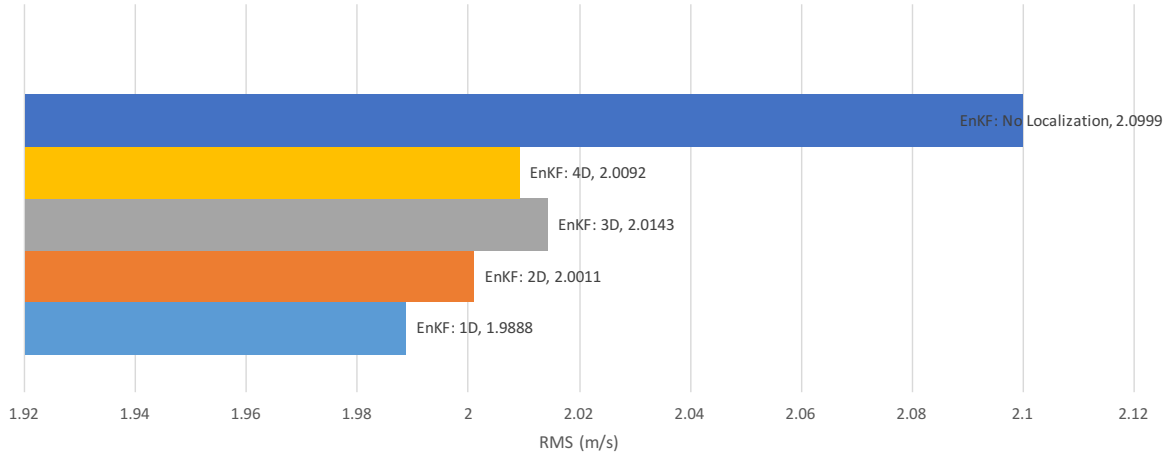


Figure 4-14: Mean flow error between the SOWFA and various localized EnKF (m/s).

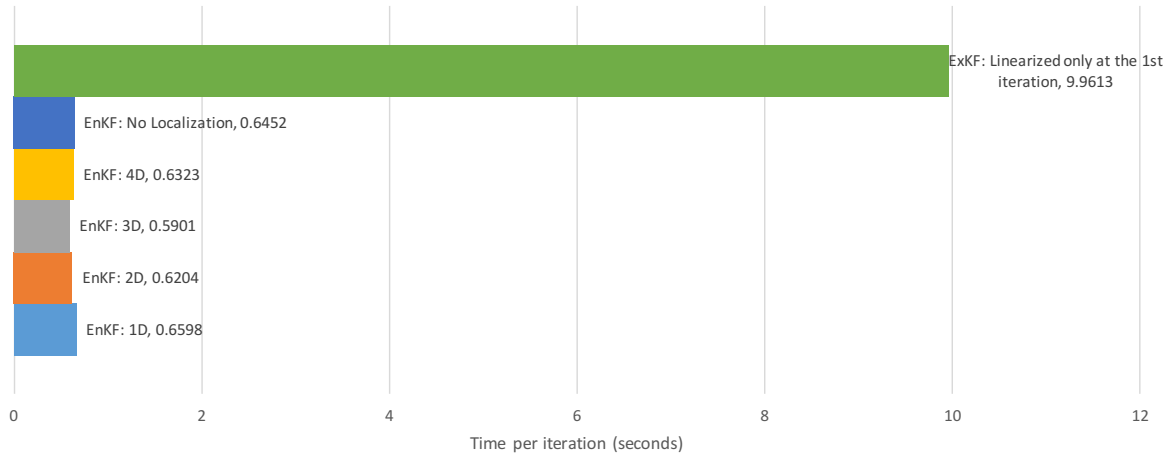


Figure 4-15: Computational complexity of the EnKF at localized for different localization size.

4-6 Distributed Architecture 1

For the simulation scenario specified in Section 4-1, distributed architecture 1 is simulated for different sizes of the subsystems. When the size of the subsystems increases beyond a certain domain size, the subsystems begin to overlap. In case of an overlap between the subsystems, there would exist more than one estimate for an overlapping state. In this scenario, as specified in the previous chapter, a data fusion algorithm can be employed to fuse the multiple estimates of an overlapping state to obtain a single estimate or one of the multiple estimates of an overlapping state can be used. In the first case, exclusive information from all the overlapping estimates will be combined to obtain a single improved estimate. While in the second case, except for the selected estimate, all the exclusive information present in the remaining estimates will be discarded. Hence, the accuracy of the estimate of the flow fields should improve upon employing fusion. Here, the subsystems overlap for the domain sizes of $3D$ and $4D$. Hence, for the domain sizes of $3D$ and $4D$, simulations are performed both with and without fusion to analyze the extent to which the accuracy of the estimates improve.

Like the simulations from the centralized architecture, all the simulations, here, are performed for a total simulation time of 1997 seconds. The plots of the flow fields at the last iteration, 1997s, for a domain size of $1D$ and $2D$ are shown in Figure 4-17. The plots of the flow fields for a domain size of $3D$ and $4D$ without fusion is displayed in Figure 4-18, and with fusion is displayed in Figure 4-19. More importantly, the ExKF in this architecture will be linearized only at the first iteration as linearization hardly has any effect on the accuracy of the estimates, see Section 4-4.

Now, on comparing the flow fields displayed in Figure 4-17, Figure 4-18 and Figure 4-19, it can be inferred that the accuracy of the estimate of the flow fields increases with the increase in the size of the subsystems. This trend continues until a certain domain size, irrespective of whether or not fusion is employed. Also, a similar response is observed on comparing the mean of the RMS error of distributed architecture 1 for different sizes of the subsystems, as displayed in Figure 4-20 and Figure 4-21. A possible reason postulated for the initial increase in the estimation accuracy is the increase in the number of states that are measurement-updated. As the size of the subsystems increases, more states will be measurement-updated. Consequently, the accuracy would increase with the size. However, after a certain distance, the accuracy of the estimate of the flow fields declines and finally, saturates. A possible reason postulated for this decline in the accuracy of the estimates is the unobservability of the newly-included states. As the size of the subsystems increase, more states are included into the subsystems for measurement-update. Generally, the unmeasured states are measurement-updated because of their correlations with the measured states. As the distance of the states increases from the observations (turbines), less information are known about these states [48]. In other words, the correlations become incapable of providing any valuable information for the accuracy of the estimates to improve. This phenomenon is what was observed in the centralized filters, as displayed in Figure 4-8, Figure 4-11, Figure 4-12 and Figure 4-13, i.e., the accuracy of the estimate of the flow fields saturate after a certain localization size. However, as it can be seen in Figure 4-17, Figure 4-18 and Figure 4-19, after a certain size, the accuracy declines in distributed architecture 1, instead of saturating. A possible reason postulated for this detrimental effect caused by the correlations is the number of observations considered by the subsystems. In the distributed architecture, each subsystem considers sensor measurements that are only located within the spatial domain of that subsystem, as opposed to using all the

sensor measurements in the centralized filters localized for the same size. For example, in a 2-turbine case, if a centralized filter, either the ExKF or EnKF, is localized for a size of $4D$, all the observations located near both the turbines will be employed. Whereas in distributed architecture 1, either of the subsystems of size $4D$ will use only the measurements that are located within their spatial domain. As a result, the accuracy of the estimate of the flow fields in the centralized architecture saturates after a certain localization size. Whereas in the case of distributed architecture 1, the correlations between the states that are far from the turbines (unobservable states) and the states that are near the turbines (observable states) cause the decline in the accuracy, as less number of measurements are used in either of the subsystems.

Furthermore, on comparing the plots of the flow fields with and without fusion as displayed in Figure 4-18 and Figure 4-19, it can be inferred that fusion improves the estimates, as expected. Also, few streaky lines of reduced estimation error are visible in distributed architecture 1, with fusion, for a subsystem size of $4D$. This could be a result of numerical instability.

Finally, on comparing the computational complexity of distributed architecture 1 for different sizes of the subsystems, as displayed in Figure 4-16, the complexity increases drastically with the increase in the size of the subsystems. However, from the plots of the flow fields, the accuracy of the estimate increases with the size of the subsystems until a certain size. Hence, the size of the subsystems should be selected such that neither the complexity nor the accuracy of the estimates is compromised too much. From the plots, a domain size of $2D$ for the subsystems is both accurate as well as computationally tractable (takes less than a second).

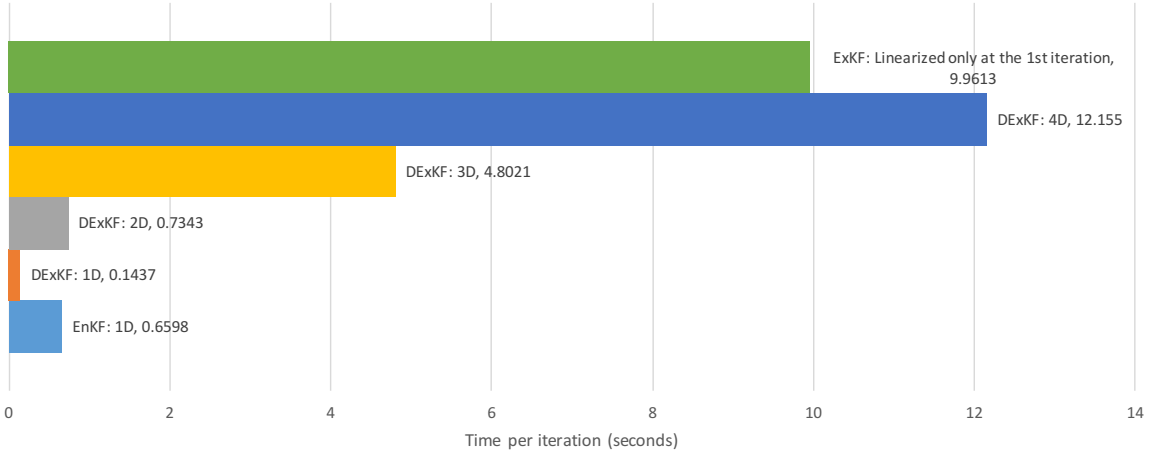


Figure 4-16: Computational complexity of distributed architecture 1 for different sizes of the subsystems.

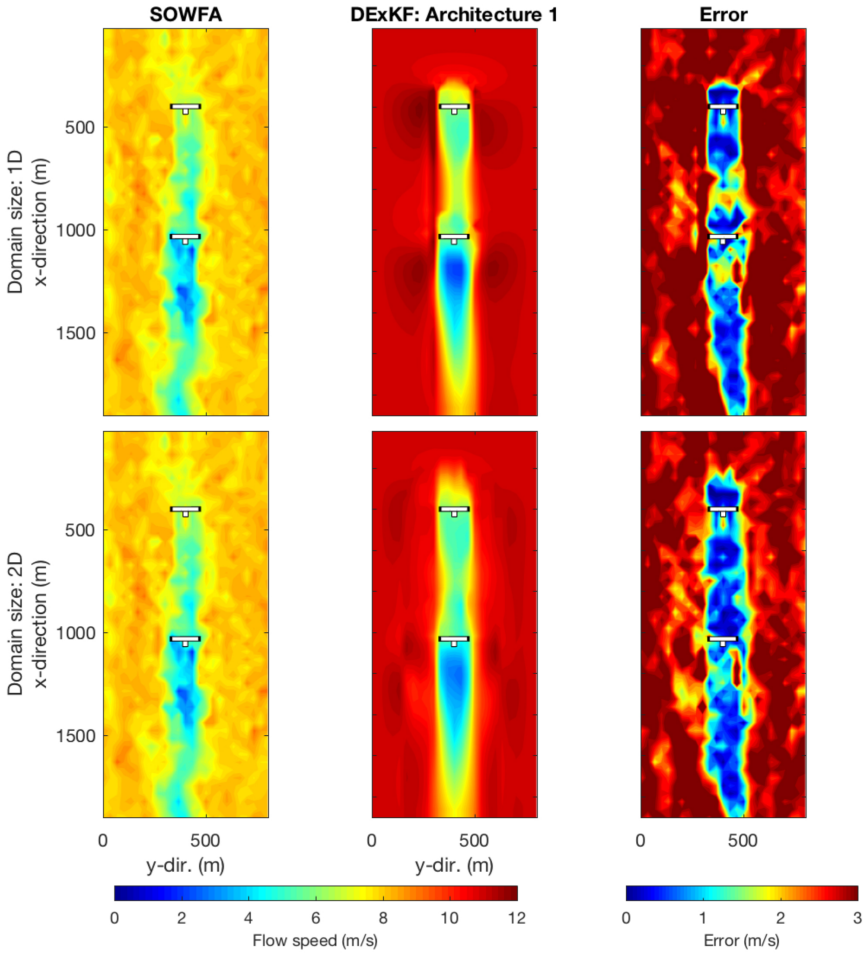


Figure 4-17: Snapshots of the longitudinal flow velocity (m/s) throughout the grid for distributed architecture 1 with a domain size of $1D, 2D$.

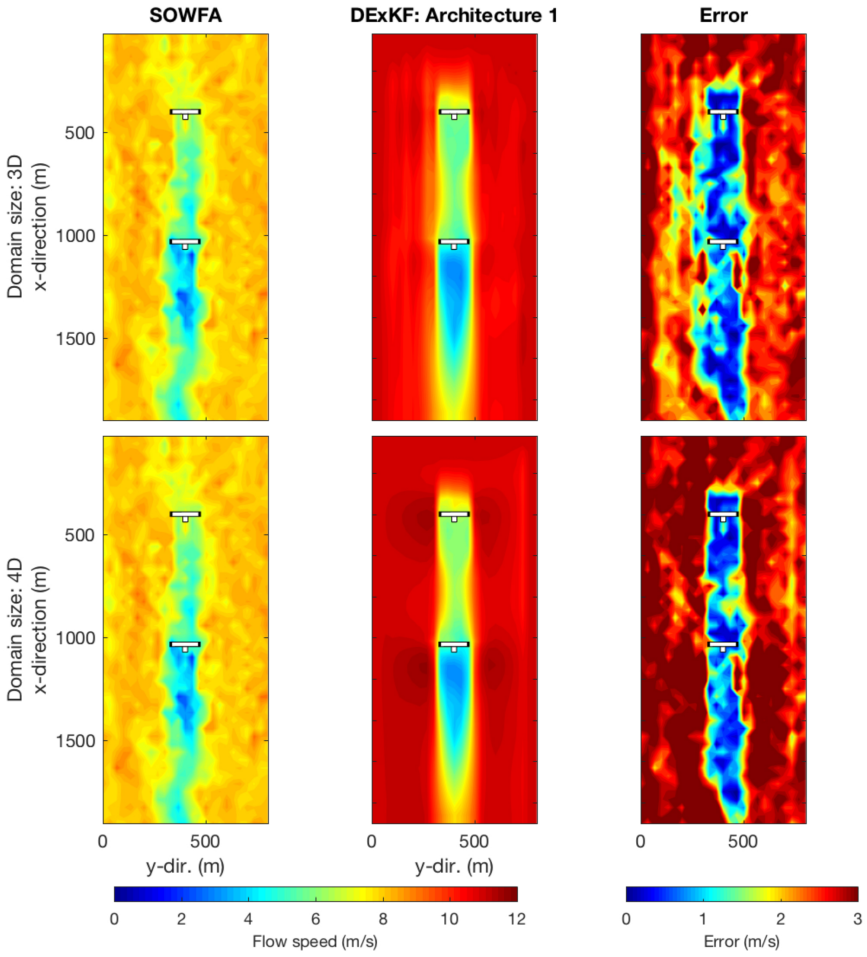


Figure 4-18: Snapshots of the longitudinal flow velocity (m/s) throughout the grid for distributed architecture 1, without fusion, for a domain size of $3D, 4D$.

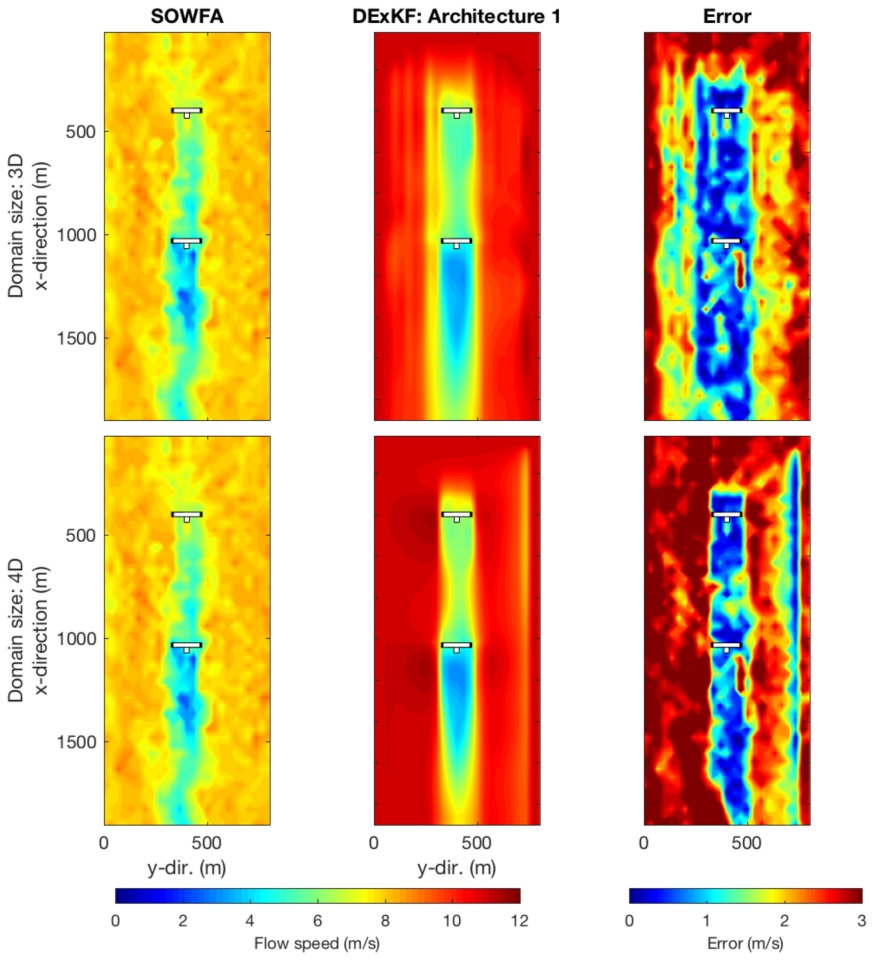


Figure 4-19: Snapshots of the longitudinal flow velocity (m/s) throughout the grid for distributed architecture 1, along with fusion, for a domain size of $3D, 4D$.

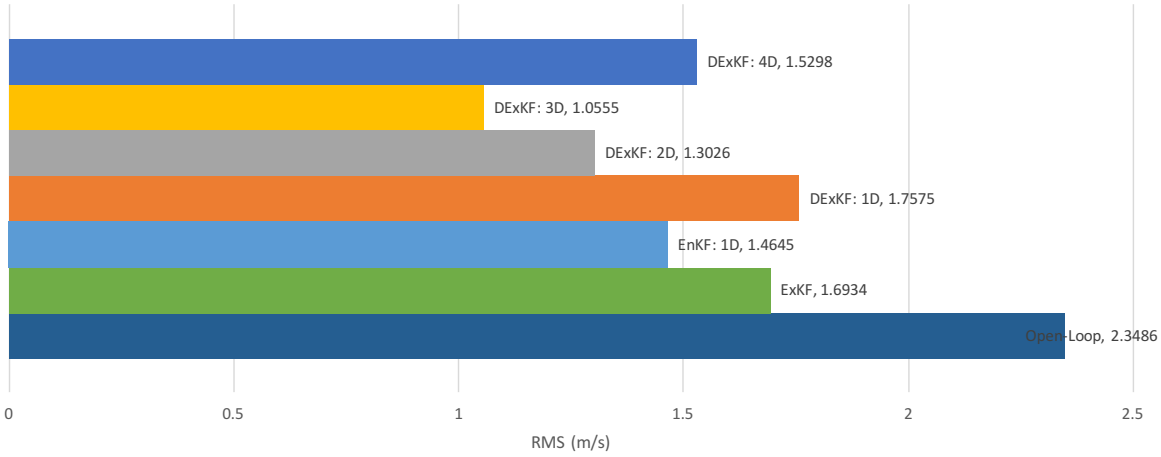


Figure 4-20: Mean centerline error between the SOWFA and distributed architecture 1 for various subsystem sizes (m/s).

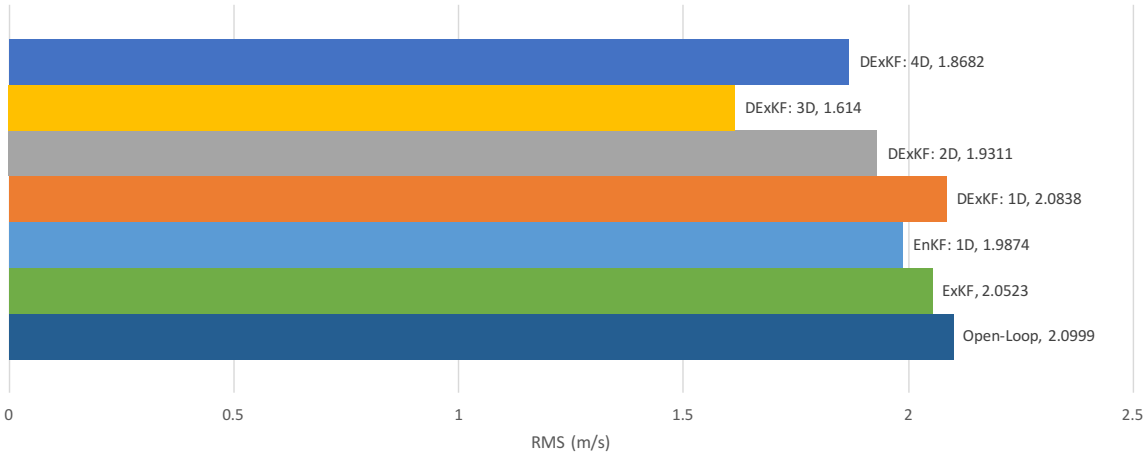


Figure 4-21: Mean flow error between the SOWFA and distributed architecture 1 for various subsystem sizes (m/s).

4-7 Distributed Architecture 2

In distributed architecture 2, a separate wind farm model is used for each turbine. Hence, the interactions between the turbines can not be modelled directly. Instead, an ad hoc correction for wake effects, using sensor measurements and fusion, need to be implemented. Here, simulations will be performed to analyze the flow fields obtained with and without fusion. Similar to the earlier simulations, all the simulations, here, are performed for a total simulation time of 1997 seconds. The plot of the flow-fields at the last iteration, 1997s, without fusion is shown in Figure 4-22, and with fusion is shown in Figure 4-23.

From the plots of the flow fields of the subsystems displayed in Figure 4-22, it can be inferred that an independent estimator used by each subsystem, without fusion, improves the estimation accuracy only around the turbine modelled by that subsystem. In other words, an estimator used by subsystem 1 improves the accuracy around turbine 1, while an estimator used by subsystem 2 improves the accuracy around turbine 2. The reason for this phenomenon is that the wind farm models used by either of the subsystems consider only one turbine and model the effects of just that turbine on the entire wind farm, though there are two turbines in the wind farm. In addition, each subsystem considers only the sensor measurements that are near the turbine that it models. Hence, in subsystem 1, upon interacting with the first turbine, the velocity of the wind flow recovers back to its free-stream flow velocity of 11 m/s and continues to remain there for the rest of the wind farm area, as subsystem 1 models only the effects of the first turbine on the wind farm area. However, in reality, the velocity of the wind flow drops once again, when it interacts with the second turbine. Hence, the estimation error of the first subsystem is minimized only for the states that are around turbine 1, as the wind farm model of subsystem 1 does not model the effects of the second turbine. Similarly, in subsystem 2, the wind flow remains at the free-stream flow velocity of 11 m/s until it interacts with the second turbine, as the first turbine is not modelled in subsystem 2. Hence, the estimation error of the second subsystem is minimized only for the states that are around turbine 2, while the error is high for the states that are around turbine 1. In short, employing

a dedicated wind farm model for each turbine results in the exclusion of the wake effects, though the primary need for a wind farm model is to model the interactions between the turbines. This issue can be tackled by employing a fusion algorithm. The flow fields shown by Figure 4-23 is the result of applying fusion. Basically, it fuses the region around turbine 1 from subsystem 1 with the region around turbine 2 from subsystem 2, thereby improving the overall estimation accuracy. This can be verified by analyzing the estimation accuracy of architecture 2 with fusion as displayed in Figure 4-24 and Figure 4-25. Also, architecture 2, with fusion, performs similar to the ExKF without localization.

Furthermore, in Section 4-1, it was introduced that assuming a smaller value for the initial velocity, like 5 m/s, will make the estimation error misleading. The reason is explained here. Since subsystem 1 does not model the effects of turbine 2, it will predict the velocity of the wind flow, behind turbine 2, to be the free-stream velocity. However, in reality, as shown by the flow field predicted by the SOWFA model in Figure 4-22, the velocity of the wind flow drops behind turbine 2 and will approximately be 5 m/s. This inability of subsystem 1 to capture the wind flow dynamics around turbine 2 needs to be conveyed by the estimation error plot. If the free-stream velocity is assumed to be larger than the actual value, like 11 m/s in Figure 4-22, subsystem 1 will predict the velocity behind turbine 2 to be 11 m/s, while the actual velocity is approximately 5 m/s as shown in Figure 4-22. Hence, the error plot will convey the fact that subsystem 1 does not model the effects of turbine 2. Instead, if the free-stream velocity was selected to be a smaller value, like 5 m/s, subsystem 1 will predict the velocity behind turbine 2 to be 5 m/s, not because it models the effect of turbine 2, but because of the free-stream velocity. In this case, the estimation error would be less, suggesting that subsystem 1 correctly predicts the effect of second turbine. However, in reality, it does not. Hence, the plot of the estimation error will be misleading. To avoid this, the free-stream velocity of the wind flow is assumed to 11 m/s for all the simulations.

Now, on comparing the computational complexity as displayed in Figure 4-26, it can be inferred that the computation complexity of architecture 2 increases because either of the estimators estimates all the states in both the wind farm models, like in the centralized architecture. Nevertheless, this architecture gives insights on the effects of considering only one turbine, and this would help in designing distributed architecture 4.

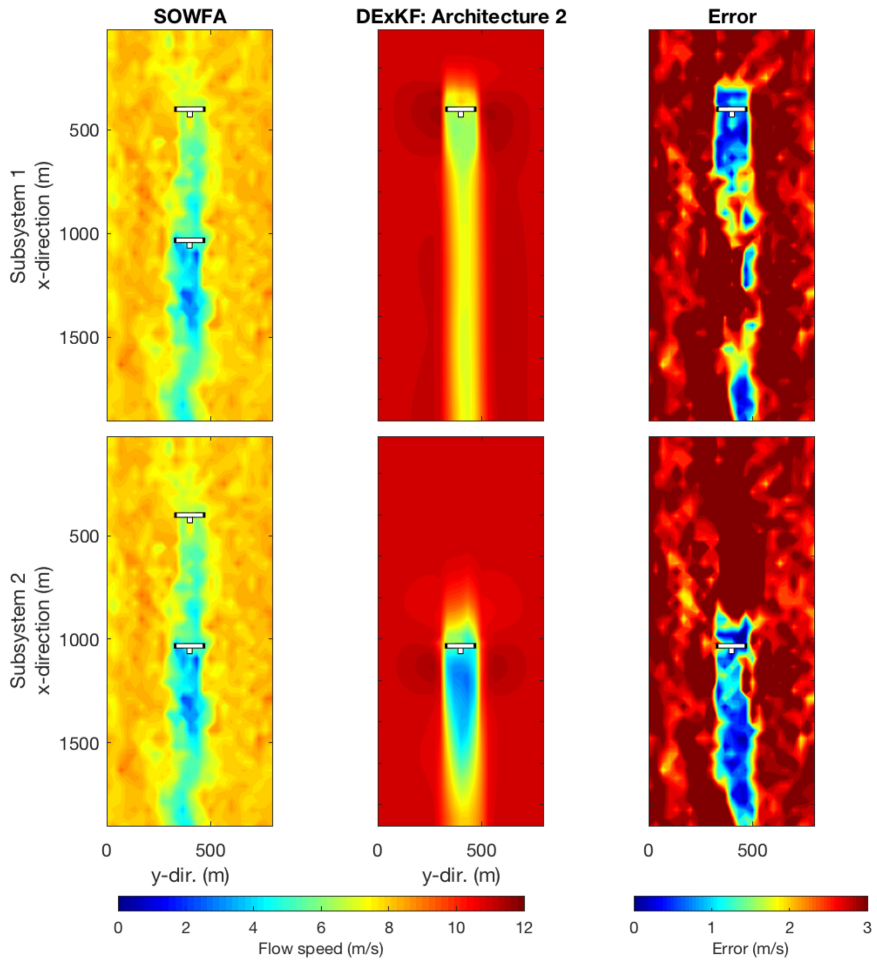


Figure 4-22: Snapshots of the longitudinal flow velocity (m/s) throughout the grid for distributed architecture 2 without fusion.

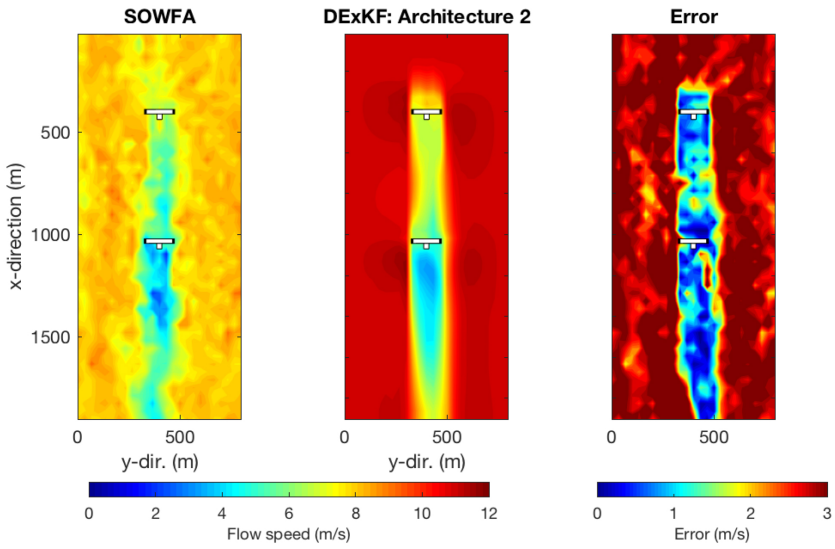


Figure 4-23: Snapshots of the longitudinal flow velocity (m/s) throughout the grid for distributed architecture 2 with fusion

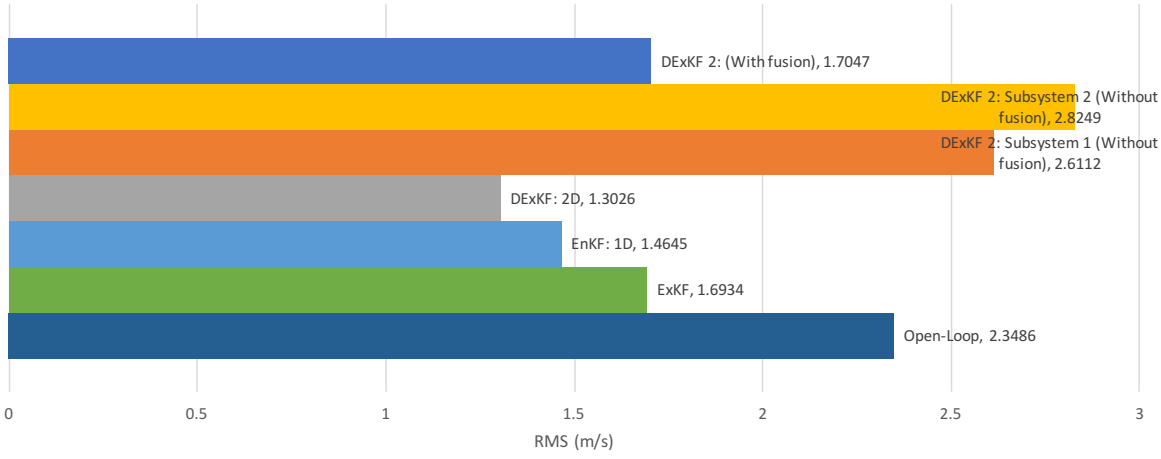


Figure 4-24: Mean centerline error between the SOWFA and distributed architecture 2, with and without fusion (m/s).

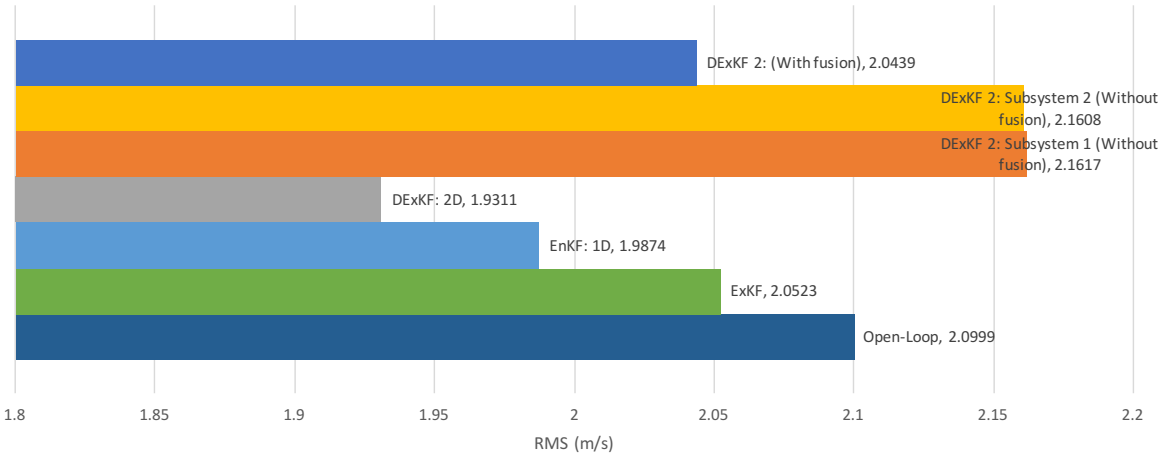


Figure 4-25: Mean flow error between the SOWFA and distributed architecture 2, with and without fusion (m/s).

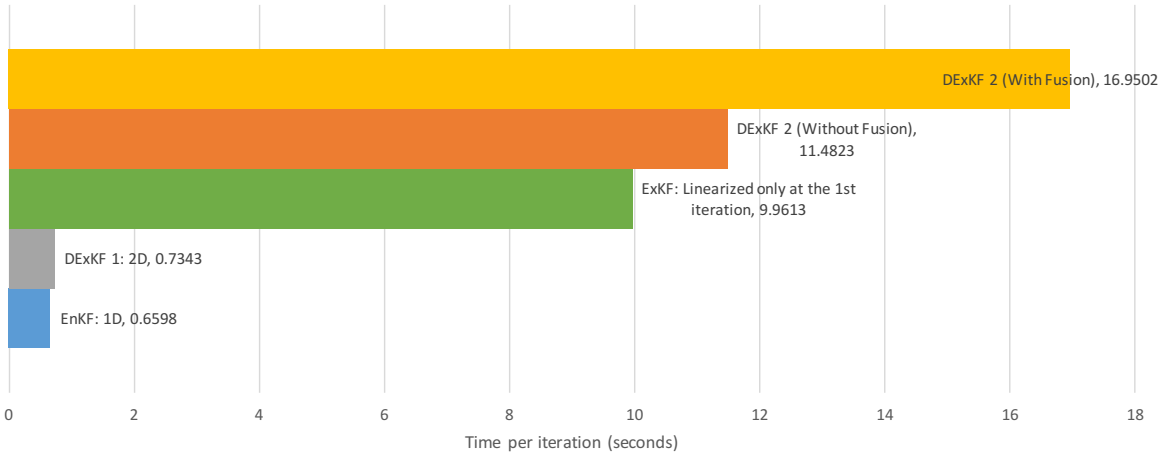


Figure 4-26: Computational complexity of distributed architecture 2 with and without fusion.

4-8 Distributed Architecture 3

Distributed architecture 3, for a subsystem size of $2D$,[7] is simulated for a total simulation time of 1997 seconds, and the plots of the flow-fields at the last iteration, 1997s, without fusion is shown in Figure 4-27 and with fusion is shown in Figure 4-28. Since distributed architecture 3 is a combination of distributed architectures 1 and 2, the plots of the estimates of the flow fields obtained for architecture 3 are almost similar to the plots obtained for architectures 1 and 2. Also, a similar response is observed on comparing the mean of the RMS error for distributed architecture 3, as displayed in Figure 4-29 and Figure 4-30. Now, on comparing the computational complexity between different architectures as shown in Figure 4-31, architecture 3 is computationally cheap. This is because only a selected number of states are estimated in architecture 3, as opposed to estimating all the states in architecture 2.

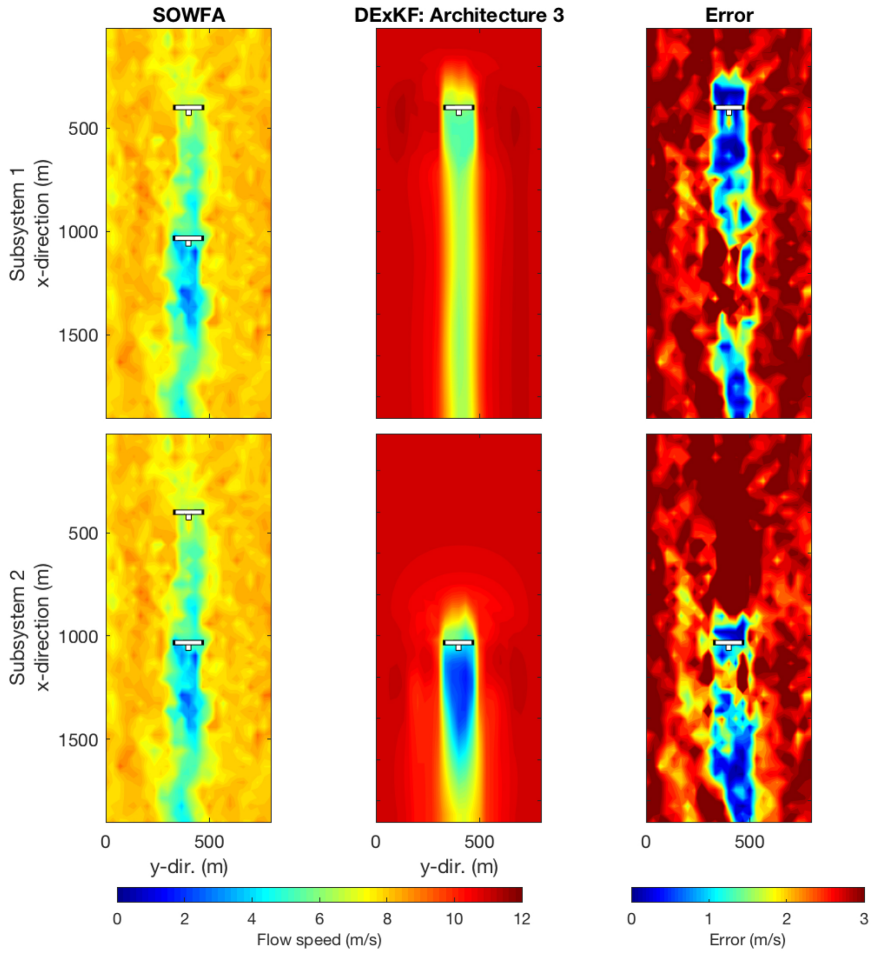


Figure 4-27: Snapshots of the longitudinal flow velocity (m/s) throughout the grid for distributed architecture 3 without fusion.

[7] Based on the computational complexity and estimation accuracy of distributed architecture 1 for different subsystem sizes, the size of the subsystem for distributed architecture 3 is selected to be $2D$.

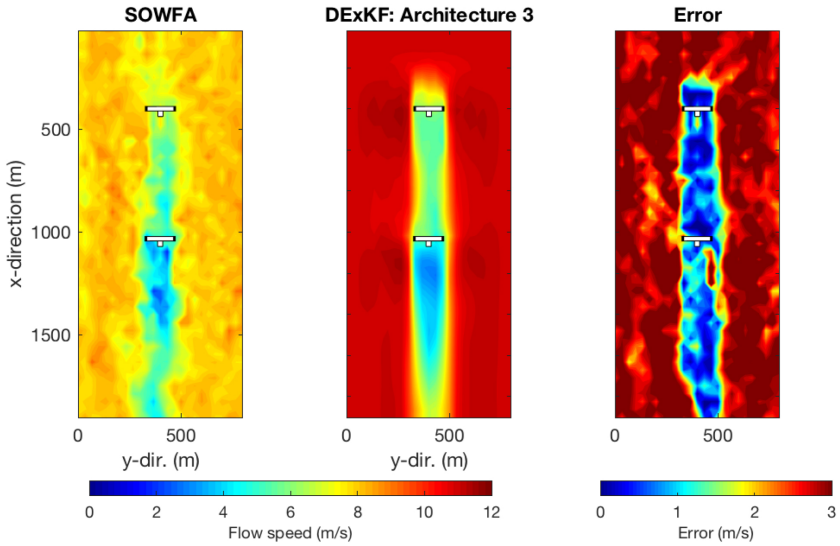


Figure 4-28: Snapshots of the longitudinal flow velocity (m/s) throughout the grid for distributed architecture 3 with fusion

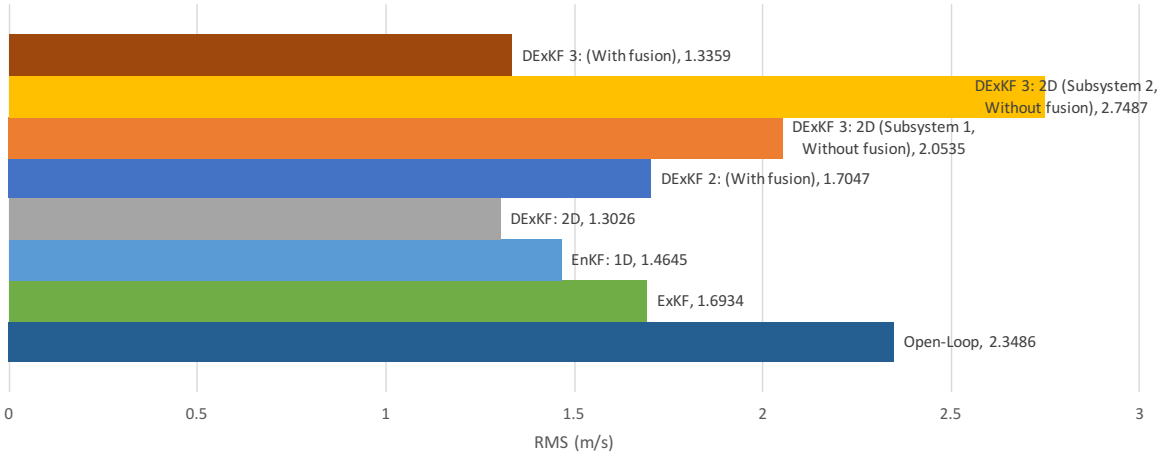


Figure 4-29: Mean centerline error between the SOWFA and distributed architecture 3, with and without fusion (m/s).

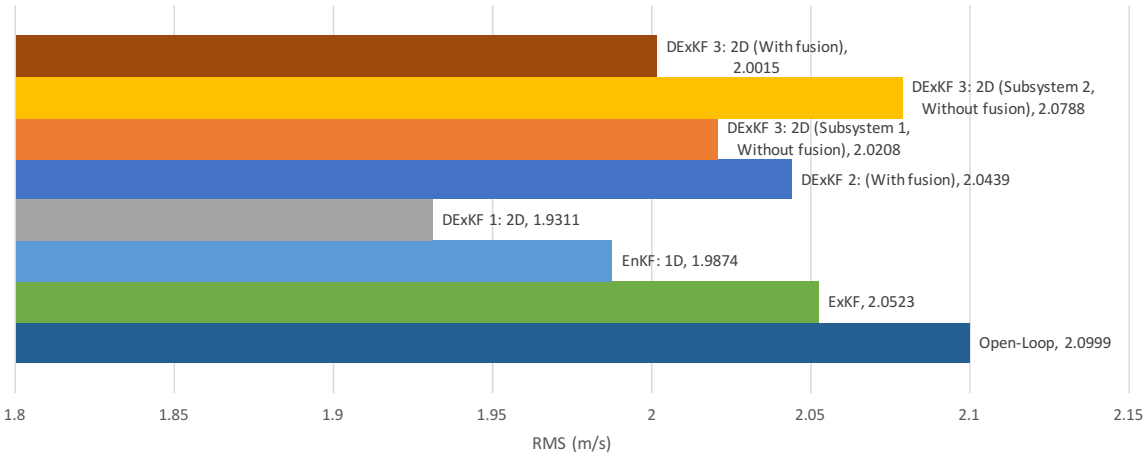


Figure 4-30: Mean flow error between the SOWFA and distributed architecture 3, with and without fusion (m/s).

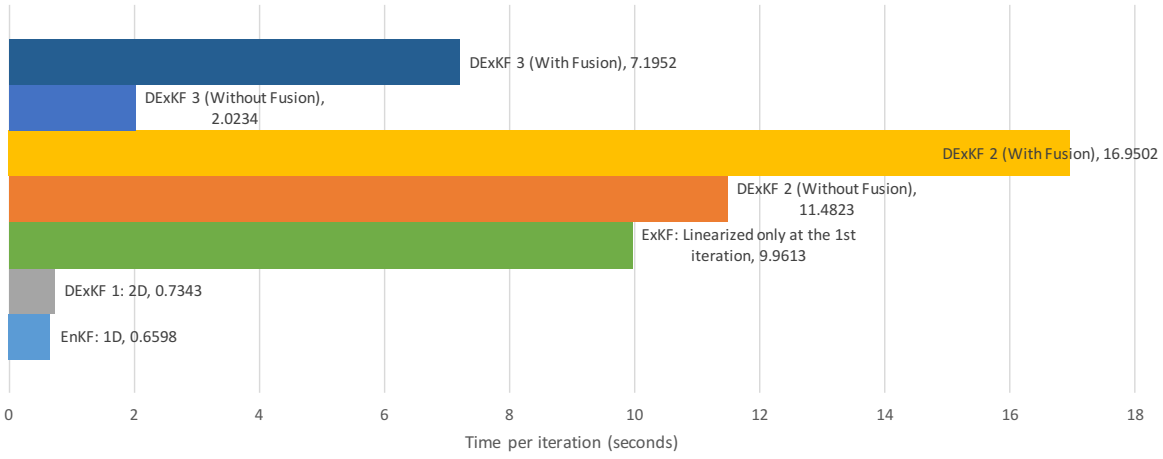


Figure 4-31: Computational complexity of distributed architecture 3, with and without fusion.

4-9 Distributed Architecture 4

Distributed architecture 4 is simulated for a total simulation time of 1997 seconds. Here, the spatial domain of the subsystems are selected to be a quadrilateral, so that they match the meshing being employed. Furthermore, the perimeters of the domain are taken to be at a distance of $4D$ from either of the turbines, so that the subsystems overlap. The plots of the flow fields at the last iteration, $1997s$, for an open-loop simulation is displayed in Figure 4-33, and a closed-loop simulation, with the ExKF, is displayed in Figure 4-34.

On comparing the plots of the flow fields between the open- and closed-loop simulations as displayed in Figure 4-33 and Figure 4-34, it can be inferred that the estimation accuracy of an open-loop model improves by employing a filter. Also, a similar response is observed on comparing the RMS error between the open- and closed-loop simulations as displayed in Figure 4-35 and Figure 4-36. Unfortunately, due to some numerical and implementation issues with fusion, and lack of time to fix the issue, fusion is not employed. Nevertheless, fusion can be employed to further improve the estimation accuracy, especially in the overlapping region.

Now, on comparing the computational complexity of distributed architecture 4 with other architectures as displayed in Figure 4-32, it can be inferred that architecture 4 is computationally cheaper than the centralized ExKF and distributed architectures 2 and 3. However, architecture 4 is still complex when compared to distributed architecture 1 and the centralized EnKF. More importantly, each iteration in architecture 4 still takes more than one second. Nevertheless, when compared to other architectures, architecture 4 has a better scope for improvement because a controller designed using architecture 4 will be computationally very cheap when compared to other architectures. This is because the size of the spatial domains over which the wind farm models predict is small. In other words, the size of the wind farm models in architecture 4 are small when compared to other architectures. However, the performance of such a controller is doubtful.

Note: Due to the lack of time, distributed architecture 4 is only in its earliest stages. Hence, the simulation plots of architecture 4 could still be improved.

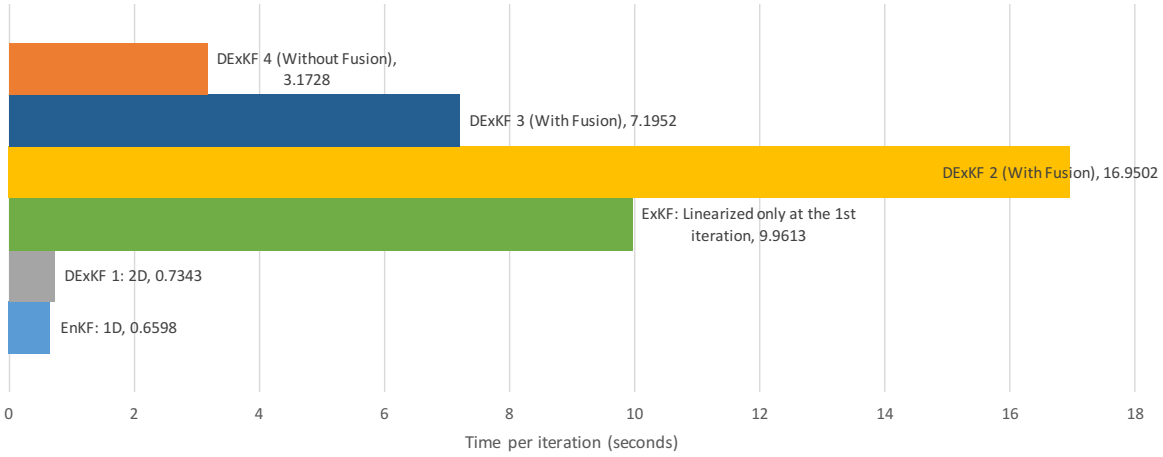


Figure 4-32: Computational complexity of distributed architecture 4.

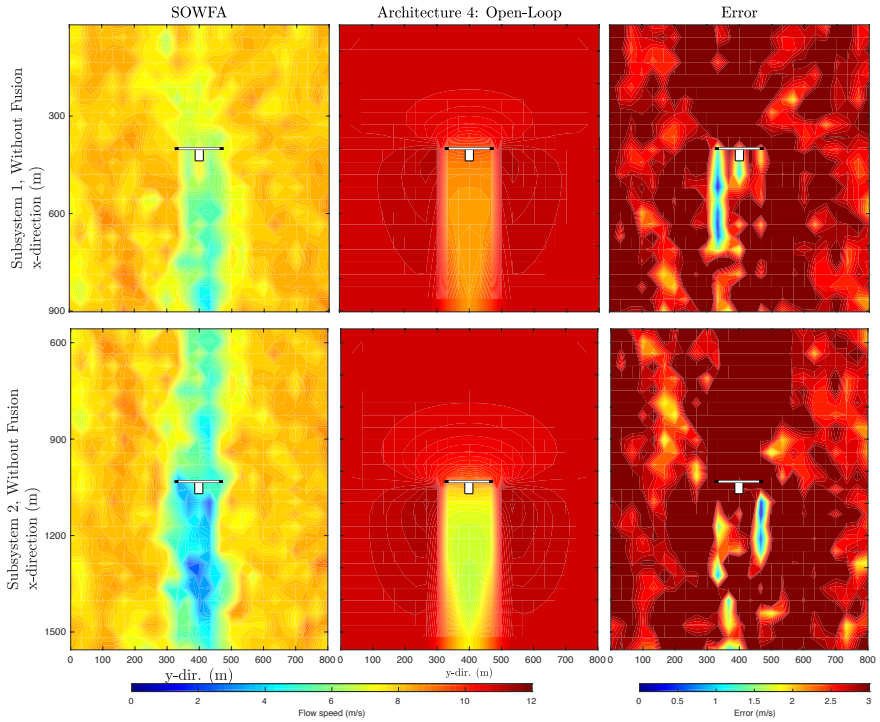


Figure 4-33: Snapshots of the longitudinal flow velocity (m/s) for an open-loop simulation following distributed architecture 4.

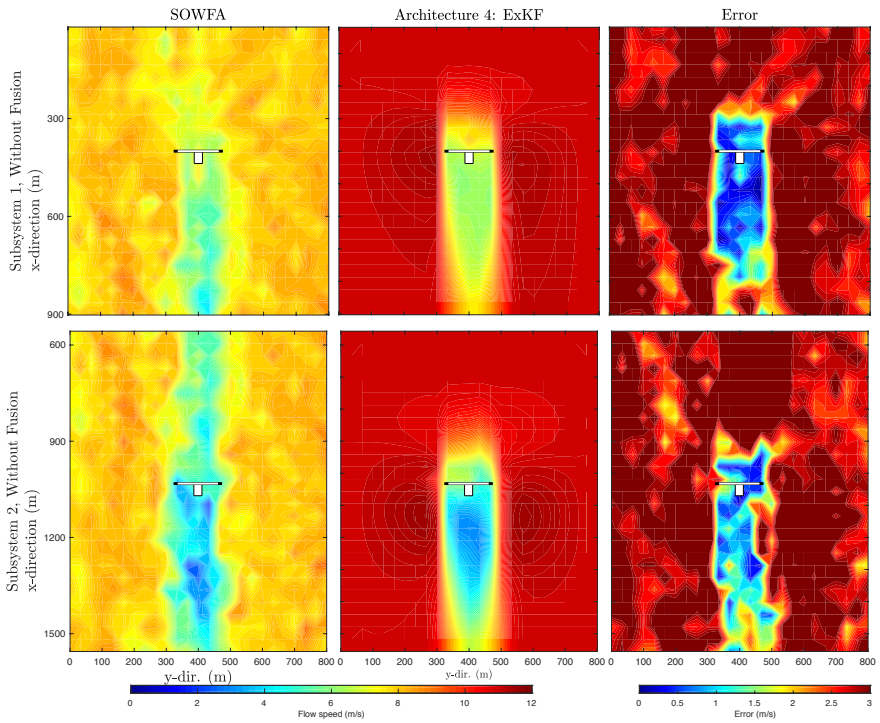


Figure 4-34: Snapshots of the longitudinal flow velocity (m/s) for theExKF, without fusion, following distributed architecture 4.

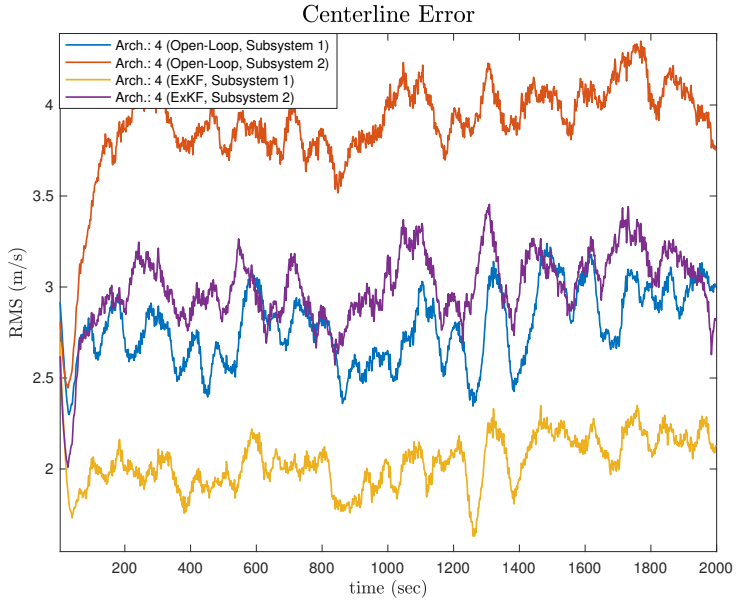


Figure 4-35: Centerline error between the SOWFA and distributed architecture 4, without fusion, over time (m/s).

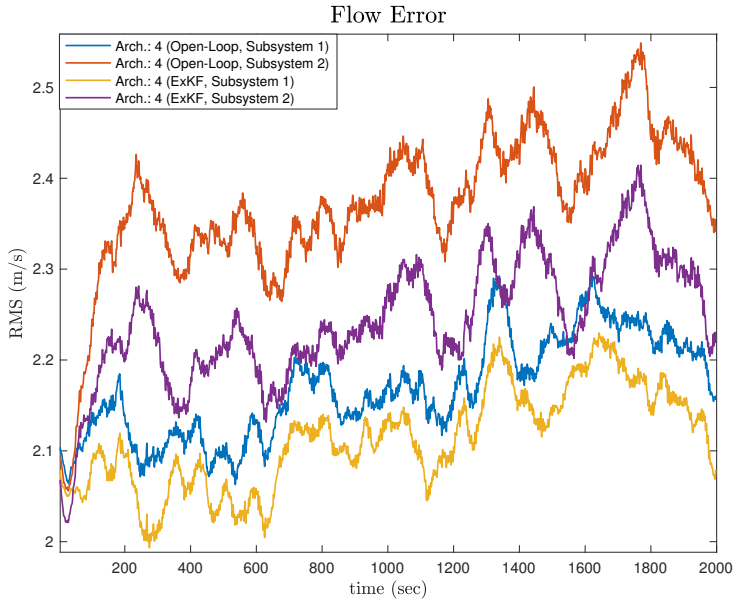


Figure 4-36: Flow error between the SOWFA and distributed architecture 4, without fusion, over time (m/s).

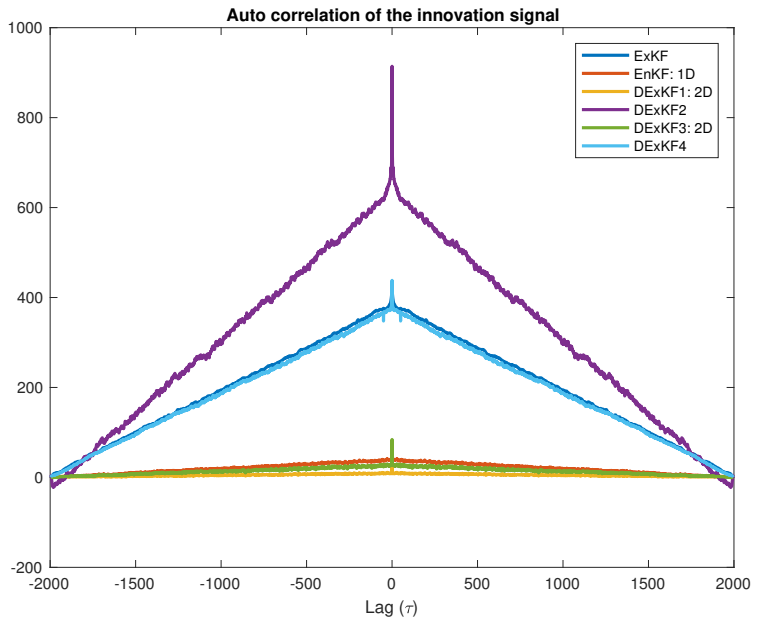


Figure 4-37: Auto correlation of the innovation signal, over time, for different filters.

4-10 Optimality of the filters

Until now, the estimation algorithms were compared in terms of the accuracy of the estimates. In this section, the optimality of the filters will be analyzed.

Basically, WFSim is a non-linear system. This implies that the Gaussianity can not be not preserved, even though the state vector and noises are assumed to be Gaussian. In other words, the true distribution of the state vector will have non-zero higher moments. However, the filters designed here will try to capture only the first two moments of the true distribution, as these are Kalman Filter (KF)-based estimation algorithms. Hence, none of the filters designed here are optimal.

In cases like this, the filters can be just checked for the minimum-variance and unbiasedness properties. These properties can be determined by investigating how close the estimated states are to the actual states. However, in reality, sensor observations are the only knowledge available on the actual state vector. Hence, the difference between the actual and estimated outputs is typically employed for testing the minimum-variance and unbiasedness of a filter, instead of using the difference between the actual and estimated states. In addition, the difference between the actual and estimated outputs is called the innovation signal (see Appendix B).

The KF is said to be minimum-variant and unbiased if the innovation signal is a white noise. This implies the KF has achieved the smallest estimation error co-variance among all the unbiased estimators. In other words, all the available information will be extracted by the filter, if the innovation signal is uncorrelated over time.

Now, on analyzing the plot of the auto correlation of the innovation signal for different filters, as displayed in Figure 4-37, it can be inferred that the innovation signals of the

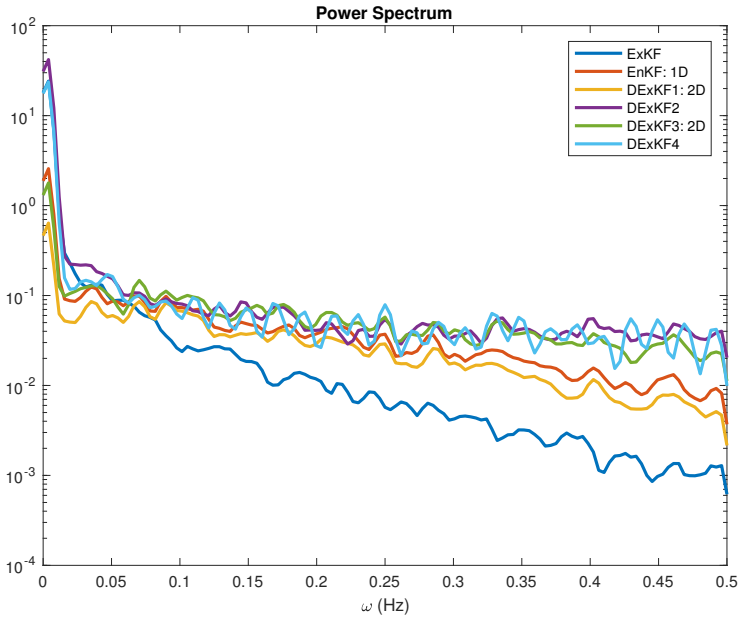


Figure 4-38: Power spectrum of the innovation signal for different filters.

centralized EnKF, and the ExKFs following distributed architectures 1 and 3 are close to being white. Also, a similar performance is observed on inspecting the power spectrum of the innovation signal of these filters, given by Figure 4-38. These responses imply that the estimation algorithms have almost achieved the smallest estimation error co-variance among all the unbiased estimators. Nevertheless, the innovation signal is not exactly white. This is because of the model mismatch between the estimator model (WFSim) and the actual plant (SOWFA), i.e., the SOWFA model, which is used for testing the performance of the estimators, is inherently different from the WFSim model. Consequently, the innovation signal is biased, and not white.

In addition, on comparing the auto correlation for the ExKFs following centralized architecture and distributed architectures 2 and 4, it can be inferred that the innovation signal of these estimators are far from being white. This implies that an estimator estimating all the states performs worse than an estimator estimating only the states that are nearby the sensor measurements. A possible reason could be the inclusion of unobservable states for measurement-update. In other words, as the distance of the states increases from the sensor measurements (turbines), less information is known about these states [48], and the cross correlation between the measured states and the unmeasured states are the only way by which these distant states will be measurement-updated. Hence, the correlation terms seem to cause detrimental effect due to the unobservability of the distant states.

Moreover, since the distant states in distributed architectures 1 and 3 are only time-propagated and the centralized EnKF directly considers the non-linear system and is also localized, the innovation signals of these estimation algorithms are far more white than that of the ExKFs following centralized architecture and distributed architectures 2 and 4.

However, it would be interesting to analyze the optimality of the filters, when an ideal WFSim

model, instead of SOWFA, will be used for testing the estimation algorithms.

4-11 Answers to the Research Questions

Numerous estimation algorithms were designed in the previous chapter for improving the accuracy of the WFSim model while maintaining the computational complexity. In this chapter, simulations were carried out to analyze the performance of these estimation algorithms. In this process, four research questions were framed. These questions are answered in this section:

1. **How often should the WFSim model be linearized for employing the ExKF in both the centralized and distributed frameworks? Can the frequency of linearization be reduced?**
   We have seen in Section 4-4 that changing the frequency of linearization does not affect the estimation accuracy of the ExKF as long as the atmospheric conditions do not change. Therefore, for a constant atmospheric condition, we conclude that the non-linear WFSim can be linearized just at the first iteration, and the obtained linearized model can be used for the remainder of the time-steps (iterations) of the ExKF simulations.

2. **What are the effects of applying localization on the centralized filters? How does the localization size affect the performance?**
   We have seen in Section 4-5 that localization helps in restricting the number of states that are measurement-updated. In the EnKF, increasing the localization length has no prominent effect on the estimation accuracy. On the other hand, the estimation accuracy of the ExKF increases with the localization size until a certain size. After that, the accuracy saturates.

3. **How small or big should the size of the subsystems in the distributed architectures be?**
   We have seen in Section 4-6 that the estimation accuracy of distributed architecture 1 increases with the size of the subsystems. After a certain size, the accuracy declines and finally, saturates. Also, the computational complexity increases with the size of the subsystems. Hence, based on the simulation results, we conclude that subsystem size of $2D$ is a good choice for this simulation scenario, as it offers both good estimation accuracy and computation tractability.

4. **What happens when no measurements or noisy measurements are available? In this case, how can the interactions be taken into account?**
   Typically, all the subsystems in a distributed architecture are implemented in parallel, and data fusion will be employed, before the start of the next iteration, to include the interactions between the turbines. However, to implement fusion, accurate observations are required. In case the observations are not available or not accurate, the subsystems can be implemented in series, and the flow field predicted by the upstream subsystems can be used as the prior for the subsequent downstream subsystems. That is, the flow predicted by the upstream subsystems for the overlapping region can be considered as the measurements for the immediate-downstream subsystems, with the downstream

subsystems estimating at a delay of a sampling instance with the upstream subsystems. The mathematics behind this idea is presented in Appendix C. Hence, the need for sensor observations can be circumvented using the method from Appendix C, to include the interactions between the turbines. However, due to the lack of time, this approach could not be implemented, and is recommended for the future work.

Chapter 5

Conclusions and Recommendations

The motivation for the thesis was to increase the overall power production of a wind farm, by implementing model-based wind farm control. To achieve the desired level of controller performance, the wind farm models needed to be accurate. Consequently, the goal of the thesis was to design an estimation algorithm to both accurately predict the wind flow in a wind farm and maintain the computational tractability. In the current research, the centralized estimation approaches were employed. Instead, in this thesis, the aforementioned objective was addressed by distributed estimation. In this thesis, simulations were carried out to test the performance of the newly-devised distributed architectures against the already-existing centralized architecture. Furthermore, the medium-fidelity wind farm model WindFarmSimulator (WFSim) was used for designing the estimation algorithms, and the high-fidelity wind farm model Simulator fOr Wind Farm Applications (SOWFA) was employed for testing the performances of these estimation algorithms.

In Section 5-1, the conclusions are outlined. Then, based on the obtained simulation results, recommendations for the future research in this field of work are presented in Section 5-2.

5-1 Conclusions

From the knowledge gained on working in the thesis, the following conclusions are made:

1. **Data-driven wind farm models can be employed for model-based wind farm control, in place of the high-fidelity models.**
   Generally, a wind farm model has to be accurate, to achieve the desired control objectives like increasing overall power production in a wind farm. Though the high-fidelity wind farm models offers high accuracy, the problem with these models is their size. Owing to the large size, these models are not suitable for real-time control. Hence, closed-loop approach, using medium-fidelity model, was presented as a solution to this problem, in this thesis. Simulations show that employing an estimator in close co-operation with a wind farm model helps it to adapt to the varying atmospheric conditions, thereby increasing the accuracy of the medium-fidelity wind farm model.

2. **Irrespective of the estimation algorithms being employed, the distributed architectures are computational tractable than the centralized architecture.**
In centralized estimation, a single estimator is used to estimate all the states of a large-scale system. This makes the computational complexity of the centralized estimation to depend upon the estimation algorithm. That is, the Ensemble Kalman Filter (EnKF), which computes the sample co-variance, takes less time per iteration to achieve the same level of accuracy as the Extended Kalman Filter (ExKF), which computes the actual co-variance. On the other hand, the distributed architecture is based on distributing a large-scale system into a number of small-scale systems, and employing a number of estimators to collectively estimate all the states in parallel. Thus, the size of the states, over which each estimator operates, is small, making the distributed estimation computationally tractable, irrespective of the estimation algorithms. In addition, simulations show that the distributed architectures are computationally cheaper than the centralized architecture.

3. **Unobservable states affect the optimality of the filters.**
In the wind farm model WFSim, the observablity of the states depends upon the distance of the states from the sensor observations. That is, as the distance of the states increases from the observations, less information is available on the states, rendering the distant states unobservable [48]. Thus, a filter estimating all the states, including the observable (nearby) and unobservable (distant) states, is incapable of reducing the mean of the estimation error to zero (biased), and the variance is not minimum. Consequently, the optimality of filter is affected by the observabililty of the states, implying that an estimator estimating only the states that are nearby the sensors, as in distributed architecture 1, performs better than an estimator estimating all the states, as in the centralized architecture.

4. **Distributed architecture 4 is recommended for closed-loop wind farm control.**
Distributed architecture 4 defines a number of small-scale wind farm models, to collective predict the wind flow throughout a wind farm. Hence, a separate controller can be implemented for each subsystem in parallel, and the computational complexity of such a controller will be cheap, as each controller will act on a small-scale wind farm model. Consequently, distributed architecture 4 is an interesting option for controller design. However, the performance of such a controller can be debatable.

5-2 Recommendations

This thesis focused on improving the accuracy of the medium-fidelity wind farm models. More importantly, it succeeded in addressing this issue, by designing estimation algorithms which offer accuracy at a tractable computational cost. If this work is continued, future researches conducted in this area can focus on answering the following questions:

1. **Effects of dynamic free-stream inflow**
The simulations carried out in this thesis assumed the velocity of the inflow wind to be constant (constant atmospheric conditions). Consequently, it was concluded that

the WFSim model can be linearized just at the first iteration for the ExKF, as the estimation accuracy would remain the same. However, in the future, a varying inflow velocity can be assumed, and its effects on the linearization and estimation accuracy of the ExKF can be studied.

2. **Performance of the distributed architectures without measurements or with noisy measurements**
   In this thesis, simulations showed that the stand-alone distributed architectures 2 and 3 do not model the interactions, and fusion algorithms using sensor measurements are employed to address this issue. Hence, in the future, the effects of noisy measurements or no measurements on the distributed architectures can be analyzed.

3. **Performance of the alternate approach proposed in Appendix C for incorporating the interactions between the turbines**
   In Appendix C, an alternate approach to data fusion for including the interactions between the turbines was proposed. However, it could not be tested due to the lack of time. Hence, the performance of this hypothesis can be tested, if this work is continued in the future.

4. **Performance of the controllers following distributed architecture 4**
   In distributed architecture 4, the size of the wind farm models are small. Hence, a model-based controller designed following this architecture will be computationally tractable. However, the performance of such a controller is doubtful. Hence, the performance of a controller following architecture 4 could be analyzed.

5. **Effects of increasing the number of sensor observations on the distributed architectures**
   For the simulations conducted in the thesis, sensors observations were placed only near the turbines. This caused the estimation accuracy of the centralized estimation algorithms to saturate after a certain localization size. Consequently, the distributed architectures were able to estimate the flow field of a wind farm to the same level of accuracy, even though it did not estimate the states that were far from the turbines. This leads to a question: If the number of sensor observations are increased and if the states that are far away are also estimated, will the distributed architectures still perform equal to the centralized architecture?

6. **Performance of the distributed architectures against an ideal WFSim model, instead of SOWFA**
   For the simulations conducted in the thesis, the performances of the estimation algorithms designed using WFSim were tested against SOWFA. Due to the inherent differences between the WFSim and SOWFA models, the inferences on the performances of the estimation algorithms were not able to be made with absolute certainty. Hence, in the future, the estimation algorithms can be tested against an ideal WFSim model, and the inferences made in this thesis can be verified.

7. **Performance of the static filters**
   Generally, on applying the Kalman Filter (KF) to an Linear time-invariant (LTI) system, the error co-variance of the estimate converges with the Kalman gain [49]. Since the error co-variance and the Kalman gain converge, they need not be computed, i.e.,

the converged Kalman gain can be employed for estimation from the time-step they converge. Consequently, the computationally tractability of the estimation algorithms can be further improved, as the resourceful steps like the computation of the error covariance and Kalman gain can be circumvented. Hence, in the future, these static filters can be implemented, and their performance in terms of optimality can be analyzed.[1]

8. **Distributing the non-linear filters**
   The distributed architectures devised in the thesis were only tested with the ExKF. In the future, these architectures can be implemented with the other non-linear filters, like the EnKF and Unscented Kalman Filter (UKF).

9. **Effects of fusion algorithms on the distributed architectures**
   There are numerous data fusion algorithms available in the literature. Hence, in the future, the effects of various fusion algorithms on the distributed architectures can be studied.

[1] Actually, these static filters were implemented during this thesis, and the initial simulations showed promising results. However, due to the lack of time, they could not be analyzed, in depth. Consequently, the results are not included in this report. Nevertheless, an in-depth analysis on this topic can be conducted in the future.

Appendix A

Basics of Wind Farm Modeling

This chapter focuses on the state-of-the-art in wind farm modeling. The chapter starts by discussing the different components of a wind farm model. It then gives an overview on each of these components in detail from Appendix A-1 to Appendix A-3.

Modeling a wind farm is more challenging than modeling a single wind turbine because of the interactions between the turbines through the wakes formed behind each turbine. In other words, a single wind turbine model predicts only the aerodynamic forces between the flow and the rotors [10], whereas a wind farm model describes the turbine interactions in addition to the aerodynamic forces [4]. In short, a wind farm model can be considered as a collection of the following elements (models):

- **Flow Model**: a mathematical model that describes the flow behaviour within a wind farm.
- **Turbulence Model**: a model that is used to close the flow models by calculating the unknown Reynolds stress terms. In others words, it describes the fluctuating part of the flow, namely the turbulence.
- **Rotor Model**: a model that describes the aerodynamic forces between the flow and the rotors.
- **Turbine Model**: a model that describes the effect of structural loading on a turbine.

Depending on the type of flow model being employed, a turbulence model may or may not be needed. Furthermore, not all wind farm models feature a turbine model. In the upcoming sections, components of wind farm model are briefly explained. For more information on wind farm models and its components, the reader is referred to the article by Sanderse *et al.* [50].

Note: Turbine modeling is not touched upon in this report, as the WindFarmSimulator (WFSim) model does not include a turbine model and also the control objective of this research does not directly involve minimizing the structural loading. However, for information on turbine modelling, the reader is referred to an article by Moriarty and Butterfield [51].

A-1 Flow Model

Flow models range from simple analytical models describing just the most dominant characteristics of a flow to complex Navier-Stokes (NS) based computational fluid dynamics (CFD) models predicting the entire flow field. Usually, flow models are classified based on the fidelity. However, in this thesis, flow models are organized based on the principle of the model as this helps in understanding the models better.

A-1-1 Kinematic Models

Kinematic models, otherwise known as parametric models, are simple steady-state models which describe only the most dominant behaviours in a wind flow. Some of the famous kinematic models are the Jensen [52] and FLOw Redirection and Induction in Steady-state (FLORIS) [53] models.

The Jensen model approximates a downstream wake as a turbulent flow by ignoring the deterministic vortices that occur in the near field. Based on this assumption, the Jensen model predicts the dimension of the expanding wake to be proportional to its distance from the turbine [52].

An extension to the Jensen model is the FLORIS model. The FLORIS model uses the Jensen model in association with a yaw-based wake deflection model to predict the static effects of misaligning the turbine yaw on the power generated. This results in a better match with the power measurements of SOWFA, a high-fidelity LES model. For more information on the Jensen and FLORIS models, the reader is referred to the articles by Jensen [52] and Gebraad *et al.* [53].

A-1-2 Direct Numerical Simulation Flow Models

The Direct Numerical Simulation (DNS) flow models are NS based CFD models. Unlike the kinematic models which describe only the most dominant wake properties like velocity deficit, these models resolve (compute) the entire flow field represented by the NS equations. Hence, the DNS flow models predict the wake more precisely than any other flow models. However, the DNS flow models are computationally impossible or theoretical for high Reynolds number flow. This is because the range of scales in the flow increases with the increase in the Reynolds number. This demands a finer mesh size to resolve the entire flow field, thereby making the computation impossible. Hence, the DNS flow models are computationally possible only for low Reynolds number flow [50].

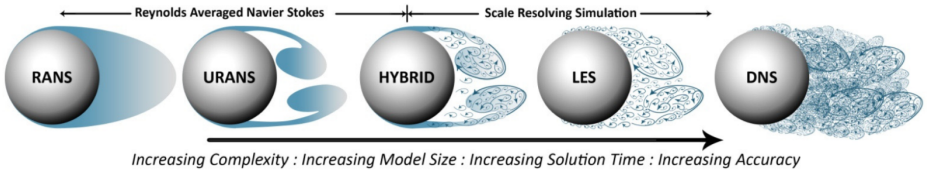


Figure A-1: NS based CFD flow models resolved using different spatial scales. [8]

In case of a high Reynolds number flow, a more practical solution to describe the entire flow field is to either resolve only the time independent part as in the Reynolds-Averaged Navier-Stokes (RANS) flow models or resolve only the large-scale eddies as is the case with the Large Eddy Simulation (LES) flow models [54]. Figure A-1 demonstrates the different techniques employed to describe the entire flow field represented by the NS equations. These methods are discussed in the upcoming sub-sections.

A-1-3 Reynolds-Averaged Navier-Stokes based Flow Models

The RANS based flow models are NS based CFD models. The basic idea behind the RANS models is that the mean of a turbulent flow is more important than its fluctuating part, as the mean of the turbulent flows are reproducible [54]. Thus, the RANS models decompose the flow into mean (time independent component) and fluctuating parts (time varying component).

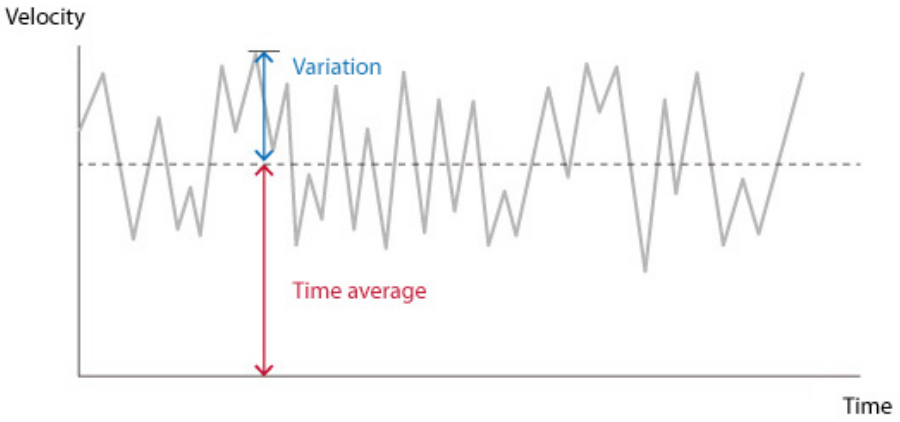


Figure A-2: The Reynolds decomposition performed on the instantaneous velocity of a turbulent flow. [9]

On decomposing an instantaneous quantity $u_i(x,t)$, we obtain

$$\mathbf{u}(\mathbf{x},t) = \underbrace{\overline{\mathbf{u}}(\mathbf{x})}_{\text{time independent}} + \underbrace{\mathbf{u}'(\mathbf{x},t)}_{\text{time varying}}, \tag{A-1}$$

where $\mathbf{u}$ can be any physical variable such as velocity or pressure, $\mathbf{x} = \begin{pmatrix} x & y & z \end{pmatrix}^T$ refers to the spatial co-ordinates, t the time, $\overline{\mathbf{u}}(\mathbf{x})$ the mean and $\mathbf{u}'(\mathbf{x},t)$ the fluctuating part. Figure A-2 demonstrates the application of the Reynolds decomposition on the flow velocity.

On analyzing Eq. (A-1) closely, the mean part is time independent, while the fluctuating part is time varying. In addition, a RANS model resolves only the mean part, while the fluctuating part is approximated by employing a turbulence model. Hence, the RANS model is computationally feasible.

For more information on the RANS flow models, the reader is referred to the book by Wilcox [55] and the lecture materials by Hulshoff [54] and Davidson [56].

A-1-4 Large Eddy Simulation based Flow Models

The LES based flow models are NS based CFD models. These models resolve the large-scale eddies, while the small-scales are modelled. In other words, the time varying components (the turbulence) are partly resolved.

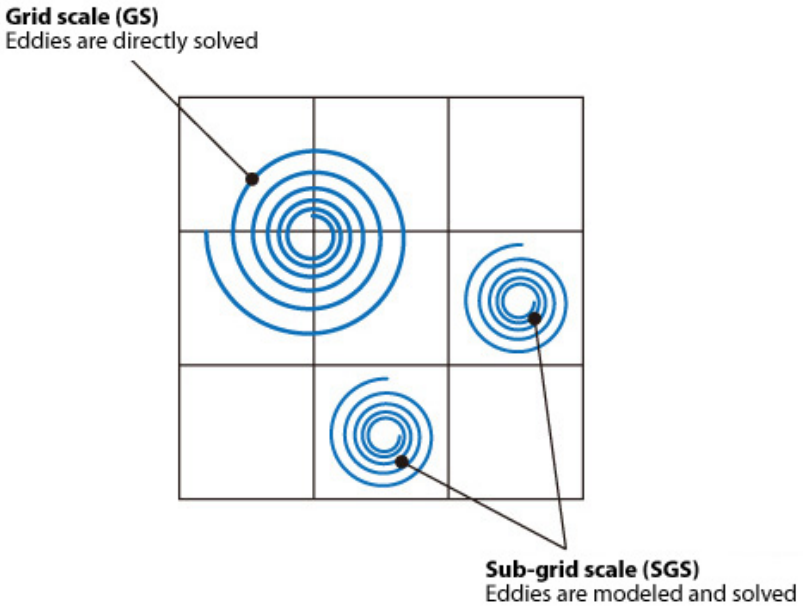


Figure A-3: Spatial decomposition performed on a turbulent flow. [9]

The LES models employ a filter to spatially decompose the flow into large- and small-scale eddies. On spatially decomposing a physical variable, we get

$$\mathbf{u}(\mathbf{x},t) = \underbrace{\tilde{\mathbf{u}}(\mathbf{x},t)}_{\text{time varying}} + \underbrace{\mathbf{u}'(\mathbf{x},t)}_{\text{time varying}} , \tag{A-2}$$

where $\mathbf{u}(\mathbf{x},t)$ refers to an unfiltered physical variable, $\tilde{\mathbf{u}}(\mathbf{x},t)$ the filtered large-scale physical variable and $\mathbf{u}'(\mathbf{x},t)$ the sub-grid physical variable. Figure A-3 demonstrates the application of spatial decomposition on a turbulent flow.

More importantly, the implementation of the LES based flow models differ significantly from that of the RANS based flow models, though these models might seem similar. In the RANS based flow models, the fluctuating part of the flow is completely modelled by the turbulence models. Thus, a single time-independent computation is sufficient to compute the time independent component (the mean) of the flow. On the other hand, in case of the LES flow models, both the parts that are being resolved and modelled are time dependent. Thus, time-domain simulations (online) are required to resolve the large-scale eddies [54]. Hence, the LES flow models are computationally complex when compared to the RANS flow models. However, the LES based flow models describe the flow more accurately than the RANS models. This is because the entire time varying component (the turbulence) is modelled in the RANS models

and none of the turbulence models represent the stress tensor robustly [50]. Thus, for applications where accuracy is essential but the LES models are not computationally viable, hybrid models are developed. These models use combination of the LES and RANS models. Some of the hybrid models are Unsteady RANS (URANS) and Detached Eddy Simulation (DES). For more information on the hybrid models, the reader is referred to the article by Hart [8] and the reader material by Hulshoff [54].

A-2 Turbulence Model

As explained in the previous section, the extent to which a flow model is resolved (computed) varies. In case of the DNS flow models, the entire flow field is resolved. Hence, it does not require a turbulence model. However, in case of the RANS and LES based models, the flow field is only partially resolved and the stress terms need to be modelled. Hence, the objective of the turbulence models is to either approximate the entire time-varying component or just the small-scale eddies, depending upon the type of flow models being employed.

A widely adopted technique to model the stresses employ the eddy viscosity assumption, postulated by Boussinesq in 1877 [50]. According to the Boussinesq assumption, the stress tensor is proportional to the mean velocity gradients with the eddy viscosity as the proportionality constant [56]. On applying the Boussinesq assumption, the stress is given by

$$\tau = \nu_T\left(\nabla\overline{u} + (\nabla\overline{u})^T\right), \tag{A-3}$$

where ν_t is the eddy viscosity.

For more information on the turbulence models, the reader is referred to the book by Wilcox [55].

A-3 Rotor Model

In the previous section, the interactions between the closely placed turbines were modelled by describing the flow field within the wind farm. The main reason behind the generation of the so-called wake interactions are actually the force that the wind flow exerts on a wind turbine and the reactive force that a turbine exerts back on the flow. These aerodynamic forces are described by the rotor models. There are two major rotor models, namely the Actuator Disk Model (ADM) and the Blade Element Model (BEM). This report restricts itself to the ADM, as it is employed in WFSim.

A-3-1 Actuator Disk Model

The ADM is a relatively simple description of the aerodynamic forces between a turbine and the flow. This model is based on the momentum theory. It represents the turbine as an infinitely thin actuator disk, as shown in fig. A-4. An actuator disc generates electrical energy by extracting part of the kinetic energy from the wind. Thus, the velocity of the downstream flow, $U_{-\infty}$, is necessarily lower than that of the upstream wind U_{∞}. According to the law of conservation of mass, the mass flow rate of an incompressible flow is same everywhere

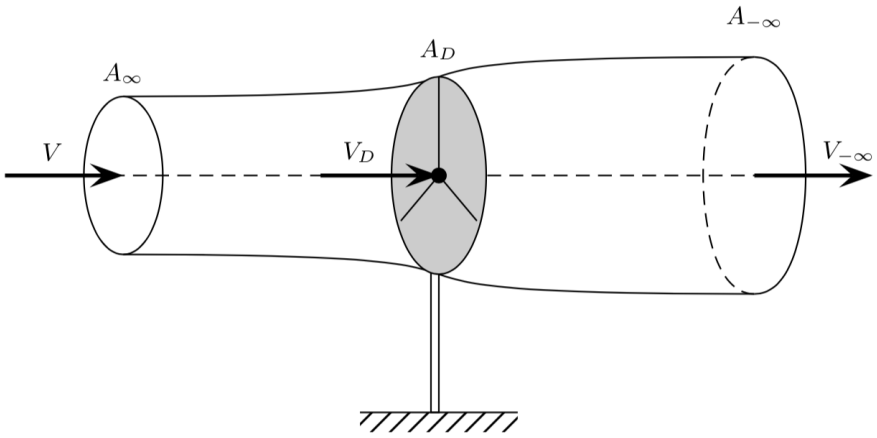


Figure A-4: The ADM predicts the aerodynamic force exerted by and on the rotor by considering the turbine as an actuator disc [10]

from a turbine. Hence, the cross-sectional area of the flow at the downstream increases to counter the decrease in the downstream flow velocity, and to maintain a constant mass flow rate everywhere. This can be represented by

$$\rho U_\infty A_\infty = \rho U_D A_D = \rho U_{-\infty} A_{-\infty}, \tag{A-4}$$

where ρ is the density of the air, U_D the velocity at the disk and $A_\infty, A_D, A_{-\infty}$ the cross-sectional area of the flow upstream of, at, and downstream of the disc, respectively.

Owing to the velocity deficit introduced as a result of the flow passing through the disk, a change in pressure is caused before and after the disc. This pressure difference in turn produces a force, F_D, on the flow, and is given by

$$F_D = (U_\infty - U_{-\infty})\rho A_D U_D. \tag{A-5}$$

The flow velocity at the actuator disk can be calculated using the upstream flow velocity, and is given by

$$U_D = (1-a)U_\infty, \tag{A-6}$$

where a is the axial induction factor. It is a measure of the amount of kinetic energy tapped by a turbine.

According to Bernoulli's principle, under steady conditions, when no work is done on the fluid, the total energy within the flow is same everywhere. As no work is done on the flow, Bernoulli's principle can be applied on both the upstream and downstream flows, and the resulting downstream flow velocity is given by

$$U_{-\infty} = (1-2a)U_\infty. \tag{A-7}$$

From the aforementioned equation, the theoretical maximum that the axial induction factor can have is $a = 0.5$. After which, the downstream velocity becomes negative. As negative

velocity is not possible, the theoretical minimum for the downstream velocity across at $a = 0.5$. Furthermore, on representing the force F_D in terms of the axial induction factor, Eq. (A-5) becomes

$$F_D = 2\rho A_D U_\infty^2 a(1-a). \tag{A-8}$$

Then, the power generated by a turbine is given by

$$\begin{aligned} P_D &= F_D U_D, \\ &= 2\rho A_D U_\infty^3 a(1-a)^3. \end{aligned} \tag{A-9}$$

A wind turbine can not completely extract the kinetic energy present in the wind. Thus, the downstream flow velocity is not a vacuum, and the efficiency of a turbine is expressed in terms of the power co-efficient C_P. It is the ratio between the power generated by a turbine, P_D, and the power available in the wind P_V. This is given by

$$\begin{aligned} C_P &= \frac{P_D}{P_V}, \\ &= 4a(1-a)^3. \end{aligned} \tag{A-10}$$

The maximum power efficiency attainable by a turbine is given by Betz limit. For more information on the derivations related to the aforementioned equations and Betz limit, the reader is referred to the book by Bianchi *et al.* [10]. Moreover, the aforementioned equations are valid only for non-yawed conditions. For more information on the yaw extension to these equations, the reader is referred to Gebraad's Ph.D. thesis [4].

An extension to the ADM is the Acutator Line Model (ALM). It is a more advanced model which is employed in high fidelity wind farm models like Simulator fOr Wind Farm Applications (SOWFA). For more information on the ALM, the reader is referred to an article by Churchfield *et al.* [30].

Appendix B

Basics of Kalman Filtering

Filtering is the process of improving the accuracy of the estimates of the state vector of a noisy dynamical system, by conditioning it on the available sensor observations. This chapter presents a brief overview of the Kalman Filter (KF), and its non-linear variants.

The chapter begins by introducing the dynamical model considered in this chapter for explaining the KF in Appendix B-1. In Appendix B-2, the general approach to the KF is outlined. The Extended Kalman Filter (ExKF) is presented in Appendix B-3. Finally, the chapter concludes by explaining the Ensemble Kalman Filter (EnKF) in Appendix B-4.

B-1 Dynamical System

Consider a dynamical discrete-time system of the form

$$x_k = f_k(x_{k-1}, u_k, w_k), \tag{B-1}$$
$$y_k = h_k(x_k, v_k), \tag{B-2}$$

where $k \in \mathbb{Z}$ is the time instant at which the states of a system is defined, $x_k \in \mathbb{R}^n$ the state vector, $u_k \in \mathbb{R}^m$ the input vector, $y_k \in \mathbb{R}^\ell$ the output vector, $w_k \in \mathbb{R}^n$ the process noise vector, and $v_k \in \mathbb{R}^\ell$ the measurement noise vector. Moreover, Eq. (B-1) and Eq. (B-2) can be referred to as state-transition and measurement (input-output) equations respectively.

B-2 Kalman Filter

The Kalman Filter is a minimum-variant unbiased estimator (MVUE), meaning the KF achieves the smallest estimation error co-variance among all the unbiased estimators. For the existence of the minimum-variant unbiased estimate, the process and measurement noises must be zero-mean white noise, i.e., the estimates must be uncorrelated with the process and measurement noises [49]. However, this assumption does not guarantee the optimality of the

estimate. In other words, there might exist a posterior distribution that matches the actual distribution of the state vector better than the posterior distribution estimated by the KF, even though the KF provides the best estimate that any unbiased estimator can achieve.

Basically, a random process can be approximated using a Gaussian random process, as there exists a unique gaussian distribution with the same mean and co-variance as that of the actual random process [57]. The KF follows this strategy, i.e., the KF approximates the actual posterior distribution of the state vector with a gaussian distribution, as it tries to match only the first two moments of the distribution. Hence, the KF estimate is optimal if and only if the following conditions are met:

1. Both process and measurement noises follow Gaussian distributions: A Gaussian distribution is the only distribution which can be completely characterized just with the first two moments, namely the mean and co-variance. On the other hand, all the other random processes require high-order moments, to accurately describe their distributions. Hence, the KF is optimal only when the noises and state vector involved in the dynamical systems are Gaussian, as the KF accurately approximates only the first two moments.

2. $f_k(x_{k-1}, u_k, w_k)$ and $h_k(x_k, v_k)$ are linear functions of the state and input vectors, with additive process and measurement noises: The Gaussianity in a dynamical system is preserved only for linear transformations and additive noises. Hence, for estimation, a dynamic system needs to be linear, as the KF requires Gaussianity for the existence of an optimal solution.

If the discrete-time dynamic system defined by Eq. (B-1) and Eq. (B-2) follows the aforementioned conditions, it can be represented as

$$x_k = F_k x_{k-1} + G_k u_k + w_k, \tag{B-3}$$
$$y_k = H_k x_k + v_k, \tag{B-4}$$

where $F_k \in \mathbb{R}^{n\times n}$, $G_k \in \mathbb{R}^{n\times m}$, and $H_k \in \mathbb{R}^{\ell\times n}$ are the system matrices, $x_k \in \mathbb{R}^n$ the state vector, $y_k \in \mathbb{R}^\ell$ the output vector, $w_k \sim \mathcal{N}(0, Q_k)$ the process noise and $v_k \sim \mathcal{N}(0, R_k)$ the measurement noise. In addition, the feed-through term in Eq. (B-4) is ignored. Moreover, the process and measurement noises are assumed to be uncorrelated with each other and with the state estimate, for the obtained state estimate to be minimum variant [49]. This can be mathematically translated as $E[v_k w_k] = 0$ and $E[(x_k - \hat{x}_k)(v_k^T \quad w_k^T)] = 0$ respectively.

The KF algorithm for a linear system, defined by Eq. (B-3) and Eq. (B-4), is given by

Kalman Filter

1. Time-Propagation

$$\overline{\hat{x}}_{k\,|\,k-1} = F_k \overline{\hat{x}}_{k-1\,|\,k-1} + G_k u_k, \tag{B-5}$$
$$P_{k\,|\,k-1} = F_k P_{k-1\,|\,k-1} F_k^T + Q_k, \tag{B-6}$$

2. Measurement-Update

$$\begin{aligned}
\tilde{y}_k &= y_k - H_k \overline{\hat{x}}_{k \mid k-1}, && \text{(B-7)}\\
P_{yy} &= R_k + H_k P_{k|k-1} H_k^T, && \text{(B-8)}\\
P_{xy} &= P_{k \mid k-1} H_k^T, && \text{(B-9)}\\
K_k &= P_{xy} P_{yy}^{-1}, && \text{(B-10)}\\
\overline{\hat{x}}_{k \mid k} &= \overline{\hat{x}}_{k \mid k-1} + K_k \tilde{y}_k, && \text{(B-11)}\\
P_{k \mid k} &= (I - K_k H_k) P_{k \mid k-1}, && \text{(B-12)}
\end{aligned}$$

where $\overline{\hat{x}}_{k-1 \mid k-1}$ is the expected value of the updated state estimate at $k-1$, $\overline{\hat{x}}_{k \mid k-1}$ the expected value of the predicted state estimate at k and $\overline{\hat{x}}_{k \mid k}$ is the expected value of the updated state estimate at k.

Typically, the aforementioned KF algorithm, defined by Eq. (B-5) to Eq. (B-12), is called the filtered estimator [49] or the current state estimator or just the measurement-update [49]. An alternate version of the KF algorithm, called the one-step-ahead predictor [49] or just the time-update [49], exists. Mathematically, the one-step-ahead predictor is given by

$$\begin{aligned}
\tilde{y}_k &= y_k - H_k \overline{\hat{x}}_{k \mid k-1}, && \text{(B-13)}\\
K_k^* &= (S_k + F_k P_{k \mid k-1} H_k^T)(R_k + H_k P_{k \mid k-1} H_k^T)^{-1}, && \text{(B-14)}\\
\overline{\hat{x}}(k+1 \mid k) &= F_k \overline{\hat{x}}(k \mid k-1) + G_k u_k + K_k^* \tilde{y}_k, && \text{(B-15)}\\
P_{k+1 \mid k} &= F_k P_{k \mid k-1} F_k^T + Q_k - (S_k + F_k P_{k \mid k-1} H_k^T) \\
&\quad (R_k + H_k P_{k \mid k-1} H_k^T)^{-1} (S_k + F_k P_{k \mid k-1} H_k^T)^T, && \text{(B-16)}
\end{aligned}$$

where $x_{k+1 \mid k}$ and $x_{k \mid k-1}$ are the one-step-ahead predicted estimates at time-steps k and $k-1$, respectively. Similarly, $P_{k+1 \mid k}$ and $P_{k \mid k-1}$ are the one-step-ahead predicted co-variances at time-steps k and $k-1$, respectively. S_k is the cross co-variance between the process and measurement noises. Mathematically, the co-variances can be represented as

$$E\left[\begin{bmatrix} w_k \\ v_k \end{bmatrix} \begin{bmatrix} w_k^T & v_k^T \end{bmatrix}\right] = \begin{bmatrix} Q_k & S_k \\ S_k^T & R_k \end{bmatrix} \Delta_{i-j}, \quad \text{(B-17)}$$

where Δ_{i-j} is the unit pulse being nonzero only for $i = j$ [49]. The aforementioned mathematical notation for the co-variances signify that the process and measurement noises are white, since the noise is uncorrelated over time.

The advantage of the one-step-ahead predictor over the filtered estimator is the reduced computational cost and parallel implementation of the estimator and controller. In case of the one-step-ahead predictor, the state vector, used for the controller design at time-step k, are updated at time-step $k-1$ itself. In other words, the state vector update for the k^{th} time-step and the controller design for the $k-1^{\text{th}}$ time-step are processed in parallel. Hence, the one-step-ahead predictor implements the estimator and controller in parallel, thereby requiring the computation time of either the estimator or controller (whichever is longer). On the other hand, in case of the filtered estimator, the the state vector, used for the controller design at time-step k, are updated only at time-step k. In other words, the state vector update

for the k^{th} time-step and the controller design for the k^{th} time-step are processed one after the other, in series. Hence, the computation time taken by the filtered estimator is the sum of the individual time taken by both the estimator and controller. Thus, the one-step-ahead predictor is computationally cheaper than the filtered estimator.

Contrarily, the advantage of the filtered estimator over the one-step-ahead predictor is the accuracy. In case of the filtered estimator, sensor observations until time-step k are available to update the state vector x_k. However, in case of the one-step-ahead predictor, sensor observations only until time-step $k-1$ are available to update the state vector x_k. Hence, the filtered estimator can yield a better estimate of the state vector than the one-step-ahead predictor, as the filtered estimator has an additional information to condition the state vector.

In this thesis, the filtered estimator version of the KF, defined by Eq. (B-5) to Eq. (B-12), is considered, as it is the accuracy of the estimate of the state vector that is important here. Moreover, the computational complexity of the estimation algorithm is anyways being taken care by the distributed architecture.

Furthermore, on applying the KF to an Linear time-invariant (LTI) system, the error covariance of the estimate converges with the Kalman gain. Mathematically, this can be represented as

$$\lim_{k\to\infty} P_{k\,|\,k} = P > 0, \tag{B-18}$$

$$\lim_{k\to\infty} K_k = K, \tag{B-19}$$

$$= PH^T(R + HPH^T)^{-1}. \tag{B-20}$$

Finally, the optimality of the KF can be investigated by analyzing the innovation signal e_k. The innovation signal is the difference between the actual and estimated output, and is given by

$$e_k = y_k - H_k\bar{\hat{x}}_{k\,|\,k}, \tag{B-21}$$

where y_k is the actual output, and $H_k\bar{\hat{x}}_{k\,|\,k}$ the estimated output.

For the estimator to be minimum-variant and unbiased, the innovation signal must uncorrelated over time, meaning the filter has extracted all the useful information. In short, the innovation signal must be a zero-mean white sequence with minimum variance. The whiteness of the innovation signal can be determined by either computing the auto co-variance or power spectrum of the innovation signal [49].

For more information on the Kalman Filter, the reader is referred to books by Verhaegen and Verdult [49] and Särkkä [58].

B-3 Extended Kalman Filter

The Extended Kalman Filter is a non-linear extension to the Kalman filter. It is based on linearizing the non-linear functions by utilizing Taylor series approximations [58].

On applying Taylor series approximations to the non-linear system with non-additive process and measurement noises of the form Eq. (B-1), we obtain

$$\begin{aligned}\hat{x}_{k\,|\,k-1} &= f_k(m_{k-1\,|\,k-1}, u_k, w_k) + [D_x f + \frac{D_x^2 f}{2!} + \frac{D_x^3 f}{3!} + ...] \\ &\quad + [D_w f + \frac{D_w^2 f}{2!} + \frac{D_w^3 f}{3!} + ...] + [D_u f + \frac{D_u^2 f}{2!} + \frac{D_u^3 f}{3!} + ...], \qquad \text{(B-22)} \\ \hat{x}_{k\,|\,k-1} &= \frac{\partial f_k}{\partial x}\Big|_{m_{k-1\,|\,k-1}} x_{k-1} + \frac{\partial f_k}{\partial u}\Big|_{u_k} u_k + \frac{\partial f_k}{\partial w}\Big|_{w_k} w_k + \text{H.O.T.}, \qquad \text{(B-23)}\end{aligned}$$

where the i^{th} term in the Taylor series is

$$\begin{aligned}\frac{D_x^i f}{i!} &= \frac{1}{i!}\Big(\sum_{j=1}^{n}(x_{k-1} - \hat{x}_{k-1\,|\,k-1})\frac{\partial}{\partial x_j}\Big)^i f_k\Big|_{m_{k-1\,|\,k-1}}, \\ \frac{D_u^i f}{i!} &= \frac{1}{i!}\Big(\sum_{j=1}^{n} u_k \frac{\partial}{\partial u_j}\Big)^i f_k\Big|_{u_k}, \\ \frac{D_w^i f}{i!} &= \frac{1}{i!}\Big(\sum_{j=1}^{n} w_k \frac{\partial}{\partial w_j}\Big)^i f_k\Big|_{w_k}.\end{aligned}$$

Similarly, on applying Taylor series approximations to Eq. (B-2) and neglecting the higher order terms (H.O.T.), the resulting state space representation is given by

$$\begin{aligned}x_k &= F_k x_{k-1} + G_k u_k + L_k w_k, \qquad \text{(B-24)} \\ y_k &= H_k x_k + M_k v_k, \qquad \text{(B-25)}\end{aligned}$$

with

$$\begin{aligned}F_k = \frac{\partial f_k}{\partial x}\Big|_{m_{k-1\,|\,k-1}} \quad G_k &= \frac{\partial f_k}{\partial u}\Big|_{u_k}, \quad L_k = \frac{\partial f_k}{\partial w}\Big|_{w_k}, \\ H_k = \frac{\partial h_k}{\partial x}\Big|_{m_{k-1\,|\,k-1}} \quad M_k &= \frac{\partial h_k}{\partial v}\Big|_{v_k},\end{aligned}$$

where $F_k \in \mathbb{R}^{n\times n}$, $G_k \in \mathbb{R}^{n\times m}$, $L_k \in \mathbb{R}^{n\times n}$, $H_k \in \mathbb{R}^{\ell\times n}$, and $M_k \in \mathbb{R}^{\ell\times\ell}$ are the system matrices, $x_k \in \mathbb{R}^n$ the state vector, $y_k \in \mathbb{R}^\ell$ the output vector, $w_k \in \mathbb{R}^n$ the process noise and $v_k \in \mathbb{R}^\ell$ the measurement noise.

Once the non-linear system equations are linearized, the KF equations, given by Eq. (B-5) to Eq. (B-12), can be applied directly. However, the ExKF is applicable only when the system is not highly non-linear, as all the high-order terms, except for the first order, is ignored during linearization.

Furthermore, for improving the accuracy of the ExKF estimates, instead of considering only the first-order term and neglecting the entire higher order terms from Eq. (B-23), second-order, third-order and other higher-order terms can also be considered, resulting in the second-order ExKF, third-order ExKF and high-order ExKF respectively. These high-order ExKF would, however, increase the computational complexity.

For more information on the ExKF, the reader is referred to books by Särkkä [58], Simon [59] & Mutambara [60].

B-4 Ensemble Kalman Filter

The EnKF is a non-linear variant of the Kalman Filter. It is widely used in weather forecasting for data assimilation, where the models are extremely large in size and non-linear with a large number of measurements.

Unlike the ExKF where Taylor series linearization is employed to approximate the actual non-linear functions f and h, the EnKF tries to match the actual probability distribution of the state vector, by drawing a set of points, at random, from the Gaussian distributions. In other words, the EnKF approximate the actual distribution of the state vector as a Gaussian distribution, since the EnKF focuses only on the first two moments. Hence, when the noises and states are highly non-Gaussian, the accuracy of the EnKF estimate will be far from optimal.

A set of randomly-selected points, called ensemble members, play an important role in the accuracy and computational complexity of the EnKF. If a large number of ensemble members are employed, the sample mean and co-variance computed by the EnKF will converge to the actual mean and co-variance, thereby improving the accuracy of the estimates. However, this also increases the computational complexity. Contrarily, employing a very few ensemble members will render the estimation algorithm incapable of capturing the true distribution. Hence, the number of ensemble members have to selected such that there is a balance between both the accuracy and computational complexity of the estimation algorithm.

The EnKF algorithm begins by generating the ensemble points for the process and measurement noises, along with the state vector. In mathematical notation, the realizations of the initial ensemble members are defined as

$$\begin{aligned}
\boldsymbol{\psi}_{0\,|\,0} &= \begin{bmatrix} \boldsymbol{\psi}^1_{0\,|\,0} & \cdots & \boldsymbol{\psi}^{N_e}_{0\,|\,0} \end{bmatrix}, \\
&= \begin{bmatrix} \overline{x}_0 & \cdots & \overline{x}_0 \end{bmatrix} + \begin{bmatrix} \xi^1_0 & \cdots & \xi^{N_e}_0 \end{bmatrix}, \\
\boldsymbol{w}_0 &= \begin{bmatrix} \boldsymbol{w}^1_0 & \cdots & \boldsymbol{w}^{N_e}_0 \end{bmatrix}, \\
&= \begin{bmatrix} \epsilon^1_0 & \cdots & \epsilon^{N_e}_0 \end{bmatrix}, \\
\boldsymbol{v}_0 &= \begin{bmatrix} \boldsymbol{v}^1_0 & \cdots & \boldsymbol{v}^{N_e}_0 \end{bmatrix}, \\
&= \begin{bmatrix} \mu^1_0 & \cdots & \mu^{N_e}_0 \end{bmatrix},
\end{aligned}$$

where $\boldsymbol{\psi}^i_{0\,|\,0} \in \mathbb{R}^n$, $\boldsymbol{w}^i_0 \in \mathbb{R}^n$, and $\boldsymbol{v}^i_0 \in \mathbb{R}^n$ are the initial ensemble members of the state vector, process and measurement noises respectively, with $i \in \{1, 2, \ldots, N_e\}$, $N_e \ll n$ and N_e the total number of ensemble members. The process $\boldsymbol{w}^i_k \in \mathbb{R}^n$ and measurement $\boldsymbol{v}^i_k \in \mathbb{R}^l$ noises are assumed to be zero-mean Gaussian noises, with Q_k and R_k as co-variances respectively. In addition, $\xi^i_0 \sim \mathcal{N}(0, P_0)$, $\epsilon^i_0 \sim \mathcal{N}(0, Q_0)$, and $\mu^i_0 \sim \mathcal{N}(0, R_0)$ are the artificial disturbances, which are included for creating the ensemble members.

The sample co-variance matrices, calculated using the ensemble members, are given by

$$\begin{aligned}
Q_e &= \frac{w_{e_k} w^T_{e_k}}{N_e - 1}, \\
R_e &= \frac{v_{e_k} v^T_{e_k}}{N_e - 1},
\end{aligned}$$

where $w_{e_k} = \begin{bmatrix}\epsilon_k^1 & \cdots & \epsilon_k^{N_e}\end{bmatrix}$, and $v_{e_k} = \begin{bmatrix}\mu_k^1 & \cdots & \mu_k^{N_e}\end{bmatrix}$ are the realizations of the process and measurement disturbances, at time k, respectively. Furthermore, as mentioned earlier, increasing the ensemble members causes the sample co-variance matrices to converge to the actual co-variance [61]. Mathematically, this is given by

$$\lim_{N_e \to \infty} \{Q_{k,e}, R_{k,e}, P_{0,e}, \hat{P}_{yy}, \hat{P}_{xy}\} = \{Q_k, R_k, P_0, P_{yy}, P_{xy}\},$$

where $Q_k, R_k, P_0, P_{yy}, P_{xy}$ refer to true auto co-variances of the process noise, measurement noise, initial state estimate, output vector and the true cross co-variance between the state and output respectively.

Once the ensemble members are generated, as shown by Eq. (B-26) to Eq. (B-28), they are propagated through the state-transition and measurement equations, to yield a set of transformed points, as shown in Eq. (B-29) and Eq. (B-30). The transformed points are then used to calculate the sample mean and co-variance. Finally, the ensemble members are updated. Thus, the EnKF algorithm for the non-linear system of the form Eq. (B-1) and Eq. (B-2) is given as

Ensemble Kalman Filter

Initialization: Defining Ensemble Members

$$\boldsymbol{\psi}_{0\,|\,0} = \begin{bmatrix}\overline{\hat{x}}_0 & \cdots & \overline{\hat{x}}_0\end{bmatrix} + \begin{bmatrix}\xi_0^1 & \cdots & \xi_0^{N_e}\end{bmatrix}, \tag{B-26}$$

$$\boldsymbol{w}_0 = \begin{bmatrix}\epsilon_0^1 & \cdots & \epsilon_0^{N_e}\end{bmatrix}, \tag{B-27}$$

$$\boldsymbol{v}_0 = \begin{bmatrix}\mu_0^1 & \cdots & \mu_0^{N_e}\end{bmatrix}, \tag{B-28}$$

Recursive Steps

1. Prediction

$$\boldsymbol{\psi}^i_{k\,|\,k-1} = f_k(\boldsymbol{\psi}^i_{k-1\,|\,k-1}, u_k, \boldsymbol{w}^i_k), \forall i = 1, \ldots, N_e \tag{B-29}$$

2. Measurement Ensemble

$$\boldsymbol{D}^i_k = h_k(\boldsymbol{\psi}^i_{k-1\,|\,k-1}, \boldsymbol{v}^i_k), \forall i = 1, \ldots, N_e \tag{B-30}$$

3. Sample Mean

$$\overline{\boldsymbol{\psi}}_{k\,|\,k-1} = \frac{1}{N_e}\sum_{i=1}^{N_e} \boldsymbol{\psi}^i_{k\,|\,k-1}, \tag{B-31}$$

$$\overline{\boldsymbol{D}}_k = \frac{1}{N_e}\sum_{i=1}^{N_e} \boldsymbol{D}^i_k, \tag{B-32}$$

4. Sample Co-variance

$$E_x = \left[\boldsymbol{\psi}^1_{k\,|\,k-1} - \overline{\boldsymbol{\psi}}_{k\,|\,k-1} \quad \cdots \quad \boldsymbol{\psi}^{N_e}_{k\,|\,k-1} - \overline{\boldsymbol{\psi}}_{k\,|\,k-1}\right], \tag{B-33}$$

$$E_y = \left[\boldsymbol{D}^1_k - \overline{\boldsymbol{D}}_k \quad \cdots \quad \boldsymbol{D}^{N_e}_k - \overline{\boldsymbol{D}}_k\right], \tag{B-34}$$

$$P_{yy} \approx \hat{P}_{yy} = \frac{1}{N_e - 1} E_y (E_y)^T, \tag{B-35}$$

$$P_{xy} \approx \hat{P}_{xy} = \frac{1}{N_e - 1} E_x (E_y)^T, \tag{B-36}$$

5. Update/ Analysis

$$K_k = P_{xy} P_{yy}^{-1}, \tag{B-37}$$

$$\boldsymbol{\psi}^i_{k\,|\,k} = \boldsymbol{\psi}^i_{k\,|\,k-1} + K_k(y_k - \boldsymbol{D}^i_k), \forall i = 1, \ldots, N_e \tag{B-38}$$

$$\overline{\hat{x}}_{k\,|\,k} = \frac{1}{N_e} \sum_{i=1}^{N_e} \boldsymbol{\psi}^i_{k\,|\,k}. \tag{B-39}$$

For more information on the EnKF, the reader is referred to articles by Evensen [62], and Gillijns [61].

Appendix C

An Alternate Approach to Measurement-Based Data Fusion for Distributed Architecture 4

In architecture 4, the interactions between the subsystems are considered explicitly, by implementing an ad hoc correction for wake effects using sensor measurements and fusion algorithms. This measurement-based data fusion methods, for including the interactions, are applicable only when the observations are available. Contrarily, when the observations are not available, the alternate method devised in this chapter can be applied.

Typically, in the measurement-based data fusion method, all the subsystems in the distributed architecture are implemented in parallel, and data fusion will be employed, before the start of the next iteration, to include the interactions between the turbines. The alternate approach, on the other hand, implements the subsystems in series, with the flow field predicted by the upstream subsystems as a prior for the downstream subsystems, i.e., the flow field predicted by the upstream subsystems for the overlapping region will be used as the measurements for the immediate-downstream subsystems, with the downstream subsystems estimating at a delay of a sampling instance with their upstream subsystems.

Generally, the sampling time of a discrete-time system is taken to be smaller than the speed at which the dynamics of the system change. In case of a dynamic wind farm model, the sampling time is taken to be smaller than the time it takes for the wind to traverse, downstream, from one turbine to the other. Hence, operating the downstream subsystems at a delay of a sampling instance with the upstream subsystems is a valid consideration, as it will necessarily take more than one sampling instance for the wind to traverse, downstream, between the consecutive subsystems. In addition, this alternate method is formulated only for a fixed wind direction, flowing perpendicular to the wind turbines in a wind farm, and a farm topology as given by Figure 3-7. This implies that the wind turbines in a farm must interact sequentially with the wind flow. That is, the wind flow will first interact with the turbines in the first row. Followed by that, the turbines in second row will interact, and so on. Hence, for such a

wind farm, this alternate approach can be applied directly. Contrarily, for a wind farm which violates this condition, changes must be made to the alternate approach before implementing.

From now on, let us consider only those wind farms which obey the aforementioned conditions. Assume N_i to be the number of rows (axis perpendicular to the wind flow direction), N_j to be the number of turbines in a row, with each row having an equal number of turbines, and N_{turb} to be the total number of turbines in a wind farm, with $N_{turb} = N_i \times N_j$. For example, for the 9-turbine wind farm displayed in Figure 3-7, the number of rows, N_i, is 3, and the number of turbines, N_j, in each row is also 3.

The alternate approach, to measurement-based data fusion, for a wind farm following distributed architecture 4, defined by Eq. (3-44) to Eq. (3-46), is given by

Subsystems in row 1 (Upstream)

1. Time-Propagation (repeated for $\forall j \in \{1, \ldots, N_j\}$, with $i = 1$)

$$\overline{\hat{X}}_{i,j,k_i \mid k_i-1} = A_{i,j}\overline{\hat{X}}_{i,j,k_i-1 \mid k_i-2} + B_{i,j}u_{i,j,k_i}, \tag{C-1}$$
$$P_{i,j,k_i \mid k_i-1} = A_{i,j}P_{i,j,k_i-1 \mid k_i-2}A_{i,j}^T + Q_{i,j,k_i}, \tag{C-2}$$

Downstream Subsystems

1. Time-Propagation (repeated for $\forall i \in \{2, \ldots, N_i\}$ and $\forall j \in \{1, \ldots, N_j\}$)

$$\overline{\hat{X}}_{i,j,k_i \mid k_i-1} = A_{i,j}\overline{\hat{X}}_{i,j,k_i-1 \mid k_i-1} + B_{i,j}u_{i,j,k_i}, \tag{C-3}$$
$$P_{i,j,k_i \mid k_i-1} = A_{i,j}P_{i,j,k_i-1 \mid k_i-1}A_{i,j}^T + Q_{i,j,k_i}, \tag{C-4}$$

2. Measurement-Update (repeated for $\forall i \in \{2, \ldots, N_i\}$ and $\forall j \in \{1, \ldots, N_j\}$)

$$Y_{i,j,k_i} = H_{i,j,k_{i-1}}\overline{\hat{X}}_{i-1,j,k_{i-1} \mid k_{i-1}-1}, \tag{C-5}$$
$$\hat{Y}_{i,j,k_i} = H_{i,j,k_i}\overline{\hat{X}}_{i,j,k_i \mid k_i-1}, \tag{C-6}$$
$$\tilde{Y}_{i,k} = Y_{i,j,k_i} - \hat{Y}_{i,j,k_i}, \tag{C-7}$$
$$R_{i,j,k_i} = P_{i-1,j,k_{i-1} \mid k_{i-1}-1}, \tag{C-8}$$
$$P_{i,j,yy} = R_{i,j,k_i} + H_{i,j,k_i}P_{i,j,k_i \mid k_i-1}H_{i,j,k_i}^T, \tag{C-9}$$
$$P_{i,j,xy} = P_{i,j,k_i \mid k_i-1}H_{i,j,k_i}^T, \tag{C-10}$$
$$K_{i,j,k_i} = P_{i,j,xy}P_{i,j,yy}^{-1}, \tag{C-11}$$
$$\overline{\hat{X}}_{i,j,k_i \mid k_i} = \overline{\hat{X}}_{i,j,k_i \mid k_i-1} + K_{i,j,k_i}\tilde{Y}_{i,j,k_i}, \tag{C-12}$$
$$P_{i,j,k_i \mid k_i} = (I - K_{i,j,k_i}H_{i,j,k_i})P_{i,j,k_i \mid k_i-1}, \tag{C-13}$$

where k is the time instant at which the actual plant is defined, and k_i the time instant at which the states of the subsystems in row i are defined, with $k = k_1$ and $k_i = k_{i-1} - T$, where T is the sampling time. Here, T is assumed to be 1 second. $\overline{\hat{X}}_{i,j,k_i \mid k_i-1}$ and $P_{i,j,k_i \mid k_i-1}$ are the expected value and co-variance of the predicted state estimate of subsystem j in row i at

k_i, respectively. Similarly, $\overline{\hat{X}}_{i,j,k_i \mid k_i}$ and $P_{i,j,k_i \mid k_i}$ are the expected value and co-variance of the updated state estimate of subsystem j in row i at k_i, respectively. Y_{i,j,k_i} is the estimate of the overlapping region of the subsystem j in row $i-1$ (upstream subsystem), and it is used as the measurements for the subsystem j in row i (downstream subsystem). $\hat{Y}_{i,j,k_i}$ is the estimate of the the overlapping region of the subsystem j in row i (downstream subsystem). R_{i,j,k_i} is the co-variance of measurement noise. Here, since the estimate of the overlapping region of the subsystems in row $i-1$ (upstream subsystem) is used as the measurements for the subsystems in row i (downstream subsystem), the co-variance of the predicted state estimate of subsystem j in row $i-1$, $P_{i-1,j,k_{i-1} \mid k_{i-1}-1}$, will be used as the co-variance of measurement noise.

Now, on analyzing the time instants at which each subsystem is defined, it can be inferred that the subsystems in row 1 are initialized together with the actual plant, while the subsystems in row i are initialized at a delay of T with the subsystems in row $i-1$. This means that the downstream subsystems operate at a delay of T units with the upstream subsystems. This is a valid consideration, as the wind flow does not interact with all the subsystems at once, but one after the other. Furthermore, the sampling time, T, is assumed to be 1 second, here. This means that the subsystems in row 1 will start to estimate from the first time-step, i.e., with the actual plant. Then, the subsystems in row 2 will start to estimate from the second time-step, i.e., one second after the subsystems in row 1, and so on.

Furthermore, on analyzing the mathematics involved in the alternate approach, it can be inferred that the subsystems in row 1 are only time-propagated, as these subsystems directly interact with the free-stream wind flow, i.e., there are no external disturbances for the wind, before it interacts with the subsystems in row 1. Hence, they would perform similar to the open-loop centralized architecture. On the other hand, from row 2 onward, the subsystems assume the incoming wind flow to be a free-stream flow. However, in reality, the inflow for the downstream subsystems are the wakes of the upstream subsystems. Hence, the interactions between the subsystems need to be explicitly considered. Earlier, the interactions were included by fusing the state estimates between the subsystems, by employing sensor measurement and data fusion algorithms. The alternate method, on the other hand, uses the estimate of the overlapping region of the upstream subsystems ($H_{i,j,k_{i-1}} \overline{\hat{X}}_{i-1,j,k_{i-1} \mid k_{i-1}-1}$) as the measurements for the downstream subsystems, Y_{i,j,k_i}. Consequently, the interactions between the subsystems are modelled by incorporating the measurement-update step, using the estimates of the upstream subsystems.

In short, the alternate approach proposed in this chapter provides a method to explicitly model the interactions between the subsystems, in the absence of the sensor observations. Hence, the subsystems in distributed architecture 4 will remain coupled with each other, and can almost predict the flow field similar to the centralized architecture. However, due to the lack of time, this alternate approach could not be implemented, and is recommend for the future work.

Bibliography

[1] IEA, *Transition to Sustainable Buildings: Strategies and Opportunities to 2050*. 2013. https://www.oecd-ilibrary.org/content/publication/9789264202955-en.

[2] T. Boden, G. Marland, and R. Andres, *Global, Regional, and National Fossil - Fuel CO_2 Emissions*. Tennessee, U.S.A.: Carbon Dioxide Information Analysis Center, Oak Ridge National Laboratory, U.S. Department of Energy, Oak Ridge, 2017.

[3] Vattenfall, *Photograph: Horns Rev 3*, (accessed May 1, 2018). https://corporate.vattenfall.dk/vores-vindmoller-i-danmark/vindprojekter/horns-rev-3/.

[4] P. Gebraad, *Data-Driven Wind Plant Control*. PhD thesis, Delft University of Technology, Dec. 2014.

[5] S. Boersma, B. M. Doekemeijer, P. M. O. Gebraad, P. A. Fleming, J. Annoni, A. K. Scholbrock, J. A. Frederik, and J. W. van Wingerden, "A tutorial on control-oriented modeling and control of wind farms," in *2017 American Control Conference (ACC)*, pp. 1–18, May 2017.

[6] S. Boersma, B. Doekemeijer, M. Vali, J. Meyers, and J. van Wingerden, "A control-oriented dynamic wind farm model: Wfsim," *Wind Energy Science*, vol. 3, no. 1, pp. 75–95, 2018.

[7] S. Boersma, P. Gebraad, M. Vali, B. Doekemeijer, and J. van Wingerden, "A control-oriented dynamic wind farm flow model: "WFSim"," *Journal of Physics: Conference Series*, vol. 753, no. 3, p. 032005, 2016.

[8] J. Hart, "Comparison of turbulence modeling approaches to the simulation of a dimpled sphere," *Procedia Engineering*, vol. 147, pp. 68 – 73, 2016. The Engineering of SPORT 11.

[9] S. Cradle, *Turbulent Flow Analysis*, (accessed May 10, 2018). http://www.cradle-cfd.com/tec/column01/018.html.

[10] R. J. M. Fernando D. Bianchi and H. D. Battista, *The Wind and Wind Turbines*, pp. 7–28. London: Springer London, 2007.

[11] F. Felix, "Growth of energy consumption and national income throughout the world," *IEEE Spectrum*, vol. 1, pp. 81–102, July 1964.

[12] S. Sorrell, "Reducing energy demand: A review of issues, challenges and approaches," *Renewable and Sustainable Energy Reviews*, vol. 47, pp. 74 – 82, 2015.

[13] K. Bithas and P. Kalimeris, *A Brief History of Energy Use in Human Societies*, pp. 5–10. Cham: Springer International Publishing, 2016.

[14] C. Zou, Q. Zhao, G. Zhang, and B. Xiong, "Energy revolution: From a fossil energy era to a new energy era," *Natural Gas Industry B*, vol. 3, no. 1, pp. 1 – 11, 2016.

[15] EIA, *International Energy Outlook 2017*. 2017. https://www.eia.gov/outlooks/ieo/.

[16] E. Nehrenheim, "Introduction to renewable energy," in *Encyclopedia of the Anthropocene* (D. A. Dellasala and M. I. Goldstein, eds.), pp. 405 – 406, Oxford: Elsevier, 2018.

[17] EIA, *Energy and the Environment Explained: Greenhouse Gases' Effect on the Climate*, August 17, 2017 (accessed April 27, 2018). https://www.eia.gov/energyexplained/index.cfm?page=environment_how_ghg_affect_climate.

[18] S. Solomon, D. Qin, M. Manning, Z. Chen, M. Marquis, K. Averyt, M. Tignor, and H. M. (eds.)., *Climate Change 2007: The Physical Science Basis: Working group I contribution to the fourth assessment report of the IPCC*, vol. 4. Cambridge University Press, 2007. http://www.ipcc.ch/publications_and_data/ar4/wg1/en/contents.html.

[19] W. Haeberli, C. Huggel, F. Paul, and M. Zemp, "13.10 glacial responses to climate change," in *Treatise on Geomorphology* (J. F. Shroder, ed.), pp. 152 – 175, San Diego: Academic Press, 2013.

[20] M. M. Q. Mirza, "Global warming and changes in the probability of occurrence of floods in bangladesh and implications," *Global Environmental Change*, vol. 12, no. 2, pp. 127 – 138, 2002.

[21] Q. Schiermeier, "Increased flood risk linked to global warming: Likelihood of extreme rainfall may have been doubled by rising greenhouse-gas levels," *Nature*, vol. 470, no. 7334, pp. 316–317, 2011.

[22] L. Alfieri, P. Burek, L. Feyen, and G. Forzieri, "Global warming increases the frequency of river floods in europe," *Hydrology and Earth System Sciences*, vol. 19, no. 5, p. 2247, 2015.

[23] A. Dai, "Drought under global warming: a review," *Wiley Interdisciplinary Reviews: Climate Change*, vol. 2, no. 1, pp. 45–65.

[24] B. Worm and H. K. Lotze, "Chapter 13 - marine biodiversity and climate change," in *Climate Change (Second Edition)* (T. M. Letcher, ed.), pp. 195 – 212, Boston: Elsevier, second edition ed., 2016.

[25] D. J. Arent, A. Wise, and R. Gelman, "The status and prospects of renewable energy for combating global warming," *Energy Economics*, vol. 33, no. 4, pp. 584 – 593, 2011. Special Issue on The Economics of Technologies to Combat Global Warming.

[26] W. Obergassel, C. Arens, L. Hermwille, N. Kreibich, F. Mersmann, H. E. Ott, and H. Wang-Helmreich, "Phoenix from the ashes: an analysis of the paris agreement to the united nations framework convention on climate change; part 1," 2015.

[27] W. contributors, *Cost of electricity by source*, (accessed May 1, 2018). https://en.wikipedia.org/wiki/Cost_of_electricity_by_source.

[28] L. Y. Pao and K. E. Johnson, "A tutorial on the dynamics and control of wind turbines and wind farms," in *2009 American Control Conference*, pp. 2076–2089, June 2009.

[29] J. P. Goit and J. Meyers, "Optimal control of energy extraction in wind-farm boundary layers," *Journal of Fluid Mechanics*, vol. 768, pp. 5–50, 2015.

[30] M. J. Churchfield, S. Lee, J. Michalakes, and P. J. Moriarty, "A numerical study of the effects of atmospheric and wake turbulence on wind turbine dynamics," *Journal of Turbulence*, vol. 13, p. N14, 2012.

[31] J. Annoni, P. M. O. Gebraad, A. K. Scholbrock, P. A. Fleming, and J. W. van Wingerden, "Analysis of axial-induction-based wind plant control using an engineering and a high-order wind plant model," *Wind Energy*, vol. 19, no. 6, pp. 1135–1150.

[32] Á. Jiménez, A. Crespo, and E. Migoya, "Application of a les technique to characterize the wake deflection of a wind turbine in yaw," *Wind Energy*, vol. 13, no. 6, pp. 559–572.

[33] T. Knudsen, T. Bak, and M. Svenstrup, "Survey of wind farm control-power and fatigue optimization," *Wind Energy*, vol. 18, no. 8, pp. 1333–1351.

[34] B. M. Doekemeijer, *Enhanced Kalman filtering for a 2D CFD Navier-Stokes wind farm model*. MSc dissertation, Delft University of Technology, 2016.

[35] B. M. Doekemeijer, S. Boersma, L. Y. Pao, and J. W. van Wingerden, "Ensemble kalman filtering for wind field estimation in wind farms," in *2017 American Control Conference (ACC)*, pp. 19–24, May 2017.

[36] B. Doekemeijer, S. Boersma, L. Pao, T. Knudsen, and J. van Wingerden, "Online Model Calibration and State Estimation for a Simplified LES Model in Pursuit of Real-Time Closed-Loop Wind Farm Controls," *Wind Energy Science*, in review.

[37] J. M. Jonkman and M. L. Buhl, "Fast user's guide," tech. rep., National Renewable Energy Laboratory (NREL), August 2005.

[38] N. S. Kumar, *De-Centralized State Estimation for Surrogate Wind Farm Models in pursuit of Closed-Loop Model-Based Control*. literature survey, Delft University of Technology, February 2018.

[39] Y. Bar-Shalom and L. Campo, "The effect of the common process noise on the two-sensor fused-track covariance," *IEEE Transactions on Aerospace and Electronic Systems*, vol. AES-22, pp. 803–805, Nov 1986.

[40] B. Noack, J. Sijs, and U. D. Hanebeck, "Fusion strategies for unequal state vectors in distributed kalman filtering," *IFAC Proceedings Volumes*, vol. 47, no. 3, pp. 3262 – 3267, 2014. 19th IFAC World Congress.

[41] S. J. Julier and J. K. Uhlmann, "A non-divergent estimation algorithm in the presence of unknown correlations," in *Proceedings of the 1997 American Control Conference (Cat. No.97CH36041)*, vol. 4, pp. 2369–2373 vol.4, Jun 1997.

[42] J. Sijs, M. Lazar, and P. P. J. v. d. Bosch, "State fusion with unknown correlation: Ellipsoidal intersection," in *Proceedings of the 2010 American Control Conference*, pp. 3992–3997, June 2010.

[43] B. Noack, J. Sijs, and U. D. Hanebeck, "Inverse covariance intersection: New insights and properties," in *2017 20th International Conference on Information Fusion (Fusion)*, pp. 1–8, July 2017.

[44] B. Noack, J. Sijs, M. Reinhardt, and U. D. Hanebeck, "Decentralized data fusion with inverse covariance intersection," *Automatica*, vol. 79, pp. 35 – 41, 2017.

[45] J. Sijs, U. Hanebeck, and B. Noack, "An empirical method to fuse partially overlapping state vectors for distributed state estimation," in *2013 European Control Conference (ECC)*, pp. 1615–1620, July 2013.

[46] B. E. Larock, R. W. Jeppson, and G. Z. Watters, *Hydraulics of Pipeline Systems*. Boca Raton, FL, USA: CRC Press, 1999.

[47] R. V. Mises, *Theory of flight*. New York,USA: Dover Publications, 1959.

[48] R. de Ruijter, *Controllability and observability of a 2D wind farm model*. MSc dissertation, Delft University of Technology, 2017.

[49] M. Verhaegen and V. Verdult, *Filtering and System Identification: A Least Square Approach*. New York, USA: Cambridge University Press, 2011.

[50] B. Sanderse, S. Pijl, and B. Koren, "Review of computational fluid dynamics for wind turbine wake aerodynamics," *Wind Energy*, vol. 14, no. 7, pp. 799–819.

[51] P. J. Moriarty and S. B. Butterfield, "Wind turbine modeling overview for control engineers," in *2009 American Control Conference*, pp. 2090–2095, June 2009.

[52] N. Jensen, "A note on wind generator interaction," tech. rep., Risø National Laboratory, 1983.

[53] P. M. O. Gebraad, F. W. Teeuwisse, J. W. Wingerden, P. A. Fleming, S. D. Ruben, J. R. Marden, and L. Y. Pao, "Wind plant power optimization through yaw control using a parametric model for wake effects - a cfd simulation study," *Wind Energy*, vol. 19, no. 1, pp. 95–114.

[54] S. Hulshoff, *CFD II Part 2: Computation and Modelling of Turbulence*. Faculty of Aerospace Engineering, Delft University of Technology, version 2015.2 ed., April 2015.

[55] D. C. Wilcox, *Turbulence Modelling for CFD*. DCW Industries, 2006.

[56] L. Davidson, *An Introduction to Turbulence Models*. Department of Thermo and Fluid Dynamics, Chalmers University of Technology, September 2017.

[57] R. E. Kalman, "A new approach to linear filtering and prediction problems," *ASME Journal of Basic Engineering*, 1960.

[58] S. Särkkä, *Bayesian Filtering and Smoothing*. New York, USA: Cambridge University Press, 2013.

[59] D. Simon, *Optimal State Estimation: Kalman, H_∞, and Nonlinear Approaches*. John Wiley & Sons, Inc, 2006.

[60] A. G. Mutambara, *Decentralized Estimation and Control for Multisensor Systems*. CRC Press LLC, 1998.

[61] S. Gillijns, O. B. Mendoza, J. Chandrasekar, B. L. R. D. Moor, D. S. Bernstein, and A. Ridley, "What is the ensemble kalman filter and how well does it work?," in *2006 American Control Conference*, June 2006.

[62] G. Evensen, "The Ensemble Kalman Filter: Theoretical formulation and practical implementation," *Ocean Dynamics*, vol. 53, no. 4, pp. 343–367, 2003.

Glossary

List of Acronyms

TU Delft	Delft University of Technology
NREL	National Renewable Energy Laboratory
KIT	Karlsruhe Institute of Technology
EIA	Energy Information Administration
IEA	International Energy Agency
GHG	greenhouse gas
LCOE	levelized cost of energy
IPCC	Intergovernmental Panel on Climate Change
AIC	Axial-induction control
WRC	Wake redirection control
WFSim	WindFarmSimulator
SOWFA	Simulator fOr Wind Farm Applications
PALM	Parallelized Large Eddy Simulation Model
FLORIS	FLOw Redirection and Induction in Steady-state
CFD	computational fluid dynamics
RANS	Reynolds-Averaged Navier-Stokes
LES	Large Eddy Simulation
DNS	Direct Numerical Simulation
URANS	Unsteady Reynolds-Averaged Navier-Stokes (RANS)

DES	Detached Eddy Simulation
FAST	Fatigue, Aerodynamics, Structures and Turbulence
ADM	Actuator Disk Model
ALM	Acutator Line Model
BEM	Blade Element Model
2D	two-dimensional
3D	three-dimensional
NS	Navier-Stokes
LTI	Linear time-invariant
KF	Kalman Filter
ExKF	Extended Kalman Filter
EnKF	Ensemble Kalman Filter
UKF	Unscented Kalman Filter
MVUE	minimum-variant unbiased estimator
H.O.T.	higher order terms
CI	Co-Variance Intersection
EI	Ellipsoidal Intersection
ICI	Inverse Co-Variance Intersection
RMS	root mean square