2020 American Control Conference
Denver, CO, USA, July 1-3, 2020

Machine Learning Control for Floating Offshore Wind Turbine Individual Blade Pitch Control

Michael B. Kane, *Member, IEEE*

Abstract — **The cost of energy from current floating offshore wind turbines (FOWTs) are not economical due to inefficiencies and maintenance costs, leaving significant renewable energy resources untapped. Co-designing lighter less expensive FOWTs with individual pitch control (IPC) of each blade could increase efficiencies, decreases costs, and make offshore wind economically viable. However, the nonlinear dynamics and breadth of nonstationary wind and wave loading present challenges to designing effective and robust IPC for each desired location and situation.**

This manuscript presents the development, design, and simulation of machine learning control (MLC) for IPC of FOWTs. MLC has been shown effective for many complex nonlinear fluid-structure interaction problems. This project investigates scaling up these component-level control problems to the system level control of the NREL 5MW OC3 FOWT. A massively parallel genetic program (GP) is developed using MATLAB Simulink and OpenFAST that efficiently evaluates new individuals and selectively tests fitness of each generation in the most challenging design load case. The proposed controller was compared to a baseline PID controller using a cost function that captured the value of annual energy production with maintenance costs correlated to ultimate loads and harmonic fatigue. The proposed controller achieved 67% of the cost of the baseline PID controller, resulting in 4th place in the ARPA-E ATLAS Offshore competition for IPC of the OC3 FOWT for the given design load cases.

I. Introduction

The total technical potential of offshore wind in the U.S. could supply the entire U.S. demand for electric energy. However, over half of this resource lies in waters deeper than 60 m [1]. The barrier lies in the current levelized cost of energy (LCOE) from floating offshore wind turbines (FOWTs) appropriate for depths greater than 60 m is three time greater than terrestrial wind turbine technologies [2]. In order to bring these costs down, FOWTs must become more efficient, lighter, and easier to deploy. Such a next generation of FOWTs present opportunities for advanced control to maximize power generation while minimizing ultimate and fatigue loads caused by dynamic wind and wave forces.

A. Overview of FOWT Individual Pitch Control

FOWT designs rely on combining two proven technologies: wind turbine towers, nacelles, and blade designs from terrestrial applications and floating foundations commonly used in the oil and gas sector. The three most common floating foundations considered for FOWTs are spar-buoy, spar-submersible, and tension leg platforms (TLPs). Spar designs typically have heave natural frequencies much less than that of the applied wave loading, while TLPs are stiffer with frequencies that lie above the wave loading dynamics. The rotational inertia from moving blades, increased (intentional) wind loading, and higher center of gravities preset new challenges for these floating foundations. Additionally, the heave and pitch movements of the floating foundation present greater ultimate and dynamic loads on the turbine components than in terrestrial applications.

M. B. Kane is with the Department of Civil and Environmental Engineering, Northeastern University, Boston, MA 02115 USA (e-mail: mi.kane@northeastern.edu).

Fortunately, these new challenges come with new opportunities for control: namely control of the aerodynamic loading of the blades through controlling generator torque and blade pitch. When operating at maximum power (i.e., Region III) the generator's maximum efficiency occurs within a narrow range of rotational speed. Therefore, torque control typically aims for constant speed. This leaves blade aerodynamic angle of attach, i.e. pitch, control to maximize energy harvesting from the wind and minimize structural loading caused by the wind and wave disturbances.

In the latest generation of turbines, mechanisms in the nacelle adjust the angle of attack of each blade, a technology called individual pitch control (IPC). The typical 10°/s bandwidth of these actuators enables controlling the thrust of each blade as it rotates. A wide variety of control design approaches have been proposed in the literature for IPC, primarily for terrestrial applications: linear quadratic Gaussian, linear quadratic regulators (LQR), multi-blade coordinate methods (MBC) in the time domain and Ziegler-Nichols, H_∞, and robust quantitative feedback theory (QFT) in the frequency domain [3].

Realization of these control algorithms rely on analytic models of the system dynamics; additionally, many of the methods require linear models. However, the rotating blades, nonstationary wind and wave loading, and nonlinear hydrodynamics of floating foundations present significant challenges to realizing these controllers for FOWTs. The most effective approaches utilize model linearization, transitioning between controllers as sea states and other conditions changes. However, even linearization struggles to be effective for rapidly changing conditions or strong nonlinearities.

B. Overview of ATLAS Challenge

With the goal of accelerating the reduction of FOWT LCOE through advanced IPC, the U.S. Department of Energy's (DoE) Advanced Research Projects Agency–Energy (ARPA-E) ran the Aerodynamic Turbines with Load

Attenuation Systems (ATLAS) competition in the spring of 2019 [4]. The Offshore Challenge was based on IPC of the NREL 5MW baseline turbine (i.e., blades and nacelle assembly) attached to the OC3 Hywind Spar floating platform [5].

A MATLAB Simulink simulation environment was provided based on the OpenFAST computer program [6] capable of modeling servo-dynamics (i.e., control system and actuators and power generation), structural dynamics (i.e. rotor dynamics, drivetrain dynamics, nacelle dynamics, tower dynamics, and platform dynamics), mooring dynamics, hydro-dynamics, and aero-dynamics. A dozen design load cases (DLCs) were provided, inspired by design standards for FOWTs, that explore the range of wind speeds, yaw error, pitch offset, wind conditions (e.g., turbulent, direction change, extreme wind shear, and extreme gusts), and dirty aerodynamics (i.e., the airfoil performance is reduced to approximate the effect of upwind turbines). A scoring function processes the simulation outputs for a given controller under the twelve DLCs, comparing the response to a baseline collective pitch control (CPC) PID controller. This cost function aims to capture the positive value of annual energy production (AEP) and maintenance costs. These costs for the blades, hub, nacelle, tower, and platform driven are assumed to be correlated to the ultimate loads and fatigue loads. The harmonic fatigue loads are calculated at frequencies associated with the floating platform pitch (0.036 Hz), once-per-revolution (1P) (0.2 Hz), tower for-aft 1-st mode (0.5 Hz), 2P (0.4 Hz), 3P (0.6 Hz), blade edgewise dynamics (0.95 Hz, 1.1 Hz, and 1.3 Hz), and drivetrain torsion (1.7 Hz). The objective of the competition is to develop a control algorithm to minimize the cost function, where a cost of 1.0 is equal to the performance of the baseline controller.

The remainder of this manuscript presents the development, design, and testing of a machine learning control (MLC) and associated design tools which aims to overcome the challenge of designing a 3DOF FOWT IPC controller for a nonlinear system subject to a wide variety of loading conditions.

C. *Overview of Machine Learning Control*

Machine Learning Control (MLC) [7] uses a genetic programming global optimization architecture to learn not just the weights of a predefined control architecture, but both the control logic architecture and weights. A tree of arithmetic operations, schematically shown in Figure 1 , specifies this control logic, a single instantiation of which forms an *individual* in the *population*. Elitism, replication, crossover, and mutation form genetic operations that when applied in a random weighted manor, lead the population to *evolve* over multiple generation. In each generation the *fitness* of each individual is computed, and the fitter individuals are more likely to have their genetic material make up the next generation.

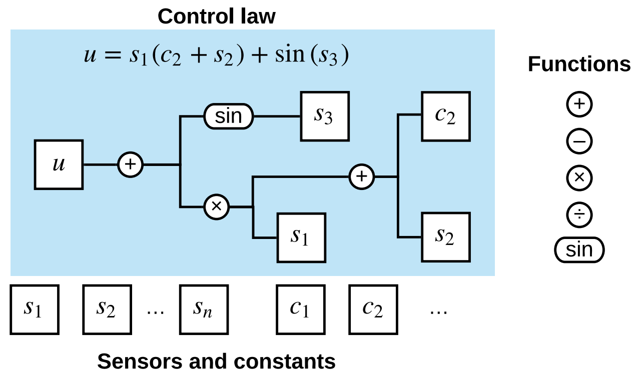

Figure 1 Schematic of an invidiual control law in the MLC population

MLC has been shown effective for many complex nonlinear fluid-structure interaction problems [7]. This project investigates scaling up these component-level control problems to the system level control of the NREL 5MW OC3 FOWT.

II. Formulation of MLC IPC

This algorithm is implemented with the open source code OpenMLC [8]. This code was modified in two significant ways for this project: full parallelization of the generation, evaluation, pre-evaluation, and evolution; and evaluation of the population against multiple environments, i.e. design load cases (DLC). OpenMLC has a parallelization option, but it only parallelized the evaluation of individuals, while the generation, pre-evaluation, and evolution processes added up to a significant amount of time when executed serially. The pre-evaluation parallelization was especially helpful, since the pre-evaluation required compiling and running an open-loop Simulink model of the control logic and pitch actuators. Evaluation of individuals in OpenMLC assumed that the environment in which individuals were tested for fitness remained constant from generation to generation; either a single DLC, or all design load cases evaluated in every generation. However, in nature, the environment in which evolutions occurs is constantly changing and evolution and diversity provide a critical element to the robustness of life. This heuristic was implemented in OpenMLC by first testing each DLC sequentially in the first generations while remembering the difficulty of each DLC, defined as the mean value of the best quarter of the population. After this initialization, the current most difficult DLC is chosen as the environment in the current generation, updating the difficulty indices as appropriate after evaluation. By evolving after each DLC instead of waiting for all DLCs to be evaluated, the space of feasible control laws is explored quicker. However, further study is required to understand and mitigate oscillations that may occur in the optimization convergence and how much *memory* may be required in the genetic operation parameters to robustly distribute genetic material across all the DLCs. After each generation, on a separate machine, the 8 best individuals were evaluated on the entire DLC suite.

As with any control design processes, parameters must be chosen and set prior to running the MLC design algorithm. Table I lists the parameters that are either particularly notable. Items marked with a * were kept as the OpenMLC defaults for similarly complex problems.

TABLE I KEY PARAMETERS OF THE MLC DESIGN ALGORITHM

Parameter Name	Parameter Value
Number of sensors[a]	31
Number of actuators[b]	3
*Population size	500
*Operations	{ (+), (−), (×), (÷), sin(·), cos(·), log(·)}
*Num. of elite individuals replicated	10
*Probability of crossover	50%
*Probability of replication	10%
*Probability of mutation	40%

a. The sensors are listed in Figure 2 .
b. The outputs are the differential IPC from the pitch specified by the baseline controller

III. MLC IPC Implementation Challenges

The primary challenge of most machine learning use-cases is gathering enough diverse data to learn the system. This is especially challenging in the case of IPC, where calculating the objective function for a full suite of design load cases takes approximately an hour on a desktop PC. A typical MLC evolves 500 individuals over 50 generations, requiring a minimum of 25,000 cost function evaluations. This minimum doesn't include individuals that were randomly generated through initial creation or evolution that resulted in unstable responses (i.e. CF>1000) and had to be culled. The following methods were developed to make the FOWT IPC MLC computationally feasible

A. *Selective sensing*

The FAST model outputs 110 potential sensors, but only 31 entered the genetic pool which were heuristically important. This reduced the number of individuals needed in the population to explore the space of admissible controls.

B. *Pre-evaluation*

A randomly generated control law is unlikely to be effective, let alone even stable. Instead of running a full DLC simulation to determine that an individual control law is grossly infeasible, new individuals were pre-evaluated by passing the sensor histories from the baseline controller through an open-loop Simulink model of the controller to assert that the controller was open-loop stable and produced feasible pitch commands (e.g. within bounds, non-constant, and within rate limits). Infeasible individuals were replaced.

C. *Variations in DLCs*

As mentioned above, each generation was tested on only one DLC, the most difficult DLC encountered up until that point. This enabled more evolution to occur within a given amount of wall-clock time and produced visual confirmation of MLC convergence.

D. *Parallelization*

The code supplied for the ATLAS competition runs all DLCs in series, taking an hour of wall-clock time. However, each of these DLC is independent of the others and are therefore parallelizable. Furthermore, the evaluation of a population is parallelizable. The challenge with the parallelization was isolating each FAST input/output and Simulink files, which was easily overcome by copying and uniquely renaming the input files before each evaluation and changing the working directory of each parallel worker to a unique temporary location.

E. *Simulink design*

Each individual's genetic program control law is converted into a string representing an equivalent Matlab command. This command is then written to a Simulink MATLAB Function block, compiled, and executed as part of the Simulink/FAST simulation.

F. *Cloud computing:*

Once the code was parallelized, the effective speed-up was limited only by the computational power available. To this end, a virtual machine (VM) with 96 vCPUs (48 logical cores) @ ~2 GHz with ~300 GB of RAM and 500 GB of HDD was run in the Cloud (both AWS and Google Cloud were tested).

These improvements enabled the submitted controllers to run for 100 generation, lasting 3-days of wall-clock computation time, or approximately 9 months of CPU-clock time. All these analyses produced over 250 GB of data, that while needed only temporarily to compute the population fitness, was saved for future analyses and development of parsimonious models that could speed up future designs.

IV. Results

The 31 sensors (along with constants) become leaf nodes on the genetic arithmetic tree, forming the control law that determines the IPC. Through evolution, the population will learn the most useful sensors for developing improved IPCs. Figure 2 shows the average of how often each sensor is used in the control logic of the 10 best overall individuals. Before being fed into the MLC control laws, each of these sensors are preprocessed by subtracting the mean of the signal from all the DLCs with the baseline controller and dividing by the variance. The following sensors are available to the algorithm:

- `RotSpeed`: Rotor rotational speed
- `Azimuth`: The azimuth (angle of rotation) of the rotor
- `BldPitch[N]`: The pitch of blades 1, 2, and 3
- `RootM[y|z]c[N]`: The moment in the Y and Z directions at the base of blade 1, 2, and 3
- `RotTorq`: The torque measured at the rotor axle

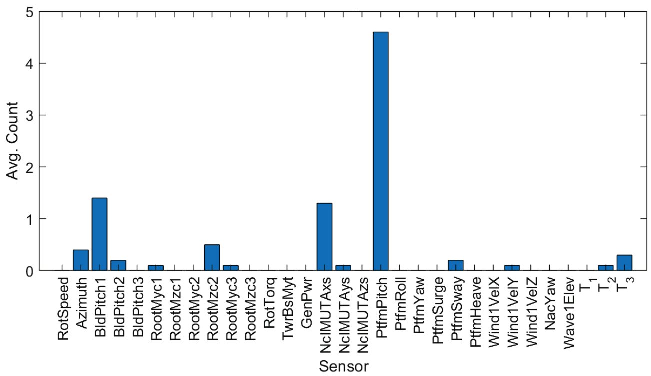

Figure 2 Histogram of sensor use by population.

- `TwrBsMyt`: The moment at the base of the tower in the Y direction
- `GenPwr`: Power produced by the generator
- `NcIMUTA[x|y|z]s`: Acceleration at the nacelle in the X, Y, and Z directions
- `Ptfm[Pitch|Roll|Yaw|Surge|Sway|Heave]`: Platform pitch, roll, yaw, surge, sway, and heave
- `Wind1Vel[X|Y|Z]`: Wind velocity the X, Y, and Z directions
- `NacYaw`: Yaw of the nacelle
- `Wave1Elev`: Wave elevation

TABLE II BEST OVERRALL IPC ALGORITHM

$$u_1 = \mathrm{PtfmPitch} - 0.142$$

$$u_2 = (\mathrm{PtfmPitch} + 6.638) + \mathrm{NcIMUTAxs} \cdot \left(\frac{-2.351 - \left(\frac{\sin\left(\log\left(\frac{\left(\frac{\cos((-9.914\,\mathrm{BldPitch1}) + \cos(-6.33))}{A1} \right)}{\cos(\cos(-1.087))} \right) \cdot \sin(8.177) \right)}{A2} \right) - \log(-1.526)}{A3 + \sin\left(\left(\frac{\log\left(\frac{5.71 + \left(\frac{\sin(\mathrm{BldPitch1})}{\left(\frac{9.668 \cdot 2.966}{\sin(\mathrm{BldPitch1})} \right) \cdot \cos(-4.704) \cdot 0.8755} \right)}{\cos(-7.143)} \right)}{-5.623} \right) + \left(\frac{\sin(\mathrm{NcIMUTAys})}{\left(\frac{A4 + \log(7.637)}{\sin(\mathrm{PtfmPitch})} \right)} \right) \right)} \right)$$

$$A1 = \log(3.704) \cdot (-5.632) + (9.005 \cdot 3.656) \cong 25.5$$

$$A2 = 7.991 + \left(\frac{\cos(-3.625)}{-4.329} \right) \cong 8.20$$

$$A3 = \log(\cos(\cos(0.258) \cdot 0.8755)) \cdot \sin(8.177) \cong -0.390$$

$$A4 = \left(\frac{\left(\frac{-2.219}{8.139} \right)}{7.001} \right) \cong -0.0389$$

$$u_3 = \left(\cos(7.34) + \left((\mathrm{PtfmPitch} - 1.231) + \left((\mathrm{PtfmPitch} + 9.974) + (\mathrm{PtfmPitch} - 0.142) \right) \right) \right)$$

Found by the MLC in generation 57 of 100 with a score 0f 0.609

Division and log functions are protected to guarantee real outputs for all inputs

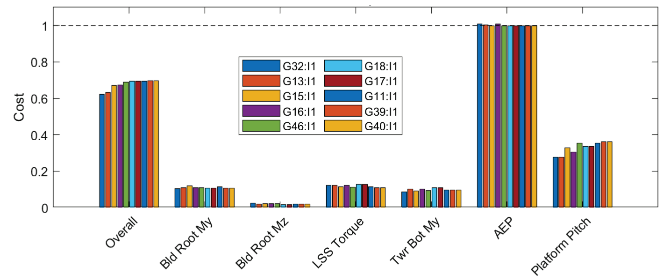

Figure 3 Absolute cost contribution per metric variable. Thebaseline achieved 1.0 Overall, which is equal to the sum of the other costs divided by the baseline normalized AEP. The baseline value for these variables are {1.00, 0.11, 0.02, 0.11, 0.11, 1.00, 0.65} respectively.

- `T[1|2|3]`: Tension in the three mooring lines
- `[X|Y|Z]`: The downwind, cross-wind, and vertical directions.

The best overall individual in last 40 generations was found in generation 67 out of 100 with an overall cost function that reduced the cost of the baseline PI CPC by 62%. Table II shows the resulting algorithm, which when summed with the baseline PI CPC yields the command for each blade. It is apparent that the controller is nearly an affine function of `PtfmPitch` plus an additional complex component for blade 2.

Figure 3 figure shows the variables that add up to the Overall cost. It is apparent that the great saving in overall cost is attributed to the reduction in platform pitch. This is why the MLC learned that `PtfmPitch` is such an important sensor. The overall costs for the 10 best overall individuals are {0.6212, 0.6316, 0.6715, 0.6724, 0.6898, 0.6950, 0.6950, 0.6954, 0.6959, 0.6959} respectively.

Figure 4 shows the convergence of the last 50 of the 100 generations genetic program evolved over. The full DLC suite was used to evaluate the 8 best individuals after each generation, and the lowest of these cost functions in each generation form the blue line. The best individual was found in generation 67/100, then in generation 68, upon evolving

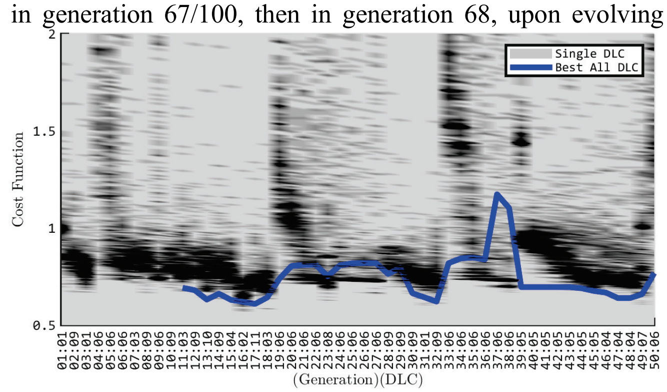

Figure 4 Evolution of population fitness histograms. *(The axes are limited to a max of 2, with many individuals lying outside this limit. The grey shading shows the histogram of the cost functions of each individual in the population, evaluated for the most difficult case prior to that generation. The full DLC suite was used to evaluate the 8 best individuals after each generation, and the lowest of these cost functions in each generation form the blue line.)*

from case `DLC120_ws19_yeNEG_s3_r2` (i.e. design load case 120 with 19.4 m/s wind speed with -10° error in yaw sensor) to case `DLC140_ws13_ye000_s0_r1_ECD` (i.e., design load case 120 with 13.4 m/s wind speed, zero yaw error, and an extreme change in direction of wind), the controller became highly unstable across the population, requiring 14 generations to even approach a similar minimum. This is characteristic of the aforementioned oscillations in the convergence of the global GP optimization problem.

V. Conclusion

This works demonstrates a proof of concept for machine learning control (MLC) for individual pitch control (IPC) of blades in floating offshore wind turbines (FOWTs), as well as highlighting challenges of MLC for control of large multi-physics systems. The approached was executed on the NREL 5MW OC3 FOWT under design load cases (DLCs) specified by the ARPA-E ATLAS competition. The genetic program ran for 100 generations with 100 individuals in each generation. The computations took 72 hours on a cloud computer with 96 vCPUs and 300 GB of RAM. The best individual created essentially an affine control law with respect to platform pitch with additional nonlinear terms using the nacelle acceleration and blade pitch. The resulting controller achieved 67% of the cost of the baseline PID controller on the training test cases, primarily by reducing the platform pitch without reducing annual energy production. In blind testing during the ATLAS competition, the controller demonstrated its robustness by achieving a mean 66.8% reduction over baseline.

The approaches of other ATLAS competitors are confidential; however, PID, quantitative feedback theory (QFT) [3], feed forward control [9], model predictive control (MPC) [10], and linear quadratic gaussian (LQG) [11] controllers have all been considered for this type of problem. With further investigation and improvements, MLC may become a viable alternative to these approaches due to its ability to automatically design controllers for each new turbine's unique site and conditions, given appropriate (measured or simulated) training data. Furthermore, the MLC approach could be modified to include internal controller states that could add integral-type action.

Convergence during the evolution was non-monotonic, resulting in the best individual being found in generation 67 of 100. This essentially resulted in 33 generations of wasted computations in hindsight. To mitigate this issue, each individual could be evaluated against a DCL chosen with a weighted probability by the case difficulty. This randomness within and across generations should dampen optimization oscillations while still favoring evaluation of difficult DLCs. With these and other performance modifications such as learned parsimonious control-design models, MLC could become a viable and valuable approach to designing complex nonlinear control laws for complex systems such as floating offshore wind turbines.

The codes for this work are available here: https://github.com/NEU-ABLE-LAB/ATLAS_Offshore_MLC.

Acknowledgment

The author thanks U.S. Advanced Research Project Agency–Energy (ARPA-E) program director Mario Garcia-Sanz and the organizers of the Aerodynamic Turbines with Load Attenuation Systems (ATLAS) competition for providing this challenge and the simulation environment.

References

[1] Advanced Research Project Agency - Energy (ARPA-E), "AERODYNAMIC TURBINES, LIGHTER AND AFLOAT, WITH NAUTICAL TECHNOLOGIES AND INTEGRATED SERVO-CONTROL (ATLANTIS)," U.S. Department of Energy, Washington, D.C., Funding Opportunity Announcement DE-FOA-0002051, Jan. 2019.

[2] T. J. Stehly, P. C. Beiter, D. M. Heimiller, and G. N. (ORCID:0000000279833050) Scott, "2017 Cost of Wind Energy Review," National Renewable Energy Lab. (NREL), Golden, CO (United States), NREL/TP-6A20-72167, Sep. 2018.

[3] L. H. Wheeler and M. Garcia-Sanz, "Wind Turbine Collective and Individual Pitch Control Using Quantitative Feedback Theory," presented at the ASME 2017 Dynamic Systems and Control Conference, 2017, p. V001T25A005-V001T25A005, doi: 10.1115/DSCC2017-5197.

[4] Advanced Research Project Agency - Energy (ARPA-E), "ATLAS: Offshore Wind Challenge | HeroX," *HeroX*, 2019. [Online]. Available: https://www.herox.com/atlas-offshore-challenge. [Accessed: 26-Sep-2019].

[5] J. Jonkman and W. Musial, "Offshore Code Comparison Collaboration (OC3) for IEA Task 23 Offshore Wind Technology and Deployment," *Contract*, vol. 303, pp. 275–3000, 2010.

[6] "OpenFAST | NWTC Information Portal." [Online]. Available: https://nwtc.nrel.gov/OpenFAST. [Accessed: 06-Jun-2019].

[7] T. Duriez, S. L. Brunton, and B. R. Noack, *Machine Learning Control-Taming Nonlinear Dynamics and Turbulence*. Springer, 2017.

[8] T. Duriez, *tduriez/OpenMLC-Matlab-2*. 2019.

[9] K. Selvam, S. Kanev, J. W. van Wingerden, T. van Engelen, and M. Verhaegen, "Feedback–feedforward individual pitch control for wind turbine load reduction," *Int. J. Robust Nonlinear Control*, vol. 19, no. 1, pp. 72–91, 2009, doi: 10.1002/rnc.1324.

[10] M. Mirzaei, M. Soltani, N. K. Poulsen, and H. H. Niemann, "An MPC approach to individual pitch control of wind turbines using uncertain LIDAR measurements," in *2013 European Control Conference (ECC)*, 2013, pp. 490–495, doi: 10.23919/ECC.2013.6669729.

[11] S. Nourdine, H. Camblong, I. Vechiu, and G. Tapia, "Comparison of wind turbine LQG controllers using Individual Pitch Control to alleviate fatigue loads," in *18th Mediterranean Conference on Control and Automation, MED'10*, 2010, pp. 1591–1596, doi: 10.1109/MED.2010.5547822.