JOURNAL OF
EMERGING INVESTIGATORS

Article

PID and fuzzy logic optimization of the pitch control of wind turbines

Han Sheng Zhou[1], Xiao Hui Wang[1]
[1] West Vancouver Secondary School, British Columbia, Canada

SUMMARY

In the future, wind turbines may become an important source of energy generation because of their scalability and ability to be constructed in most places on Earth. To meet net-zero greenhouse gas emissions by 2050, emissions need to be reduced globally. This can be achieved in part by making renewable energy cheaper than fossil fuels. The energy production of wind turbines may be increased by integrating a smart controller that can optimize the pitch angles of the turbine. A smart controller would also reduce maintenance and servicing costs while increasing the turbine's functional lifespan. By integrating proportional-integral-derivative (PID) and fuzzy logic controllers, which are used in many industrial control systems, the pitch control of a wind turbine could be optimized to generate the most electricity while avoiding overshoot. We hypothesized that the fuzzy-PID controller, which uses both fuzzy and PID controller algorithms, would be the most optimal for electricity generation by having the shortest settling and rise time with very little overshoot and settling error. We modeled a proportional-integral (PI) controller, a proportional-derivative (PD) controller, a PID controller, 7 and 9 membership function fuzzy controllers, and a fuzzy-PID controller in Simulink. In general, we observed that fuzzy logic-based controllers rose and settled slower than PID-based controllers with less overshoot and steady-state error. Overall, the PID controller had a fast rise time with little overshoot and appeared to be the most optimal for electricity generation, but the PI controller provided the best life span for the wind turbine by having zero overshoot with a slightly slower rise time.

INTRODUCTION

Even with the recent development of more efficient renewable energy sources, the global community is still behind pace in reaching net-zero emissions by the 2050 goal established by the Paris Agreement (1). With the total amount of wind turbine energy generation forecasted to more than double by 2028 compared to 2023, continued wind turbine refinement and development will likely be crucial in meeting our net-zero emissions by the 2050 goal (2). In addition to simply building more wind turbines, improving the efficiency of turbines can also boost their energy output.

In general, the main problem facing wind turbines is unpredictable wind speeds. For example, if the wind is blowing too hard and exceeds the "cut-out speed" for a wind turbine, major stress is put on the blades and inner mechanisms, which causes friction and damage to the internal bearings and gears. On the other hand, if the wind is below the "cut-in speed" of the wind turbine, it will not be able to generate sufficient electricity. Only when the wind is blowing at a speed between the "cut-in speed" and the "cut-out speed" will the wind turbine be allowed to turn and generate electricity. In order to reduce stress on the internal mechanisms, a smart controller can stall the wind turbine (increasing the angle of attack of the rotor blades) so the turbine catches less wind and therefore turns less (3).

The field of wind turbine pitch control optimization has already explored various methods of control. Different types of controllers have been designed for wind turbines to ensure the continuity of operation even when the external wind speed rises higher than the rated value by adjusting the pitch rotor angles. Most notably, research has been conducted on different variations of PID controllers. A PID controller is a closed-loop control system used in automation to regulate processes and systems. PID controllers are employed in applications such as temperature control, speed control, and position control (4).

A PID controller is made up of three components: the proportional (P), integral (I), and derivative (D) terms, not all of which must be used. For example, one can choose to forgo the D term, resulting in a proportional-integral (PI) controller. The P term is an output directly corresponding to the current error (the distance between the desired position and the current position), while the I term is the summation of all past errors over time (4). The I term reduces the steady-state error by continuously adjusting the output based on the accumulated error, ensuring the controller can help the system reach its desired setpoint, even with constant disturbances (4). Finally, the D term considers the rate of change of the error with respect to time and reduces the overshoot of the system, predicting future behavior based on the current rate of change (4). The output of the PID controller is the sum of the P, I, and D terms, each multiplied by their respective coefficients (K_p, K_i, and K_d), which are tuned to achieve the most optimal system performance (Eq.1) (4).

$$u(t) = K_p \cdot e(t) + K_i \cdot \int_{-\infty}^{\infty} e(t)dt + K_d \cdot \frac{de}{dt} \qquad \text{(Eq.1)}$$

Another controller investigated for this purpose is the fuzzy controller, which utilizes fuzzy logic to do computations. Fuzzy logic control is a control system based on human reasoning processes instead of mathematical models like PID controllers. Fuzzy logic control deals with uncertainty and imprecision in a human-like manner by using linguistic

https://doi.org/10.59720/24-178

variables instead of numeric values for inputs and outputs (5). Numeric inputs are "fuzzified" and entered into a fuzzy ruleset (a list of if-then statements that dictates how the input linguistic values correspond to the output linguistic values) and then "defuzzified" back into numeric values, which are the output (5). During the fuzzification process, the fuzzy ruleset makes calculations based on membership functions, which specify the degree to which a given input belongs to a set on a range of 0–1 (5). A fuzzy logic controller is most useful when the system dynamics are complex and unpredictable, which wind streams often are (5).

Notable research has been conducted on fuzzy-PID controllers (6–9). Fractional order fuzzy-PID controllers featuring tunable integral and differential orders on top of a normal PID algorithm have been tested, resulting in better controller performance at the cost of more computational power requirements (10, 11). Furthermore, other research groups have developed adaptive PID controller models with more accurate rotor pitch outputs than conventional PID controllers (12–14). Interestingly, using mathematical analysis to identify stable system boundaries and optimal operation points before integrating the PID systems allows for quicker computation of ideal PID values (15). Also, instead of adjusting PID gains linearly, tuning them non-linearly generates more efficient feedback values and precise models (16, 17). On a similar note, back-propagation (BP) PID controllers combine a BP neural network with a PID algorithm to self-adjust their own weightings and makes the tuning process a lot easier (18). The controllers modelled so far have contributed to the rapid deployment of wind turbines in offshore and mountaintop settings, where wind conditions are unpredictable.

The efficiency of controllers is measured using the following four metrics: rise time, settling time, steady-state error, and overshoot. Rise time is the time it takes for the controller to achieve 100% of the desired value, while settling time is the time it takes for the controller to arrive at a constant, non-changing value. Overshoot is defined as the percentage the peak of the controller goes above the desired value, and steady-state error is the difference between the constant, non-changing value and the desired value, also expressed as a percentage (19).

This paper will focus on rotor pitch control for a Leitwind LTW77 wind turbine by building and tuning PID and fuzzy controllers to maximize electricity generation. The Leitwind LTW77 turbine has a "cut-out speed" of 25 m/s and a "cut-in speed" of 3 m/s (20). Only when the wind is blowing between 3 m/s to 25 m/s will the Leitwind LTW77 be allowed to turn and generate electricity.

We hypothesized that a fuzzy-PID controller would be the most optimal wind turbine controller by having the shortest settling and rise times with the least overshoot and steady-state error, as it first minimizes the overshoot and steady-state errors through the use of a PID controller, then inserts those values into a fuzzy controller. By modelling a PI controller, a PD controller, a PID controller, 7 and 9 membership function fuzzy controllers, and a fuzzy-PID controller in MATLAB and Simulink, we were able to compare their rise time, settling time, steady-state error, and overshoot against each other. PID and fuzzy controllers were chosen for this study as they do not require cumbersome numerical calculations and are better suited for controlling constantly changing and unpredictable operating conditions. In general, we found that fuzzy logic-based controllers settled slower than PID-based controllers but had less overshoot. We found that the PID controller was the best all-around controller for fast rise time and low overshoot. If the longevity of a wind turbine is valued instead, the PI controller should be implemented as it had zero rotor pitch overshoot despite having a slightly slower rise time.

RESULTS

To find the most optimal controller for wind turbine electricity generation, we simulated each controller in a Simulink program from MATLAB (21, 22). We obtained the time-domain values of rise time, settling time, overshoot, and steady-state error for each controller's performance against a unit-step function in order to best judge their efficiency for power production.

No Controller

First, we simulated a wind pitch control system without any controllers as a control simulation to compare our other controllers against. We constructed a Simulink model of our Leitwind LTW77 **(Figure 1A)**. We added a slight electrical delay of 0.005 s to every controller model to account for the information processing done by the wind sensor. We simulated the unit-step response time of the controller for 2.5 s **(Figure 1B)**. The no controller turbine was found to have a rise time of 0.989s, a settling time of 1.861s, an overshoot of 0.3%, and a steady-state error of 75% **(Table 1)**.

PD, PI, PID Controllers

Similarly, we recorded the unit-step response time of a Simulink model for the Leitwind LTW77 wind turbine with a PD controller **(Figure 1C)**. The PD controller was found to have a rise time of 0.026s, settling time of 0.286s, overshoot of 0.9%, and steady-state error of 0.7% observed from the response graph were tabulated **(Table 1)**. We also constructed a Simulink model for the Leitwind LTW77 wind turbine with the implementation of a PI controller and graphed the unit-step response time **(Figure 1E)**. The PI controller had a rise time of 0.815s, a settling time of 3.709s, an overshoot of 0%, and a steady-state error of 0% **(Table 1)**. Finally, we constructed a Simulink model for our Leitwind LTW77 including a PID controller with its unit-step response time (Figure 1G). The PID controller was found to have a rise time of 0.235s, a settling time of 0.772s, an overshoot of 0.3%, and a steady-state error of 0% **(Table 1)**. To compare the efficiency of our tested Leitwind LTW77 PID controllers, we plotted them on a graph **(Figure 4)**.

We compared the no-controller model, the PI controller, the PD controller, and the PID controller using the following metrics: rise time, settling time, overshoot, and steady-state error. We found that the no-controller had the slowest rise time at 0.989s, while the PD controller had the fastest rise time at 0.026s. Meanwhile, we discovered that the PID controller was the second fastest at 0.235s, and the PI controller had a rise time of 0.815s. Regarding settling time, the PD controller was the fastest at 0.286s while the PI controller was slowest at 3.709s. The no-controller had a settling time of 1.861s while the PID controller took 0.722s to settle. Looking at overshoot, the PI controller was most accurate with 0% overshoot, while the PD controller was most inaccurate with 9% overshoot. The PD and PID controller both had a 0.3% overshoot. Finally, for

https://doi.org/10.59720/24-178

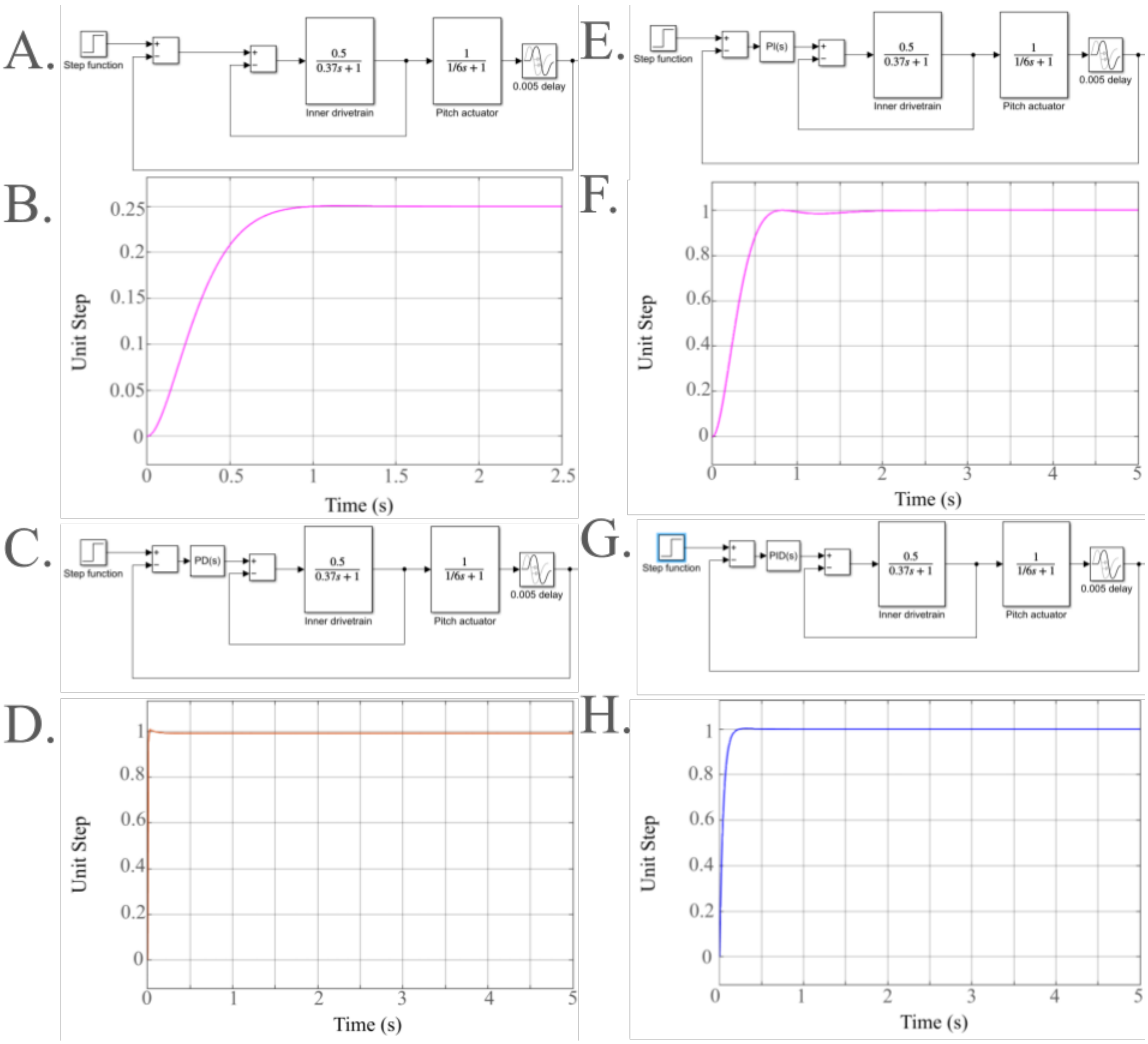


Figure 1: Simulink model and unit-response time graph for no controller, PD controller, PI controller, and PID controller trials. The Simulink model and unit-response time graph for the Leitwind LTW77 wind turbine with (A,B) no controller, (C,D), a PD controller, (E,F) a PI controller, and (G,H) a PID controller is shown. The diagram displays a closed-loop control system modeling the inner drivetrain and pitch actuator of the Leitwind LTW77 wind turbine with a 0.005 second delay from a step-function. All the data were exported from the MATLAB simulation (23).

steady-state error, the PI and PID controller were both steady with 0% error, while the no-controller model had the worst error at 75%. The PD controller had an error of 0.7% **(Table 1)**. Overall, we discovered that the PID controller did second-best or best throughout all the metrics.

7-MF, 9-MF, and Fuzzy-PID Controllers

We constructed the 7-MF fuzzy controller as a subsystem and then implemented it into our Leitwind LTW77 wind turbine model **(Figure 2A, 2B)**. We designed the fuzzy controller to be scalable (it produces a value between zero and one for K_p, K_i, and K_d), and we obtained a value of 6.22 through trial and error for the multiplier constant. We accomplished this by repeatedly adjusting the scalable constant, making note of rise-time each time until the changes resulted in minimal optimization of rise-time. The unit-step response of the 7-MF fuzzy controller wind turbine pitch control system was measured **(Figure 2C)**. The 7-MF fuzzy controller was found to have a rise time of 3.949s, a settling time of 3.949s, an overshoot of 0.1%, and a steady-state error of 0% **(Table 1)**.

Similarly, we also constructed the 9-MF fuzzy controller as a subsystem and then implemented it into the Leitwind LTW77 wind turbine model **(Figure 1D, 1E)**. We found the multiplier constant through trial and error to be 5.90. We recorded the unit-step response of the 9-MF fuzzy controller wind turbine pitch control system. The 9-MF fuzzy controller

https://doi.org/10.59720/24-178

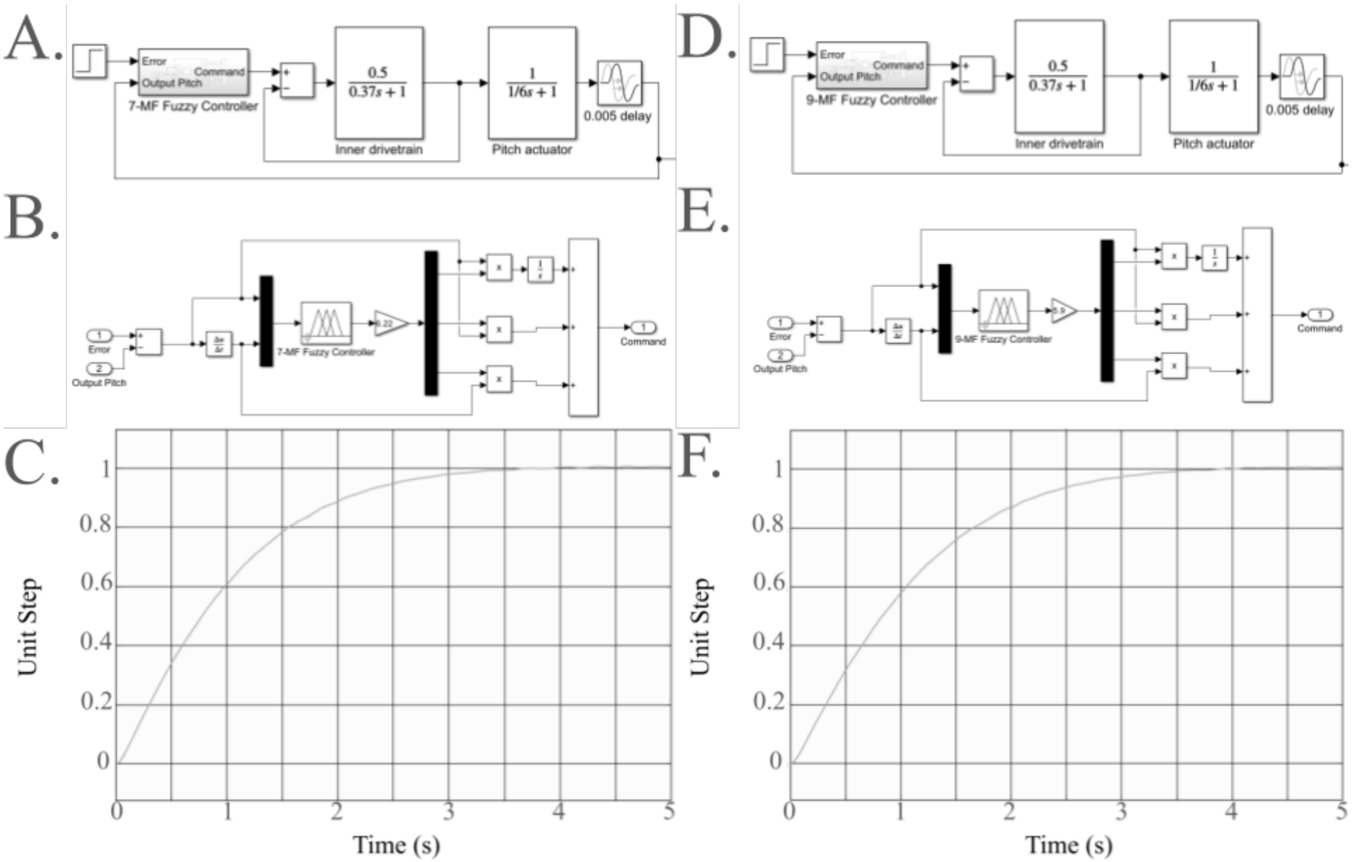


Figure 2: Simulink model, fuzzy controller subsystem and unit-response time graph of 7-MF and 9-MF fuzzy controller. The Simulink model, fuzzy controller subsystem, and unit-response time graph for the Leitwind LTW77 wind turbine with (A,B,C) a 7-MF fuzzy controller and (D,E,F) a 9-MF fuzzy controller is shown. The Simulink model (A,D) displays a closed-loop control system modeling the inner drivetrain, pitch actuator, and fuzzy subsystem of the Leitwind LTW77 with a 0.005 second delay from a step-function. The closed-loop fuzzy control subsystem (B,E) is modeled, with their respective multiplier constants of 6.22 and 5.9 found through trial and error. We extracted the data shown from the MATLAB simulation and fuzzy logic toolbox (23, 34).

was found to have a rise time of 3.880s, a settling time of 3.880s, an overshoot of 0%, and a steady-state error of 0% **(Table 1)**. Compared to the 7-MF fuzzy controller, the 9-MF fuzzy controller lessened rise time and settling time by 0.069s, while the steady-state error was reduced to zero.

We observed an interesting pattern observed from our two fuzzy logic controllers: they settled quickly with very little overshoot and steady-state error. Furthermore, the inclusion of additional membership functions was found to result in faster rise time, faster settling time, less overshoot, and less steady-state error.

For the fuzzy-PID controller, we constructed three subsystems: one for the wind turbine plant, one for the PID controller, and one for the fuzzy controller **(Figure 3B-D)**. Then, we compiiled these three subsystems into our Leitwind LTW77 Simulink model **(Figure 3A)**. We constructed the fuzzy-PID controller by using both a PID controller and a fuzzy controller, then multiplying and subsequently summing their results. We chose the 9-MF fuzzy controller, as we found it to be more accurate than the 7-MF fuzzy controller based on a comparison of their time-domain values. We used a saturation block to ensure the output did not exceed the values of -6 and 6, as that was the maximum rate of change of our pitch actuator. We recorded the unit-step response of the fuzzy-PID controller wind turbine pitch control system **(Figure 3E)**. The fuzzy-PID controller was found to have a rise time of 0.605s, a settling time of 2.366s, an overshoot of 1.2%, and a steady-state error of 0% **(Table 1)**.

To compare the efficiency of our Leitwind LTW77 fuzzy controllers, we plotted them on a graph (Figure 4). We compared the 7-MF fuzzy controller, 9-MF fuzzy controller, and fuzzy-PID controller by their rise time, settling time, overshoot, and steady-state error. Analyzing rise time, the fuzzy-PID controller was fastest with 3.949s, while the 7-MF fuzzy controller took the longest with 3.949s. The 9-MF fuzzy controller took 3.880s. Regarding settling-time, the fuzzy-PID controller was fastest with 2.366s, while the 7-MF fuzzy controller was slowest with 3.949s. The 9-MF fuzzy controller took 3.880s. For overshoot, both the 7-MF and 9-MF fuzzy controllers performed the best with 0% overshoot, while the fuzzy-PID controller did the worst with a 1.2% overshoot. Finally, for steady-state error, both the 9-MF and fuzzy-PID controller did the best with 0% error, while the 7-MF fuzzy controller did worse with a steady-state error of 0.1%. Overall, we found that the fuzzy-PID controller performed the best in three out of four metrics: rise time, settling time, and steady-state error.

Implementation of PID Controller

We then implemented the PID controller into the wind

https://doi.org/10.59720/24-178

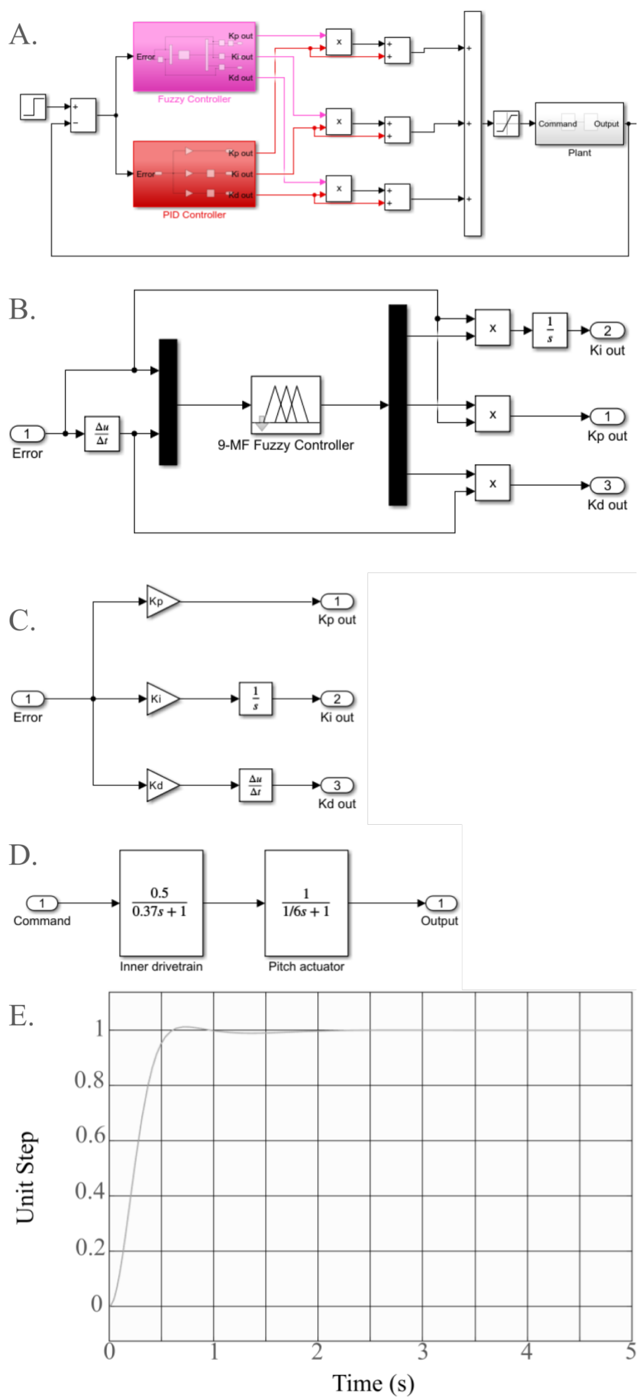


Figure 3: Simulink model, unit-response time graph, and time-domain of the fuzzy-PID controller. A) The Simulink model with a fuzzy and PID subsystem is shown, with the results from the subsystems multiplied and then summed together inside a closed-loop control system. B) The fuzzy subsystem of the 9-MF fuzzy control system was used, as it had a faster rise time and settling time. C) The PID subsystem is shown, with the Kp value staying constant, the Ki value being integrated, and the Kd value being differentiated. D) The inner drivetrain and pitch actuator components of the Leitwind LTW77 model is modeled with transfer functions. E) The unit-response time graph of the fuzzy-PID controller is displayed. We used the MATLAB simulation and fuzzy logic toolbox to display the diagrams (23, 34).

Controller	Rise time (s)	Settling time (s)	Overshoot (%)	Steady-state error (%)
None	0.989	1.861	0.3	75
PI	0.815	3.709	0	0
PD	0.026	0.286	0.9	0.7
PID	0.235	0.722	0.3	0
7-MF Fuzzy	3.949	3.949	0	0.1
9-MF Fuzzy	3.880	3.880	0	0
Fuzzy PID	0.605	2.366	1.2	0

Table 1: Time-domain specifications of none, PI, PD, PID, 7-MF and 9-MF fuzzy, and fuzzy-PID. Rise time is the time taken for the controller to achieve 100% of its desired value. Settling time is the time it takes for the controller to reach a constant, non-changing value. Overshoot is the percentage the peak of the controller overshoots the desired value. Steady-state error is the difference between the constant, non-changing value and the desired value expressed as a percentage. Out of all the controllers, we determined the PID controller to be the most optimal, for its fast rise time and low overshoot. However, the PI controller should be implemented to allow the longest lifespan for a wind turbine as it had zero overshoot despite having a slightly slower rise time. MF = membership function.

turbine model, which we obtained from MATLAB and modified with the manufacturer specifications of the Leitwind LTW77 (23). We found the theoretical pitch angle and the PID-controlled pitch angle to be almost identical given the simulated wind input, which considered both the cut-in and cut-out wind speeds **(Figure 5A, 5B)**. Furthermore, we observed the difference to never exceed +/- 0.3 degrees **(Figure 5C)**. As the control accuracy of the pitch turbine was within one degree, we believe we have succeeded in constructing an accurate and fast-reacting pitch controller.

Overall, we found that the PID controller was the best all-around controller for fast rise time and low overshoot. However, if the longevity of a wind turbine is valued instead, the PI controller should be implemented as it had zero rotor pitch overshoot despite having a slightly slower rise time.

DISCUSSION

In this study, our goal was to find the most optimal controller for electricity generation for a Leitwind LTW77 wind turbine. Analyzing the experiment results, we found that the PID controller was the best controller for electricity generation, featuring a fast rise time and insignificant overshoot. However, some commercial wind turbines are built with longevity in mind. In that case, the PI controller would be the most optimal as it has zero rotor pitch overshoot despite having a slightly slower rise time.

In general, we found the PID-based controllers to have faster rise and settling times compared to fuzzy controllers, but this came at the expense of more overshoot and/or steady-state error. We discovered that the fuzzy-PID controller displayed characteristics observed in both PID controllers and fuzzy controllers, having relatively fast rise and settling times but larger overshoot values with no steady-state error.

Of the controllers we examined, the PID controller was the most optimal one for a Leitwind LTW77 wind turbine model,

JOURNAL OF
EMERGING INVESTIGATORS

https://doi.org/10.59720/24-178

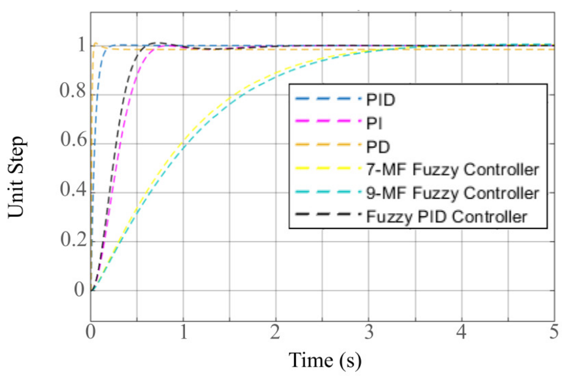


Figure 4: Comparison of the unit-step response time of all controllers. All controllers are shown in unit-step response time graphs. The PI, PD, PID, and fuzzy-PID showcase a fast initial rise-time that overshoots, while the 7-MF and 9-MF fuzzy-controllers reach the unit-step gradually with no overshoot. Overall, the PD controller had the fastest rise time at the cost of the highest overshoot. The PID controller had the second fastest rise time with very minimal overshoot, making it the most optimal controller out of all the controllers tested. MF = membership function. We graphed the data in a MATLAB simulation (23).

as it had the second fastest rise time and settling time after the PD controller. However, the PD controller had a large overshoot and steady-state error, while the PID controller had a slight overshoot of 0.3% and no steady-state error. The overshooting of rotor pitch blades in wind turbines may cause tensions along the blade, which over time requires more preventive care and maintenance services (24). The cost of these services can far exceed the extra power generated by a fast-responding but overshooting controller. Therefore, one could also implement a controller with no overshoot, as this would prevent wear-and-tear of the wind turbine and decrease the frequency of maintenance needed. To summarize, if one wants a wind turbine to generate as much power as possible, the PID controller should be implemented. However, if one values longevity in a wind turbine and wants to save on maintenance costs, the PI controller should be implemented. The 9-MF fuzzy controller also had no overshoot, but it had a slower rise time of 3.880s compared to the 0.815s of the PI controller.

Furthermore, we compared the rise time, settling time, overshoot, and steady-state error of PID, fuzzy, and fuzzy-PID controllers to the results of a similar study by Silpa Baburajan (6). Comparing our PID controller to theirs, our controller reduced the rise time by 3.975s, and the PID controller's settling time was reduced by 97% from 27s to 0.722s. We also reduced overshoot by 97% from 11.8% to 0.3% while steady-state error remained the same. Comparing the 7-MF fuzzy controller in the Baburajan study to our 9-MF fuzzy controller, our 9-MF controller reduced rise time by 43% from 6.81s to 3.880s. In addition, we reduced settling time from 25s to 3.880s, a decrease of 84%. We nullified the overshoot from 0.5% to 0%, while steady-state error stayed at 0%. Finally, we minimized the rise time of the fuzzy-PID controller by 0.025s from 0.63s to 0.605s, and the settling time was diminished by 70% from 8s to 2.366s. However, overshoot increased from 0.02% to 1.2%. Again, the steady-state error remained the same at 0%. Overall, we believe our controllers exhibited

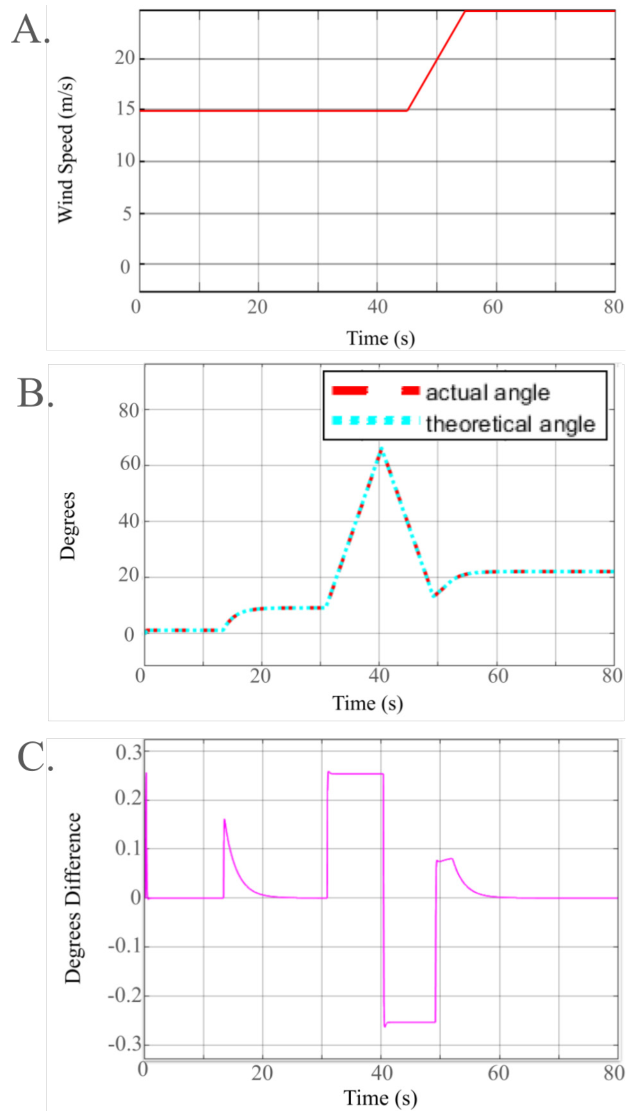


Figure 5: Comparison of the theoretical and measured angle of the PID controller when tested against the showcased wind profile and their angle difference over time. A) The simulated wind model used for testing is graphed (15 m/s from 0s to 45s, increases by 1 m/s from 45s to 55s, then stabilizes at 25 m/s from 55s to 80s). B) The theoretical angle of the rotor pitch blades are superimposed with the PID controller's angle, and they appear to almost overlap. C) The angle difference between the theoretical and measured angle of the PID controller is graphed, and the difference oscillates but never goes beyond +/- 0.3 degrees. Relationship graph between time and angle over eighty seconds. We graphed the data in a MATLAB simulation (23).

improved performance compared to a similar design (6).

As we did not have access to an actual Leitwind LTW77 wind turbine, we were not able to test these controllers on a real-life system. Therefore, we could not directly evaluate the physical effectiveness of these controllers and instead relied on our MATLAB simulations to mimic the process. The MATLAB wind turbine model is not a perfect replica of an actual wind turbine, and there will always be signal noise that is not accounted for in the simulation from sources like friction, temperature, humidity, and the change of wind

https://doi.org/10.59720/24-178

streams. In addition, the controllers implemented in Simulink were assumed to be perfect controllers. Although we did account for a 0.005 s delay, in actual turbines this value differs for different makes and models and also depends on the hardware and software used to control the turbine (25).

Future research may implement these controllers on physical wind turbines and subsequently fine-tune them based on temperature and humidity. These environmental factors should be investigated separately to determine an equation to allow the controller to adjust to different conditions. These results could then be implemented in a more comprehensive controller. We anticipate that this would not only increase the efficiency of electricity generation but also demonstrate our continuing ability to further optimize wind turbine rotor pitch control systems.

MATERIALS AND METHODS

For this study, we used the Simulink program from MATLAB to model the wind turbine pitch (21, 22). We built all controllers (both PID and fuzzy) off the MATLAB default wind turbine model (23).

We calculated the power from a wind turbine using (2) (26).

.

$$P = \frac{1}{2} \cdot C_p(\lambda, \beta) \cdot \rho \cdot A \cdot V^3 \qquad \text{(Eq. 2)}$$

Where p=1.225kg m^3 is the density of air, A= π r^2 is the area swept out by the rotor blades, V is the wind velocity, C(λ,β) is the coefficient of performance, with β being the pitch angle and λ being the tip-speed-ratio. The coefficient of performance returns a value between 0 and 0.593 according to Betz's Law (27).

To model the wind turbine, we derived two transfer functions: one for the drivetrain (gearbox + generator) and one for the pitch actuators (28). The pitch actuator can be modeled by letting β be our initial angle, β_f be our final pitch angle, and T_β be our time interval.
The change in pitch angle is given by (Eq. 3.0):

$$\frac{d\beta}{dt} = \frac{\beta_f - \beta}{T_\beta} \qquad \text{(Eq. 3.0)}$$

$$T_\beta \frac{d\beta}{dt} = \beta_f - \beta \qquad \text{(Eq. 3.1)}$$

Applying a Laplace transformation, we have

$$T_\beta \cdot s \cdot \frac{\beta}{s} = \frac{\beta_f - \beta}{s} \qquad \text{(Eq. 3.2)}$$

$$T_\beta s \cdot \beta = \beta_f - \beta \qquad \text{(Eq. 3.3)}$$

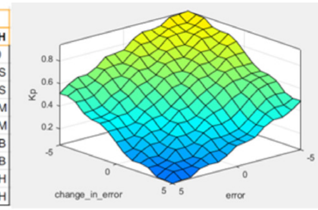
$$T_\beta s \cdot \beta + \beta = \beta_f \qquad \text{(Eq. 3.4)}$$

$$\beta(T_\beta s + 1) = \beta_f \qquad \text{(Eq. 3.5)}$$

$$\frac{\beta}{\beta_f} = \frac{1}{T_\beta s + 1} \qquad \text{(Eq. 3.6)}$$

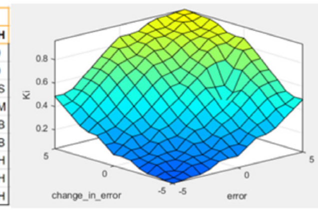
A.

Kp	Change In Error								
Error	NH	NB	NM	NS	0	PS	PM	PB	PH
NH	PH	PH	PB	PB	PM	PS	PS	0	0
NB	PH	PB	PB	PM	PM	PS	PS	0	NS
NM	PB	PB	PM	PM	PS	PS	0	NS	NS
NS	PB	PM	PM	PS	PS	0	NS	NM	NM
0	PM	PM	PS	PS	0	NS	NS	NM	NM
PS	PM	PS	PS	0	NS	NS	NM	NM	NB
PM	PS	PS	0	NS	NS	NM	NM	NB	NB
PB	PS	0	NS	NS	NM	NM	NB	NB	NH
PH	0	0	NS	NS	NM	NB	NB	NH	NH



B.

Ki	Change In Error								
Error	NH	NB	NM	NS	0	PS	PM	PB	PH
NH	NH	NH	NH	NB	NB	NM	NM	NS	0
NB	NH	NH	NB	NB	NM	NM	NS	0	0
NM	NH	NB	NB	NM	NM	NS	0	PS	PS
NS	NB	NB	NM	NM	NS	0	PS	PM	PM
0	NB	NM	NM	NS	0	PS	PM	PM	PB
PS	NM	NM	NS	0	PS	PS	PM	PB	PB
PM	NM	NS	0	PS	PM	PM	PB	PB	PH
PB	NS	0	PS	PM	PM	PB	PB	PH	PH
PH	0	0	PS	PM	PB	PB	PH	PH	PH



C.

Kd	Change In Error								
Error	NH	NB	NM	NS	0	PS	PM	PB	PH
NH	PM	PS	NB	NH	NH	NH	NB	PS	PM
NB	PS	NS	NB	NH	NB	NB	NM	NS	0
NM	0	NS	NB	NM	NS	NS	NS	NS	0
NS	0	NS	NS	NM	NS	NS	NS	NS	0
0	0	NS	NS	NS	NS	NS	NS	NS	0
PS	0	0	0	0	0	0	0	0	0
PM	0	0	0	0	0	0	0	0	0
PB	PH	NS	NS	PS	PS	PS	PS	PS	PH
PH	PH	PB	PM	PS	0	PS	PS	PS	PH

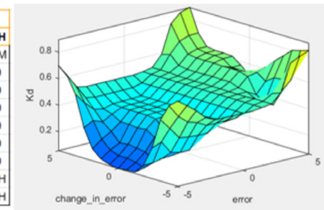


Figure 6: Fuzzy rule sets and their respective rule surface diagrams. The fuzzy rule sets and rule surface diagrams are simulated for the PID controller coefficient variables of A) K_p, B) K_i, and C) K_d. The K_p surface rule diagram has a negative slope from –5 to 5 as a large error or change in error warrants a significant adjustment. The inverse is true for the K_i as a large change in the summation of error warrants a minor adjustment. However, for K_d a significant change in the rate of error would result in a relatively minor adjustment overall, as the rate of error is very subject to change and we intend to construct a stable controller. We used the MATLAB fuzzy logic toolbox to display the surface rule diagrams (34). NH = negative huge, NB = negative big, NM = negative medium, NS = negative small, PS = positive small, PM = positive medium, PB = positive big, PH = positive huge.

Which is our required transfer function. The time interval T_β can be determined from (Eq. 3.1) as shown.

$$T_\beta \frac{d\beta}{dt} = \beta_f - \beta \qquad \text{(Eq. 3.1)}$$

$$T_\beta = \frac{(\beta_f - \beta)}{\frac{d\beta}{dt}} \qquad \text{(Eq. 3.7)}$$

In (Eq. 3.1), β_f is the control accuracy of the pitch angle in degrees, approximated to be one degree. In addition, dβ/dt is the rate of change of pitch angle in degrees per second, and for most wind turbines this number is around three to ten degrees (29). As more detailed manufacturer specifications are inaccessible for our wind turbine, we used a median value of six for the rate of change of the wind turbine angle.

Thus, our time interval is T_β=1/6. Our transfer function for our pitch actuator is given by (Eq. 4).

$$\frac{1}{T_\beta s + 1} = \frac{1}{\frac{1}{6}s + 1} \qquad \text{(Eq. 4)}$$

For the drivetrain's transfer function, Baburajan derives a mathematical mode of the internal gearboxes of the wind

turbine hub and then applies a Laplace transformation, which simplifies to (Eq. 5) (6).

$$\frac{0.5}{0.37s+1} \quad \text{(Eq. 5)}$$

We tested the controllers on the Leitwind LTW77, a medium-sized on-shore wind turbine with three blades (20). According to the manufacturer specifications, the Leitwind LTW77 has a rated power of 1500 kW, a cut-in speed of 3.0 m/s, cut-out speed of 25.0 m/s, and a rated wind speed of 15.0 m/s. The diameter of the blades was listed as 76.7 m, creating a swept area of 4608.0 m². The rotor max speed was stated as 17.8 rotations per minute, with a tip speed of 71 m/s.

PI Controller

We tuned the PI controller using the Transfer Function Based PID Tuner App in MATLAB, giving us values of P=3.4560, I=9.8915 (30). We tested the controller against a unit-step function so its rise time, rise time, settling time, overshoot, and steady-state error could be measured.

PD Controller

We also tuned the PD controller using the Transfer Function Based PID Tuner App in MATLAB, giving us values of P=392.9361, D=29.0666, and N=26977.47233 (30). We tested the controller against a unit-step function so its rise time, rise time, settling time, overshoot, and steady-state error could be measured.

PID Controller

We tuned the PID controller using the Ziegler-Nichol method in MATLAB, giving us values of P=28.6056, I=71.0408, D=2.7618, and N=2576.3286 (31). We tested the controller against a unit-step function so its rise time, rise time, settling time, overshoot, and steady-state error could be measured.

Fuzzy Controllers

We built two fuzzy logic controllers: one with seven membership functions (7-MF) and one with nine membership functions (9-MF) in an attempt to improve upon the 7-MF controller and to decide if more membership functions would yield better results (6). We used the fuzzy logic toolbox in MATLAB/Simulink (32). We tested the fuzzy controllers against a unit-step function so their rise time, rise time, settling time, overshoot, and steady-state error could be measured.

To create a fuzzy logic control system, the first step is fuzzification. In this step, we defined two inputs and three outputs for the fuzzy controller. The two inputs were error and change in error, while the three outputs were K_p, K_i, and K_d, which are the proportional, integral, and derivative weights. We used Gaussian membership functions for inputs and outputs. For the inputs, we selected a range from -5 to 5, which can later be scaled to any size by multiplying the output by a constant. Scaling the output is the same as scaling the input, so it will not cause any computation errors. We divided this range into nine equal segments: negative huge (NH), negative big (NB), negative medium (NM), negative small (NS), zero (0), positive small (PS), positive medium (PM), positive big (PB), and positive huge (PH).

The second step is to construct the fuzzy rule base. We constructed tables for the fuzzy inputs and outputs for K_p, K_i, and K_d, and we entered the linguistic rules into the fuzzy logic toolbox rule editor in MATLAB to extract their surface rule diagrams **(Figure 6)**.

Finally, we defuzzified our fuzzy output by converting from a linguistic value in our fuzzy set to a numerical value. There are several methods of defuzzification, but for our model, we used centroid defuzzification. The centroid of a fuzzy set is found by treating the area as a solid of consistent density and finding its gravitational center using (Eq. 6).

$$\frac{\sum_i \mu(x_i)\cdot x_i}{\sum_i \mu(x_i)} \quad \text{(Eq. 6)}$$

In (Eq. 6), $\mu(x_i)$ is the membership point for xi in a fuzzy set.

Fuzzy PID-Controller

Using the Ziegler–Nichols method of tuning PID algorithms in MATLAB, we obtained and implemented values of K_p=2.8, K_i=7.1, and K_d=0.28 for the Simulink model (31). We tested the controller against a unit-step function so its rise time, rise time, settling time, overshoot, and steady-state error could be measured.

Received: June 15, 2024
Accepted: August 25, 2024
Published: January 28, 2025

REFERENCES

1. IPCC. *Summary for Policymakers — Global Warming of 1.5 °C.* Accessed 16 Aug. 2024.
2. Clean Energy Canada. *A Renewables Powerhouse.* Accessed 9 Apr. 2024.
3. Dvorak, Paul. "How Turbulent Winds Abuse Wind Turbine Drivetrains." *Windpower Engineering & Development.* Accessed 9 Apr. 2024
4. "Control Tutorials for MATLAB and Simulink - Introduction: PID Controller Design." *Introduction: PID Controller Design.* Accessed 25 Sept. 2024.
5. Pavithra, V., and V. Jayalakshmi. "Chapter 2 - Smart Energy and Electric Power System: Current Trends and New Intelligent Perspectives and Introduction to AI and Power System." *Smart Energy and Electric Power Systems*, edited by Sanjeevikumar Padmanaban et al., Elsevier, 2023, pp. 19–36.
6. Baburajan, Silpa. *Pitch Control of Wind Turbine through PID, Fuzzy and Adaptive Fuzzy-PID Controllers.*
7. B. S. Mouna and A. Sakly. "Smoothing Wind Power Fluctuations by Particle Swarm Optimization-Based Pitch Angle Controller." *Transactions of the Institute of Measurement and Control*, vol. 41, no. 3, Feb. 2019, pp. 647–56. https://doi.org/10.1177/0142331218764594.
8. Boukhezzar, B., et al. "Multivariable Control Strategy for Variable Speed, Variable Pitch Wind Turbines." *Renewable Energy*, vol. 32, no. 8, July 2007, pp. 1273–87. https://doi.org/10.1016/j.renene.2006.06.010.
9. Civelek, Zafer, et al. "Control of Pitch Angle of Wind Turbine by Fuzzy Pid Controller." I*ntelligent Automation & Soft Computing*, vol. 22, no. 3, July 2016, pp. 463–71.

https://doi.org/10.59720/24-178

https://doi.org/10.1080/10798587.2015.1095417.

10. A. Gambier and Y. Y. Nazaruddin. "Collective Pitch Control with Active Tower Damping of a Wind Turbine by Using a Nonlinear PID Approach." *IFAC-PapersOnLine*, vol. 51, no. 4, Jan. 2018, pp. 238–43. https://doi.org/10.1016/j.ifacol.2018.06.072.
11. A. Gambier and Y. Y. Nazaruddin. "Nonlinear PID Control for Pitch Systems of Large Wind Energy Converters." *2018 IEEE Conference on Control Technology and Applications (CCTA)*, 2018, pp. 996–1001. https://doi.org/10.1109/CCTA.2018.8511531.
12. Habibi, Hamed, et al. "Adaptive PID Control of Wind Turbines for Power Regulation With Unknown Control Direction and Actuator Faults." *IEEE Access*, vol. 6, 2018, pp. 37464–79. https://doi.org/10.1109/ACCESS.2018.2853090.
13. K. Johnson and L. Fingersh. "Adaptive Pitch Control of Variable-Speed Wind Turbines." *45th AIAA Aerospace Sciences Meeting and Exhibit*, American Institute of Aeronautics and Astronautics, 2007. https://doi.org/10.2514/6.2007-1023.
14. Macedo, A. V. A., and W. S. Mota. "Wind Turbine Pitch Angle Control Using Fuzzy Logic." *2012 Sixth IEEE/PES Transmission and Distribution: Latin America Conference and Exposition (T&D-LA)*, IEEE, 2012, pp. 1–6. https://doi.org/10.1109/TDC-LA.2012.6319100.
15. Maroufi, Oussama, et al. "Hybrid Fractional Fuzzy PID Design for MPPT-Pitch Control of Wind Turbine-Based Bat Algorithm." *Electrical Engineering*, vol. 102, no. 4, Dec. 2020, pp. 2149–60. https://doi.org/10.1007/s00202-020-01007-5.
16. Musyafa, Ali. "Implementation of Pitch Angle Control Base Particle Swarm Optimization on Horizontal Axis Wind Turbine at Low Rate Wind Speed." *Advances in Natural and Applied Sciences*, July 2015.
17. Perng, Jau-Woei, et al. "Optimal PID Controller Design Based on PSO-RBFNN for Wind Turbine Systems." *Energies*, vol. 7, no. 1, 1, Jan. 2014, pp. 191–209. https://doi.org/10.3390/en7010191.
18. Qi, Yishuang, and Qingjin Meng. "The Application of Fuzzy PID Control in Pitch Wind Turbine." *Energy Procedia*, vol. 16, Jan. 2012, pp. 1635–41. https://doi.org/10.1016/j.egypro.2012.01.254.
19. Yu, Zhenpeng, et al. "Performance Assessment of PID Control Loops Subject to Setpoint Changes." *Journal of Process Control*, vol. 21, no. 8, Sept. 2011, pp. 1164–71. https://doi.org/10.1016/j.jprocont.2011.06.012.
20. Bauer, Lucas. *Leitwind LTW77 1500 - 1,50 MW - Wind Turbine*. Accessed 5 May 2024.
21. MATLAB. *Simulink*. Accessed 5 May 2024.
22. The MathWorks, Inc. (2022). MATLAB version: 9.13.0 (R2022b). Accessed: Apr 18, 2024
23. *Wind Turbine - MATLAB & Simulink*. Accessed 5 May 2024.
24. S. Abdolzadeh, and S. M. A. Mohammadi. "Optimized Designing of PID Controller for Variable Speed Wind Turbine Systems Using PSO Algorithm." *Ciência e Natura*, vol. 37, Dec. 2015, p. 397. https://doi.org/10.5902/2179460X20870.
25. Wright, A. D., and L. J. Fingersh. *Advanced Control Design for Wind Turbines; Part I: Control Design, Implementation, and Initial Tests*. NREL/TP-500-42437, 927269, 1 Mar. 2008, p. NREL/TP-500-42437, 927269. https://doi.org/10.2172/927269.
26. Ochieng, Reccab M, et al. "Physical Formulation of the Expression of Wind Power." *Advances in Energy Research*, vol. 1, Nova Science Publishers, Inc., 2009, pp. 107–113.
27. Albring, W. "A. Betz, Introduction to the Theory of Flow Machines. XVI + 281 S. m. 168 Fig. Oxford/London/Edinburgh/New York/Toronto/Paris/Braunschweig 1966. Pergamon Pres. Preis Geb. 75 s. Net." *ZAMM - Journal of Applied Mathematics and Mechanics / Zeitschrift Für Angewandte Mathematik Und Mechanik*, vol. 47, no. 2, Jan. 1967, pp. 140–41. https://doi.org/10.1002/zamm.19670470224.
28. Funkjoker23. *English: Doubly Fed Induction Generator (DFIG) Connected to a Wind*. 10 Dec. 2011. Own work, *Wikimedia Commons.*
29. Tejaswi, Y. Bhanu Chandra M. E. Josyula s l. "The Pitch Angle Control of Variable Speed Wind Turbine Using PID Controller." *International Journal of Scientific Research and Management (IJSRM)*, vol. 3, no. 11, 11, Nov. 2015, pp. 3728–33. https://doi.org/ 10.18535/ijsrm/v3i11.12.
30. *Tune PID Controllers - MATLAB*. Accessed 5 May 2024.
31. J. G. Ziegler and N. B. Nichols. "Optimum Settings for Automatic Controllers." *Journal of Dynamic Systems, Measurement, and Control*, vol. 115, no. 2B, June 1993, pp. 220–22. https://doi.org/10.1115/1.2899060.
32. *Fuzzy Logic Toolbox - Simulink*. Accessed 5 May 2024.