Lukas Robinig, BSc

Model based control of a fan-actuated pendulum

Master's Thesis

to achieve the university degree of
Master of Science

submitted to
Graz University of Technology

Supervisor
Assoc.Prof. Dipl.-Ing. Dr.techn. Markus Reichhartinger

Co-Supervisor
Dipl.-Ing. BSc Marijan Palmisano

Institute of Automation and Control
Head: Univ.-Prof. Dipl.-Ing. Dr.techn. Martin Horn

Graz, 2020

Affidavit

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly indicated all material which has been quoted either literally or by content from the sources used. The text document uploaded to TUGRAZonline is identical to the present master's thesis.

Date	Signature

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche kenntlich gemacht habe. Das in TUGRAZonline hochgeladene Textdokument ist mit der vorliegenden Masterarbeit identisch.

Datum	Unterschrift

Acknowledgment

I want to thank Prof. Markus Reichhartinger and Marjian Palmisano for their guidance through this thesis. Furthermore, I want to thank Josef Pucher for the quick adjustments made on the physical model.

A companion through most of my study time was the *Dynamobauzeichensaal*, which gave me a lot of pleasure, and I would like to thank everyone who helped me with advice and support. The *Dynamobauzeichensaal* has always been a safe haven, after returning from my stays abroad, and I have never lost contact with my fellow students.

I would also like to thank my family and friends for their support. And last but not least I would like to give a very special thanks to my better half, who has always supported me. The input from all of you, and those who I forgot, has helped me a lot to achieve my goals.

THANK YOU.

Abstract

The aim of this thesis is to compare different model-based control concepts applied to a fan driven pendulum. The laboratory setup of the pendulum is analyzed and a mathematical model is determined using measurement data from experiments. The mathematical model forms the basis for the used control concepts. A cascaded control system is used, whereby the inner control circuit controls the speed of a Brushless DC-Motor (BLDC) and the outer circuit controls the angle of the pendulum arm. Three nonlinear control concepts namely, the exact linearization, backstepping and adaptive backstepping are investigated. The results show that adaptive backstepping provides the best control behavior. It turns out that the achievable accuracy only can be improved if more accurate measurement devices are used.

Keywords: **model-based control**, **nonlinear control**, **backstepping**

Kurzfassung

Das Ziel dieser Arbeit ist es, verschiedene modellbasierte Regelungskonzepte zu vergleichen und mögliche Verbesserungen des derzeit verwendeten Propeller betriebenen Pendels aufzuzeigen. Der bestehende Laboraufbau wird analysiert und die mathematischen Zusammenhänge werden anhand von Experimenten ermittelt. Das ermittelte mathematische Modell bildet die Grundlage für die verwendeten Regelungskonzepte. In dieser Arbeit wird ein kaskadierter Regelkreis vorgeschlagen. Der innere Regelkreis regelt die Drehzahl, des zur Aktuierung des Pendels verwendeten Brushless DC-Motors (BLDCs) und der äußere Kreis regelt den Winkel des Pendelarms. Für die Winkelregelung werden die drei nichtlinearen Regelkonzepte: exakte Linearisierung, Backstepping und adaptives Backstepping untersucht. Die Ergebnisse zeigen, dass das adaptive Backstepping das beste Regelverhalten erreicht. Um den Toleranzbereich der resultierenden Regelung zu reduzieren, muss die Genauigkeit der Winkel- und Geschwindigkeitsmessung erhöht werden.

Keywords: **modelbasierte Regelung**, **nichtlineare Regelung**, **Backstepping**

Contents

List of Figures

List of Tables

List of Abbreviations

ADC Analog Digital Converter

BEMF Back Electro Magnetic Force

BLDC Brushless DC-Motor

BR Baudrate

CHR Chien, Hrones, Reswick

rt reference tracking

da disturbance attenuation

DAC Digital Analog Converter

ESC Electronic Speed Control

FDP Fan Driven Pendulum

IDE Integrated Development Environment

MCU Microcontroller

PWM Pulse Width Modulation

RPM Revolution per Minute

RPS Revoltion per Second

SISO Single Input Single Output

SP Smith Predictor

TI Texas Instruments

USB Universal Serial Bus

ZN Ziegler Nichols

ZOH Zero Order Hold

List of Symbols

$\mathbf{A}$	Dynamic matrix of the linear system	
$\mathbf{b}$	Control vector	
c_p	Power coefficient	
$\mathbf{C}$	Observation vector	
C(s)	Controller transfer function	
C_0	Control transfer function SP	
$c_{1,2,3}$	System parameters	
d_p	Diameter propeller	m
e	error	
$f(\mathbf{x}, u)$	Function in dependence of $\mathbf{x}$ and u	
F_p	Thrust	N
g	Acceleration of gravity	$\frac{\mathrm{m}}{\mathrm{s}^2}$
$\mathbf{g(x)}$	Function in dependence of x	
$ge(x_1)$	Ground effect	
G(s)	Transfer function	
G_r(s)	Reference transfer function	
h_a	Height of pendulum arm	cm
$h(x)$	Function in dependence of x	
H	Pitch	m
J	Moment of inertia	kgm^2
J	Jacobi matrix	
k	Slope of rpm relation	
k_p	Proportional gain PID-control	
k_i	Integral gain	
$k_{1,2,\cdots,n}$	Control parameters	
k_{BS}	Control parameter backstepping	
k_R	Friction coefficient	
K	Gain PT1-element	
K_{15}	Gain of step response 15% PWM	
K_C	Adjustment variable SP	

l_a	Length of pendulum arm	cm
l_p	Displacement axis to propeller	m
l_s	Centroidal distance	m
$L_f h(x)$	Lie-derivative of h with respect to f	
$L_g h(x)$	Lie-derivative of h with respect to g	
m	Mass of pendulum	kg
n	Speed of propeller	$\frac{1}{s}$
n_{ir}	Measured speed of propeller	$\frac{1}{s}$
n_{tf}	Measured speed phase sensor	$\frac{1}{s}$
P	Plant transfer function	
$\hat{P}$	Copy of plant transfer function	
$\hat{P}_0$	Copy of plant transfer function without delay	
Q(s)	Closed loop transfer function	
r	Radius of the propeller	cm
r(t)	Reference value	
t	Time	s
T	Torque	N m
Tc_n	Sample time of n-control	s
T_d	Delay time	s
$\hat{T}_d$	Estimated delay time	s
ΔT_d	Difference of the delay time	s
ΔT_{dm}	Maximal difference of the delay time	s
T_{dp}	Period of the damped oscillation	s
T_g	Thrust with ground effect	N
T_∞	Thrust with out ground effect	N
T_s	Evaluation time of speed n	s
T_{PT1}	Time constant PT1- element	s
T_{PTn}	Time constant PTn- element	s
T_{T1}	Time constant filter	s
$T_{63\%}$	Estimate of time constant of plant 63%-Method	s
u	Control-, manipulation variable	
v	control law	
V(**x**)	Lyapunov function	
$\bar{u}$	Control variable at operation point	
$\bar{x}$	State variable at operation point	
$x_{1,2,\cdots,n}$	State variables	
y	Output of the controlled system	
z	Distance to ground	m
$z_{1,2,\cdot,n}$	Error variables	

α	Angular acceleration	$\frac{\text{rad}}{\text{s}^2}$
$\beta_{0,\cdots,i}(x_1,\cdots,x_n)$	Function multiplied by u	
Γ	Positive definite matrix	
δ_d	Damping factor	
δ	Relative degree	
λ	Solution of derivation	
φ	Angle	rad
$\varphi_{0,\cdots,i}(x_1,\cdots,x_n)$	Known functions	
$\phi_{1,\cdots,n-1}$	Virtual control	
ω	Angular velocity	$\frac{\text{rad}}{\text{s}}$
ω_0	Angular frequency harmonic oscillation	$\frac{\text{rad}}{\text{s}}$
ω_d	Angular frequency damped oscillation	$\frac{\text{rad}}{\text{s}}$
ω_{0sp}	Angular frequency where $\lvert Q(s)\rvert < \frac{1}{2}$	
ρ	Air density	$\frac{\text{kg}}{\text{m}^3}$
ρ_{20}	Air density by 20°C	$\frac{\text{kg}}{\text{m}^3}$
σ	Step function	
τ_d	Derivation time	s
τ_i	Integration time	s
ϑ	Phase displacement	
Θ	Uncertain system parameters	
$\tilde{\Theta}$	Difference of uncertain system parameter	
$\hat{\Theta}$	Estimated uncertainties	

1. Introduction

The first time probably most of us have heard about the pendulum equation was in secondary school physics class. It is a common topic to explain oscillation processes, from harmonic to damped harmonic oscillation [1, Chap. 15.5, 15.6]. In the field of control engineering, the pendulum is a very popular model for testing control concepts and investigating their performance. Examples are: The inverse pendulum [2] and the propeller-levitation arm model [3].

The aim of this work is to combine the broad knowledge of physics, programming, mathematics and control engineering to provide a functioning propeller-levitated arm model, which is called Fan Driven Pendulum (FDP) in this thesis.

With the FDP, different linear and nonlinear control concepts and their performance can be evaluated. Examples for possible control concepts are: PID-control, exact linearisation, backstepping, sliding mode control [4], etc. In addition, this thesis offers the opportunity to bridge the gap between computer simulation and a *real-life* problem with the associated challenges. This can help students to motivate themselves for more complex topics in the field of nonlinear control engineering and provides a platform to test what they have learned [5]. A further motivation is to provide a robust *hands-on* model for students at the *Institute of Automation and Control* at TU-Graz.

This work is divided into five sections. The first section gives a brief overview of the thesis. In the second part, the mechanical structure, the physically occurring effects and the resulting mathematical model are presented. The third chapter discusses the methods used and the necessary theory. In the fourth part, the simulation and the implementation of the control on the real model are examined. At last, the results of this thesis are discussed.

2. System Description

In this chapter, all facts relevant to the model are presented. These are: the mechanical structure, the used components, the occurring physical effects and the resulting mathematical system description.

2.1. Model setup

The mechanical model of the FDP is provided by the *Institute of Automation and Control*. The scheme of the FDP is shown in Figure 2.1 .

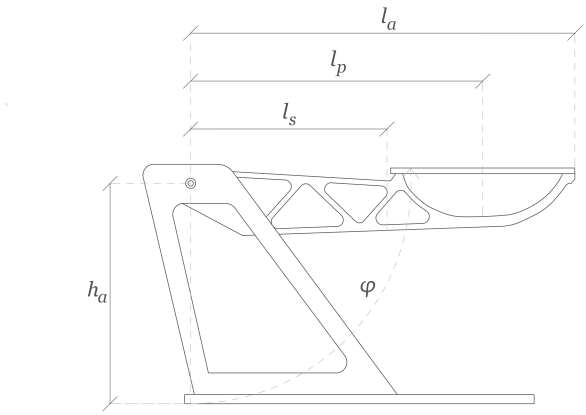

Figure 2.1.: Scheme of the FDP. All distances needed for the calculation of the model parameters are provided in this scheme. h_a is the distance to the ground, l_s is the distance to the center of gravity, l_p is the distance from the pivot point to the mounted motor and l_a the length of the pendulum arm.

All necessary distances for the system description are shown in this figure. The height from the ground to the pivot point is h_a. The distance l_s represents the distance from the pivot point to the center of gravity of the arm. l_a is the length of the pendulum arm. The BLDC motor is mounted at the distance l_p to the pivot point. This motor has an APC propeller connected to its shaft. The thrust necessary to lift the pendulum arm is generated by the rotating propeller. The BLDC motor needs an AC voltage to spin. The MCU provides the ESC with the necessary Pulse Width Modulation (PWM) signal to generate the AC voltage. The MCU used for measuring and control purposes is a 32 bit 100MHz AMR Cortex-M3 micro controller from NXP.

A hall sensor is used to measure the angle. Due to the mechanical construction, a direct measurement is not possible, therefore an indirect measurement with a gear factor of four is used. The speed of the rotor is measured with a phase sensor and an infrared sensor. Since the infrared sensor has a limited measuring range, high speeds cannot be measured and therefore it is not used for control. The relation between the actual speed of the rotor and the speed measured by the phase sensor is presented in Section 2.3.

The only mechanical changes realized during the elaboration of the present thesis were: the replacement of the bearing from the pivot point of the model, and the extension of the range of the angular limitations. The range of the moving arm goes from 67° to 150°.

Conversely to the few mechanical changes, the provided C++ code needed a complete makeover. The coding for the FDP software is done in an online development environment from Mbed. Mbed is a rapid prototyping platform for embedded systems, which provides necessary libraries for the used components. The maximum speed of the serial connection is limited to a Baudrate (BR) of 115200 bauds due to the imprecise oscillator of the MCU. This speed is sufficient for the exchange of the measured data.

Detailed information regarding the used components can be found in Appendix A.1.

2.2. Mathematical Model

As starting point for the mathematical model, the pendulum equation from [6, P. 7] is taking under consideration.

$$\begin{aligned} \dot{x_1} &= x_2 \\ \dot{x_2} &= -\frac{g}{l_s}\sin(x_1) - \frac{k_R}{m}x_2 + \frac{1}{m \cdot l_s^2}T \end{aligned} \tag{2.1}$$

In Equation 2.1, x_1 is the angle φ, x_2 is the angular velocity ω and $\dot{x_2}$ is the angular acceleration α. The variable g is the gravitational acceleration. The mass of the pendulum is represented with the variable m, k_R is the friction coefficient and T represents the torque provided by the propeller.

The moment of inertia is defined in Equation 2.2

$$J = m \cdot l_s^2 \tag{2.2}$$

The torque T from Equation 2.1 is generated by the rotating propeller. The following relations for APC-Propeller were determined by experiments, as described in [7].

$$c_p = 0.0856 \cdot \frac{H}{d_p} - 0.0091 \tag{2.3}$$

In Equation 2.3, c_p is the resulting power factor of the propeller, H is the pitch of the propeller and d_p is the diameter of the propeller.

$$P_m = c_p \cdot \rho \cdot \frac{n}{60}^3 \cdot d_p^5 \tag{2.4}$$

The mechanical power P_m is calculated in Equation 2.4, where c_p is the power factor, ρ is the air density, n is the speed of the propeller in $\frac{1}{min}$ and

d_p is the diameter.

$$\begin{aligned} F_p &= 0.67 \cdot \sqrt[3]{\frac{\rho}{2} \cdot \pi \cdot d_p^2 \cdot P_m^2} \\ &= 0.67 \cdot \rho \sqrt[3]{\frac{\pi}{2} c_p^2} (\frac{n}{60})^2 \cdot d_p^4 \end{aligned} \tag{2.5}$$

The thrust F_p, generated by the propeller, is calculated with Equation 2.5.

The generated thrust F_p is proportional to the square of the speed and to the fourth power of the radius of the propeller.

- $F_p \propto n^2$
- $F_p \propto r^4$

The mechanical power needed to turn the propeller, is proportional to the third power of the speed and the fifth power of the radius of the propeller.

- $P_m \propto n^3$
- $P_m \propto r^5$

Inserting F_p from Equation 2.5 for the thrust T in Equation 2.1 yields Equation 2.6.

$$\begin{aligned} \dot{\varphi} &= \omega \\ \dot{\omega} &= -\frac{g}{l_s} \sin(\varphi) - \frac{k_R}{m} \omega + \frac{1}{m \cdot l_s^2} \cdot 0.67 \cdot \rho \sqrt[3]{\frac{\pi}{2} c_p^2} n^2 \cdot d_p^4 \cdot l_p \end{aligned} \tag{2.6}$$

Another physical effect that occurs due to the rotation of the propeller has to be added to Equation 2.6. The so called ground effect, also referred to as hovering effect in the literature, is increasing the thrust of the propeller in the area close to the ground (see [8]). The thrust is increased by an air cushion generated by the rotation of the propeller. The ground effect is described by Equation 2.7.

$$\frac{T_g}{T_\infty} = \frac{1}{1 - \frac{r^2}{16 \cdot z^2}} \tag{2.7}$$

The variables used in the previous equation are the thrust T_g, which represents the thrust increased by the ground effect; the thrust T_∞, which is the generated thrust by the propeller without the ground effect; and z which is the distance to the ground.

The current distance z to the ground of the FDP depends on the angle φ and is described in Equation 2.8.

$$z = h_a - l_a \cdot \cos(\varphi) \tag{2.8}$$

$$ge(\varphi) = \frac{16 \cdot (h_a - l_a \cdot \cos(\varphi))^2}{16 \cdot (h_a - l_a \cdot \cos(\varphi))^2 - r^2} \tag{2.9}$$

The ground effect ge(φ) of the model can be seen in Equation 2.9. Furthermore, the effect in respect to the elevation of the arm can be seen in Figure 2.2. The effect has a big impact on the thrust near the ground. If the distance to the ground is smaller than the radius of the propeller, the thrust increases by up to 28%. The impact of the ground effect decreases with increasing distance to the ground. At the point where z reaches twice the diameter of the rotor, the increase of the thrust is below 0.5% and can be neglected.

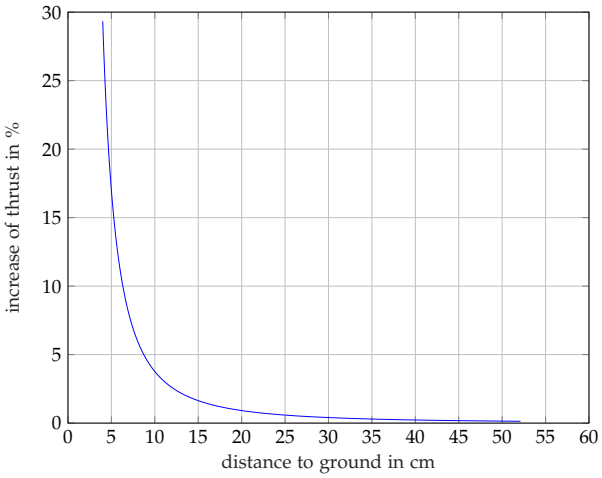

Figure 2.2.: The Ground effect of the FDP with respect to the distance between the propeller and the ground.

The ground effect is taken into account by multiplying the thrust F_p from Equation 2.5 with ge(φ) (Equation 2.9). Inserting the resulting thrust for the thrust T in Equation 2.1 yields

$$\begin{aligned}\dot{\varphi} &= \omega \\ \dot{\omega} &= -\frac{g}{l_s}\sin(\varphi) - \frac{k_R}{m}\omega + \frac{16\cdot(h_a - l_a\cdot\cos(\varphi))^2}{16\cdot(h_a - l_a\cdot\cos(\varphi))^2 - r^2}\cdot\frac{1}{m\cdot l_s^2}\cdot 0.67\cdot\rho\sqrt[3]{\frac{\pi}{2}c_p^2}n^2\cdot d_p^4\cdot l_p\end{aligned} \tag{2.10}$$

which can be written as

$$\begin{aligned}\dot{x_1} &= x_2 \\ \dot{x_2} &= -c_1\cdot\sin(x_1) - c_2\cdot x_2 + ge(x_1)\cdot c_3\cdot u\end{aligned} \tag{2.11}$$

The manipulation variable u = n^2. The constants c_1, c_2 and c_3 can be found in Equation 2.12.

$$\begin{aligned}c_1 &= \frac{g}{l_s} \\ c_2 &= \frac{k_R}{m} \\ c_3 &= \frac{1}{m\cdot l_s^2}\cdot 0.67\cdot\rho\cdot d_p^4\sqrt[3]{\frac{\pi}{2}c_p^2}\cdot l_p\end{aligned} \tag{2.12}$$

2.3. Relation between the real speed and the measured speed of the propeller

Since the built-in infrared sensor can only measure low speeds and the phase sensor can not measure the real speed but the speed of the rotating field, the relationship between the measured speed of the rotating field and the speed of the rotor is investigated in this section. This relation is determined experimentally. For this purpose, the measurement of the infrared sensor was verified with an external tachometer to be sure to measure the real speed of the propeller. Due to the two bladed rotor, two pulses are measured

per revolution by the infrared sensor. The phase sensor measures six pulses per revolution due to the six poles of the BLDC.

The speed relevant for controlling the angle φ is in the range from 160 to 175 Revoltion per Second (RPS).

The MCU is connected to Matlab via a serial bidirectional connection. The BR for the serial connection is set to 115200 bauds and the parity bit is odd. The value of the PWM-signal used to control the BLDC is sent from Matlab to the MCU. The acquired data is the measured speed of the infrared sensor and the phase sensor. Furthermore, the PWM-signal and the time are added to the measured data and are sent to the PC.

For this experiment, the speed of the propeller is calculated every second by adding up the recorded pulses and dividing them by the number of pulses per revolution, depending on the sensor.

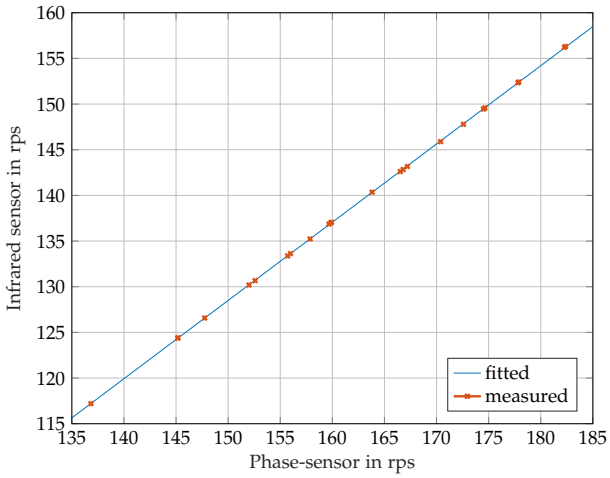

Figure 2.3.: Relation of the real speed of the propeller and the measured speed. The fitted curve displayed in blue and the measured points are given in red.

This experiment shows a linear relationship (Figure 2.3) between the real speed of the propeller and the measured speed of the phase sensor. The slope k and the offset d of the linear relation was computed from the measured data in Figure 2.3. It turned out that the offset d can be neglected due to its low value of d= -0.016RPS because the accuracy of speed measurement by

the phase sensor is ± 0.166RPS when calculated every second. The speed of the propeller n_{ir} can be calculated from the measured speed of the rotating field n_{tf} by:

$$\begin{aligned} n_{ir} &= k \cdot n_{tf} \\ n_{ir} &= 0.8568 \cdot n_{tf} \end{aligned} \tag{2.13}$$

2.4. Transfer function of BLDC-Motor and ESC

The open loop method was used to determine the behavior of the BLDC combined with the ESC. The input of the system is a PWM-signal and the output is the speed of the propeller. The connection between the MCU and the PC was established with a Matlab script. A PWM value of 15% is sent to the MCU, the BR is set to 115200 bauds and the parity bit is odd. 300 samples were measured for each step response, which correspond to a time of approximately 3.5s. This process was repeated 20 times. The mean of the measured step responses is calculated and is used for the further evaluation.

From the measured data, it can be seen that the transfer function has a PT1 behavior with a time delay.

This behavior is described in Equation 2.14 where G(s) is the transfer function, K is the gain, T_{PT1} is the time constant and T_d is the time delay of the system.

$$G(s) = \frac{y(s)}{u(s)} = \frac{K}{1 + s \cdot T_{PT1}} \cdot e^{-s \cdot T_d} \tag{2.14}$$

The impulse response g(t) in the time domain is given by Equation 2.15.

$$g(t) = K \cdot e^{-\frac{t-T_d}{T_{PT1}}} \cdot \delta(t - T_d) \tag{2.15}$$

The corresponding step response n(t) is shown in Equation 2.16.

$$n(t) = K \cdot (1 - e^{-\frac{t-T_d}{T_{PT1}}}) \cdot \sigma(t - T_d) \tag{2.16}$$

The 63%-rule presented in [9] was used to evaluate the parameters K and T_{PT1} of the transfer function.

The results of the evaluation can be seen in Figure 2.4. The parameter

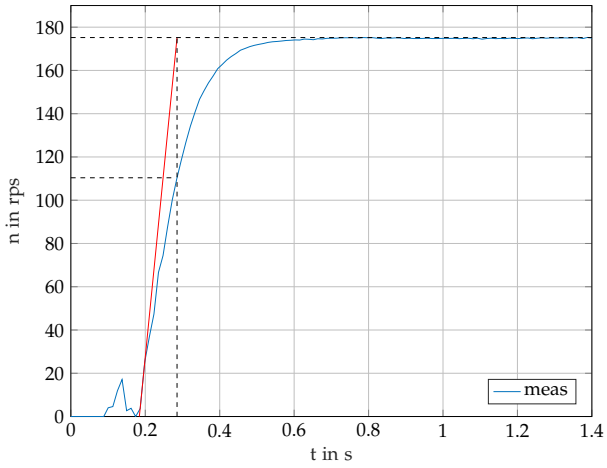

Figure 2.4.: The open loop method is used to determine the System parameters of the BLDC with the ESC. The mean of the 20 measured step responses is the blue curve. The red straight line is used to determine the time constant with the 63% method.
The resulting system parameters are: $T_d = 184.1ms$, $K_{15} = 175.2$ and $T_{15} = 101.3ms$

$K_{15} = 175.2$ must be converted due to the fact that no unit step has been used to obtain the step response.

The conversion can be seen in Equation 2.17.

$$K = \frac{K_{15}}{15} \cdot 100 = 1168 \tag{2.17}$$

In Figure 2.5, the mean of the measured signals and the step response can be seen.

The ripple of the measured signal, before the curve starts to rise, can be explained by measurement uncertainties due to the fast evaluation of the speed of the phase sensor (10ms).

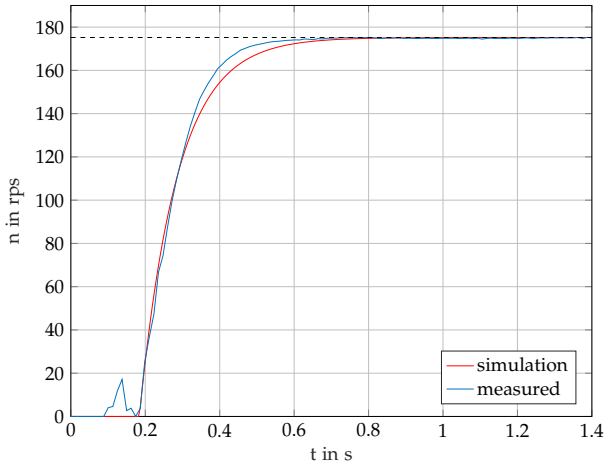

Figure 2.5.: The mean of the 20 measured step responses is displayed in the blue curve. The red curve is the step response of the transfer function of the BLDC by 15% PWM.

The transfer function which is used for the simulation of the speed controller can be seen in Equation 2.18.

$$G(s) = \frac{1168}{1 + s \cdot 0.1013} \cdot e^{-s \cdot 0.1841} \tag{2.18}$$

An unexpected result of that experiment is that the BLDC and ESC have a varying time delay T_d. The range of the time delay is 180ms to 245 ms. Most time delays occurred in the range of 180 ms to 191.5ms. This is also reflected in the calculated mean of the time delay T_d= 184.1ms.

2.5. Model verification

The starting point for the evaluation is the assumption that the pendulum arm is in balance. Balance means that the angle of the arm is constant which yields to the following relation (see Equation 2.19).

$$x_1 = const \Rightarrow x_2 = 0, \dot{x_2} = 0 \tag{2.19}$$

The following relation is obtained by combining Equation 2.19 with Equation 2.11:

$$u = \frac{c_1}{c_3 \cdot ge(x_1)} \cdot \sin(x_1) \tag{2.20}$$

The mathematical model is verified with an experiment, where a Smith Predictor (SP) is implemented on the MCU in order to control the speed of the propeller. Due to the fact that the transfer function of the BLDC is of first order, the used controller in the SP is a PI-controller (see Section 3.3). The parameters of the controller are:

$$k_i = 0.00265$$
$$k_p = 0.00072$$

The range of the reference values of the speeds is between 160 RPS and 176 RPS. Furthermore, two points with an angle greater than 90 degrees are recorded. For this experiment, the evaluation time T_s of the speed is set to 170ms and the sample time Tc_n of the controller is 5ms. The evaluation time T_s is the time used to determine the speed. The recorded pulses from the sensor are divided by the number of pulses per revolution and the evaluation time to obtain the speed.

$$n = \frac{\text{pulses}}{6 \cdot T_s} \tag{2.21}$$

The connection between the MCU and the PC was established with a Matlab script. The reference value of n is sent to the MCU, the BR is set to 115200 bauds and the parity bit is odd. Three runs of this experiment were executed. For each run, 600 samples per reference value are recorded.

In this experiment, no perfect resting position is found. For the evaluation, the points with constant speed are determined and the corresponding angles are summed up and averaged. These values are used to approximate the

function $n(\varphi)$ by a polynomial of third order using the Matlab command *polyfit*.

During the verification process it was found that the mathematical model (Equation 2.10) had to be adapted to reality. It was found that the transition where the force must decrease to keep the arm in balance is 105 degrees instead of 90 degrees as expected. For this shift of the peak of the sine to the right side, $\vartheta = 15°$ is introduced. This behavior is probably due to the geometry of the pendulum arm used. In addition, the calculated power factor c_p of the propeller was adjusted until the approximated curve and the mathematical model overlapped. c_p was changed from the calculated value $c_p = 0.0694$ to a value of $c_p = 0.0498$. The adapted model is given by:

$$\begin{aligned} \dot{x_1} &= x_2 \\ \dot{x_2} &= -c_1 \cdot \sin(x_1 - \vartheta) - c_2 \cdot x_2 + ge(x_1) \cdot c_3 \cdot u \end{aligned} \tag{2.22}$$

The resulting relation between the angle and the speed of the propeller is shown in Figure 2.6.

To illustrate the influence of the ground effect, the adjusted model was plotted with and without the ground effect. This influence can be seen in the range from 70° to 100° by the occurring gap between the blue and red line. The yellow curve is the approximated model obtained from the measured data. The red curve is the adjusted mathematical system model to fit to the approximated curve including the ground effect. The blue line is the adjusted mathematical system without the ground effect. The black dots are the measured data and the green dots are the measured data which are used for the approximation. The two black dots were not used for the approximation because the pendulum arm is not moving free in this area.

2.6. Determination of the friction coefficient k_R

Starting point for the determination of the friction coefficient is the following second order differential equation (see [10, P. 59-61]).

$$J \cdot \dot{\dot{\varphi}} + k_R \cdot \dot{\varphi} \cdot l_s^2 + m \cdot g \cdot \sin(\varphi) \cdot l_s = 0 \tag{2.23}$$

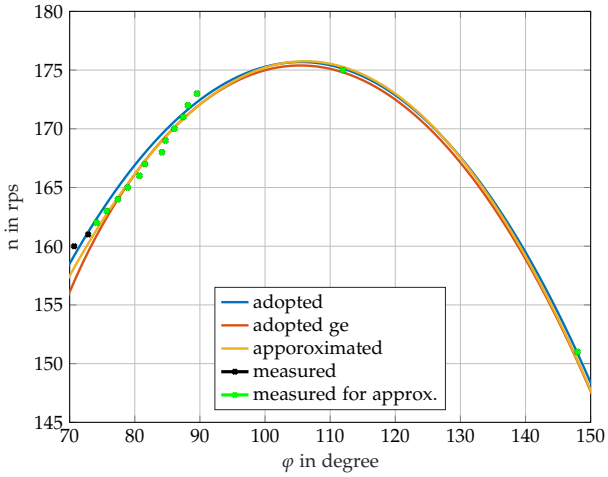

Figure 2.6.: Approximated model of the FDP. The yellow curve is the approximated model obtained from the measured data. The red curve is the adjusted mathematical system model to fit to the approximated curve including the ground effect. The blue line is the adjusted mathematical system without the ground effect. The black dots are the measured data and the green dots are the measured data which are used for the approximation.

Equation 2.23 can be simplified to Equation 2.24 by using the approximation for small angles that $\sin(\varphi) \approx \varphi$ and by dividing Equation 2.23 by the moment of inertia. The simplification for small angles holds for angles smaller than 10°.

$$\begin{aligned} \ddot{\varphi} + \frac{k_R}{m} \cdot \dot{\varphi} + \frac{g}{l_s} \cdot \varphi = 0 \\ \ddot{\varphi} + 2\delta_d \cdot \dot{\varphi} + \omega_0^2 \cdot \varphi = 0 \end{aligned} \tag{2.24}$$

The differential Equation 2.24 is solved using the ansatz $\varphi = A \cdot e^{\lambda \cdot t}$. The resulting values for λ for the decay process $\delta_d^2 < \omega_0^2$ are given in equation 2.25.

$$\begin{aligned} \lambda_{1,2} &= -\delta_d \pm i \cdot \sqrt{\omega_0^2 - \delta_d^2} \\ &= -\delta_d \pm i \cdot \omega_d \end{aligned} \tag{2.25}$$

The damped angular frequency ω_d can be determined from the decay process of the pendulum.

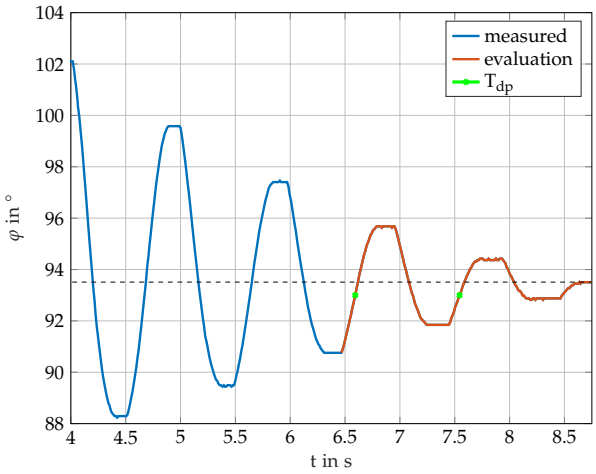

Figure 2.7.: The damped harmonic oscillation of the FDP. The blue curve is the measured data from the decay process; the red curve is the part which was used for the evaluation of the damped angular frequency ω_d; the period time T_{dp} is measured between the two green points.

In Figure 2.7, the damped harmonic oscillation of the FDP is shown. The blue curve is the measured data from the decay process; the red curve is the part which was used for the evaluation of the damped angular frequency ω_d; the period time T_{dp} is measured between the two green points and the resulting period time of the damped oscillation is $T_{dp} = 0.9514s$.

$$2 \cdot \delta_d = \frac{k_R}{m} \tag{2.26}$$

$$\omega_0^2 = \frac{g}{l_s} \tag{2.27}$$

$$\omega_d = \sqrt{\omega_0^2 - \delta_d^2} \tag{2.28}$$

$$\omega_d = \frac{2 \cdot \pi}{T_{dp}} \tag{2.29}$$

The damped angular frequencies ω_d and the harmonic angular frequency ω_0 can be calculated with equations 2.27 and 2.29. By transforming the Equation

Symbol	Value	Unit
ω_d	6.604	$\frac{\text{rad}}{\text{s}}$
ω_0^2	50.308	$\frac{\text{rad}}{\text{s}}$
δ_d	2.587	
k_R	1.216	

Table 2.1.: Resulting values of the friction coefficient calculation.

2.28, the damping coefficient δ_d can also be calculated. The resulting friction coefficient k_R can be computed by transforming Equation 2.26.

The results are shown in Table 2.1.

With the parameter k_R, the constants: c_1, c_2 and c_3 in 2.22 can be calculated (Equation 2.12). The parameters used for the calculations of the constants can be found in Appendix A.1.

The resulting values of the constants can be seen in 2.30.

$$\begin{aligned} c_1 &= 50.3077 \\ c_2 &= 5.1745 \\ c_3 &= 0.002207 \end{aligned} \tag{2.30}$$

2.7. Limitations

Due to the mechanical design and the components used, several limitations have to be taken into account when designing the control of the propeller speed and the controller of the angel:

1. The transmission speed is limited to 115200 BR via the Universal Serial Bus (USB), due to imprecision of internal clock of the MCU.
2. The pendulum can be moved in the angular range from 67° to 150° (Figure 2.1).
3. The PWM signal is limited to a range of 1% to 20% due to the limitation of the current by the power supply unit.
4. The accuracy of the phase sensor depends on the evaluation speed of the recorded pulses. Slower speed leads to a more precise measurement and vice versa.
5. The backlash of the cogs lead to an inaccuracy of the angular measurement of 1.3° every time that the pendulum changes the moving direction.
6. The varying time delay T_d, which occurs due to the used components.

3. Theory and Methods

In this chapter, the different control concepts and their application to the FDP are discussed.

3.1. PID-Control

The PID-Controller is a common controller in industrial applications [11, P. 257]. It consists of:

- A proportional gain, which responds immediately to an error e.
- An integral part, which responds to the integral of the error e.
- A differential part which responds to de derivative of the error e.

The general form of the PID-control in the time domain is given by Equation 3.1.

$$u(t) = k_p \left(1 + \frac{1}{\tau_i} \int_0^t e(\tau) d\tau + \tau_d \cdot \frac{de(t)}{dt}\right) \tag{3.1}$$

In this equation: u(t) is the control variable; e(t) is the error; k_p is the proportional gain; τ_i is the integration time and τ_d the derivation time. By transforming Equation 3.1 with the Laplace transformation, it turns out that this equation is not realizable due to the fact that the denominator has a lower order than the numerator. By adding a T1-element to the differential part, the realizable PID-control equation is obtained. The transfer function of the realizable PID-controller C(s) is shown in Equation 3.2.

$$C(s) = \frac{u(s)}{e(s)} = k_p \left(1 + \frac{1}{\tau_i \cdot s} + \frac{s \cdot \tau_d}{1 + s \cdot T_{T1}}\right) \tag{3.2}$$

The time constant T_{T1} has to be chosen such that the DT1-element has D behavior in the relevant frequency range. Setting $\tau_d = 0$ yields the PI-control

$$C(s) = k_p + \frac{k_i}{s} \tag{3.3}$$

where $k_i = \frac{k_p}{\tau_i}$.

For stable systems, experiments can be performed to acquire plant parameters which are needed to determine the control parameters k_p, τ_i, τ_d.

Such experiments are:

- The open loop method, also known as step response method;
- The closed loop method

For the open loop method, a step is applied as input of to the system and the system response is measured. From the measured response, the system behavior and the system parameters can be determined. The gain K of the plant is defined as the steady state of the response. Several methods for determining the rise time T_{PTn} are known in the literature, two of them are mentioned here:

- The method of the inflection point tangent;
- The 63%-method

The method of the inflicting point tangent has a lower accuracy than the 63% method. For the 63%-method, T_{PTn} is the time required to reach 63% of the final value K.

This method was used for the determination of the transfer function of the BLDC in Section 2.4.

For the closed loop method, the controller of the plant is a proportional gain. This gain has to be increased until the system starts to oscillate. From this measured response, the critical gain k_{crit} and the critical period time T_{crit} of the oscillation can be obtained.

For the calculation of the control parameters, there are many existing rules. In appendix B.1, the methods to calculate the parameters with Ziegler Nichols and Chien, Hrones, Reswick (CHR) are displayed. The calculated

parameters usually have to be adjusted in order to obtain the desired control behavior.

3.2. Linearization in the vicinity of the operation point

For controlling nonlinear systems with linear control concepts, the system needs to be linearized around the point of operation.

The nonlinear system is given by Equation 3.4.

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{f}(\mathbf{x}, u) \\ y &= h(\mathbf{x}) \end{aligned} \tag{3.4}$$

The bold symbols are multidimensional variables and functions.

$$\mathbf{J}_{\mathbf{x}=\bar{\mathbf{x}},u=\bar{u}} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \vdots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix}_{\mathbf{x}=\bar{\mathbf{x}},u=\bar{u}} = \mathbf{A} \tag{3.5}$$

By the calculation of the Jacobian matrix $\mathbf{J}$ (Equation: 3.5) and entering the values $\bar{\mathbf{x}}$ and $\bar{u}$ of the operation point, the system matrix $\mathbf{A}$ of the linear system is obtained. Vector $\mathbf{b}$ and vector $\mathbf{C}$ are calculated by partial differential equations (Equation 3.6).

$$\mathbf{b} = \begin{bmatrix} \frac{\partial f_1}{\partial u} \\ \vdots \\ \frac{\partial f_n}{\partial u} \end{bmatrix}_{\mathbf{x}=\bar{\mathbf{x}},u=\bar{u}} \qquad \mathbf{C} = \begin{bmatrix} \frac{\partial h}{\partial x_1} & \cdots & \frac{\partial h}{\partial x_n} \end{bmatrix}_{\mathbf{x}=\bar{\mathbf{x}},u=\bar{u}} \tag{3.6}$$

The result is the linear system described in Equation 3.7.

$$\begin{aligned} \Delta\dot{\mathbf{x}} &\approx \mathbf{A}\Delta\mathbf{x} + \mathbf{b}\Delta u \qquad \Delta\mathbf{x}(t) = \mathbf{x}(t) - \bar{\mathbf{x}} \\ \Delta y &\approx \mathbf{C}\Delta\mathbf{x} \end{aligned} \tag{3.7}$$

Design methods for linear systems can now be applied to the linearized model.

3.3. Smith-Predictor

The information in this section was mainly gathered from [12, P.224–237.]. The SP is a powerful method to increase the performance of the closed-loop control with occurring time delay. In Figure 3.1, the structure of the SP can be seen.

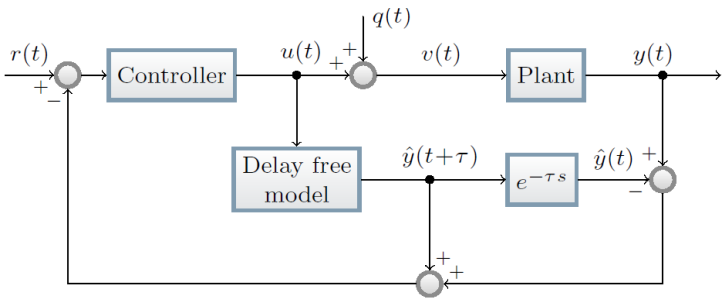

Figure 3.1.: The structure of the Smith Predictor (Source: [13, P. 1270]). For the ideal case that the plant and the delay free model are exactly the same, the closed control loop behaves like the delay free model.

The closed-loop transfer function $G_r(s)$ from r to y is given by:

$$G_r(s) = \frac{y}{r} = \frac{C_0 \cdot P}{1 + C_0 \cdot \hat{P}_0 + C_0 \cdot P - C_0 \cdot \hat{P}} \tag{3.8}$$

With the transfer functions: P(s) as the real plant; $\hat{P}_0(s)$ is the copy of plant P(s) without time delay; $\hat{P}(s)$ is the copy of the model with time delay and $C_0(s)$ is the controller. Q(s) is defined as the closed loop transfer function $G_r(s)$ for the ideal case, divided by the occurring time delay $e^{\tau s}$. The ideal case is given if $P(s) = \hat{P}(s)$.

$$Q(s) = \frac{C_0 \cdot \hat{P}_0}{1 + C_0 \cdot \hat{P}_0} \tag{3.9}$$

In the case that the system model is known and the uncertainties occur only in the time delay, the following relations apply. Q(s) is practically stable if:

$$lim_{\omega\to\infty}|Q(j\omega)| < \frac{1}{2}, \tag{3.10}$$

is fulfilled. If Q(s) is strictly proper the system is practically stable. Practically stable means that the desired system can oscillate within an acceptable range near the point of operation.

Now the uncertainty in the time delay T_d is introduced.

$$\Delta T_d = T_d - \hat{T}_d \tag{3.11}$$

$\hat{T}_d$ is the used lag time in the SP.

The closed loop system is asymptotically stable for any ΔT_d if:

$$|Q(j\omega)| < \frac{1}{2} \qquad \forall \qquad \omega \geq 0, \tag{3.12}$$

is satisfied. The closed loop is asymptotically stable for $|\Delta T_d| < \Delta T_{dm}$ if :

$$|Q(j\omega)| \leq 1 \quad \forall \quad \omega \geq \qquad \& \qquad lim_{\omega\to\infty}|Q(j\omega)| < \frac{1}{2}, \tag{3.13}$$

with the rough estimate of ΔT_{dm} is given by:

$$\Delta T_{dm} = \frac{\pi}{3\omega_{0sp}} \tag{3.14}$$

where ω_{0sp} is a frequency above which $|Q(j\omega)| < \frac{1}{2}$ is satisfied.

The *classical* SP can just be used for stable systems. If the following four points are fulfilled a zero stead-state error for reference steps and for step disturbances is displayed:

1. The plant is asymptotically stable;
2. The plant does not contain a pure differentiator;
3. The primary controller includes an integrator;

4. The closed-loop transfer function is asymptotically stable (Equations 3.12, 3.13) and practically stable (Equation 3.10).

The SP is not in general robust to parameter and time delay uncertainties, but if the closed-loop stability of Q(s) is still maintained despite occurring uncertainties and the other three points are fulfilled, a zero steady-state error is displayed.

Massive improvements of reference tracking can be achieved with a SP. On the other hand, it is less effective in disturbance attenuation.

For first and second order plants, there are some guidelines on how to set the control parameters. Due to the fact that the transfer function of the BLDC (Equation 2.14) is of first order, the guide lines for first order systems will be mentioned in this section. If all poles and zeros of the system are in the left half plane, the following structure of $C_0(s)$ can be used:

$$C_0(s) = \frac{K_c}{s} \cdot \hat{P_0}^{-1} \tag{3.15}$$

For a system description of a first order filter (PT1-element), the controller C_0 is a classical PI-Controller (see Equation 3.3). The control parameters of k_p and τ_i depend on the parameters of the plant and are given in the following equation.

$$k_p = \frac{K_c \cdot T_{PT1}}{K} \qquad \& \qquad \tau_i = T_{PT1} \tag{3.16}$$

If the parameters are chosen like in Equation 3.16 suggested, the close-loop transfer function leads to:

$$Q(s) = \frac{K_c}{s + K_c} \tag{3.17}$$

To provide stability of Q(s), the variable K_c needs to be positive for the ideal case. Due to model uncertainties and the relative error in the time delay, there is an upper limit given for K_c (see [12, P. 231]).

A tuning rule for K_c is set to:

$$K_c = \frac{3}{\hat{T}_d} \tag{3.18}$$

An interesting fact is that, $K_c < 3$ assures stability for a $\pm 100\%$ mismatch of the time delay.

Another approach to calculate the parameters for the PI control is from Hägglund [14].

$$k_p = \frac{1}{K} \qquad \& \qquad \tau_i = T_{63\%} \tag{3.19}$$

With $T_{63\%}$, the time constant of the open loop step response is evaluated with the 63% method and K is the constant gain of the step response of the system.

If the time delay is bigger than the time constant of the plant, the smith predictor outperforms the classical PID-control.

A result of the use of the smith predictor is that higher values for the control parameters can be used in comparison to PID- or PI-control without SP.

3.4. Exact linearization

The references for this section are [15, P.287-293] and [6, P. 505-521].

In this section, a Single Input Single Output (SISO)-Systems, like in Equation 3.20, is considered.

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{f(x)} + \mathbf{g(x)} \cdot u \\ y &= h(\mathbf{x}) \end{aligned} \tag{3.20}$$

The bold symbols are multidimensional variables and functions. If the system is given by the nonlinear controllable canonical form, it is a flat system. A flat system means that all inputs and state variables can be

written as a function of the output and its derivatives. The system (Equation 3.20) is flat if the relative degree δ of the output y is equal to the system order n. Therefore we introduce the Lie derivative:

$$\begin{aligned} \dot{y} =& \frac{\partial h(x)}{\partial x}(f(x) + g(x)u) \\ \dot{y} =& L_f h(x) + L_g h(x) \cdot u \end{aligned} \tag{3.21}$$

Where:

$$L_f h(x) = \frac{\partial h(x)}{\partial x} f(x) \tag{3.22}$$

is the Lie-derivative of h with respect to f. For higher order derivatives of y the following relation for the Lie derivatives is given.

$$\begin{aligned} L_g L_f h(x) =& \frac{\partial L_f h(x)}{\partial x} g(x) \\ L_f^2 h(x) =& L_f L_f h(x) = \frac{\partial L_f h(x)}{\partial x} h(x) \end{aligned} \tag{3.23}$$

To determine the relative degree, the output y is repeatedly derived with respect to the time. The relative degree δ is reached when, the manipulated variable u can be expressed as a function of the output y and its derivatives for the first time.

$$y^{(\delta)} = L_f^{\delta} h(x) + L_g L_f^{\delta - 1} h(x) \cdot u \tag{3.24}$$

If the relative degree δ=n, where n is the system order, the system is flat. Flat systems are controllable and the exact linearization can be used. Furthermore, flat systems can be written in the strict feedback form. In the case that $\delta < n$ the system can be written in the Byrnes-Isidori canonical form with an external and an internal dynamic of the system. The internal dynamic has no influence on the output y, but if it is not stable it causes instability for the whole system.

The goal of the exact linearization is to eliminate the nonlinear parts of the system with the control variable u and to use a linear control law v

to stabilize the corresponding linear system (Equation 3.25). The resulting linear system has the system order δ.

$$\begin{aligned} \dot{\xi} =& \mathbf{A}\xi + \mathbf{b}v \\ y =& C\xi \end{aligned} \tag{3.25}$$

where $\xi \in \mathbb{R}^{\delta}$ is the state vector of the external dynamics. It consists of the states $\xi_1 = y$, $\xi_2 = \dot{y}$, ..., $\xi_\delta = y^{(\delta-1)}$.

By eliminating the nonlinear terms with the control variable u, the resulting linear system (Equation 3.25) is given in the Brunovsky-canonical form. To eliminate the nonlinearities, Equation 3.24 equals to v and u yields

$$u = \frac{1}{L_g L_f^{\delta-1} h(x)} \left(-L_f^{\delta} h(x) + v \right) \tag{3.26}$$

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \\ 0 & 0 & 0 & \cdots & 0 \end{bmatrix}, \mathbf{b} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix} \tag{3.27}$$

The matrix (**A**-**b** **K**) needs to be a Hurwitz matrix. **K** is a row vector with the control parameters $(k_1, k_2, \cdots, k_\delta)$. For the first and second order case, this is satisfied if the control parameters (k_1, k_2) are positive.

3.4.1. Exact linearization FDP

To determine if the pendulum system description is a flat system, the relative degree of the system needs to be calculated.

The mathematical system is given by the Equation 3.28

$$\begin{aligned} \dot{x_1} &= x_2 \\ \dot{x_2} &= -c_1 \cdot \sin(x_1 - \vartheta) - c_2 \cdot x_2 - c_3 \cdot ge(x_1) \cdot u \\ y &= x_1 \end{aligned} \tag{3.28}$$

When the manipulated variable u, as a function of output y and its derivatives, appears for the first time, the relative degree δ has been reached.

$$\begin{aligned} \dot{y} &= \dot{x_1} = x_2 \\ \ddot{y} &= \dot{x_2} = -c_1 \cdot \sin(x_1 - \vartheta) - c_2 \cdot x_2 + c_3 \cdot ge(x_1) \cdot u \end{aligned} \tag{3.29}$$

The relative degree $\delta = 2$ is the same as the system order n=2. The FDP system is flat and controllable. A full compensation is performed by choosing u (Equation 3.30) with the linear control law $v = -k1 \cdot x_1 - k2 \cdot x_2$.

$$u = \frac{1}{c_3 \cdot ge(x_1)}(c_1 \cdot \sin(x_1 - \vartheta) + c_2 \cdot x_2 - k1 \cdot x_1 - k2 \cdot x_2) \tag{3.30}$$

The parameters k1 and k2 are chosen such that a satisfying performance is achieved in the simulation in Matlab Simulink.

3.5. Backstepping

The reference for elaborating this section is [15, P.400-404]. Compared to exact linerization, which requires a precise model and eliminates all nonlinearities, even if they have an advantage for the control behavior, the backstepping method can deal with model uncertainties. Furthermore, nonlinearities can be used for a better control behavior, which can lead to a smaller control variable.

Backstepping can be used if the system is a flat system, as defined in Section 3.4. The system description for integrator backstepping is given by the strict feedback form in Equation 3.31.

$$\begin{aligned} \dot{\mathbf{x}_1} &= \mathbf{f}_1(\mathbf{x}_1) + \mathbf{h}_1(\mathbf{x}_1) \cdot x_2 \\ \dot{x_2} &= f_2(\mathbf{x}_1, x_2) + h_2(\mathbf{x}_1, x_2) \cdot u \end{aligned} \tag{3.31}$$

The bold symbols are multidimensional variables and functions. For the strict feedback form $f_1(0)$=0.

$x_2 = \phi(\mathbf{x}_1)$ is a virtual control law of the first line of Equation 3.31 with $\phi(0) = 0$, that leads to an asymptotically stable equilibrium $\mathbf{x}_{1,R} = \mathbf{0}$ of the first system. Furthermore, a Lyapunov function $V(\mathbf{x}_1)$ is used to guarantee stability for the first line of Equation 3.31.

The direct method of the stability criterion from Lyapunov is shown in Equation 3.32 [15, P. 85].

$$\begin{aligned} V(\mathbf{x}_1) &\succ 0 \\ \dot{V}(\mathbf{x}_1) &\prec 0 \\ \lim_{\|\mathbf{x}_1\| \to \infty} V(\mathbf{x}_1) &= \infty \end{aligned} \tag{3.32}$$

If all points of Equation 3.32 are fulfilled, then the controlled system is globally asymptotically stable. The resulting control law u, for a SISO system (Equation 3.31), is given in Equation 3.33, where $k_2 > 0$ leads to an asymptotic stable equilibrium $[\mathbf{x}_{1,R} \quad x_{2,R}] = \mathbf{0}$.

$$\begin{aligned} u = \frac{1}{h_2(\mathbf{x}_1, x_2)} \Bigg(& \frac{\partial \phi(\mathbf{x}_1)}{\partial \mathbf{x}_1} (\mathbf{f}_1(\mathbf{x}_1) + \mathbf{h}_1(\mathbf{x}_1) x_2) - \frac{\partial V(\mathbf{x}_1)}{\partial \mathbf{x}_1} \mathbf{h}_1(\mathbf{x}_1) - \\ & k_2 \cdot (x_2 - \phi(\mathbf{x}_1)) - f_2(\mathbf{x}_1, x_2) \Bigg) \end{aligned} \tag{3.33}$$

The resulting Lyapunov function for the controlled system is given by Equation: 3.34.

$$V(\mathbf{x}_1, x_2) = V(\mathbf{x}_1) + \frac{1}{2} \cdot (x_2 - \phi(\mathbf{x}_1))^2 \tag{3.34}$$

3.5.1. Backstepping FDP

The starting point for this method is Equation 3.28. The error $z_1 = x_1 - r_t$ ($x_1 = z_1 + r_t$) is introduced, where r_t is the reference value. With a constant reference value ($r_t = const$) the derivative leads to $\dot{z_1} = \dot{x}_1 - \dot{r}_t$ with $\dot{r}_t = 0$. The error z_1 and the relation for x_1 are used to obtain the system description which is used for the backstepping method.

$$\begin{aligned} \dot{z_1} &= x_2 \\ \dot{x_2} &= -c_1 \cdot \sin(x_1 - \vartheta) - c_2 \cdot x_2 - c_3 \cdot ge(x_1) \cdot u \end{aligned} \tag{3.35}$$

The virtual control and the Lyapunov function are given by $\phi(z_1) = -k_1 \cdot z_1$ and $V(z_1) = \frac{1}{2} \cdot z_1^2$.

$$\begin{aligned} V(z_1) &= \frac{1}{2} z_1^2 > 0 \\ \dot{V}(z_1) &= z_1 \dot{z_1} = -k_1 \cdot z_1^2 < 0 \qquad \forall \qquad k_1 > 0 \end{aligned} \tag{3.36}$$

By inserting the corresponding variables and functions in Equation 3.33, we get the control law for the FDP (Equation 3.37).

$$u = \frac{1}{c_3 \cdot ge(x_1)} \left(c_1 \sin(x_1 - \vartheta) + c_2 x_2 - k_1 \cdot x_2 - z_1 - k_2 (x_2 + k_1 \cdot z_1) \right) \tag{3.37}$$

The Lyapunov function of the controlled system is given in Equation 3.38 with $z_2 = x_2 - \phi(z_1)$.

$$V_I(z_1, z_2) = \frac{1}{2} z_1^2 + \frac{1}{2} z_2^2 \tag{3.38}$$

3.6. Adaptive backstepping

The references for this section are [16]–[18]. They all conclude that every model has his uncertainties in its parameters due to modeling errors and simplifications. Those uncertainties, that occur due to model imprecision, are called parametric or structured uncertainties in the literature. For the case of parametric uncertainties of strict feedback systems, adaptive backstepping is one option to increase the control performance and preserve global stability.

In this part the adaptive backstepping is introduced in the general case for a n-th order system The starting point is backstepping, which was introduced in Section 3.5. The uncertain parameters of the system are described by Θ and are introduced to the system here:

$$\begin{aligned} \dot{x}_i &= x_{i+1} + \varphi_i^T(x_1, \cdots, x_i)\Theta \\ \dot{x}_n &= \varphi_0(x_1, \cdots, x_n) + \varphi_n^T(x_1, \cdots, x_n)\Theta + \beta_0(x_1, \cdots, x_n)u \\ y &= x_1 \end{aligned} \tag{3.39}$$

If one line of the system has no uncertain parameter, the control law for this step is determined in the manner of the general backstepping algorithm. In a line with parametric uncertainties, the Lyapunov function of the controlled system without uncertainties has to be extended by this uncertainties. An example is the extended Lyapunov function for the case i=1 with uncertain parameters in this equation with the transformation in error coordinates $z1 = x1 - r_t$. (Equation 3.40).

$$V(z_1, \hat{\Theta}) = V(z_1) + \frac{1}{2}\tilde{\Theta}^T\Gamma^{-1}\tilde{\Theta} \tag{3.40}$$

Where Γ is positive definite and $\tilde{\Theta} = \Theta - \hat{\Theta}$ where $\hat{\Theta}$ is the estimated uncertainty.

By derivation of the Lyapunov function (Equation 3.40), the parameter update law can be obtained ($\dot{V}(z_1, \hat{\Theta}) \quad \rightarrow \quad \dot{\hat{\Theta}}$).

The above process needs to be repeated until the system order n is reached. By combining all obtained adaption laws, the resulting control law for the n-th order system can be calculated.

The following connections are taken from the above mentioned procedure (the exact derivation can be found in [16, P. 58-66]).

Starting with a coordinate transform in error coordinates:

$$\begin{aligned} z_1 &= x_1 - r \\ z_k &= x_k - \phi_{k-1}(x_1, \cdots, x_{k-1}, \hat{\Theta}) \qquad 2 \leq k \leq n \end{aligned} \tag{3.41}$$

The virtual control for the k-th step is given by:

$$\begin{aligned} \phi_k(x_1, \cdots, x_k, \hat{\Theta}) = -z_{k-1} + \sum_{i=1}^{k-1} \frac{\partial \phi_{k-1}}{\partial x_i} x_{i+1} + \frac{\partial \phi_{k-1}}{\partial \hat{\Theta}} \tau_k - \\ \omega_k^T \hat{\Theta} + \left(\sum_{i=1}^{k-2} z_{i+1} \frac{\partial \phi_i}{\partial \hat{\Theta}} \right) \Gamma \omega_k - k_k z_k \end{aligned} \tag{3.42}$$

The functions ω_k and τ_k are given by the Equations 3.43 and 3.44.

$$\omega_k(x_1, \cdots, x_k, \hat{\Theta}) = \varphi_k - \sum_{i=1}^{k-1} \frac{\partial \phi_{k-1}}{\partial x_i} \varphi_i \tag{3.43}$$

$$\tau_k(x_1, \cdots, x_k, \hat{\Theta}) = \tau_{k-1} + \Gamma \omega_k z_k = \Gamma \sum_{i=1}^{k} \omega_i z_i \tag{3.44}$$

The law to update the uncertain parameters is given by:

$$\dot{\hat{\Theta}} = \tau_n(\mathbf{x}, \hat{\Theta}) = \Gamma W \cdot z \qquad with \quad W(\mathbf{z}, \hat{\Theta}) = \begin{bmatrix} \omega_1, & \cdots, & \omega_n \end{bmatrix} \quad z = \begin{bmatrix} z_1 \\ \vdots \\ z_n \end{bmatrix} \tag{3.45}$$

The adaptive control law is given by:

$$u = \frac{1}{\beta_0(\mathbf{x})}\Bigg[- z_{n-1} - \varphi_0(\mathbf{x}) + \sum_{i=1}^{n-1} \frac{\partial \phi_{n-1}}{\partial x_i} x_{i+1} - \omega_k^T \hat{\Theta} + \frac{\partial \phi_{n-1}}{\partial \hat{\Theta}} \tau_n + \left(\sum_{i=1}^{n-2} z_{i+1} \frac{\partial \phi_i}{\partial \hat{\Theta}} \right) \Gamma \omega_n - k_n z_n \Bigg] \tag{3.46}$$

If all uncertainties appear only in the last equation of the system description, those uncertainties are called *matched* uncertainties.

3.6.1. Adaptive backstepping for the FDP

The starting point for the adaptive backstepping is Equation 3.35. By introducing the error $z_2 = x_2 - \phi(z_1)$, the resulting system is given by Equation 3.47.

$$\begin{aligned} \dot{z_1} &= z_2 + \phi(z_1) \\ \dot{z_2} + \dot{\phi}(z1) &= -c_1 \sin(x_1 - \vartheta) - c_2 \cdot x_2 + c_3 \cdot ge(x_1) \cdot u := v \end{aligned} \tag{3.47}$$

The control law for v is given due to the backstepping algorithm (Equation 3.33) where $h_2(\mathbf{x}_1, x_2) = 1$ and $f_2(\mathbf{x}_1, x_2) = 0$.

$$v = -k_2 \cdot z_2 - \frac{\partial V(z_1)}{\partial z_1} h1(z_1) + \dot{\phi}(z_1) \tag{3.48}$$

The parameters c_1 and c_2 should now be considered as uncertain parameters. The resulting system with matched parametric uncertainties is shown in Equation 3.49.

$$\begin{aligned} \dot{z_1} &= z_2 + \phi(z_1) \\ \dot{z_2} + \dot{\phi}(z1) &= W \cdot \Theta + c_3 \cdot ge(x_1) \cdot u := v \\ \textit{with} \quad & W = \begin{bmatrix} \omega_1 & \omega_2 \end{bmatrix} = \begin{bmatrix} -\sin(x_1 - \vartheta), & -x_2 \end{bmatrix} \\ \textit{and} \quad & \Theta = \begin{bmatrix} \Theta_1 \\ \Theta_2 \end{bmatrix} = \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} \end{aligned} \tag{3.49}$$

The control law u for the system (Equation 3.49) is given by:

$$u = \frac{1}{c_3 \cdot ge(x_1)}(v - W \cdot \hat{\Theta}) \tag{3.50}$$

The resulting closed loop description of the system with unknown parametric uncertainties leads to Equation 3.51.

$$\begin{aligned} \dot{z_1} &= z_2 + \phi(z_1) \\ \dot{z_2} &= W \cdot \tilde{\Theta} - k_2 \cdot z_2 - \underbrace{\frac{\partial V(z_1)}{\partial z_1} h_1(z_1)}_{z_1} \end{aligned} \tag{3.51}$$

The Lyapunov function of the controlled system for known uncertainties is given in Equation 3.38. This Lyapunov function needs to be extended by the relation with the uncertainties, that leads to Equation 3.52.

$$V_{II}(z_1, z_2, \hat{\Theta}) = V_I(z_1, z_2) + \frac{1}{2}\tilde{\Theta}^T \Gamma^{-1} \tilde{\Theta} \quad with \quad \Gamma^{-1} = \left(\Gamma^{-1}\right)^T > 0 \tag{3.52}$$

The parameter adaption law is calculated using the time derivative of V_{II}.

$$\begin{aligned} \dot{V}_{II}(z_1, z_2, \hat{\Theta}) =& z_1 \cdot (z_2 + \phi(z_1)) - k_2 \cdot z_2^2 - z_2 \cdot z_1 + z_2 \cdot W \cdot \tilde{\Theta} + \\ & \frac{1}{2}\left[\dot{\tilde{\Theta}}^T \Gamma^{-1} \tilde{\Theta} - \tilde{\Theta}^T \Gamma \dot{\tilde{\Theta}}\right] \quad with \quad \dot{\tilde{\Theta}} = \dot{\Theta} - \dot{\hat{\Theta}} \\ & where \quad \dot{\Theta} = 0 \end{aligned} \tag{3.53}$$

$$\dot{V}_{II}(z_1, z_2, \hat{\Theta}) = \underbrace{-k_1 \cdot z_1^2 - k_2 \cdot z_2^2}_{\dot{V}_I\big|_{\Theta=0}} + z_2 \cdot W \cdot \tilde{\Theta} + \tilde{\Theta}^T \Gamma^{-1} \dot{\tilde{\Theta}} \tag{3.54}$$

$$\dot{V}_{II}(z_1, z_2, \hat{\Theta}) = \dot{V}_I\big|_{\Theta=0} + \underbrace{(z_2 \cdot W \cdot \tilde{\Theta})^T}_{z_2 \cdot \tilde{\Theta}^T W^T} + \tilde{\Theta}^T \Gamma^{-1} \dot{\hat{\Theta}} \tag{3.55}$$

$$\dot{V}_{II}(z_1, z_2, \hat{\Theta}) = \dot{V}_I\big|_{\Theta=0} + \tilde{\Theta}^T \cdot \underbrace{(z_2 \cdot W^T - \Gamma^{-1} \dot{\hat{\Theta}})}_{=0} \tag{3.56}$$

Equation 3.56 leads to the adaption law in Equation 3.57.

$$\dot{\hat{\Theta}} = z_2 \cdot \Gamma \cdot W^T \tag{3.57}$$

The final control law is given for u by Equation 3.58.

$$u = \frac{1}{c_3 \cdot ge(x_1)} (-k_2 \cdot z_2 - z_1 - k_1 \cdot x_2 - W \cdot \hat{\Theta}) \tag{3.58}$$

The matrix Γ is chosen to be positive definite and symmetric.

$$\Gamma = \begin{bmatrix} a & b \\ b & d \end{bmatrix} \tag{3.59}$$

There are two possibilities to achieve the above mentioned properties. With coupling the uncertainties with each other or without coupling. The coupling can be achieved, if the value b is not zero, and the resulting matrix is positive definite. For no link between the two uncertain parameters, b is zero and the remaining values are positive to guarantee a positive definite matrix Γ.

4. Simulation and Implementation

In this section, the simulation and the code implementation on the MCU are presented.

The programs used for the simulation and evaluation are MATLAB R2018b including Simulink and CoolTerm as a serial port terminal. For the implementation on the MCU, the development environment Mbed is used. The program, which runs on the MCU, measures data, evaluates them and transfers the data via USB over a serial connection to the PC. The program is structured in a main part, where the functions for measurements, evaluation and control purposes are called with timers and interrupts in the main function. The code was implemented in C++ and a flowchart of the program can be found in Appendix B.5.

The angle is measured with a Hall-sensor, while the speed of the propeller is measured twice: once with an infrared sensor, which can only measure low speeds; the second measurement is obtained using a phase sensor which is measuring the Back Electro Magnetic Force (BEMF) of the BLDC motor. The PWM-output of the MCU is connected to the ESC, which provides the AC voltage for the motor. The ESC is a converter, that consist of 6 Transistors, which switch on and off corresponding to a pattern, to generate the three phase voltages.

The following settings for the serial connection apply to all implementations:

- BR = 115200 bauds
- Data bits = 8
- Stop bit = 1
- Parity bit = odd

4.1. n-Controller

There are several limitations for the n-controller like:

The measurement accuracy of the speed of the propeller is $\pm$ 0.98RPS, for an evaluation time of 170ms. This is due to the sensor which is measuring the change of the back EMF. One revolution consists of 6 changes. The second limitation is the restrain of the current to 4A, due to the power supply, and the resulting limitation of the PWM to 0% to 20%.

With that limitations and the evaluated transfer function (Eq.: 2.14) of the BLDC, the PI-controller and the smith predictor are implemented.

A starting point for the evaluation of the n-controller is the *open loop method*. The parameters k_p and k_i can be calculated from the system parameters of the transfer function G(s) (Equation 2.18).

Two different methods, namely the Ziegler Nichols and the Chien, Hrones, Reswick (CHR) are used to design the PI-controller. The method after CHR distinguishes between aperiodic and 20% overshot behavior of the controlled plant. Furthermore, it distinguishes between reference tracking (rt) and disturbance attenuation (da). For the control of the speed, the rt was used. The calculation of the corresponding control parameter can be seen in Appendix B.1. The system parameters used for the calculation are obtained in Section 2.4.

- K= 1168
- T_{PT1}= 101.3ms
- T_d=184.1ms

The resulting parameters for k_p and k_i can be found in Table 4.1. The last row of this table has the resulting parameters for the SP, which are used for the control of the angle φ.

Two different cases are examined. In the first one, the control sample time Tc_n is less than the evaluation time of the speed measurement. In the second one, the sampling time is the same as the measurement evaluation time. For the first case, the sampling time Tc_n is set to 5 ms due to the intention to provide a fast inner control loop for the angular control. The evaluation time

	k_p	k_i
Ziegler Nichols	0.00042	0.00069
CHR-aperiodic- rt	0.00017	0.0014
CHR-20% oscillation- rt	0.00028	0.0028
SP	0.00038	0.00237

Table 4.1.: Control parameters k_p and k_i for the n-control. The control parameters are calculated with the method of Ziegler Nichols, CHR and the adjustment rule for the SP.

T_s of the measured speed is 170ms. For the second case, the sample time of the controller Tc_n and the evaluation time T_s of the measured speed are set to 100ms. The evaluation time of 100ms leads to a measurement uncertainty of the speed of $\pm$ 1.66RPS.

The simulation and implementation part will be explained for a general case, if further differences besides the sample time occur, they will be mentioned.

4.1.1. Simulation

The solver ode1 (euler), with the fixed step size Tc_n is used for all simulations. The controllers have been designed without uncertainties in the simulation. For further investigation, the uncertainties of the measured signals are added to the simulation.

PI-Control

The PI-Controller was implemented in the parallel form. The control loop with the parameters from Table 4.1 was simulated in Simulink. The system model can be seen in Equation 2.18.

$$G(s) = \frac{1168}{1 + s \cdot 0.1013} \cdot e^{-s \cdot 0.1841}$$

The Simulink connection diagram can be seen in Figure 4.1.

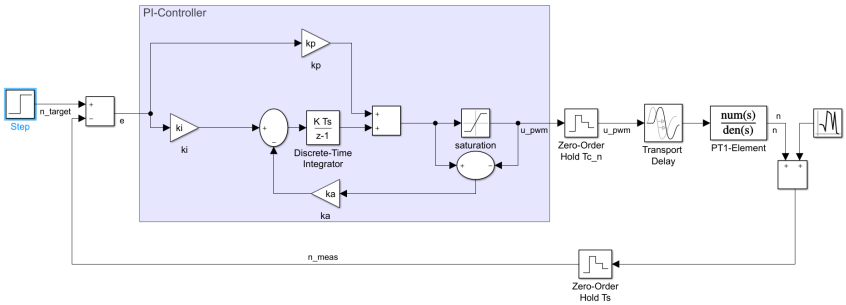

Figure 4.1.: The Simulink connection diagram of the PI-n-control. In the blue area, the discrete PI-controller with anti wind up is shown, the sample time is set to Tc_n. The sample time for the Zero Order Hold (ZOH) after the PI-controller is the same as the control sample time Tc_n. This ZOH simulates a DAC. The second ZOH has the sample time of the evaluation speed of the speed measurement. The system is simulated as a transport delay, with the delay time of T_d=184.1ms, multiplied with the transfer function of the BLDC (Equation 2.18).

The simulation is divided into a discrete part (controller), and a continuous part (system) which are separated with the zero order hold. In the blue area, the discrete PI-controller with anti wind up is shown, the sample time is set to Tc_n. For the discrete PI controller, the continuous integrator ($\frac{1}{s}$) is replaced by the discrete integrator ($\frac{Tc_n}{z-1}$). The anti-windup is realized as a feedback of the difference before and after the saturation, this difference is multiplied by k_a which is the gain for the anti-windup. The sample time for the ZOH after the PI-controller is the same as the control sample time Tc_n. This ZOH simulates a DAC. The second ZOH has the sample time of the evaluation speed T_s of the speed measurement. The system is simulated as a transport delay, with the time delay of T_d=184.1ms, multiplied with the transfer function of the BLDC (Equation 2.18)

The sampling times are:

- First case: $Tc_n = 5ms$ and $T_s = 170ms$
- Second case: $Tc_n = 100ms$ and $T_s = 100ms$

Smith-Predictor

For the implementation on MCU, the SP must be discretized. For this reason, the continuous system (Equation 2.18) needs to be transformed to a discrete system. The transform from the Laplace-domain to the z-domain was performed with the *c2d* command in Matlab.

The sample time for the transform is Tc_n. The resulting transfer function for the first case can be seen in Equation 4.1 and for the second case in Equation 4.2.

$$G(z) = \frac{56.25}{z - 0.9518} \cdot z^{-37} \tag{4.1}$$

$$G(z) = \frac{732.8}{z - 0.3726} \cdot z^{-2} \tag{4.2}$$

The Simulink connection diagram can be seen in Figure 4.2.

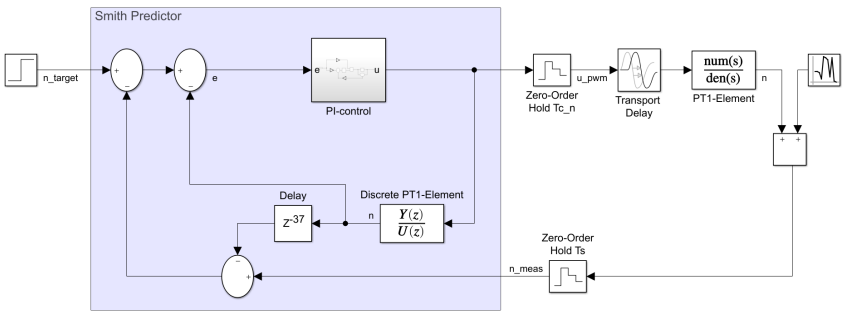

Figure 4.2.: The simulink connection diagram of the SP-n-control. The SP is shown in the blue area. It consists of a discrete copy of the system, a discrete delay and a PI-controller. The first Zero Order Hold (ZOH) after the SP has the same sample time as the control sample time Tc_n. This ZOH simulates an DAC. The second ZOH has the sample time of the evaluation speed T_s of the speed measurement.

4.1.2. Implementation

The discrete part of the simulation (blue area in Figure 4.1 and Figure 4.2) is used for the code generation. Therefore, the *Embedded Coder* from Simulink generates the C++ code of the selected area. The settings for the embedded coder can be found in appendix B.3. The generated code for the n-controller is migrated to the program of the MCU. For this reason a new function, with the name *n_control*, is created and the generated variables from the embedded coder are changed in accordance to the variables used in the program. The online compiler from *Mbed* is used to generate the binary file for the MCU. The program is loaded via USB to the MCU and is tested for the expected outcome with *CoolTerm*. The data is exchanged via a bidirectional serial connection.

4.1.3. Evaluation

For the evaluation, the communication between the MCU and the PC is established with the Matlab script *communication_n_control.m*. The code to establish the serial connection with the mentioned Matlab script is shown in Appendix B.4. A serial bidirectional connection is used to send a reference value of $n_{reference}$ to the MCU and to receive data in the following order: Angle in rad; PWM signal; measured speed of propeller in RPS; $n_{reference}$ and time in μs. The reference value needs to be sent in the following format to be recognized by the MCU:

```
'n: %f\n'
```

The reference variable $n_{reference}$ is in the range of 165 to 175RPS with a step size of 2RPS. For each reference value, 400 samples are measured. The measured data is saved in different Matlab files for each method (see Appendix B.2).

The simulation time for the comparison is set to 10s. For the comparison between reality and simulation, the uncertainty of the phase sensor is added to the simulation. This uncertainty is simulated with a random number generator with normal Gaussian distribution. The mean is set to zero and the variance σ is set to 0.16 for the evaluation time of 170ms and σ=0.45 for

the evaluation time of 100ms. The random number is added to the output of the system.

4.2. φ - Control

A cascaded control loop is used to control the angle of the pendulum arm. The structure of the control loop is shown in Figure 4.3. There you can see the Simulink connection diagram of the control loop. The green part is the continuous system. The yellow part is the n-control loop with the sample time Tc_n. The blue part is the angular control loop with the sample time Tc_φ. The sample times of the ZOH blocks are corresponding to the sample time of the inner and outer control loop.

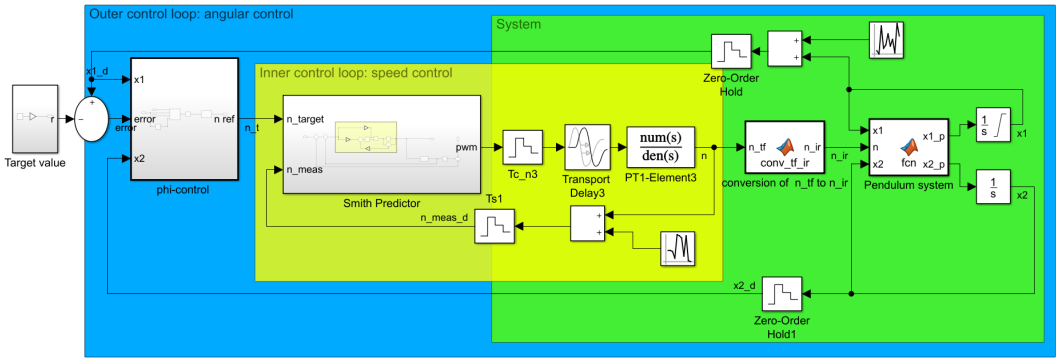

Figure 4.3.: The used cascaded control structure. The green part is the continuous system. The yellow part is the n-control loop with the sample time Tc_n. The blue part is the angular control loop with the sample time Tc_φ.

Four different controllers are tested and analyzed for the angular control of the FDP. Those four controllers are:

- PI-Control;
- Exact linearization;
- Backstepping and
- Adaptive Backstepping

All of these controllers are studied for two different cases. The sample time of the n-control for the first case is $Tc_n = 5ms$ and the sample time of the φ-control is $Tc_\varphi = 100ms$. In every 170ms, the measured speed of the propeller is provided to the n-control. In this case, the measurement uncertainty of the speed is $\pm$ 0.98RPS. The n-controller is a SP with the following control parameters:

- $k_i = 0.00256$
- $k_p = 0.00016$

The system parameters for the discrete copy of the system are given in Equation 4.1.

For the second case, the sample time of the n-control is $Tc_n = 100ms$ and the sample time of the φ-control is $Tc_\varphi = 800ms$. In every 100ms, the measured speed of the propeller is provided to the n-controller. In this case, the measurement uncertainty of the speed is $\pm$ 1.66RPS. The n-controller is a SP with the following control parameters:

- $k_i = 0.00237$
- $k_p = 0.00038$

The system parameters for the discrete copy of the system are given in Equation 4.2.

Additionally to the previous mentioned limitation for the n-control, the backlash of the cogs leads to an inaccuracy of the angular measurement every time the pendulum arm changes directions. The backlash is 1.3°.

Furthermore, the investigation focuses on angles between 75 and 90°, because the settling time of the n-controller is to slow, to be able to control angles bigger than the transmission point as explained in Section 5.3.

4.2.1. Simulation

The system is divided into two parts: the speed behavior of the motor and the behavior of the pendulum depending on the applied torque. The speed behavior is described with the transfer function G(s) (Equation: 2.18) and the behavior of the FDP is described with two differential equations (Equation

2.22). Due to the fact that the measured speed and the actual speed differ (see Section 2.3), a conversion block needs to be introduced between the transfer function of the BLDC and the system description of the FDP. The integrator for x_1 has the initial condition set to $x_{1,0} = 67°$and the saturation was activated. The upper limit is set to 150° and the lower bound to the initial condition $x_{1,0}$, which is the rest position of the pendulum. The initial condition for the integrator of x_2 is set to $x_{2,0} = 0$. The two ZOHs, which represent an Analog Digital Converter (ADC) in the green area of Figure 4.3, have the same sample time as the cycle time Tc_φ of the angular controller. The ZOHs in the yellow area have the same sample time as sample time Tc_n of the n-control. The measurement uncertainty of the speed sensor is simulated with a random number generator with normal distribution. The mean μ is set to 0 and the variance σ is set to 0.16 for the evaluation time of 170ms and to σ=0.45 for the evaluation time of 100ms. The backlash of the cogs is simulated with a uniform distribution with the limit $\pm$0.0209 rad, which is $\pm 1.3°$.

The yellow part of Figure 4.3 represents the n-controller. This part was explained in Section 4.1.

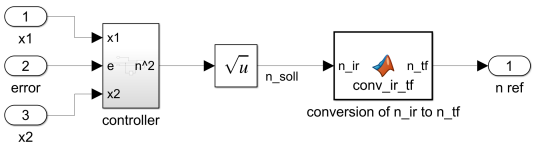

Figure 4.4.: The phi-control subsystem consists of a controller, a square root function and a conversion from the real speed of the fan to the measured speed.

The subsystem *phi-control* in the blue area is now explained in more detail. The Figure 4.4 shows the layer of the subsystem phi-control. This subsystem consists of a controller, a square root function and a conversion from the real speed of the fan to the measured speed and provides the reference value of the speed for the inner control loop. If not mentioned otherwise, the controller is replaced by the respective algorithm.

For the controller design, the uncertainties are not considered in the simulation. The used control algorithms are explained in Chapter 3.

PI-Control

For the PI-control, the standard PID-controller block from Matlab Simulink replaces the phi-control subsystem. The output saturation and the anti wind up are activated. The limits are in the range of 60 to 200RPS. The control parameters: k_i, k_p and k_a are determined heuristically. This means that the parameters are adjusted until the desired behaviour is achieved in the simulation. The resulting control parameters for the case with the sample times $Tc_n = 5ms$ and $Tc_\varphi = 100ms$ are:

$$k_i = 22 \quad , \quad k_p = 20 \quad \text{and} \quad k_a = 2$$

For the slower controller, with the sample times $Tc_n = 100ms$ and $Tc_\varphi = 800ms$, the control parameters are:

$$k_i = 25 \quad , \quad k_p = 35 \quad \text{and} \quad k_a = 2$$

Exact linearization

The implemented control law is given below.

$$u = \frac{1}{c_3 \cdot ge(x_1)}(c_1 \cdot \sin(x_1 - \vartheta) + c_2 \cdot x_2 - k1 \cdot x_1 - k2 \cdot x_2)$$

The parameters are chosen based on simulation results.

The parameters used for the exact linearization for the first case are:

$$k_1 = 15 \quad \text{and} \quad k_2 = 7$$

The parameters used for the second case are:

$$k_1 = 17 \quad \text{and} \quad k_2 = 5$$

Backstepping

The implemented control law is shown below:

$$u = \frac{1}{c_3 \cdot ge(x_1)} \Big(c_1 \sin(x_1 - \vartheta) + c_2 x_2 - k_3 \cdot x_2 - z_1 - k_{BS}(x_2 + k_3 \cdot z_1) \Big)$$

The parameters used for the first case are:

$$k_1 = 4 \quad \text{and} \quad k_{BS} = 3.1$$

The parameters used for the second case are:

$$k_1 = 5.9 \quad \text{and} \quad k_{BS} = 3.2$$

Adaptive backstepping

For adaptive backstepping, four cases are simulated. In the first case, the uncertain system parameters c_1 and c_2 are not linked to each other via the matrix Γ. In the second case, the uncertain parameters c_1 and c_2 are coupled via the matrix Γ. The control law for this two cases is given below:

$$u = \frac{1}{c_3 \cdot ge(x_1)} (-k_{BS} \cdot z_2 - z_1 - k_1 \cdot x_2 - W \cdot \hat{\Theta})$$

The update law is given by:

$$\dot{\hat{\Theta}} = z_2 \cdot \Gamma \cdot W^T$$

Where W=$\begin{bmatrix} -\sin(x_1 - \vartheta) & -x2 \end{bmatrix}$ and $\hat{\Theta}$=$\begin{bmatrix} \hat{c}_1 \\ \hat{c}_2 \end{bmatrix}$.

The matrix Γ needs to be positive definite. For the first case, the value b of Γ (Equation 3.59) is set to zero. For the second case, the value of b is not zero and this values have to be determined such that the matrix Γ is

positive definite. In the third case, all three system parameters c_1, c_2 and c_3 are uncertain parameters. In this case, the parameters are not coupled via the matrix Γ. In the fourth and last case, the parameters are coupled via the matrix Γ. This leads to the following control law:

$$u = \frac{1}{ge(x_1)}(-k_{BS} \cdot z_2 - z_1 - k_1 \cdot x_2 - W \cdot \hat{\Theta})$$

The update law and W are the same for all four cases. $\hat{\Theta}$ changes for the last two cases to $\hat{\Theta}=\begin{bmatrix} \frac{c1}{c3} \\ \frac{c2}{c3} \end{bmatrix}$.

All found control parameters for the faster cascaded control are presented below. The sample times are $Tc_n = 5ms$ and $Tc_\varphi = 100ms$. The control parameters for the first case are:

$$\begin{aligned} k_1 &= 3.5 \quad \text{and} \quad k_2 = 3.5 \\ \hat{\Theta}_0^T &= \begin{bmatrix} c_1 - 5 & c_2 \end{bmatrix} \\ \Gamma &= \begin{bmatrix} 1.2 & 0 \\ 0 & 0.05 \end{bmatrix} \end{aligned}$$

The control parameters for the second case are:

$$\begin{aligned} k_1 &= 4.5 \quad \text{and} \quad k_2 = 3.5 \\ \hat{\Theta}_0^T &= \begin{bmatrix} c_1 - 5 & c_2 \end{bmatrix} \\ \Gamma &= \begin{bmatrix} 1.2 & 0.2 \\ 0.2 & 0.05 \end{bmatrix} \end{aligned}$$

The control parameters for the third case are:

$$\begin{aligned} k_1 &= 900 \quad \text{and} \quad k_2 = 5 \\ \hat{\Theta}_0^T &= \begin{bmatrix} 22000 & 0 \end{bmatrix} \\ \Gamma &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \end{aligned}$$

The control parameters for the fourth case are:

$$\begin{aligned} k_1 &= 900 \quad \text{and} \quad k_2 = 6 \\ \hat{\Theta}_0^T &= \begin{bmatrix} 22000 & 0 \end{bmatrix} \\ \Gamma &= \begin{bmatrix} 1 & 0.2 \\ 0.2 & 0.5 \end{bmatrix} \end{aligned}$$

Below are all control parameters for the slower cascaded control with the sample times of $Tc_n = 100ms$ and $Tc_\varphi = 800ms$. For the first case, with the uncertain parameters c_1 and c_2 and without coupling, the resulting control parameters are given by:

$$\begin{aligned} k_1 &= 4 \quad \text{and} \quad k_2 = 4.7 \\ \hat{\Theta}_0^T &= \begin{bmatrix} c_1 - 5 & c_2 \end{bmatrix} \\ \Gamma &= \begin{bmatrix} 1.2 & 0 \\ 0 & 0.05 \end{bmatrix} \end{aligned}$$

For the second case, the control parameters are:

$$\begin{aligned} k_1 &= 4.5 \quad \text{and} \quad k_2 = 3.8 \\ \hat{\Theta}_0^T &= \begin{bmatrix} c_1 - 5 & c_2 \end{bmatrix} \\ \Gamma &= \begin{bmatrix} 1.2 & 0.2 \\ 0.2 & 0.05 \end{bmatrix} \end{aligned}$$

The control parameters for the third case are:

$$\begin{aligned} k_1 &= 900 \quad \text{and} \quad k_2 = 6.7 \\ \hat{\Theta}_0^T &= \begin{bmatrix} 22000 & 0 \end{bmatrix} \\ \Gamma &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \end{aligned}$$

The control parameters for the fourth case are:

$$\begin{aligned} k_1 &= 900 \quad \text{and} \quad k_2 = 7.6 \\ \hat{\Theta}_0^T &= \begin{bmatrix} 22000 & 0 \end{bmatrix} \\ \Gamma &= \begin{bmatrix} 1 & 0.2 \\ 0.2 & 1 \end{bmatrix} \end{aligned}$$

4.2.2. Implementation

For each of the four control algorithms, the phi-control subsystem from Figure 4.3 was used to perform the code generation via the *Embedded Coder* from Simulink. The generated C++ code for the corresponding control concept of the φ-control is migrated to the program of the MCU. For this reason, a new function with the name *phi_controller* is created and the program is saved for each control concept individually. The generated variables from the code generation needed to be changed accordingly to the variables used in the program. The online compiler from *Mbed* is used to generate the binary file for the MCU. The program is loaded to the MCU via USB and is tested for the expected outcome with *CoolTerm*. The data is exchanged via a bidirectional serial connection.

4.2.3. Evaluation

For the evaluation, the communication between the MCU and the PC is established with the Matlab script *communication_phi_control.m*. A serial bidirectional connection is used to send the reference value r_t from the PC to the MCU. The reference value needs to be sent in the following format to be recognized from the MCU:

```
'r: %f\n'
```

The data sent from the MCU to Matlab for the PI-controller, exact linearization and backstepping are the following: the first value is the angle x_1; the second value is the PWM-signal; the third value is the measured speed of the phase sensor; the fourth value is the reference speed and the last value is

the time in ms. In the data sent for adaptive backstepping, the PWM-signal is replaced by the estimates of the uncertain parameters. The reference values for this evaluation were 75, 80, 85 and 90°. For each reference value the resulting angle was measured for a minimum time of 60s. The length of the recorded sample is not the same for each reference value due to the varying transmission time of the respective measurement. For the first case, with the faster control loop, the recorded data is between 1000 and 1600 samples per angle. In the second case, with the slower control loop the samples are in a range of 600 to 800 samples per angle.

For the comparison, between the measured data and the simulation, the uncertainties for the speed measurement and the angle sensor are added to the simulation. The simulation time is 80s.

5. Results and Outlook

5.1. Model verification

In this section, the observations during system model verification from Section 2.5 are mentioned and probable reasons are given.

The first thing to mention is, that the entire system description refers to the warm state of the components used. For this reason, all experiments and measurements are recorded twice. The second recorded data is used for evaluation purposes. The different behavior for the cold and warm state of the FDP, results from the components used for the motor and the ESC.

The behavior of the pendulum, also depends on whether it is placed close to a wall or freely in the room. This can be explained by the rotation of the propeller: the rotation creates an airflow that is reflected by the wall and causes interference. All control concepts are evaluated using the system description in Section 2.5, this system description was recorded while the pendulum was placed on a table far from a wall.

5.2. n-control

In this section, the results of the n-controller evaluation are presented and discussed. The influence of the difference between the actual dead time of the system and the dead time used in the SP is shown in Figure 5.1.

For this simulation, the dead time $\hat{T}_d$ of the SP is equated with the evaluated dead time (T_d=184.1ms) from Section 2.4. The dead time of the system is varied by $\pm$ 40%. The blue line is the ideal case, where $T_d = \hat{T}_d$. The red

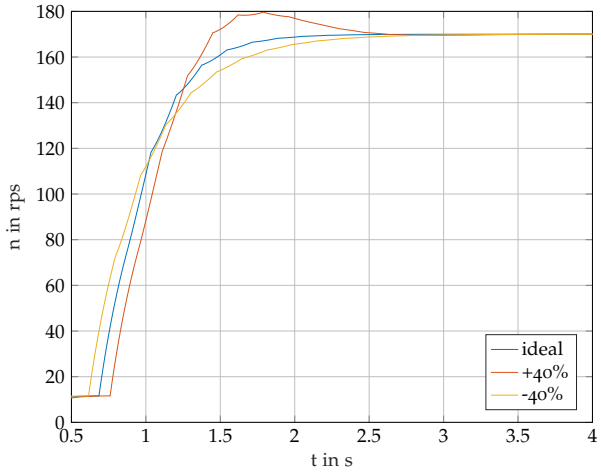

Figure 5.1.: The behavior of the SP for varying time delays. For this simulation, the dead time $\hat{T}_d$ of the SP is set to T_d=184.1ms. The dead time of the system is varied by $\pm$ 40%. The blue line is the ideal case, where $T_d = \hat{T}_d$. The red line is the case where $T_d > \hat{T}_d$. The yellow line is the case where $T_d < \hat{T}_d$.

line is the case where $T_d > \hat{T}_d$. This leads to an overshoot in the output. The yellow line is the case where $T_d < \hat{T}_d$. This results in a longer settling time, but no overshoot occurs.

5.2.1. Pi-Control

The control parameters calculated with CHR, for the case of 20% overshot, can not be used for the control of the speed of the fan, because they cause a high control variable u at the start, which leads to a too high current that the power supply can not provide and the system shuts down.

The method of Ziegler Nichols (ZN) leads to a slow control response. The reference value is reached within 10 seconds. In this case, there is no big difference between the different control approaches described in Figure 5.2.

With the aperiodic approach of CHR, the best result with the calculated parameters, without any adjustment of the parameters, is achieved. In this case, a difference between the two PI-controllers with different values for

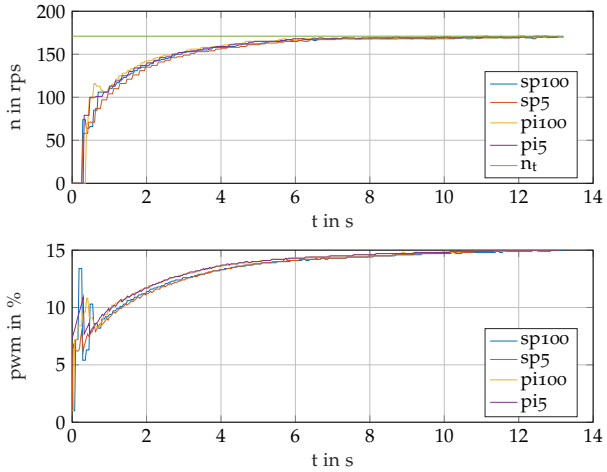

Figure 5.2.: The first figure shows the measured control response of a reference value of 171 RPS for the Ziegler Nichols Method. The second plot shows the control variable u. The blue line is the SP for the case with Tc_n=100ms. The red line is the SP for Tc_n=5ms. The yellow line is the PI controller for Tc_n=100ms. The purple line is the PI controller for Tc_n=5ms.

Tc_n can be detected. The faster controller, with Tc_n=5 ms, needs longer to reach the reference value. In the second case with a slower sample time of the controller, but with the same sample time as the measured signal, the reference value can be reached faster with a slight overshot (Figure 5.3).

The simulation and the measured curves have only small deviations (Figure 5.4). If the deviations are bigger, it can be assumed that this is due to the varying time delay in the system.

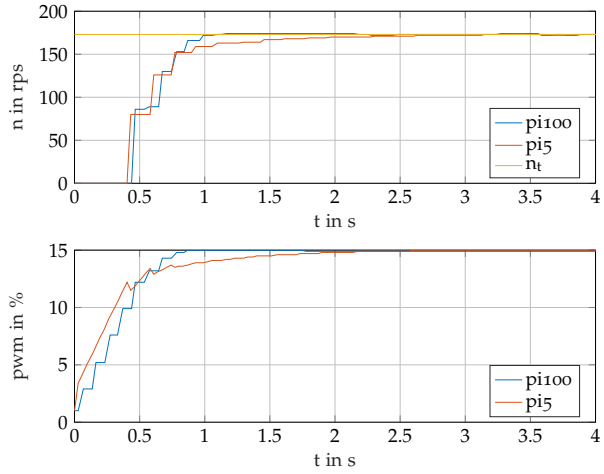

Figure 5.3.: Comparison of the PI controllers with the sampling times of Tc_n=5ms and Tc_n=100ms, for which the CHR method for the aperiodic case with a reference value of n_t=173RPS is used. The first plot shows the measured speed of the propeller and the second plot shows the control variable u. The blue line is the PI controller with the sample time of Tc_n=100ms. The red line is the PI-controller with the sample time of Tc_n=5ms.

5.2.2. Smith-Predictor

The first control parameters for the SP are calculated with the approach of Hägglund (Equation 3.19), which leads to k_p=0.00085 and k_i=0.0085. Those control parameters can not be used in the controller due to the used power supply. Instead, the approach of Palmor (Equation 3.16) is used. The case which is able to guarantee stability for all uncertainties in the time delay with $K_c < 3$ is used and leads to k_p=0.00026 and k_i=0.0026.

The control parameters are found by iterative changes of the parameters in the simulation and testing on the model. The final parameters for the sample time Tc_n=100ms are $k_p = 0.00038$ & $k_i = 0.00237$ and $k_p = 0.00016$ & $k_i = 0.0025$ for the sample time Tc_n=5ms.

In Figure 5.5a, the comparison between the simulation and the measured signal for Tc_n=5ms is shown. In this figure, it can be seen that the simulation and the measured signal show the same behavior. This leads to the conclusion that the time delay in the simulation and the varying time delay are in

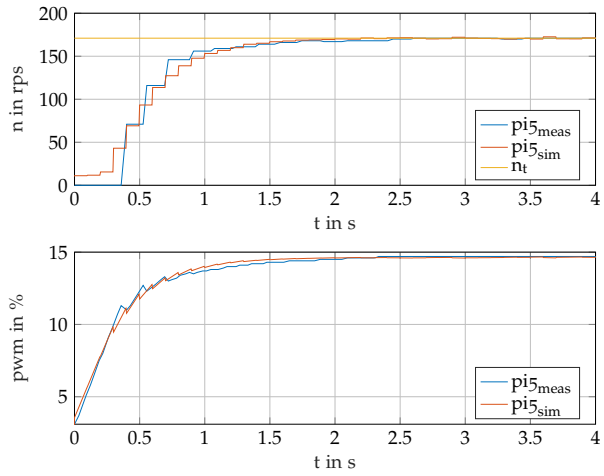

Figure 5.4.: Comparison of the PI-control (Tc_n=5ms) measured data with the simulation. The first plot shows the measured speed of the propeller and the second plot shows the control variable u. The blue line is the measured PI controller. The red line is the simulation of the PI-controller.

the same range. In Figure 5.5b, it can be seen that the simulation has a bigger deviation than in Figure 5.5a. This can be explained due to the varying time delay. The varying time delay has an influence of the performance of the SP. In Figure 5.5b the occurred time delay of the system is bigger than the time delay used in the simulation (this is evident from the rise of the blue curve).

Figure 5.6 shows the measured data of the PI-Control and the SP for Tc_n=100ms, with the same control parameter (Table 4.1, Line 2). In the first diagram, the output (the speed of the fan) is plotted over the time. In the second diagram, the manipulation variable, in the form of the PWM-signal, is shown. It can be seen, that the SP takes the dead time into account for the calculation of the actuating variable, resulting in a smaller control variable as for the PI controller.

This shows that the SP can use higher values for the control parameters to achieve better control performance.

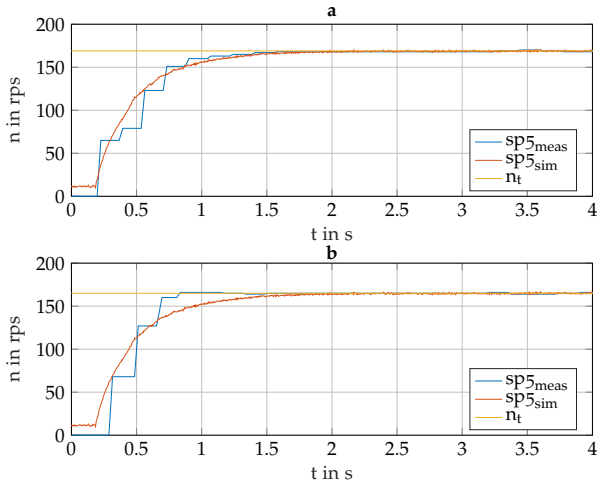

Figure 5.5.: The behavior by different occurring time delays for the SP (Tc_n=5ms) of the measured data, in comparison to the simulation. The blue line is the measured data, the red line is the simulated data and the yellow line is the reference value. In Figure a, the case for $T_d = \hat{T}_d$ is shown. In Figure b, the case for $T_d > \hat{T}_d$ is shown.

5.2.3. Chosen controller for the inner control loop

The decision upon which n-controller is chosen for the inner control loop was determined with the least error square sum method (Equation 5.7) and the condition that no overshot occurs. This additional condition is based on the angular control. It is necessary for the case that the angle is close to the transition point. A higher speed of the motor would cause the arm to tilt.

$$e = \Sigma(n_t - n_{meas})^2 \tag{5.1}$$

The least error square sum was calculated for:

- SP with CHR20 parameters and a sample time of Tc_n=5ms.
- SP with SP parameters and a sample time of Tc_n=100ms.
- SP with SP parameters and a sample time of Tc_n=5ms.
- PI-controller with CHR aperiodic parameters and a sample time of Tc_n=100ms.
- PI-controller with SP parameters and a sample time of Tc_n=100ms.

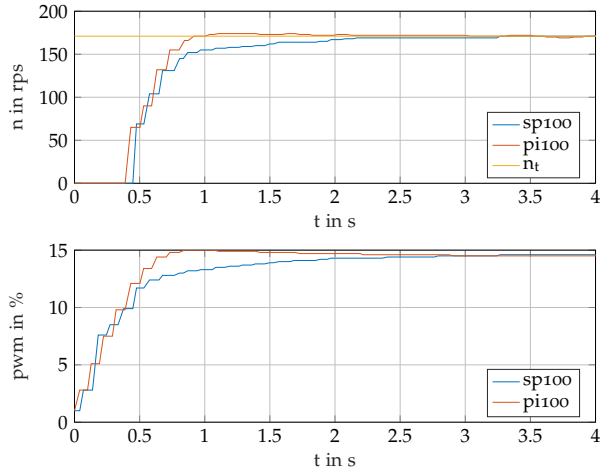

Figure 5.6.: Comparison of the SP with the PI-Controller with CHR aperiodic (Tc_n=100ms). The first plot shows the measured speed of the propeller and the second plot shows the control variable u. The blue line is the SP and the red line is the PI-controller.

- PI-controller with SP parameters and a sample time of Tc_n=5ms.

For this calculation all six measured reference values, 165RPS to 175RPS with a step size of 2RPS, where taken into account. The sum was calculated for the first rise of the output plus 140 samples, which is approximately equivalent to 4.5 seconds.

The measured data for the reference value of 173RPS can be seen in Figure 5.7a.

The three best results are: the SP with the SP parameters for a sample time of 100ms, the PI-controller with the parameters of CHR for the aperiodic case for a sampling time of 100ms and the PI-controller with the same parameter as the SP for a sampling time of 5ms. In Figure 5.7b, the three lines of the measured signals for a reference speed of 175RPS are displayed. The lines all start at the same time because the time delay was eliminated for this evaluation. Due to the second condition with no overshoot, the SP is chosen as controller for the rotor speed n. Using a less conservative controller is not possible because this could lead to an oscillation of the arm.

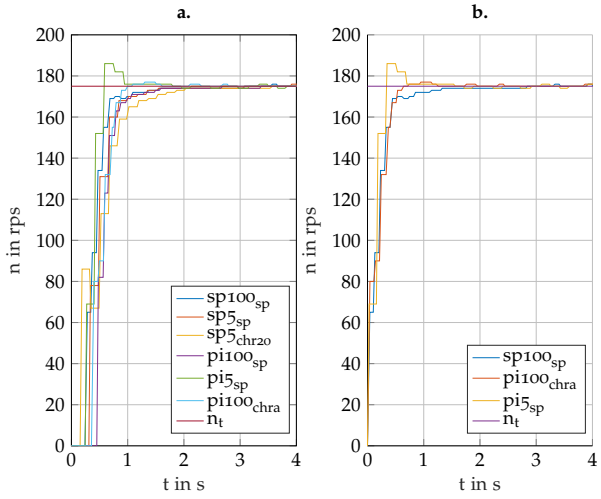

Figure 5.7.: Figure a shows the measured data for all cases for which the least square error calculation was performed. Figure b shows the three best results, where the blue line is the SP for Tc_n=100ms. The red line is the PI-controller with the parameters of CHR for the aperiodic case with the sample time of Tc_n=100ms. The yellow line is the PI-controller with the same parameters as the SP with a sample time of Tc_n=5ms.

5.3. φ-Control

In this section, the results of the angular control are presented.

When controlling the angle, it had to be specified that no overshooting of the set speed is permissible. This is achieved by a more conservative design of the speed controller. Due to the different dead times, the parameters of the SP are designed for a dead time of +40% of the mean dead time (T_d=184.1ms). This is a compromise that must be made to prevent overshooting. If the dead time is bigger than +40%, overshooting will occur.

The tipping point is at the angle of 105°, and to remain in a steady state larger than the tipping point, a lower propeller speed than at the tipping point is required. Due to the limitations of the n-controller, the minimum time without overshoot until the stationary speed is reached is two seconds. The settling time of the n-controller is limited due to the differing time delays and the used power supply. The final settling time can alternate

due to the occurring varying time delay. Since the n-controller always lags behind the reference variable, no angles larger than the tipping point can be controlled.

For the distinction between the two cases, the faster and slower cascaded control, the indices sp5 and sp100 are introduced. The index sp5 stands for cascade control with the sample times Tc_n=5ms and Tc_φ=100ms. Index sp100 represents the cascaded control with the sample times Tc_n=100ms and Tc_φ=800ms.

It turns out that no better control results can be achieved with the faster cascaded control.

5.3.1. PI-control

The first intention for the angular control was to use a PID controller. This was not possible because of the uncertainties that occur, as the derivative part would provide a too high actuating variable u and that leads to oscillations within the limits of the arm. For this reason, the PI controller was used instead. In this case, the simulation deviates very much from reality (Figure 5.8a). This leads to the conclusion that additional uncertainties occur, which are not considered in the simulation. For the second case, with the sampling times Tc_n=100ms and Tc_φ=800ms, the arm oscillates around the setpoint with an angular deviation of $\pm 5°$. In addition, some outliers can be observed. The result is, that it is not well suited to control the angle of the pendulum.

5.3.2. Exact linearization

With the exact linearization, the angles 85° and 90° can be controlled within $\pm$ 3.3°. For the angles 85° and 90°, the simulation and the measured data show a similar behavior. This can be seen in Figure 5.8b.

The measured data of the faster and slower cascaded control are shown in Figure 5.9. The faster cascaded control, with the sampling times Tc_n=5ms and Tc_φ=100ms, is the blue curve. The slower cascaded control, with the sampling times Tc_n=100ms and Tc_φ=800ms, is the red curve. The reference

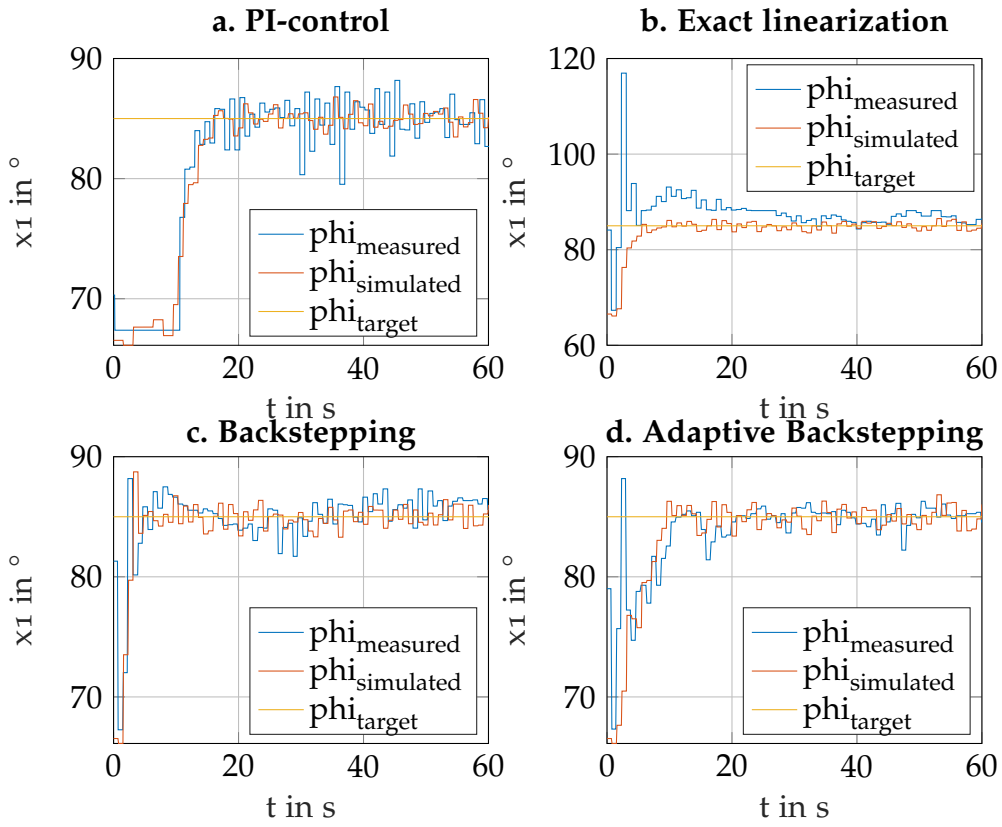

Figure 5.8.: Comparison between the measured and the simulated data for the slower cascaded control (sp100). The reference value is 85° for all plots. The blue lines are the measured data; the red lines are the simulated data with all mentioned uncertainties; the yellow lines are the reference value. In Figure a, the PI control is displayed. In Figure b, the exact linearization is plotted. In Figure c, Backstepping can be seen and in Figure d, adaptive backstepping can be seen.

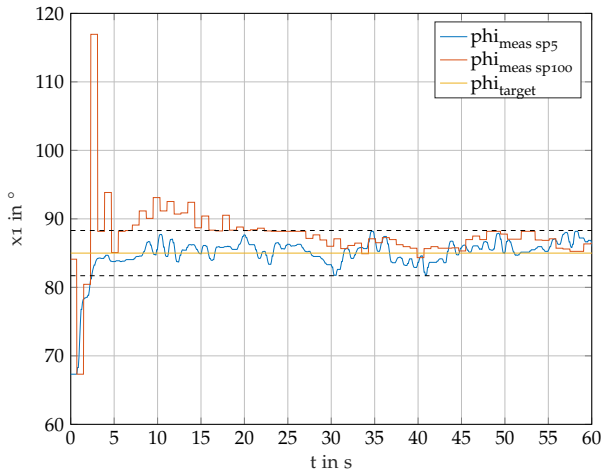

Figure 5.9.: Comparison between the two cases of exact linearization at an angle of 85°. In this figure, the measured values and the reference value of the control are shown. The faster cascaded control with the sample times Tc_n=5ms and Tc_φ=100ms is the blue curve. The slower cascaded control with the sample times Tc_n=100ms and Tc_φ=800ms is the red curve. The reference value is marked as yellow.

value is marked yellow. The high exceeding of the slower case at the beginning can be explained by the varying time delay. The time delay that occurred is bigger than the time delay used for the n-control. Despite the varying time delay, the angle can be controlled after 25s in the range of $\pm 3.3°$.

Angles below 85° do not show good control behavior, which can be explained by model uncertainties. These deviations can be caused by simplifications (averaging, mathematical simplifications) during the modeling process and the uncertainties of the measurements. Throughout the adjustment process of the control parameters, the simulation without any disturbances revealed that, for a beginning oscillation at higher angles, the parameter k2 must be increased and the parameter k1 must be decreased to obtain a zero steady-state error of the angle φ. The mentioned increase and decrease lead to a more dynamic manipulated variable u. The simulation also shows that the set control parameters do not support the full range of motion of the arm. For example, control parameters that show good control behavior at higher angles will result in oscillation at lower angles. A

compromise must be found to cover a broad range.

5.3.3. Backstepping

Compared to the previous control concepts, the backstepping algorithm provides a better result. With this method, a control of the angle in the range between 80° to 90° can be achieved. The faster control loop has a more agile behavior, although it does not lead to a better result. This is due to the fact that, the speed controller outputs a new controlled variable 42 times without knowing the current state of the system. Therefore, the compromise of shortening the measurement evaluation time from 170ms to 100ms and setting the control cycle time to the same value proves to be effective. A similar result, with a higher measurement uncertainty and a slower control time, can thus be achieved.

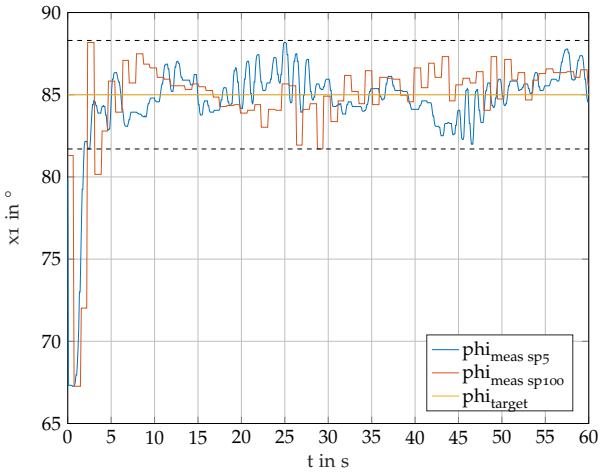

Figure 5.10.: Comparison between the two cases of backstepping at an angle of 85°. The fast cascaded control with the sample times Tc_n=5ms and Tc_φ=100ms is the blue curve. The slower cascaded control with the sample times Tc_n=100ms and Tc_φ=800ms is the red curve. The reference value is marked as yellow.

The measured data for the first (blue curve) and second case (red curve) are plotted in Figure 5.10. This figure shows the control behavior for an angle of 85°. The two cases show a similar result, which is in the range of $\pm$3.3° of the reference value r.

The simulation shows that larger angles than the tipping point can be controlled theoretically. For this purpose, parameter k_1 must be increased and parameter k_2 must be decreased. The resulting parameters for controlling an angle of 110° with the slower cascade control can be found with the simulation (k_1=10 and k_2=0.3). The simulation shows that with these parameters, the settling time for the reference value increases for angles below 110°. On the real model, this leads to oscillations due to the slow controller. This was also tested for the faster controller and led to the same result. This can be explained since the settling time of the n-controller is in the same range.

5.3.4. Adaptive backstepping

The best results can be achieved with adaptive backstepping. The four cases studied are the following:

1. The uncertain parameters c_1 and c_2 without coupling via the matrix Γ.
2. The uncertain parameters c_1 and c_2 with coupling via the matrix Γ.
3. The uncertain parameters c_1, c_2 and c_3 without coupling via the matrix Γ.
4. The uncertain parameters c_1, c_2 and c_3 with coupling via the matrix Γ.

The respective variant with the coupling of the uncertain parameters proves to be the better one. The reference values of 80°, 85° and 90° are reached in a tolerance band of$\pm$3°. At an angle of 75°, the settling time increases.

Considering the three system parameters c_1, c_2 and c_3 as uncertainties, shows the best result for angles greater than 75° when compared to the other applied control concepts. The comparison of the second and the fourth case for an angle of 85° is shown in Figure 5.11. The blue line is the measured signal of the angle for the second case. The red line is the measured data for the fourth case. This diagram shows that the controller for the fourth case reaches the settling point faster, than for the second case.

For an angle of 75°, the second case with the uncertain parameters c_1 and c_2 gives the best result.

In Figure 5.12, the first and second case for a reference angle of 90° is shown. This figure includes the measured data for the cascaded control with

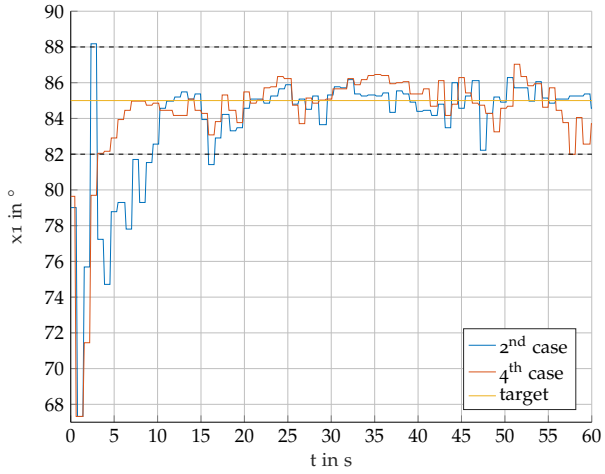

Figure 5.11.: Comparison of case two and four of adaptive backstepping for a reference angle of 85°. The blue line is the measured signal of the angle for the second case. The red line is the measured data for the fourth case. The yellow line is the reference value of 85°.

the uncertain parameters c_1 and c_2. The sample times are Tc_n=100ms and Tc_φ=800ms. The blue line is the measured data from adaptive backstepping with not coupled uncertain parameters. The red line is the coupled case. The yellow line is the reference value of the angle. In Figure 5.12a, the measured response of the control is displayed. In Figure 5.12b, the change of the uncertain parameter c_1 is shown. In Figure 5.12c, the change of the uncertain parameter c_2 is shown. Due to the occurring disturbances in the control loop, the adapted uncertainties do not reach a constant value. This can be seen in Figure 5.12, plot b and c. The influence of the coupling is also visible. The main change between these cases can be seen in diagram c, where the uncertain parameter c_2 changes more with coupling.

The initial values of the uncertainties have a big influence on the settling time. If the uncertainty is in the range of the expected final value, the reference value is reached faster. This behavior is shown in Figure 5.13. The blue curve has the starting values for both uncertainties at $\hat{\Theta}_0 = \begin{bmatrix} 0 & 0 \end{bmatrix}^T$. The red curve has the starting values for the uncertainties at $\hat{\Theta}_0 = \begin{bmatrix} c_1 & c_2 \end{bmatrix}^T$. The yellow curve shows the behavior for the starting values $\hat{\Theta}_0 = \begin{bmatrix} c_1 - 5 & c_2 \end{bmatrix}^T$. If the uncertain values are to close to the final value, this results in an

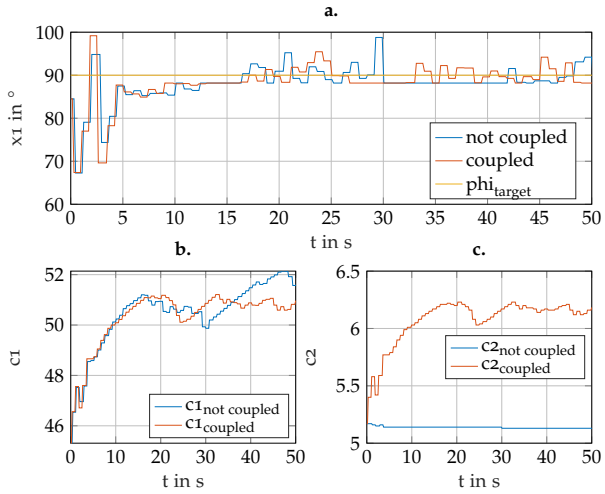

Figure 5.12.: Comparison between the first and second case for the slower cascaded control at an angle of 90°. This figure includes the measured data for the cascaded control with the uncertain parameters c_1 and c_2. The sample times are Tc_n=100ms and Tc_φ=800ms. The blue line is the measured data from adaptive backstepping with not coupled uncertain parameters. The red line is the coupled case. The yellow line is the reference value of the angle. In Figure a, the measured response of the control is displayed. In Figure b, the change of the uncertain parameter c_1 is shown. In Figure c, the change of the uncertain parameter c_2 is shown.

overshoot of the controlled output (red line).

The matrix Γ and the control parameter k_1 have a direct influence on the rate of change of the uncertainties. This can be seen from the law of adaptation (Equation 3.57), when $z_2 = x_2 + k_1 z_1$ is used.

$$\dot{\hat{\Theta}} = (x_2 + k_1 z_1) \cdot \Gamma \cdot W^T \tag{5.2}$$

The control parameter k_2 is used to set the final behavior of the control loop. This parameter can increase or decrease the influence of the error z_1 on the control behavior. This can be seen from Equation 3.58, if $z_2 = x_2 + k_1 z_1$ is used.

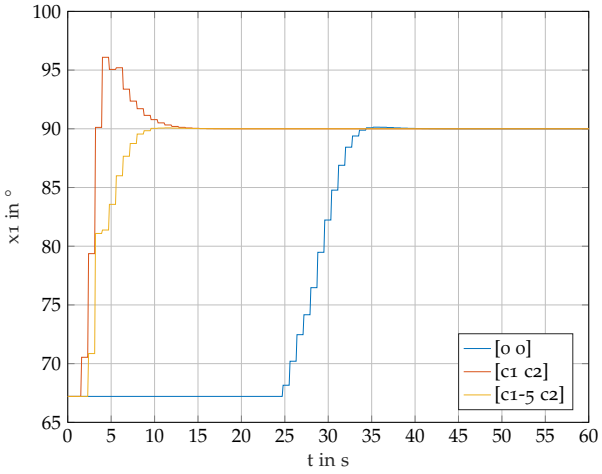

Figure 5.13.: Behavior of the adaptive backstepping for different starting values of the uncertainties. The blue curve has the starting values for both uncertainties at $\hat{\Theta}_0 = \begin{bmatrix} 0 & 0 \end{bmatrix}^T$. The red curve has the starting values for the uncertainties at $\hat{\Theta}_0 = \begin{bmatrix} c_1 & c_2 \end{bmatrix}^T$]. The yellow curve shows the behavior for the starting values $\hat{\Theta}_0 = \begin{bmatrix} c_1 - 5 & c_2 \end{bmatrix}^T$.

$$u = \frac{1}{c_3 \cdot ge(x_1)}(-k_2 \cdot (x_2 + k_1 z_1) - z_1 - k_1 \cdot x_2 - W \cdot \hat{\Theta})$$

For the case that k_2 is smaller than one, the influence of the error is reduced. On the other hand, for k_2 bigger than one, the influence of the error is increased.

5.4. Conclusion

In this thesis different model based control concepts applied to the Fan Driven Pendulum (FDP) are compared.

The results of the exact linearization show that the used model has some parameter uncertainties and, therefore, a robust algorithm able for coping with uncertainties was used for controlling the pendulum. The best results for controlling the angle could be achieved with the adaptive backstepping

algorithm. With this algorithm, uncertain parameters are adapted during the control process.

In Figure 5.14, the non-linear control concepts for the four reference values (75°=a, 80°=b, 85°=c and 90°=d) are presented. The lines represent the achieved position signal for the corresponding control algorithm. The blue line represents the exact linearization; the red line represents backstepping; the yellow line represents adaptive backstepping with the uncertain parameters c_1 and c_2; the purple line represents adaptive backstepping with the uncertain parameters c_1, c_2 and c_3; and the green line is the reference value. In these plots, it can be seen that the yellow line (adaptive backstepping with the uncertain parameters c_1 and c_2) reaches all four reference values within an error band of $\pm 3°$ with some outliers.

From the investigation of the slow (Tc_n=100ms, Tc_φ=800ms) and fast (Tc_n=5ms, Tc_φ=100ms) cascaded control, it was not possible to detect any advantage of the faster cascaded control, nor a disadvantage of the slower control. This can be explained by a similar settling time of the speed signal of the fan in closed loop operation, for both cases of 2 seconds. To achieve a satisfying control behavior, the inner control loop needs to reach the settling point before a new reference value is sent from the outer control loop. In the case of the laboratory setup of the pendulum it was not possible, due to its used components. Due to the slow settling time of the speed controller, it is also not possible to control angles greater than 105°. In order to achieve a better behavior of the angular control, the settling time of the n-control must be shortened. To achieve a faster control, the range of the PWM-signal can be increased by using a power supply that is able to provide enough current and voltage for the BLDC. Furthermore, the accuracy and the speed to measure the motor speed should be increased. For this purpose there are BLDCs with integrated speed measurement, which is carried out with Hall sensors.

Furthermore, the occurring temperature behavior was not considered in the mathematical model, because all experiments were recorded in the warm state.

The MCU should be exchanged to a Matlab Simulink compatible device, in order that the whole program can be implemented in Simulink. This reduces

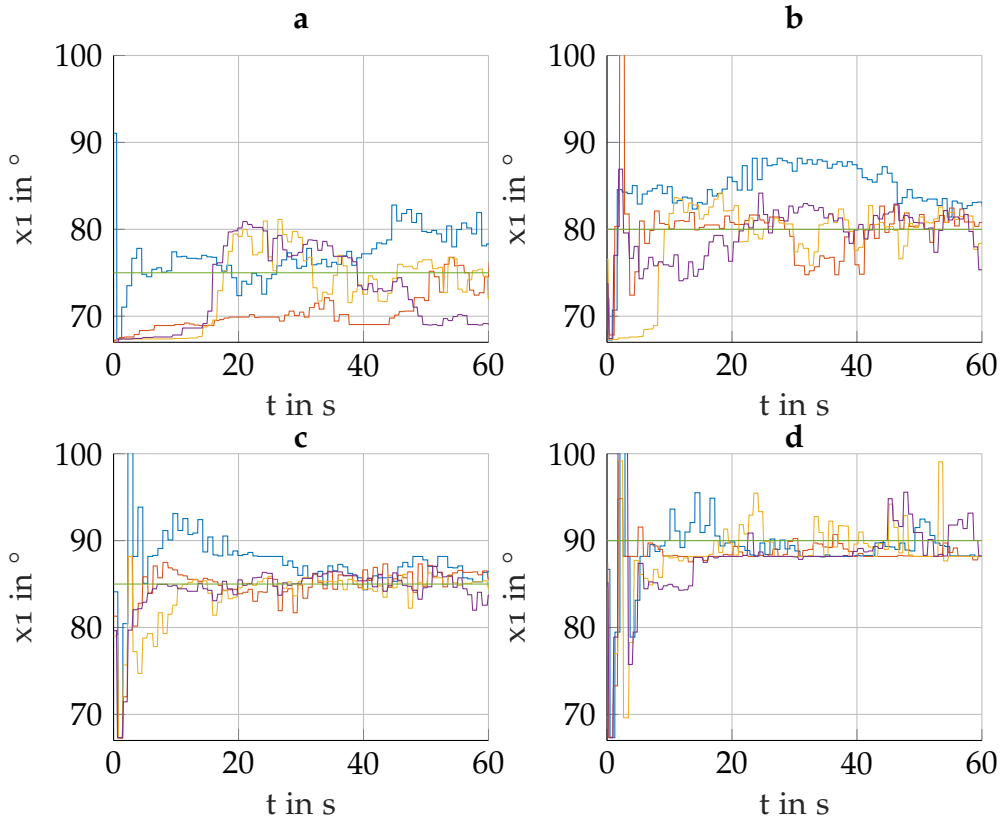

Figure 5.14.: Comparison of all algorithms for the angular control. The lines represent the achieved position signal for the used control algorithm. The blue line is the result for the exact linearization; the red line is the result for backstepping; the yellow line is the result for adaptive backstepping with the uncertain parameters c_1 and c_2; the purple line is the result for adaptive backstepping with the uncertain parameters c_1, c_2 and c_3; and the green line is the reference value.

errors which, can occur during the migration process form the generated code.

In order to use the fan actuated pendulum for further research and also as a laboratory model some changes are suggested in Section 5.5.

5.5. Outlook

Some components must be replaced in order to use the Fan Driven Pendulum (FDP) as a reliable laboratory model. This means that the microcontroller should be Simulink compatible and fully programmable in Simulink. This eliminates all errors, that can occur during the migration process. Furthermore, control parameters can be changed in a more efficient way. Some alternatives are:

- Arduino
- STM32
- TI C2000
- TI C6000

For this mentioned types of MCUs, support packages are available for Simulink. The code for the project is automatically generated in Simulink with the embedded coder. It is necessary to have an Integrated Development Environment (IDE), to build the binary file for the MCU. For instance, for the Texas Instruments (TI) controllers, the *TI Code Composer Studio* is an option.

For the extension of the PWM-signal range and to supply the BLDC with sufficient current, a power supply unit that can provide peak currents up to 30A, should be used.

Furthermore, the measurement of the angle needs to be adjusted. Due to the backlash of the indirect measurement, an inaccuracy of 1.3° by a direction change of the arm movement occurs. In a modified setup the backlash needs to be reduced, to stay with the indirect measurement and the hall sensor. An other option is to use an incremental encoder for a direct angular measurement.

The occurring varying time delay and the temperature dependency of the model is probably caused by the used ESC. The varying time delay can be explained due to the tolerances of the used transistors in the converter. By exchanging the ESC for a more precise ESC, the variety of the time delay can be reduced. There are several providers of ESCs, like for example: NXP and Infineon.

To get a more precise measurement of the motor's speed, it is suggested to exchange the motor for a BLDC with internal speed measurement (for this reason hall sensors are used). This BLDC can also have a lower rate of change than the used one. This leads to an increase of the range of the used PWM-signal, due to the higher supply voltage needed by the BLDC. An other alternative would be an external sensor that can measure speeds of up to 10500RPM of the propeller. For this case the existing BLDC could be still used.

Appendix A.

Model

In this part of the Appendix, additional information regarding the model is provided.

A.1. Model components

The following components are used in the FDP:

- Micro-controller NXP (LPC176)
- Mbed development board (C12832)
- BLDC-motor (a2212/6t 2200kV)
- Propeller (APC E 6055)
- Motorcontroller ESC
- Currentsensor (ACS712-ELCTR-05B-T)
- Fan speed measurement:
  - Infrared sensor plus comparator (H2010, LM393)
  - Phasesensor (BXA76013-BEASTX)
- Angular measurement Hall sensor
- Power supply (12V, 5A)

A.2. Mathematical model constants

In this section the mathematical model constants, for the calculation of the model, are given in the table below.

Symbol	Value	Unit
c_p	0.0498	
d_p	0.1524	m
g	9.81	$\frac{m}{s^2}$
h_a	0.23	m
H	0.1397	m
k_R	1.216	
k	0.8568	
l_a	0.38	m
l_p	0.288	m
l_s	0.195	m
m	0.235	kg
r	0.0762	m
ρ_{20}	1.2041	$\frac{kg}{m^3}$
ϑ	15	°

Table A.1.: Constant values for the calculation of the model parameters of the FDP.

The parameters of Table A.1 are used for the calculation of the differential equation of the system description (Equation 2.22). To calculate parameter c_1 the following parameters are needed: the gravitational acceleration g and the distance l_s, to the center of gravity. For the parameter c_2, the mass m of the pendulum arm and the friction coefficient k_R are needed. The constant values of the power factor c_p, the diameter of the propeller d_p, the pitch H of the propeller and the air density at 20°C ρ_{20}, are used in Equation 2.5 to calculate the thrust F_p generated by the propeller. The values relevant for the calculation of the ground effect ge(x_1) are, the height of the pivot point h_a, the length of the pendulum arm l_a and the radius r of the propeller. Furthermore, the distance form the pivot point to the center of gravity l_s, the mounted distance l_p of the BLDC and the mass m of the pendulum arm, are needed for the calculation of parameter c_3.

Appendix B.

Controller

In this part additional information needed for the controller implementation is provided.

B.1. Tables

This section provides the adjustment rules from Ziegler Nichols (ZN) and Chien, Hrones, Reswick (CHR) for the PI- controller used in Section 4.1.

Voraussetzung	Regler	Reglerparameter
Approximation der Regelstrecke durch ein PT_1T_t-Glied	P	$k_P = \frac{1}{k_s}\frac{T}{T_t}$
	PI	$k_P = \frac{0,9}{k_s}\frac{T}{T_t},\ T_I = 3,33\,T_t$
	PID	$k_P = \frac{1,2}{k_s}\frac{T}{T_t},\ T_I = 2\,T_t,\ T_D = 0,5\,T_t$
Kritische Verstärkung und Periodendauer sind bekannt	P	$k_P = 0,5\,k_{krit}$
	PI	$k_P = 0,45\,k_{krit},\ T_I = 0,85\,T_{krit}$
	PID	$k_P = 0,6\,k_{krit},\ T_I = 0,5\,T_{krit},\ T_D = 0,12\,T_{krit}$

Figure B.1.: Ziegler Nichols adjustment rules, Source [9, P. 470]

Reglereinstellung nach *Chien, Hrones , Reswick*		Aperiodischer Regelverlauf		Regelverlauf mit 20% Überschwingung	
Regler	Parameter	Führung	Störung	Führung	Störung
P	$K_{PR} K_{PS} \frac{T_u}{T_g}$	0,3	0,3	0,7	0,7
PI	$K_{PR} K_{PS} \frac{T_u}{T_g}$	0,35	0,6	0,6	0,7
	T_n	$1{,}2 \cdot T_g$	$4 \cdot T_u$	$1{,}0 \cdot T_g$	$2{,}3 \cdot T_u$
PID	$K_{PR} K_{PS} \frac{T_u}{T_g}$	0,6	0,95	0,95	1,2
	T_n	$1{,}0 \cdot T_g$	$2{,}4 \cdot T_u$	$1{,}35 \cdot T_g$	$2{,}0 \cdot T_u$
	$\frac{T_v}{T_u}$	0,5	0,42	0,47	0,42

Figure B.2.: Chien, Hrones, Reswick Parameter adjustment rules, Source [19, P. 227]

B.2. Matlab files

The following files contain measured data of the n-control that is used for the evaluation. The corresponding control parameters can be found in Table 4.1, except for sp_opt_tc_5.mat the control parameters are $k_i = 0.00256$ and $k_p = 0.00016$.

- pi_chr20_tc_5.mat
- pi_chr20_tc_100.mat
- pi_chra_tc_5.mat
- pi_chra_tc_100.mat
- pi_zn_tc_5.mat
- pi_zn_tc_100.mat
- pi_opt_tc_5.mat
- pi_opt_tc_100.mat
- sp_chr20_tc_5.mat
- sp_chr20_tc_100.mat
- sp_chra_tc_5.mat
- sp_chra_tc_100.mat
- sp_zn_tc_5.mat
- sp_zn_tc_100.mat
- sp_opt_tc_5.mat
- sp_opt_tc_100.mat

For each measured file exists an corresponding simulation file.

The used files for the evaluation of the angular control are given below. The corresponding control parameters can be found in Section 4.2. sp5 indicates the faster cascaded control with the sample times Tc_n=5ms and

Tc_φ=100ms. sp100 indicates the slower cascaded control with the sample times Tc_n=100ms and Tc_φ=800ms.

- pi_control_kp20_ki22_sp5.mat
- pi_control_kp35_ki25_sp100_2.mat
- exact_lin_k1_15_k2_7_sp5_var2_2.mat
- exact_lin_k1_17_k2_5_sp100_var2_2.mat
- Backstepping__k1_4_kbs_3_1_sp5_var2_2.mat
- Backstepping__k1_5_9_kbs_3_2_sp100_var2_2.mat
- adaptive_Backstepping_c1_c2_1_sp5_2.mat
- adaptive_Backstepping_c1_c2_1_sp100_2.mat
- adaptive_Backstepping_c1_c2_1_sp5_link_2.mat
- adaptive_Backstepping_c1_c2_2_sp100_link_2.mat
- adaptive_Backstepping_c1_c2_c3_1_sp5_2.mat
- adaptive_Backstepping_c1_c2_c3_1_sp100_2.mat
- adaptive_Backstepping_c1_c2_c3_2_sp5_link_2.mat
- adaptive_Backstepping_c1_c2_c3_1_sp100_link_2.mat

For each measured file exists an corresponding simulation file with the mentioned uncertainties.

B.3. Embedded Coder

For the code generation of the implemented controller in Matlab, the embedded coder is used. For this reason the discrete controller is copied in a new file and the following steps are performed:

1. Open the *Model Configuration Parameters* in Simulink. In the menu *Solver*, in the field *Solver selection*, the type needs to be changed to *Fixed-step* (with the sample time of the controller) and the solver is set to *discrete (no continuous states)*.
2. In the menu *Data Import/Export*, make sure that in the Section *Save to workspace or file* nothing is selected.
3. In the menu *Hardware Implementation*, select the corresponding hardware. For the used MCU select under *Device Vendor*, NXP and under *Device-Type*, Cortex-M3.

4. In the menu *Code Generation* the final adjustments need to be made:
    - In the field *Target Selection*, the *System Target File* needs to be changed to *ert.tlc Embedded Coder* and the *Language* can be selected between *C* and *C++*.
    - In the field *Build Process* select *Generate code only* and *Package code and artifacts*. Furthermore the *Building configuration* can be selected. For the generated code, *Faster Runs*, was selected.
    - In the submenu *Optimize* the *Default parameter behavior* can be selected between *Inlined* or *Tuneable*. Due to the necessary change of the control parameters in the adjustment phase, *Tuneable* was selected.

After the above steps, the model can be saved and build. The generated code can be migrated to Mbed.

B.4. Communication with the MCU

In this section, how to establish the connection between the MCU and the PC is shown. This is shown for the two used programs, Matlab and Coolterm.

To establish a connection with CoolTerm the following steps need to be performed. After opening CoolTerm the serial port needs to be configured in the options. For this reason in the *Serial Port Options* the *Baudrate* needs to be changed to 115200 and the *Parity* is set to odd. The *Terminal Mode* needs to be changed to *Line Mode* to be able to send and receive data. With a click on *Connect* the serial connection is established and the serial port terminal can be used to send reference values to the MCU.

The following code shows how the serial connection is set up with a Matlab script. This script was used, to exchange data between the PC and the MCU. The reference values and the format of the send reference value, were changed accordingly to the investigated controller (n-control or φ-control).

```
%clear up all before opening connection
clear all;close all;clc; delete(instrfind);

%% initialization (Baudrate, error checking)
BR=115200;
lpc1768=serial('COM4','Baudrate',BR);% for win use
lpc1768.Parity='odd';
lpc1768.Terminator='LF';
fopen(lpc1768); %opens serial connection
% set data to be send
n=170:1:173;
n_send=zeros(1,length(n)*2);
n_send(1:2:end)=n;

status_ser=lpc1768.status; % to check if something is wrong

for i=1:length(n_send)
    for j=1:400
        if(lpc1768.status=='open')% sends data
            fprintf(lpc1768,'n: %f\n',n_send(i),'async'); % asyncronu
        end
        if (lpc1768.status=='open') % receives data
            Data_in{j,i}=fscanf(lpc1768);
        else
            disp('Error')
        end
    end
end

fprintf(lpc1768,'n: %f\n',1,'async'); % turn of BLDC
file_name='Name';
save(file_name,'Data_in'); %Saves data to file
if (lpc1768.status=='open') %closes connection to mcu
    fclose(lpc1768);
    delete(lpc1768);
    clear lpc1768;
end
```

B.5. Flowchart of the implemented code

The flowchart shows the program structure of the implemented C++ code. The used functions are called in the main program by interrupts and timer interrupts. The interrupt to receive data, blocks all other interrupts until no more data is received. To detect the rising edge of the speed sensor an interrupt is used. The controllers are called by timer interrupts with the corresponding sampling time (Tc_n, Tc_φ). The evaluation of the speed is started by a timer with the time T_s. The lowest priority has to send data, which is send when the MCU has no other tasks to process. The minimum time that has to elapse between new data is send is set to T_{send}. The value of T_{send} was changed in accordance to the used controller.

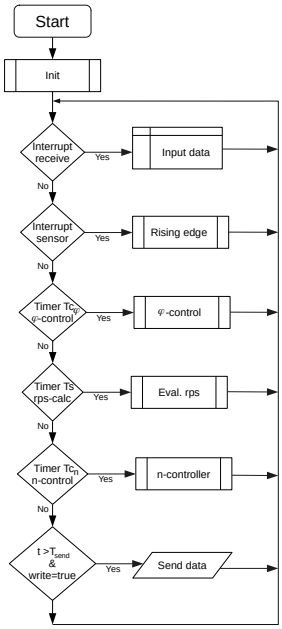

Figure B.3.: The flowchart of the implemented C++ program.

Bibliography

[1] R. A. Serway and J. John W. Jewett, *Physics for Scientists and Engineers with Modern Physics*. Brooks/Cole CENGAGE Learning, 2014, ISBN: 978-1-133-95405-7 (cit. on p. 1).

[2] V. Kurdekar and S. Borkar, "Inverted pendulum control: A brief overview," *International Journal of Modern Engineering Research (IJMER)*, vol. 3, no. 5, pp. 2924–2927, Sep. 2013 (cit. on p. 1).

[3] J. D. Steinmeyer and J. K. White, "Hardware as for teaching modeling and control," *Mechanical Engineering*, no. ME-16-JUN6, pp. 12–16, Jun. 2016 (cit. on p. 1).

[4] H. Kızmaz, S. Aksoy, and A. Mühürcü, "Sliding mode control of suspended pendulum," *Modern Electric Power Systems , Wroclaw, Poland*, 2010 (cit. on p. 1).

[5] T. Huba, T. Malatinec, and M. Huba, "Propeller-pendulum for non-linear uavs control," *International Journal of Online Engineering (iJOE)*, 2013 (cit. on p. 1).

[6] H. H. Khalil, *Nonlinear Systems*, 3rd ed. Prentice Hall, 2002, ISBN: 0-13-067389-7 (cit. on pp. 4, 24).

[7] H. Schenk, "Der standschub von propellern und rotoren," Jul. 2002 (cit. on p. 4).

[8] I. Cheeseman and W. E. Bennett, "The effect of the ground on a helicopter rotor in forward flight," *Aeronautical Research Council*, 1957 (cit. on p. 5).

[9] J. Lunze, *Regelungstechnik 1*, 10. Springer-Verlag Berlin Heidelberg, 2014, ISBN: 978-3-642-53908-4 (cit. on pp. 10, 72).

[10] K. Magnus, K. Popp, and W. Sextro, *Schwingungen*, 10th ed. Springer-Viewef, 2016, ISBN: 978-3-658-13821-9 (cit. on p. 13).

[11] M. Horn and N. Dourdoumas, *Regelungstechnik*. Pearson Studium, 2004, ISBN: 3-8273-7059-0 (cit. on p. 18).

[12] Z. Palmor, *The Control Handbook , Chapter Time delay compensation: Smith Predictor and its modificaions*. CRC-Press, 1996 (cit. on pp. 21, 23).

[13] F. S. S. de Oliveira, F. O. Souza, and R. M. Palhares, "Pid tuning for time-varying delay systems based on modified smith predictor," *IFAC PapersOnLine 50-1*, pp. 1269–1274, 2017 (cit. on p. 21).

[14] T. Hägglund, "An industrial dead-time compensating pi-controller," *Control Eng. Practice*, vol. 4, no. 6, pp. 749–756, 1996 (cit. on p. 24).

[15] J. Adamy, *Nichtlineare Systeme und Regelungen*, 2nd ed. Springer, 2014, ISBN: ISBN 978-3-642-45012-9 (cit. on pp. 24, 27, 28).

[16] M. Rios-Bolivar, "Adaptive backstepping and sliding mode control of uncertain nonlinear systems," Ph.D. dissertation, University of Sheffield, 1997 (cit. on pp. 30, 31).

[17] X. Zheng, P. Li, H. Li, and D. Ding, "Adaptive backstepping-based ntsm control for unmatched uncertain nonlinear systems," *Journal of Systems Engineering and Electronics*, vol. 26, no. 3, pp. 557–564, Jun. 2015 (cit. on p. 30).

[18] A. J. Koshkouei and A. S. I. Zinober, "Adaptive backstepping control of nonlinear systems withunmatched uncertainty," *Proceedings of the IEEE Conference on decision and Control*, 2000 (cit. on p. 30).

[19] S. Zacher and M. Reute, *Regelungstechnik für Ingenieure*. Springer, 2014, ISBN: 978-3-8348-2216-1 (cit. on p. 73).