

ÇANKAYA UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
ELECTRONIC AND COMMUNICATION SCIENCE

MASTER THESIS

DEVELOPMENT OF A DC-MOTOR CONTROL
LABORATORY EXPERIMENT

ARDM H.MOHAMMED ALI
KAHYA

NOVEMBER, 2012

**DEVELOPMENT OF A DC-MOTOR CONTROL
LABORATORY EXPERIMENT**

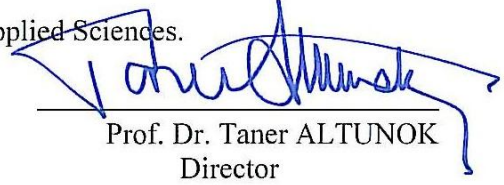
**ARDM H.MOHAMMED ALI
KAHYA**

NOVEMBER , 2012


Title of the Thesis : **DEVELOPMENT OF A DC-MOTOR CONTROL
LABORATORY EXPERIMENT**

Submitted By: **ARDM H.MOHAMMED ALI**

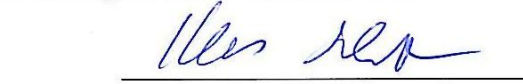
Approval of the Graduate School of Natural and Applied Sciences.


Prof. Dr. Taner ALTUNOK
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.


Prof. Dr. Celal Zaim ÇİL
Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.


Assoc. Prof. Dr. Klaus Werner SCHMIDT
Supervisor

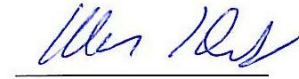
Examination Date : November 8, 2012

Examining Committee Members


Assist. Prof. Dr. Ulaş Beldek (Çankaya Univ.)



Assoc Prof. Dr. Klaus Werner SCHMIDT (Çankaya Univ.)



Buğra Sağlam (TAI)



STATEMENT OF NON-PLAGIARISM

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name : **ARDM H.MOHAMMED ALI**

Signature :



Date :

8/11/2012

ABSTRACT

DEVELOPMENT OF A DC-MOTOR CONTROL LABORATORY EXPERIMENT

ARDM H.MOHAMMEDALI

M.Sc., Department of Electronic and Communication Engineering

Supervisor: Assoc. Prof. Dr. Klaus Werner SCHMIDT

NOVEMBER 2012 52 Pages

Control laboratory experiments are usually expensive and do not allow much flexibility regarding controller designs and controller implementation. Because of this reason, the subject of this thesis is the development of a low-cost DC motor speed control experiment that can be used for various controller designs and different controller implementations. The thesis both describes the hardware components of the experiment and performs controller designs using the pole placement method, root locus method, symmetrical optimum and magnitude optimum method. In addition it is shown that control architectures such as disturbance forward can be used for the DC motor speed control. All controller designs are evaluated by simulations in Matlab/Simulink and are applied to the hardware experiment in the form of discrete-time control algorithms.

Keywords: DC motor, laboratory experiment, control system design, discrete-time control, microcontroller.

ÖZ

**DC-MOTOR KONTROL GELİŞİMİ
LABORATUVAR DENEY**

ARDM H.MOHAMMEDALI
KAHYA

Yüksek Lisans, Elektronik ve Haberleşme Bölümü
Tez Yöneticisi : Doç. Dr. Klaus Werner Schmidt

NOVEMBER 2012, 52 sayfa

Kontrol laboratuvar deneyleri genellikle pahalıdır ve denetleyici tasarımları ve denetleyici çok esnek izin vermezler. Bu nedenle, bu tezin konusu, çeşitli kontrol tasarımları ve farklı kontrol uygulamaları için kullanılacak düşük maliyetli bir DC motor hız kontrol, hem deney donanım bileşenleri hem de kutup yerleştirme yöntemi, kök yer eğrisi metodu, simetrisi uygun ve büyüklüğü uygun yöntemi denetleyici tasarımları. Buna ek olarak, ileri bozukluk gibi kontrol mimarilerinin DC motor hız kontrolü için kullanılacağı gösterilmiştir. Tüm kontrolör tasarımları Matlab / Simulink de ve kesikli zaman kontrol algoritmaları şeklinde donanım deneylerine uygulanmıştır.

Anahtar: DC motor, laboratuvar deney, kontrol sistemi tasarımı, ayrık zamanlı kontrol, mikrodenetleyici

ACKNOWLEDGEMENTS

In the name of Allah : The Most Gracious, the Most Merciful

I would like to express my profound gratitude to my supervisor, Assoc. Prof. Dr. Klaus Werner Schmidt , who has supported me throughout my thesis with his patience and knowledge.

Special thanks is also given to University of Cankaya for awarding me this opportunity to improve my knowledge and experiences.

Last but not least, my deepest gratitude goes to my beloved parents and my fabulous wife for their endless support, prayers and encouragement during my studies.

TABLE OF CONTENTS

ABSTRACT	iii
ÖZ	v
ACKNOWLEDGMENTS	vi
TABLE OF CONTENTS	vii
CHAPTERS	
INTRODUCTION	1
1 DESCRIPTION OF THE DC MOTOR SPEED CONTROL SYSTEM	3
1.1 OVERVIEW OF THE DC MOTOR CONTROL EXPERIMENT	3
1.2 SERVO DC MOTOR	5
1.3 DRIVER	6
1.4 ENCODER	7
1.5 MICROCONTROLLER (PIC16F877A)	8
1.6 SERIAL COMMUNICATION	10
1.7 SECOND MOTOR WITH LOAD	11
1.8 EXAMPLE MEASUREMENT	13
1.8.1 Pulse Measurmenet for Encoder	13
1.8.2 PWM Measurmenet and duty cycle	14
1.8.3 DC-Motor Step Response	16
1.8.4 DC-Motor Step Response with Load Generator	17
2 CONTROL METHODS FOR THE DC-MOTOR EXPERIMENT	18
2.1 DC MOTOR MODELING	18
2.2 FEEDBACK LOOP	21
2.3 POLE PLACEMENT	22
2.4 ROOT LOCUS	24

2.5	SYMMETRICAL OPTIMUM METHOD	24
2.6	MAGNITUDE OPTIMUM METHOD	25
2.7	DISTURBANCE FEED FORWARD	26
2.8	CONTINUOUS AND DISCRETE-TIME CONTROL	27
3	CONTROLLER DESIGN FOR THE DC-MOTOR EXPERIMENT . . .	29
3.1	DC MOTOR MODEL IDENTIFICATION	29
3.2	APPLICATION OF POLE PLACEMENT	32
3.3	APPLICATION OF ROOT LOCUS DESIGN	37
3.3.1	Proportional Control	37
3.3.2	PI Control	39
3.4	APPLICATION OF SYMMETRICAL OPTIMUM DESIGN . . .	40
3.5	APPLICATION OF MAGNITUDE OPTIMUM DESIGN	43
3.6	DISTURBANCE REJECTION	44
3.7	APPLICATION OF DISTURBANCE FEED-FORWARD CON- TROL	46
	CONCLUSIONS	49
	REFERENCES	51
	CURRICULUM VITAE	52

LIST OF TABLES

TABLES

Table 1.1	Voltage levels on RS232 and MAX232	10
Table 3.1	Parameters of $G_a(s)$	31

LIST OF FIGURES

FIGURES

Figure 1.1	Schematic diagram of the DC motor speed control system. . . .	4
Figure 1.2	Basic parts of a DC motor.	5
Figure 1.3	L293D pin description.	6
Figure 1.4	Encoder for the DC motor speed control system.	7
Figure 1.5	PIC16F877A pin layout	8
Figure 1.6	MAX232	10
Figure 1.7	DB-9 pin connector for RS232	11
Figure 1.8	DC Motor with load generator.	12
Figure 1.9	Block diagram including the DC motor with load generator. . . .	12
Figure 1.10	Encoder pulses measured by the PIC microcontroller.	13
Figure 1.11	Pulse width modulation.	14
Figure 1.12	Duty Cycle	15
Figure 1.13	Speed measurement	16
Figure 1.14	Step response with load generator.	17
Figure 2.1	DC motor with load momentum.	19
Figure 2.2	Block Diagram of the DC motor.	20
Figure 2.3	Transformed block diagram of the DC motor.	20
Figure 2.4	Feedback control loop.	21
Figure 2.5	Disturbance feedforward controller architecture.	27
Figure 3.1	Step response of the DC motor.	30
Figure 3.2	Comparsion of step response measurement and simulation	31

Figure 3.3	Pole placement control with slow closed-loop poles.	34
Figure 3.4	Pole placement control with fast closed-loop poles and $T = 0.05$	36
Figure 3.5	Pole placement control with fast closed-loop poles and $T = 0.02$	36
Figure 3.6	Root locus plot for proportional control.	37
Figure 3.7	Root locus design for proportional control and $T = 0.02$	38
Figure 3.8	Root locus plot for PI control.	39
Figure 3.9	Root locus design for PI control.	40
Figure 3.10	Symmetrical optimum design for $T = 0.01$	42
Figure 3.11	Symmetrical optimum design for $T = 0.02$	42
Figure 3.12	Magnitude optimum design for $T = 0.01$	44
Figure 3.13	Transformed block diagram of the DC motor	44
Figure 3.14	Disturbance step response measurement.	45
Figure 3.15	Disturbance step response measurement for pole placement control with slow poles.	46
Figure 3.16	Disturbance step response measurement for pole placement control with fast poles.	47
Figure 3.17	Application of disturbance feedforward.	48

INTRODUCTION

Education in control systems is compulsory in many engineering curricula such as Electrical and Electronics Engineering, Mechanical Engineering and Mechanical Engineering. In the related courses, the analysis and design of control systems is studied mostly on a theoretical level. Frequently, experimental control laboratories are only offered for a smaller number of students in the form of elective courses. In many cases, this is also due to the high cost and the limited usability of control equipment.

In view of the above discussion, the subject of this thesis is the development of a low-cost DC motor speed control laboratory experiment that can be used in compulsory control courses. The system is designed to support analytical modeling, which enables the use of a large variety of control design methods [5, 13, 4, 8, 14, 1]. In addition, the system is suitable for the application of disturbances in order to validate disturbance rejection of different controller design methods. The main components of the system are a DC motor with optical encoder, a load generator, a PIC 16F877A microcontroller and a L293D motor driver.

In the practical part of the thesis, the hardware setup is described in detail and sample measurements are taken for illustration. In the theoretical part of the thesis, first, an analytical model of the DC motor system is developed. Based on step response measurements of the DC motor system, the parameters of this model are identified for the later controller design. Various controller design methods are applied including pole placement [5], root locus design [13], symmetrical optimum and magnitude optimum [18]. All controller designs are accompanied by Matlab/Simulink simulations and a comparison to measurements of the real hardware setup. All controllers are converted to discrete-time control algorithms in order to support implementation on the PIC microcontroller. In addition, the disturbance feedforward architecture [5] is applied in order to make use of direct measurements of a load disturbance.

The organization of the thesis is as follows. In Chapter 1, the basic properties of

the DC-motor speed control system are described. In addition, this chapter provides details of the hardware setup used in this thesis. Regarding Chapter 2, the analytical model of the DC motor is developed and the methods that are used to design DC motor controllers are explained. Chapter 3 applies the described methods to the hardware setup and provides simulations as well as measurements from the DC motor experiment.

CHAPTER I

DESCRIPTION OF THE DC MOTOR SPEED CONTROL SYSTEM

In this chapter, the hardware setup for the DC motor control experiment is explained. Section 1.1 gives an overview of the experiment. The DC motor component is described in Section 1.2 and its driver circuit is given in Section 1.3. Section 1.4 elaborates the working principle of the speed encoder used in the experiment and in Section 1.5, the features of the microcontroller for the controller realization are outlined. The data exchange between the microcontroller and a PC is described in Section 1.6 and an extension of the DC motor experiment by a load generator is explained in Section 1.7. Finally, sample measurements using the different components of the experimental setup are shown in Section 1.8.

1.1 OVERVIEW OF THE DC MOTOR CONTROL EXPERIMENT

The DC motor experiment requires several electronic components in order to perform measurements, process measurement data and actuate the motor. This section gives an overview of the required components. A general overview is shown in Figure 1.1.

The DC motor speed depends on its input voltage. The higher the voltage, the higher the motor speed. The easiest and most power efficient way to adjust the input voltage is the usage of pulse width modulation (PWM) together with a motor driver circuit. This solution is also chosen in our experiment as can be seen in the figure. A driver circuit L293D is connected to the DC motor and receives a PWM input from a PIC microcontroller. In order to perform feedback control, an encoder

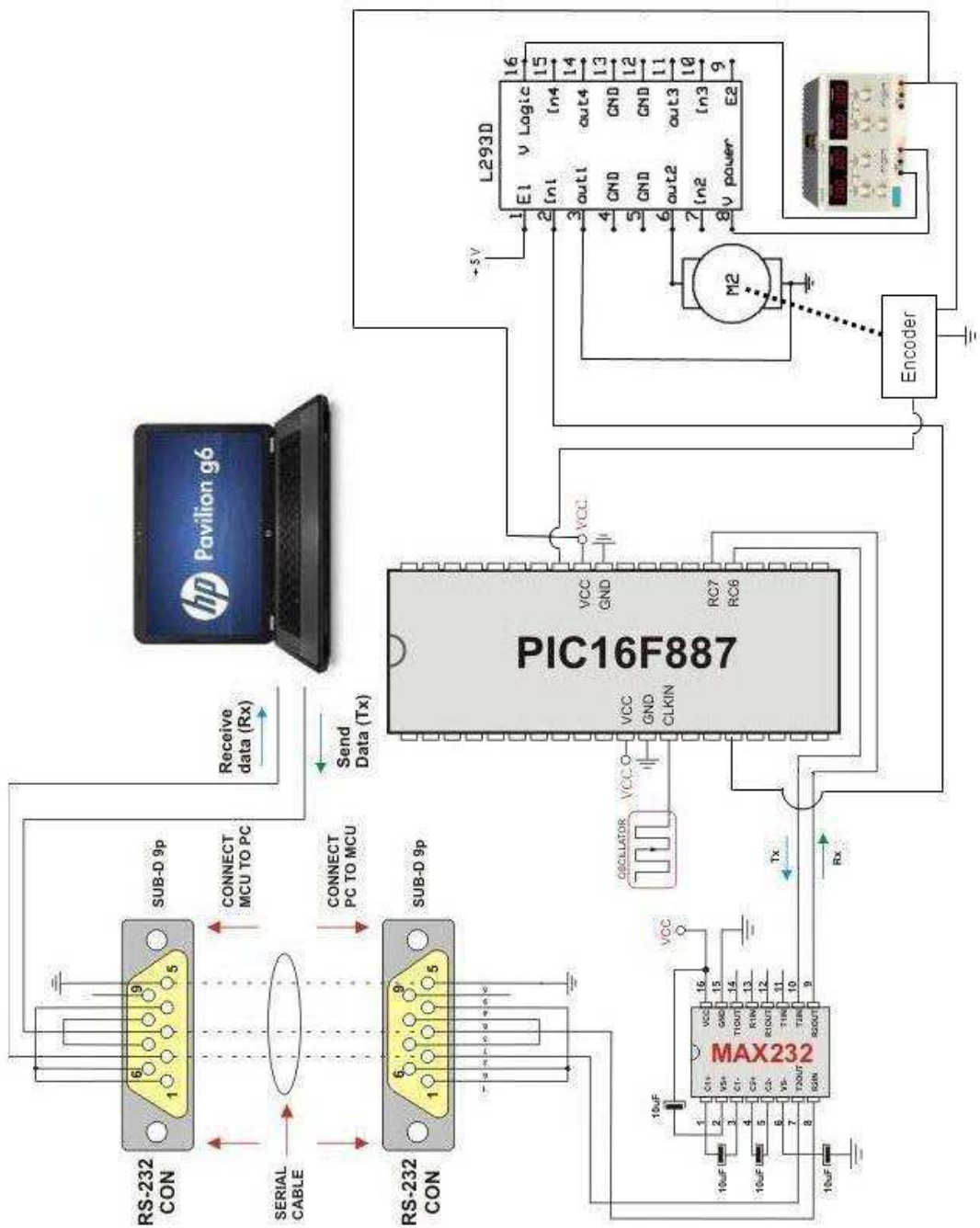


Figure 1.1: Schematic diagram of the DC motor speed control system.

sensor is used for measuring the DC motor speed. This encoder is also connected to the PIC microcontroller. The PIC microcontroller is mainly used to implement the control algorithm in the experiment. It receives the sensor data from the encoder and computes the characteristic values of the PWM signal for the motor driver. Finally, the PIC microcontroller is connected to a PC using serial communication via RS232. This enables the processing of measurement and control input data using the software Matlab.

1.2 SERVO DC MOTOR

The main function of each DC Motor is to take electrical energy as input and transform it into mechanical energy as an output through the interaction two magnetic fields. The basic parts of a DC motor are shown in Figure 1.2. There is (1) the magnets part (stator) that is considered as the non-turning part, (2) the rotating shaft (rotor) that is considered as the turning part, the rotor windings that produce a magnetic field depending on the supplied voltage.

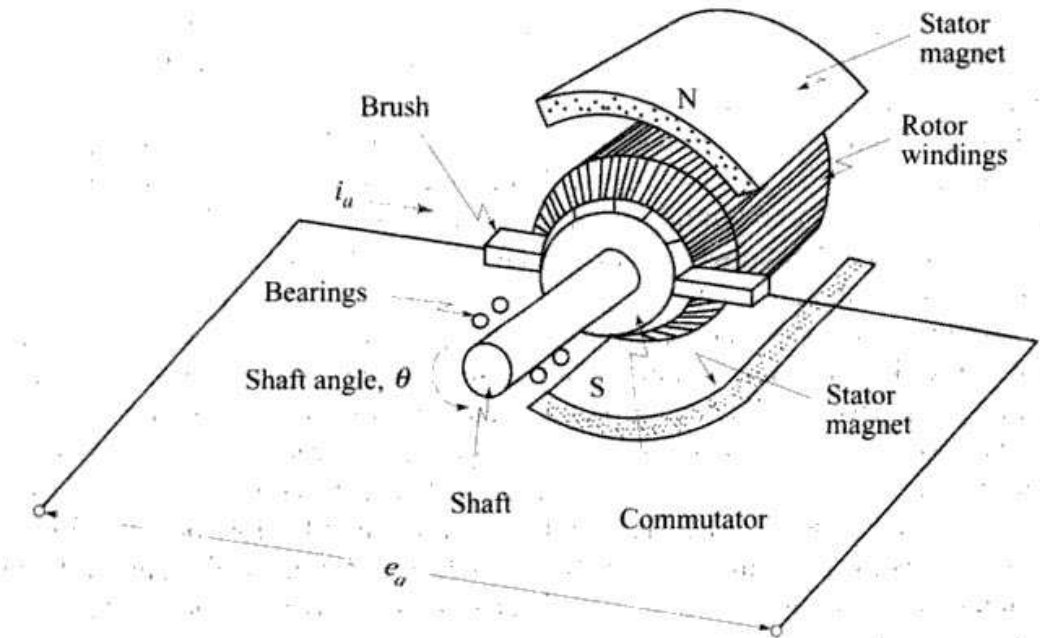


Figure 1.2: Basic parts of a DC motor.

1.3 DRIVER

The power for driving the DC motor is supplied by the driver circuit L293D [16]. It is a 16-pin DIP integrated circuit that is designed to provide output currents up to 600mA at voltages from 4.5 V to 36 V. The peak output current is 1.2A per channel. The L293D contains two built-in H-bridge driver circuits and can hence be used to drive two motors simultaneously.

Figure 1.3 illustrates the pin layout of the L293D driver. In order to drive a single DC motor, it is connected between pins OUTPUT1 and OUTPUT2. Then, the pins INPUT1 and INPUT2 can be used to determine the motion of the DC motor. It stops if both pins are high (5V) or low (0V). Otherwise, the DC motor turns in clockwise or counter-clockwise direction. Power for the DC motor is supplied via pin Vs (12V for our motor) and the logic voltage is supplied at pin Vss (5V). The driver circuit is activated if the pin Enable1 is 5V.

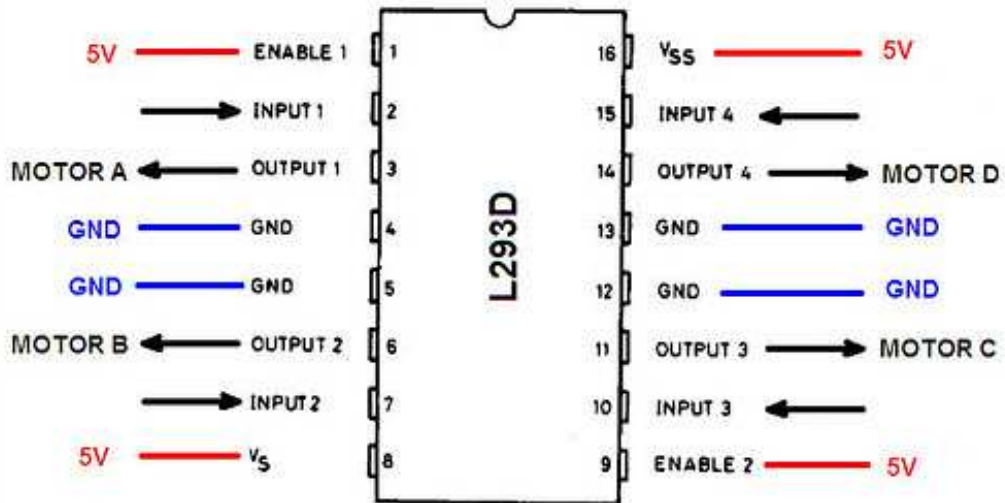


Figure 1.3: L293D pin description.

1.4 ENCODER

An encoder as shown in Figure 1.4 is used to measure the speed of the DC motor. The type of encoder is called rotary encoder because it is used to convert the rotary motion into electrical pulses. Such encoder is used in many practical applications such as industrial control, robotics, special purpose photographic lenses [3]. In our experiment, we use the encoder to measure the speed of the DC motor. It has 24 slots and when the encoder rotates, the photo sensor in the encoder counts the slots by generating voltage pulses at its output pins. With this operating principle, it is possible to determine the speed in rad/sec from the number of pulses that are counted per time unit. If we write N_T for the number of pulses per time unit T and ω for the rotational speed, we obtain the following equation.

$$\omega = \frac{N_T}{T}. \quad (1.1)$$

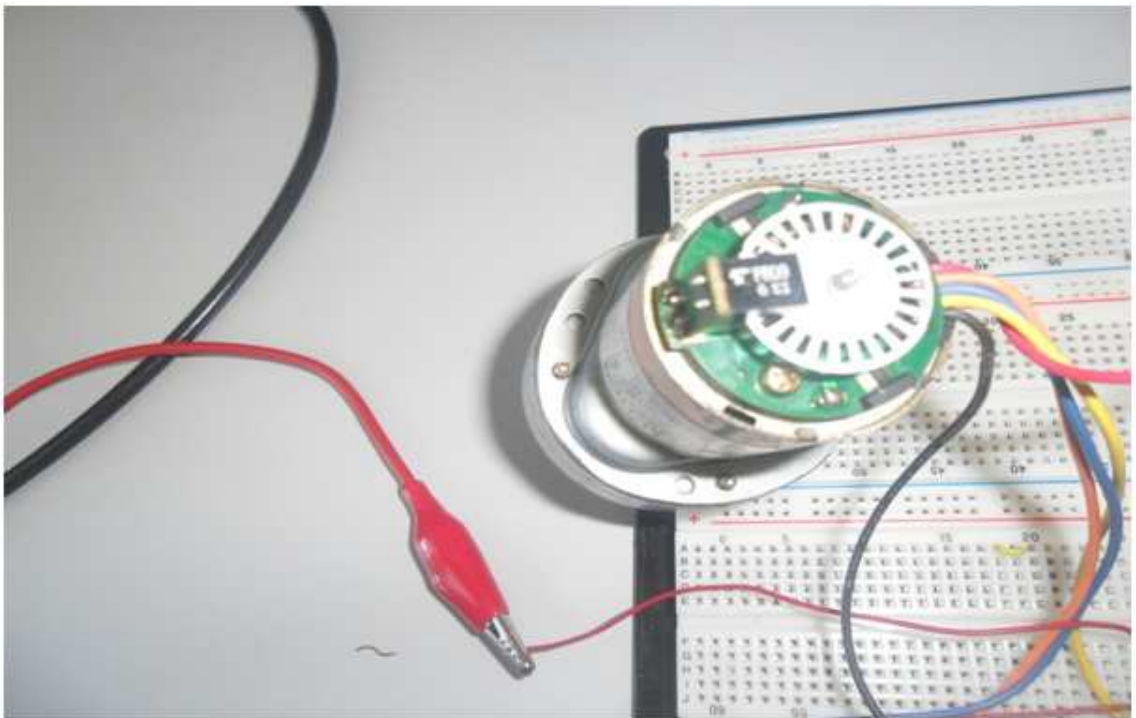


Figure 1.4: Encoder for the DC motor speed control system.

1.5 MICROCONTROLLER (PIC16F877A)

Microcontrollers are used today widely in embedded applications such as the modern car (antilock braking System (ABS), central locking, electrical mirror and seat adjustments, etc) toys, telephones, household appliances as well as peripherals for computer systems. Basically, microcontrollers integrate the fundamental resources available in a microprocessor system such as the CPU, memory, and I/O resources in a single chip. In our thesis we use the Microcontroller PIC 16F877A [11] that can be considered as one of most popular models of the PICMicro family microcontrollers. The PIC 16F877A have 40 pins as show in Figure 1.5 but it is not necessary to use all the pins of pic16f877a to control the speed of the DC motor.

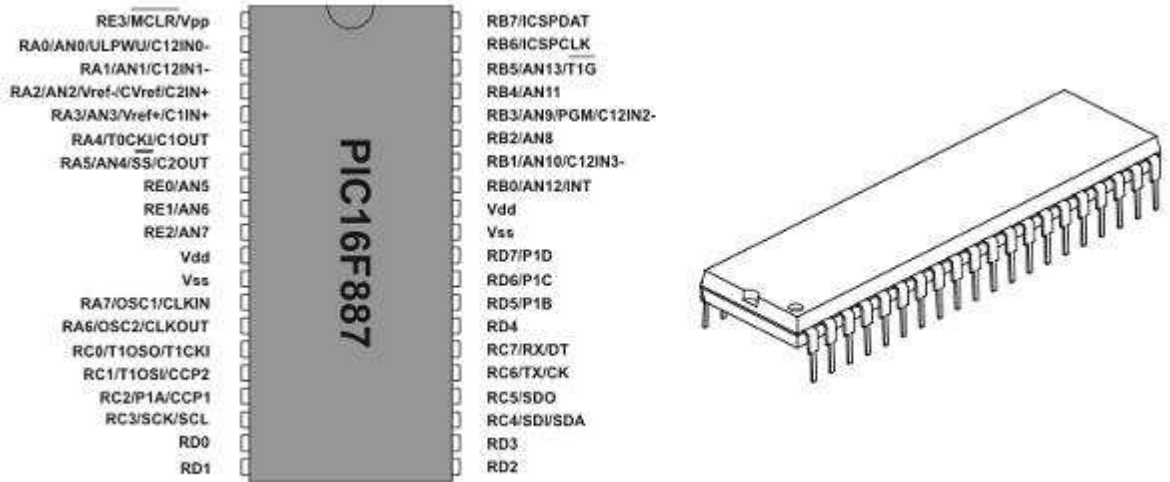


Figure 1.5: PIC16F877A pin layout

Besides the standard pins for logic voltage supply, ground and flash programming, our experiment uses further pins for PWM signal generation, interrupt measurement and serial communication. A PWM signal can be generated using the pin RC2/P1A/CCP1 of the PIC. Hence, this pin is connected to the input of the DC motor driver in order to perform speed control. The following code is used to set the duty cycle of the PWM signal.

```
PORTC = 0xff;          // PORTC is output
```

```

PWM2_Init(8000);    // Initialize with oscillator frequency 8MHz
PWM2_Start();      // Start the PWM routine
PWM2_Set_Duty(150); // Set the duty cycle to 150 (out of 255)

```

Pin RB0/AN12/INT is used to generate an interrupt whenever a pulse is detected at the pin. Hence, this pin is connected to the encoder in order to detect the encoder pulses. The following interrupt routine is implemented on the PIC to count pulses.

```

unsigned int count=0; // This count use in interrupt routine
void interrupt()     // enter when falling edge is detected
{
    count++;          // increment counter
    INTCON.INTF=0;   // clear interrupt flag
}

```

Pins RC7 and RC6 are used to perform serial communication. Because of that reason, they are connected to the driver circuit MAX232. In our experiment, measurement data and control input data are transmitted via this serial communication. The following code is used for this purpose.

```

char* input[7];
char* text[9];
char* output[7];
char $uart_rd$;
void main()
{
    PORTC = 0xff; // PORTC is output
    UART1_Init(9600); // Initialize serial communication 9600 baud
    intToStr(60,output); // output value 60
    UART1_Write_Text(output); // Transmit output value
    intToStr(150,input); // input value 150
    UART1_Write_Text(input); // Transmit input value
}

```

1.6 SERIAL COMMUNICATION

Measurement data are transmitted by the PIC microcontroller using pin RC7 and RC6 on a TTL voltage level. However, serial communication with RS232 requires a voltage level between -15 V and 15 V. The conversion of the voltage level is performed by a MAX232 integrated circuit [17]. The MAX232 is an integrated circuit that is mostly used to convert RS232 voltage level to be compatible with TTL level and vice versa as is shown in Table 1.1.

RS-232	TTL to/from MAX232	Logic
-15V ... -3V	+2V ... +5V	1
+3V ... +15V	0V ... +0.8V	0

Table 1.1: Voltage levels on RS232 and MAX232

The MAX232 contains two receivers (converts from RS-232 to TTL voltage levels) and two drivers (converts from TTL logic to RS232 voltage levels) as is show in Figure 1.6.

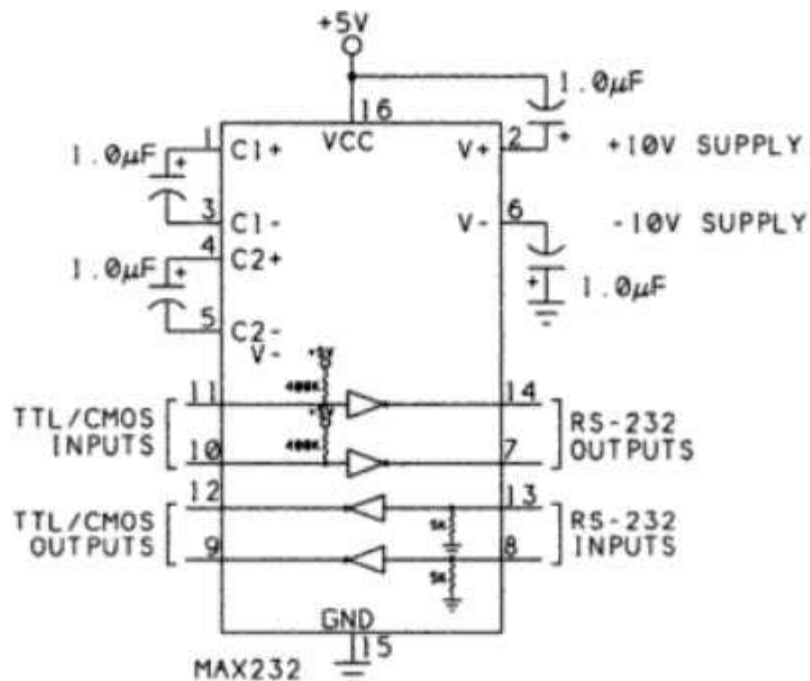


Figure 1.6: MAX232

In our experiment, we use pins 11 and 14 for the transmission of data from the

PIC to a PC. The connection between MAX232 and the PC is done by RS232. The RS232 serial interface communications standard has been widely used today to transfer data serially between two devices. The data is transmitted in one direction over two wires, **Tx** is labeled as data going out while **Rx** is labeled as data coming in. In addition, a third wire **GND** is needed as ground. Figure 1.7 illustrates pin description for the RS232 serial port (not all pins need to be used in our experiment).

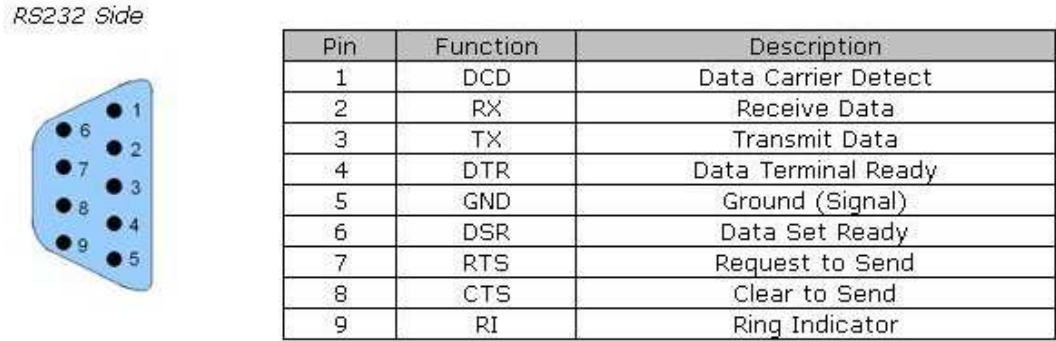


Figure 1.7: DB-9 pin connector for RS232

1.7 SECOND MOTOR WITH LOAD

One objective of our experiment is to achieve a constant DC motor speed even there is a load. We use a second DC motor that is operated in generator mode as load. It is mechanically connected to the first DC motor as is shown in Figure 1.8. A load resistance of 57Ω is chosen.

In addition, the following block diagram in figure 1.9 shows the interaction between the different components of the overall setup. In particular, the load can be switched on or off.

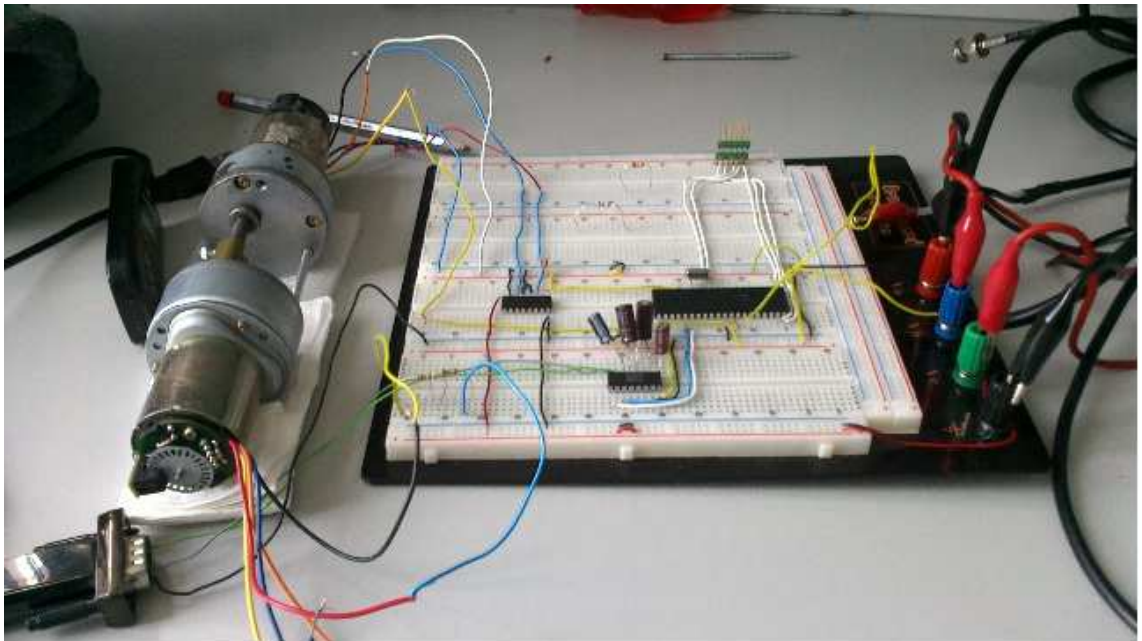


Figure 1.8: DC Motor with load generator.

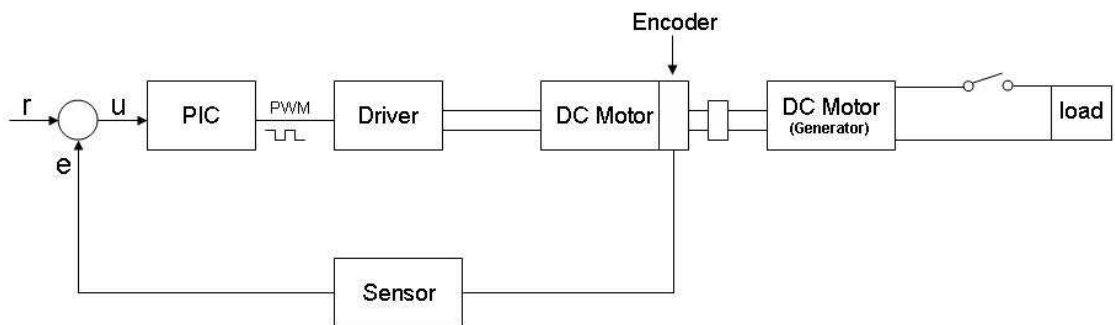


Figure 1.9: Block diagram including the DC motor with load generator.

1.8 EXAMPLE MEASUREMENT

This section shows different measurements from the experimental setup in order to illustrate the basic functionality of the DC motor experiment. Section 1.8.1 and 1.8.2 illustrate the functionality of the encoder and the PWM unit. Section 1.8.3 to 1.8.4 show different step responses. In these experiments, measurement data are transmitted from the PIC to a PC and then processed using Matlab.

1.8.1 Pulse Measurement for Encoder

Figure 1.10 shows a snapshot of the encoder output for a constant speed of the DC motor measured by an oscilloscope. Each pulse generates an interrupt on the PIC microcontroller as is explained in Section 1.5.

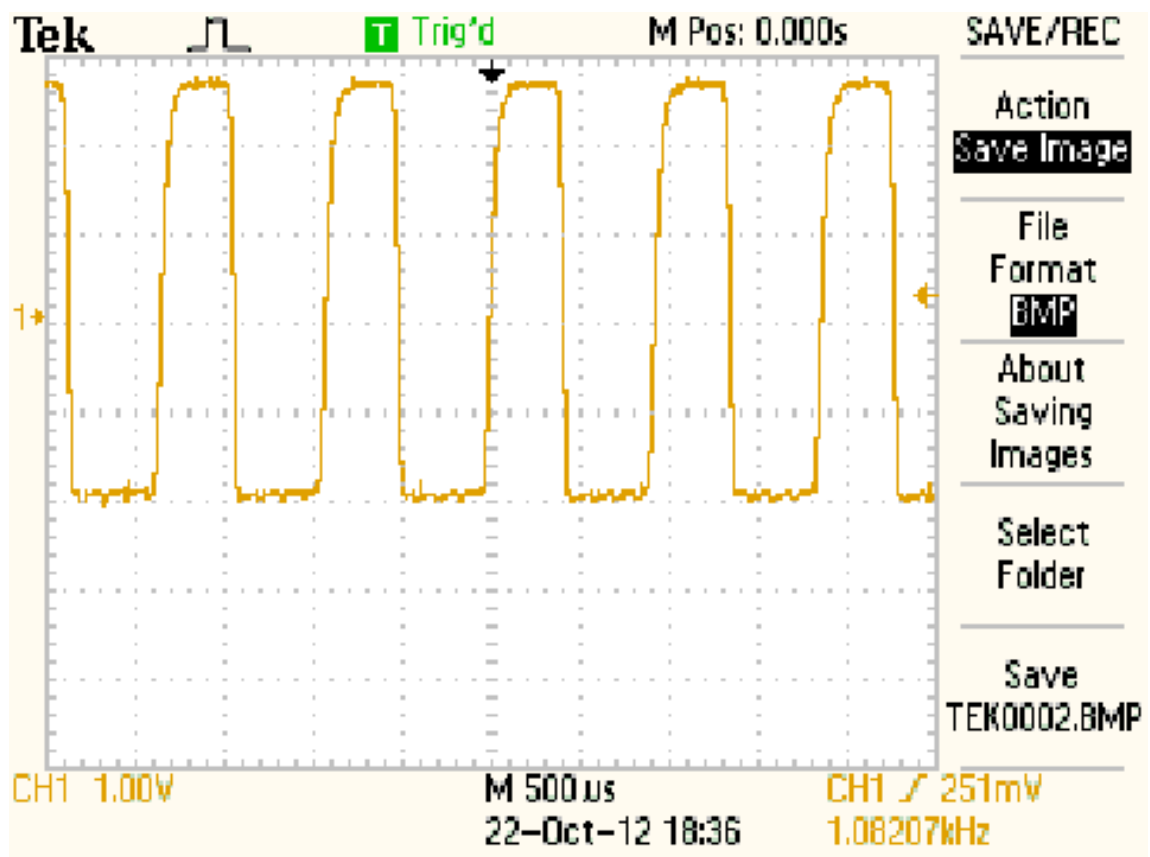


Figure 1.10: Encoder pulses measured by the PIC microcontroller.

1.8.2 PWM Measurement and duty cycle

The PIC microcontroller uses PWM in order to control the speed of the DC motor. That is, the signal given to the driver circuit is a square wave, whose duty cycle determines the level of the output voltage. Here, the term duty cycle refers to the percentage of one cycle (time T_c) during which the signal level is high (time T_{on}). Since the frequency is held constant while the on-off time is varied, the duty cycle of PWM is determined by the pulse width. The expression of duty cycle is determined by:

$$Duty\ cycle = \frac{T_{on}}{T_c} \times 100\% \quad (1.2)$$

Figure 1.12 illustrates the meaning of the duty cycle.

Figure 1.11 shows a PWM measurement from our experimental setup for a duty cycle value of 40%.

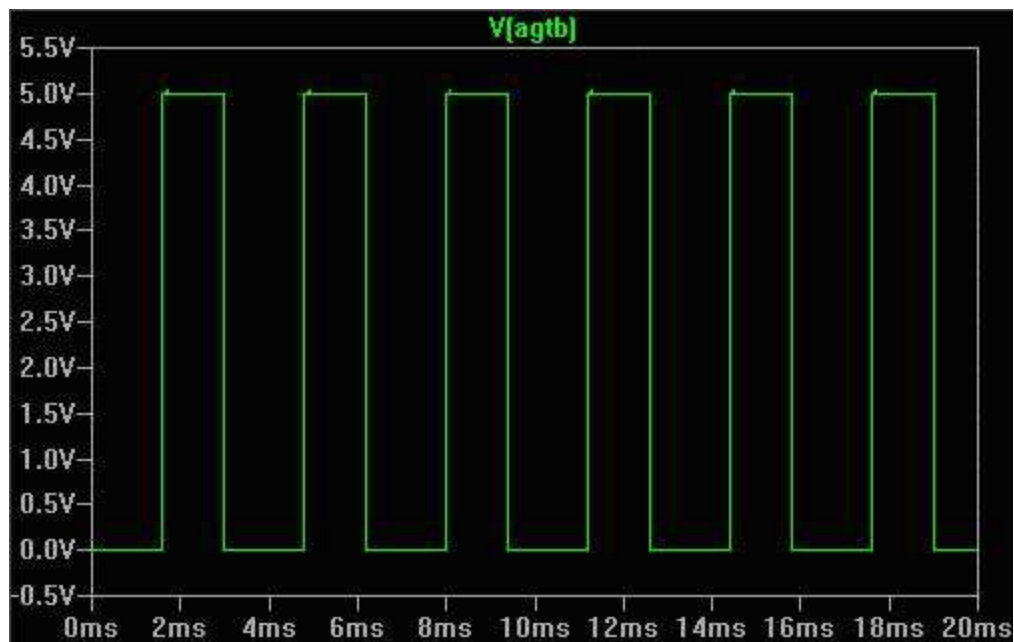


Figure 1.11: Pulse width modulation.

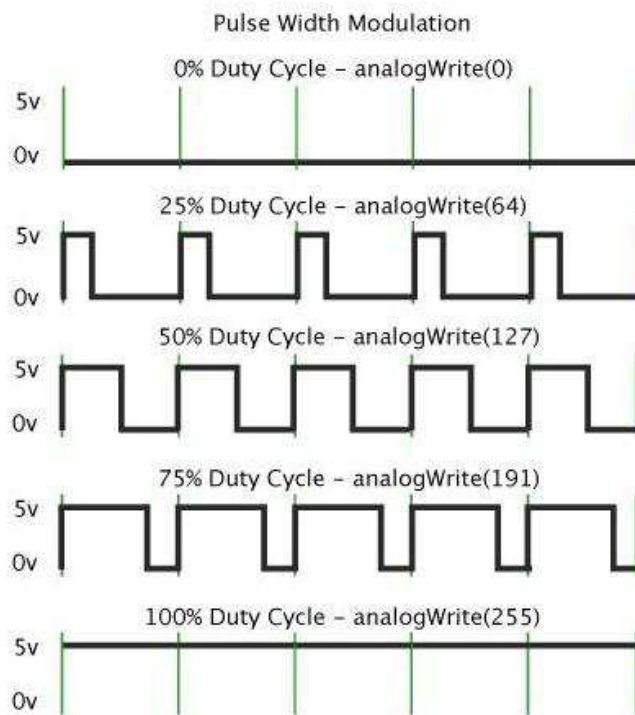
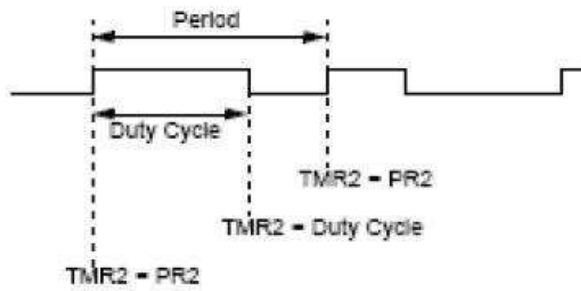


Figure 1.12: Duty Cycle

1.8.3 DC-Motor Step Response

In the first experiment, a step response of the single DC motor is recorded. To this end, the duty cycle of the PWM unit is set from 0 to 191 in the PIC program and the motor speed (in terms of counted pulses from the encoder) is measured every 50 ms. Figure 3.2 shows the resulting step response measurement. It can be seen

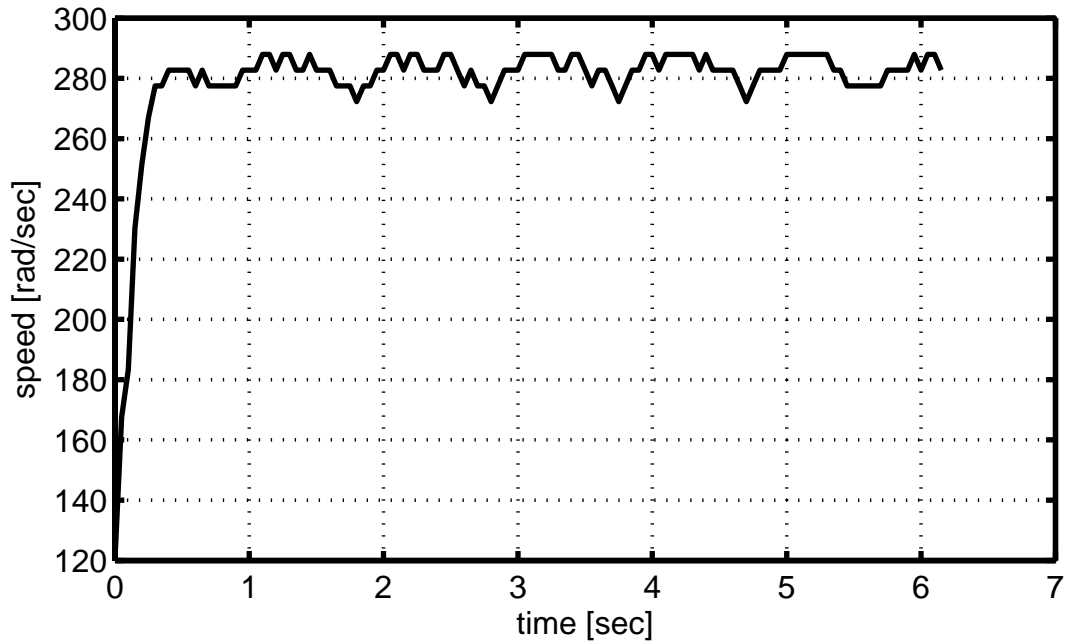


Figure 1.13: Speed measurement

from the figure how the speed of the DC Motor is effected by changing the duty cycle value. The speed increases to a value of 65 pulses per 50 ms after a short rise time of less than 0.5 sec. The speed can be converted to the classical representation of angular velocity ω in rad/sec by considering that one round (2π) corresponds to 24 pulses as described in Section 1.4. Denoting the number of pulses counted as p , we can write

$$\omega = \frac{p \cdot 2\pi}{24 \cdot 0.05 \text{ sec}} \quad (1.3)$$

Hence, 65 pulses per 50 ms corresponds to 340.3 rad/sec or $f = \frac{\omega}{2\pi} (\frac{rev}{sec})$.

1.8.4 DC-Motor Step Response with Load Generator

We next study the step response for the experimental setup with two DC motors, whereby the second motor works as generator. First a step of the duty cycle from 0 to 191 is performed and later (around time 10sec) a the load is switched on. Figure 1.14 shows that the speed increases to a value of around 54 rounds per 50 ms after the first step and decreases to about 43 rounds per 50 ms after the load step. 6mm

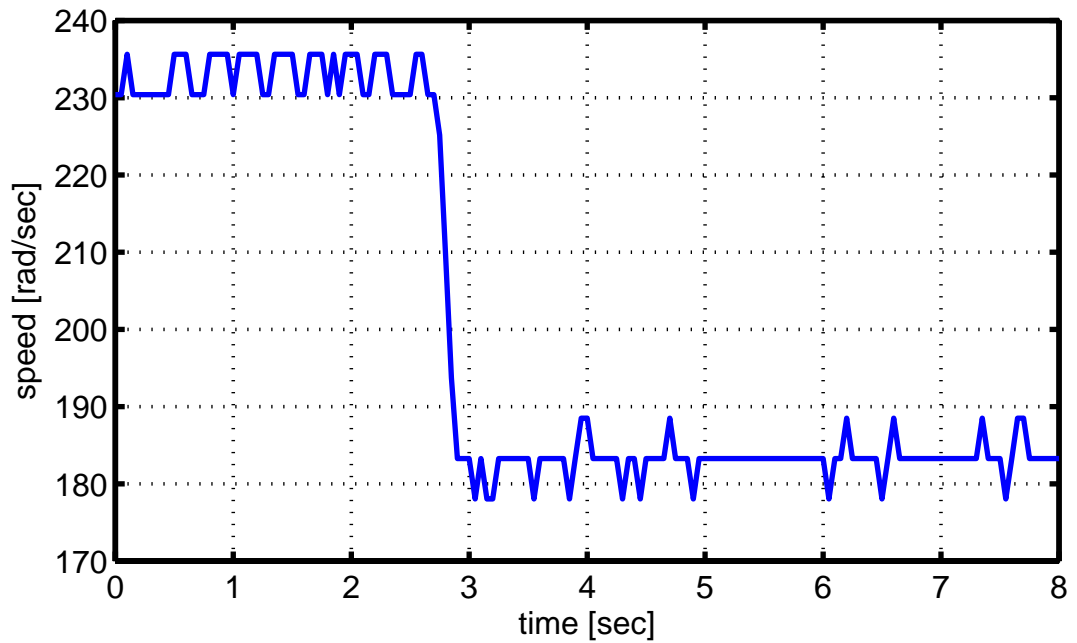


Figure 1.14: Step response with load generator.

CHAPTER II

CONTROL METHODS FOR THE DC-MOTOR EXPERIMENT

During this chapter we cover the following topics. In section 2.1, the analytical model of the DC motor is described and in Section 2.2 the basic feedback control loop is recalled. Section 2.3 explains the pole placement method that is used to design controllers for LTI systems and Section 2.4 outlines the root locus design method. The symmetrical optimum and magnitude optimum control methods are explained in Section 2.5 and Section 2.6, respectively. Section 2.7 points out the disturbance feedforward method that is used to reduce the effect of measurable load disturbances and finally Section 2.8 presents methods for the conversion of continuous time controllers to discrete time.

2.1 DC MOTOR MODELING

We first derive a model of the DC motor. It is based on the following schematic in Figure 2.1.

As can be seen in the figure, the model description is divided into an electrical component (so-called armature circuit) and a mechanical component (so-called motor shaft). The relevant variables in the armature circuit are the input voltage u , the armature current i_a , the armature resistance R_a , the armature inductance L_a as well as the voltage $u_M = c\Phi_F \cdot \omega$ that is induced by the motor motion ($c\Phi_F$ is the so-called motor constant). The motor shaft is driven by the motor torque $T_M = c\Phi_F \cdot i_a$, the load torque T_L and the momentum $(J_M + J_L) \cdot \dot{\omega}$ (we write $J_a = J_M + J_L$ in the

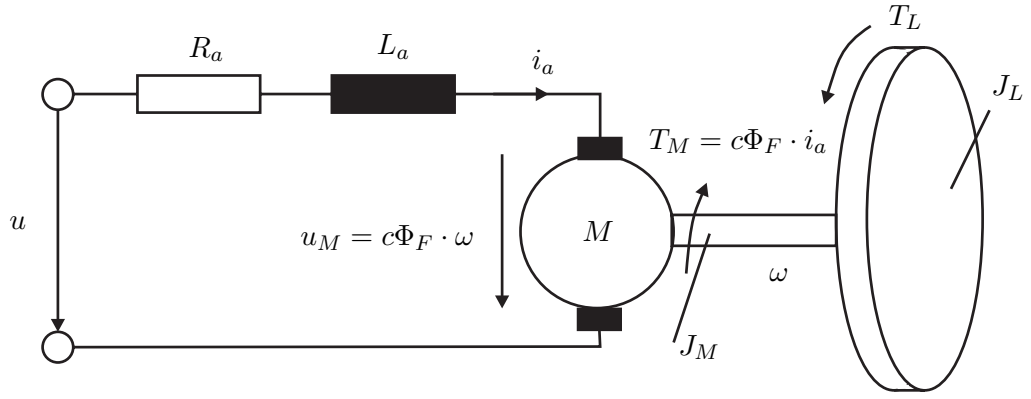


Figure 2.1: DC motor with load momentum.

future) that is induced by the inertia of the rotating masses (J_M and J_L are the moments of inertia of the motor shaft and the load).

Based on the above description, it is now possible to derive a state space model of the DC motor using the following equations.

Conservation of torque

$$J_a \frac{d}{dt} \omega = T_M - T_L \quad (2.1)$$

Motor torque equation

$$T_M = c\Phi_F \cdot i_a \quad (2.2)$$

Voltage equation

$$L_a \frac{d}{dt} i_a = u - R_a i_a - u_M \quad (2.3)$$

Induced voltage

$$u_M = c\Phi_F \cdot \omega \quad (2.4)$$

Combining the equations (2.1) to (2.4), and choosing the state variables i_a and ω , the following state equations are obtained.

$$\frac{d}{dt} i_a = \frac{1}{L_a} (-R_a i_a - c\Phi_F \cdot \omega + u) \quad (2.5)$$

$$\frac{d}{dt} \omega = \frac{1}{J_a} (c\Phi_F \cdot i_a - T_L) \quad (2.6)$$

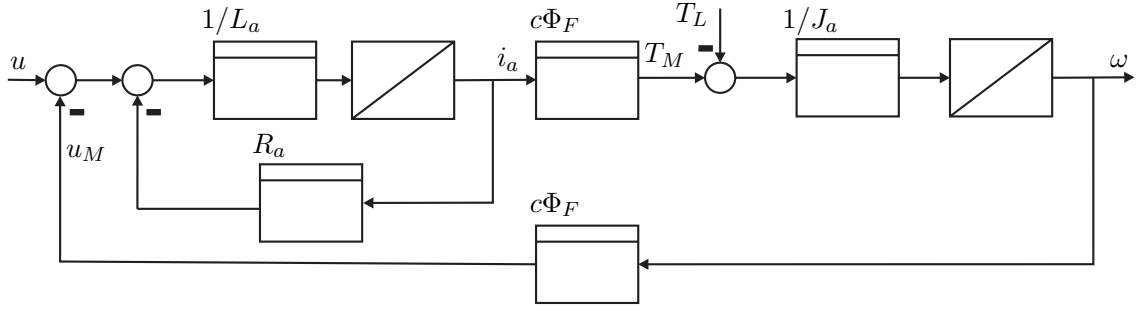


Figure 2.2: Block Diagram of the DC motor.

In addition, Figure 2.2 shows the block diagram of the DC motor model.

Using classical block diagram simplification rules, the simplified block diagram in Figure 2.3 is obtained.

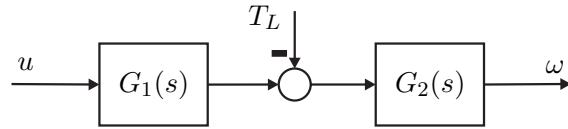


Figure 2.3: Transformed block diagram of the DC motor.

The transfer functions G_1 and G_2 in this block diagram evaluate as

$$G_1(s) = \frac{c\Phi_F/R_a}{1 + s L_a/R_a} \quad (2.7)$$

$$G_2(s) = \frac{R_a}{c\Phi_F^2} \cdot \frac{1 + s L_a/R_a}{1 + s (J_a \cdot R_a)/c\Phi_F^2 + s^2 (L_a \cdot J_a)/c\Phi_F^2} \quad (2.8)$$

2.2 FEEDBACK LOOP

The DC motor speed control requires the use of feedback. Based on the speed measurement by the encoder, our goal is to adjust the PWM signal that is considered as the input of the DC motor. The basic feedback control loop is shown 2.4.

Practically, the controller is realized on the PIC microcontroller in the form of a

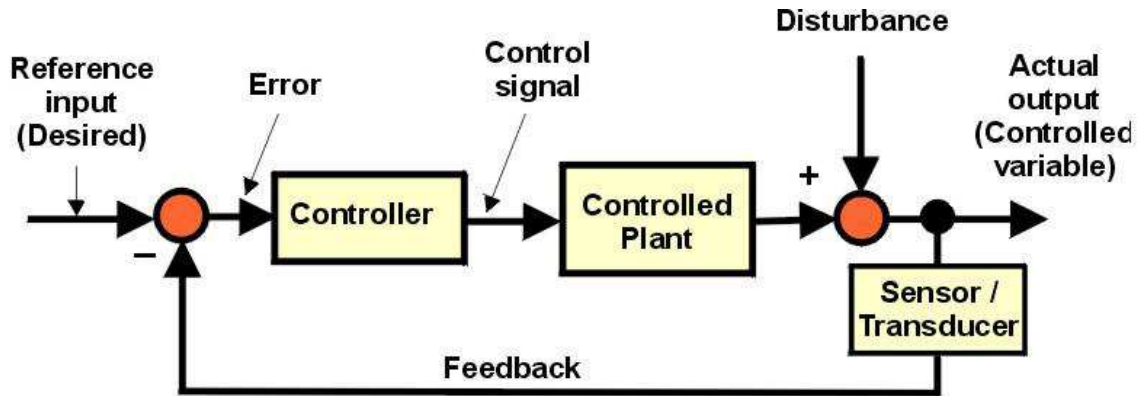


Figure 2.4: Feedback control loop.

discrete-time control algorithm. Based on the measurement of the encoder at pin 33, the PIC microcontroller adjusts the PWM duty cycle. The PWM signal at pin 16 then controls the output voltage of the driver circuit L293D.

2.3 POLE PLACEMENT

Pole placement method is one of the classical controller design methods for LTI systems in the basic feedback control loop [9] that is shown in Figure 2.4. It can be applied to both continuous-time systems and discrete-time systems. The basic idea of the pole placement is to first choose the desired poles of the closed-loop transfer function of the system according to given performance specifications. That is, the denominator polynomial $Q(s)$ of the closed-loop system is pre-determined. Then, the controller transfer function $C(s)$ is directly computed. This method is studied in the courses ECE 441 (continuous-time systems) and ECE 438 (discrete-time systems).

Suppose that we have a closed-loop system described by the rational transfer function:

$$G(s) = \frac{B(s)}{A(s)}, \quad (2.9)$$

where $B(s)$ is the numerator polynomial

$$B(s) = b_n s^n + b_{n-1} s^{n-1} + \dots + b_0 \quad (2.10)$$

and $A(s)$ is the denominator polynomial of the system:

$$A(s) = a_n s^n + a_{n-1} s^{n-1} + \dots + a_0 \quad (2.11)$$

n is called the degree of the system. The controller transfer function is

$$C(s) = \frac{P(s)}{L(s)} \quad (2.12)$$

with the controller numerator

$$P(s) = P_m s^m + P_{m-1} s^{m-1} + \dots + P_0 \quad (2.13)$$

and the controller denominator

$$L(s) = l_m s^m + l_{m-1} s^{m-1} + \dots + l_0 \quad (2.14)$$

m is the controller degree. Now assume that $Q(s)$ represents the desired closed-loop polynomial. The task is to compute the coefficients of $P(s)$ and $L(s)$ such that the closed-loop polynomial $Q(s)$ is obtained. This is achieved by the design equation

$$Q(s) = A(s)L(s) + B(s)P(s). \quad (2.15)$$

In the lecture, basically two versions of the pole placement design are discussed: pole placement without additional requirements and pole placement with integral action.

- For the pole placement method without additional requirements, the controller degree is usually chosen as $m = n - 1$. This means, if the degree of the system is $n = 1$, the controller type will be proportional controller $m = 0$ but when the system order is larger ($n > 1$), then the controller type will be a lead/lag compensator in the following form

$$C(s) = \frac{p_0 + \dots + p_{n-1} s^{n-1}}{l_0 + \dots + l_{n-1} s^{n-1}} \quad (2.16)$$

with $l_0 \neq 0$. The only disadvantage of this method is that it does not lead to an integral controller. That is, the feedback loop will generally show a non-zero steady-state error for reference and disturbance steps.

- Pole placement with integral action solves this problem by increasing the controller degree. The choice is now $m = n$ and the controller transfer function :

$$C(s) = \frac{p_0 + \dots + p_n s^n}{(l_0 + \dots + l_{n-1} s^{n-1}) s} \quad (2.17)$$

is used. Evaluating the design equation leads to integral control. Depending on the system degree n , different controller types are achieved. For example, $n = 1$ leads to PI-control and $n = 2$ leads to PID-control.

2.4 ROOT LOCUS

Root locus design is a graphical technique for the closed-loop design in the basic feedback control loop. The root locus allows us to find the poles of the closed-loop system by starting from the open-loop system's poles and zeros. Using this information it is possible to perform controller design based on the root locus for both stability and transient system response. The root locus method can be applied to systems of arbitrary order, however, the root locus can become very complex for systems of degree $n = 4$. The root locus design is explained in ECE 388 (Automatic Control), ECE 441 (Control System Design) and ECE 438 (Digital Control).

The basic steps in applying the root locus method are as follows.

- Determine the open loop transfer function with a free gain parameter K :
 $G_o(s) = K C(s) G(s)$.
- Sketch the root locus plot of $G_o(s)$ (this can be done manually or using Matlab).
- Move the closed-loop poles to desired locations according to the closed-loop specification.
- Determine the controller gain K .
- Simulate the feedback loop with the designed controller and verify if the closed-loop behavior is as desired.

2.5 SYMMETRICAL OPTIMUM METHOD

The (Kessler's) symmetrical optimum method is used for controller designs that lead to reasonable responses both for reference steps and disturbance steps. Usually, responses are fast with zero steady-state error but with overshoot. The symmetrical optimum method requires plant models that can be represented in the following form.

$$G(s) = \frac{K}{(1 + sT_1) \cdots (1 + sT_n)(1 + s\tau)}. \quad (2.18)$$

In this model, T_1, \dots, T_n are time constants that are large compared to the single time constant τ . If the plant model is accordingly, a controller of the following form is used.

$$C(s) = K_p \frac{(1 + s T_p)^n}{s (1 + s \tau_p)^{n-1}}, \quad (2.19)$$

where

- $n + 1$ is the order of the plant
- K_p is the controller gain
- T_p is the numerator time constant
- τ_p is the denominator time constant that is usually chosen as $\tau_p < 0.1 T_p$

Using the plant parameters n, T_1, \dots, T_n, τ and K , the design equation for the symmetric optimum controller is given by

$$K_p = \frac{1}{2 K \tau} \frac{T_1 \cdots T_n}{(4 n \tau)^n} \quad (2.20)$$

$$T_p = 4 n \tau \quad (2.21)$$

In summary, the symmetrical optimum enables the design of PID-type controllers for systems with several large time constants and one small time constant. It has a straightforward design equation, and leads to closed loops with fast responses to reference and disturbance steps. The symmetrical optimum is studied in ECE 441 (Control System Design).

2.6 MAGNITUDE OPTIMUM METHOD

The magnitude optimum method is usually used to achieve good reference tracking. It is based on the assumption that we want to obtain $T(s) \approx 1$ for the complementary sensitivity of the basic feedback loop.

The magnitude optimum can be applied to stable time-lag plants that can be represented in the form

$$G(s) = \frac{1}{A(s)} = \frac{1}{a_0 + \cdots + a_n s^n} \quad (2.22)$$

The controller transfer function is chosen as an integral controller with

$$C(s) = \frac{P(s)}{2s} = \frac{p_0 + \dots + p_m s^m}{2s} \quad (2.23)$$

Then it is desired to find the numerator coefficients of the controller transfer function. We basically study the cases of PI control ($P(s) = p_0 + p_1 s$) and PID control ($P(s) = p_0 + p_1 s + p_2 s^2$). For these cases, closed-form formulas for the controller parameters are as follows.

PI-control: $P(s) = p_0 + p_1 s$

$$p_0 = a_0 \frac{a_1^2 - a_0 a_2}{a_1 a_2 - a_0 a_3} \quad (2.24)$$

$$p_1 = a_1 \frac{a_1^2 - a_0 a_2}{a_1 a_2 - a_0 a_3} - a_0 \quad (2.25)$$

PID-Control $P(s) = p_0 + p_1 s + p_2 s^2$

$$D = \begin{vmatrix} a_1 & -a_0 & 0 \\ a_3 & -a_2 & a_1 \\ a_5 & -a_4 & a_3 \end{vmatrix} \quad (2.26)$$

$$p_0 = \frac{1}{D} \begin{vmatrix} a_0^2 & -a_0 & 0 \\ -a_1^2 + 2a_0 a_2 & -a_2 & a_1 \\ a_2^2 + 2a_0 a_4 - 2a_1 a_3 & -a_4 & a_3 \end{vmatrix} \quad (2.27)$$

$$p_1 = \frac{1}{D} \begin{vmatrix} a_1 & a_0^2 & 0 \\ a_3 & -a_1^2 + 2a_0 a_2 & a_1 \\ a_5 & a_2^2 + 2a_0 a_4 - 2a_1 a_3 & a_3 \end{vmatrix} \quad (2.28)$$

$$p_2 = \frac{1}{D} \begin{vmatrix} a_1 & -a_0 & a_0^2 \\ a_3 & -a_2 & -a_1^2 + 2a_0 a_2 \\ a_5 & -a_4 & a_2^2 + 2a_0 a_4 - 2a_1 a_3 \end{vmatrix} \quad (2.29)$$

2.7 DISTURBANCE FEED FORWARD

Disturbance feedforward is a method that can be used if a disturbance, that acts on the plant, can be measured [1]. In that case, it is possible to directly react to the

disturbance. We consider the block diagram in Figure 2.5, that has the structure of a DC motor speed control system. This disturbance is generated as the load torque L .

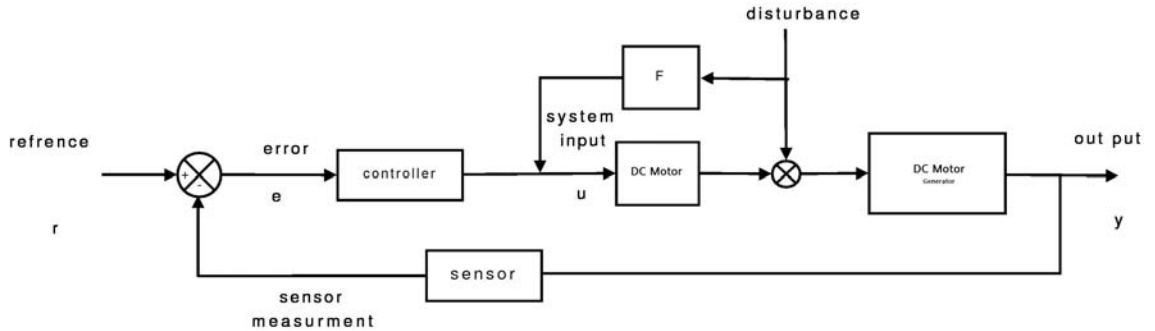


Figure 2.5: Disturbance feedforward controller architecture.

If we want to remove the disturbance before its effect to the output, we must fulfill the following equation according to the block diagram.

$$-1 + F(s) G_1(s) = 0 \quad (2.30)$$

Here, $F(s)$ is the disturbance feedforward transfer function and $P_{1(s)}$ is the DC Motor transfer function. Solving for $F(s)$ leads to

$$F(s) = \frac{1}{G_1(s)} \quad (2.31)$$

If $F(s)$ in (2.31) is not proper, it is generally multiplied by a lag transfer function. An example of this is shown in Section 3.7. Disturbance feedforward is part of the lecture ECE 441 (Control System Design).

2.8 CONTINUOUS AND DISCRETE-TIME CONTROL

In many cases, controllers are designed in the s-domain but realized on a digital computer such as a microcontroller or programmable logic controller (PLC). That is, the continuous-time controller transfer function has to be converted to a discrete-time (digital) representation. We use three classical approximation methods to perform this task.

- Euler method when:

$$s = \frac{z - 1}{T} \quad (2.32)$$

- Euler backward method when:

$$s = \frac{z - 1}{zT} \quad (2.33)$$

- Trapeziodal method when:

$$s = \frac{2z - 1}{T(z + 1)} \quad (2.34)$$

where T is the sampling period. This technique is studied in detail in the course ECE 438 (Digital Control). After performing the integral approximation, the result is a transfer function in the z -domain of the form

$$C(z) = \frac{U(z)}{E(z)} = \frac{b_n z^n + b_{n-1} z^{n-1} + \dots + b_1 z + b_0}{z^n + c_{n-1} z^{n-1} + \dots + c_1 z + c_0} \quad (2.35)$$

In order to implement such controller on a microcontroller, the z -transfer function is converted to a discrete-time difference equation as follows

$$\begin{aligned} U(z)(z^n + c_{n-1} z^{n-1} + \dots + c_1 z + c_0) &= E(z)(b_n z^n + b_{n-1} z^{n-1} + \dots + b_1 z + b_0) \\ U(z)\left(1 + \frac{c_{n-1}}{z} + \dots + \frac{c_1}{z^{n-1}} + \frac{c_0}{z^n}\right) &= E(z)\left(b_n + \frac{b_{n-1}}{z} + \dots + \frac{b_1}{z^{n-1}} + \frac{b_0}{z^n}\right) \end{aligned}$$

Considering that the factor $1/z$ in the z -domain corresponds to a time delay of one sampling time, we get the following difference equation.

$$u_k = -c_{n-1} u_{k-1} - \dots - c_1 u_{k-n+1} - c_0 u_{k-n} + b_n e_k + b_{n-1} e_{k-1} + \dots + b_1 e_{k-n+1} + b_0 e_{k-n} \quad (2.36)$$

CHAPTER III

CONTROLLER DESIGN FOR THE DC-MOTOR EXPERIMENT

In this chapter, Section 3.1 performs the identification of the DC motor model parameters. Section 3.2 to 3.5 apply different controller design methods to the DC motor experiment. Disturbance rejection is discussed in Section 3.6 and the disturbance feedforward method is described in Section 3.7.

3.1 DC MOTOR MODEL IDENTIFICATION

The mathematical model of the DC motor is explained in Section 2.1 and it is described by the transfer function

$$G(s) = \frac{\Omega(s)}{U(s)} = \frac{1}{c\Phi_F} \frac{1}{1 + \frac{R_a J_a}{c\Phi_F^2} s + \frac{L_a J_a}{c\Phi_F^2} s^2} \quad (3.1)$$

Here, U is the Laplace transform of the control input voltage u and Ω is the Laplace transform of the output angular velocity ω . In addition, it has to be considered that the actual control input in our experiment is not the input voltage but the duty cycle of the PWM signal that is provided by the PIC. Furthermore, it has to be noted that the actual sensor measurement on the PIC is not the angular velocity but the number of pulses per time unit. We consider the relationship between duty cycle δ and input voltage u as

$$u = K_u \delta = 12 V/255 \delta. \quad (3.2)$$

and the relationship between angular velocity ω and pulse count p per time unit T as

$$p = K_\omega \omega = \frac{24T}{2\pi} \omega. \quad (3.3)$$

Hence, the actual plant transfer function can be written as

$$G_a(s) = \frac{P(s)}{\Delta(s)} = K_u G(s) K_\omega, \quad (3.4)$$

In order to determine the plant model, the unknown parameters R_a , J_a , L_a and $c\Phi_F$ have to be identified. The resistance R_a of the DC motor can be directly measured and evaluates to 20Ω . The other parameters are determined using a step response measurement of the DC motor with an input duty cycle step from 120 to 200. Output measurements are taken with a sampling time of $T = 0.05$ sec. The step response measurement is shown in Figure 3.1.

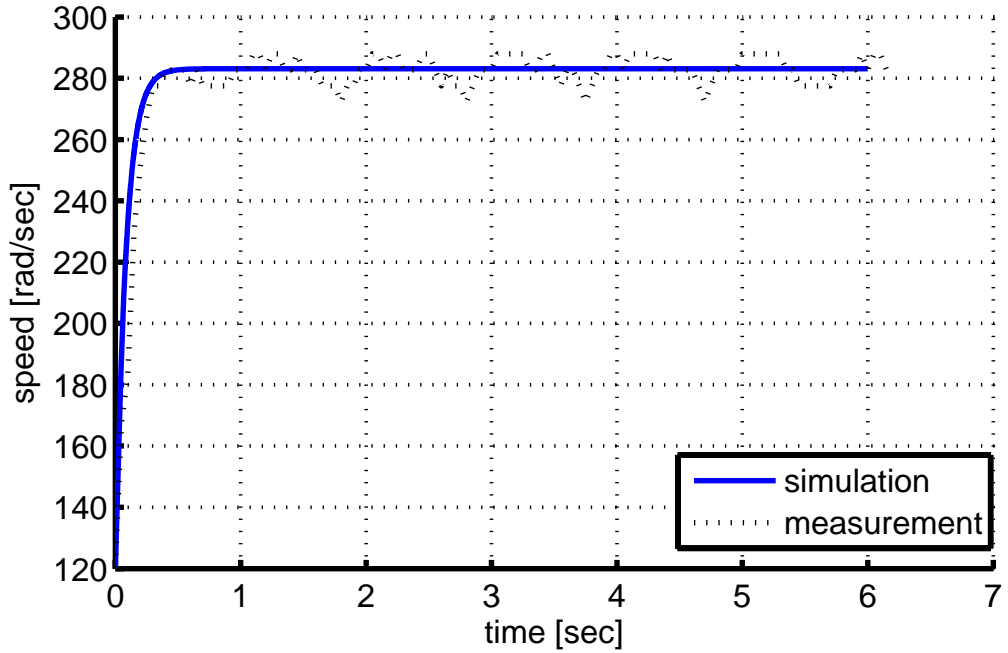


Figure 3.1: Step response of the DC motor.

In the next step, the identification tool of Matlab is used to determine the transfer function corresponding to the measured step response. In accordance with the DC motor model, we choose a second-order lag transfer function and obtain

$$G_a(s) = \frac{687.5}{s^2 + 218.5s + 2545}$$

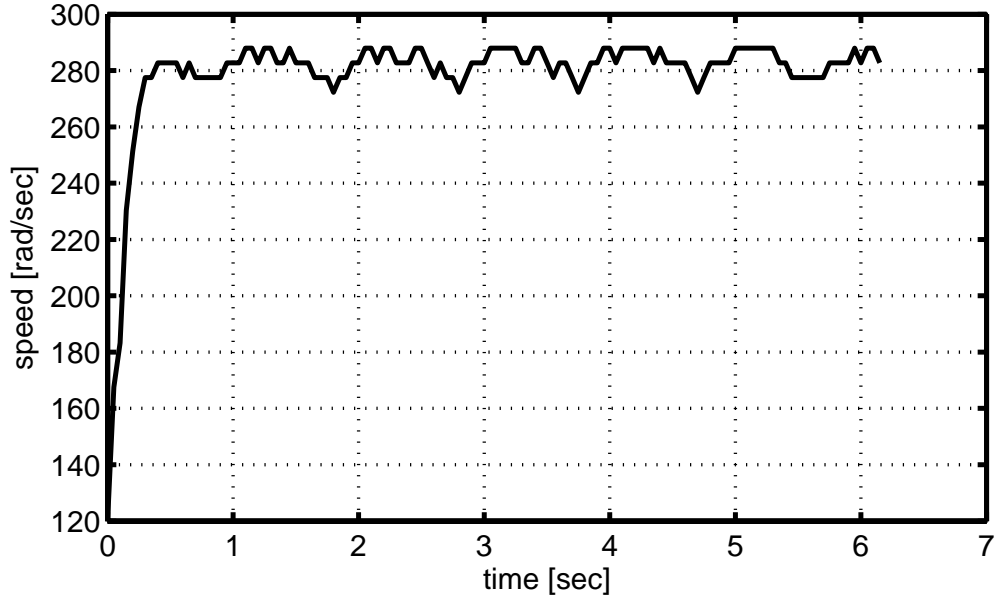


Figure 3.2: Comparison of step response measurement and simulation

Hence, the missing parameters can be computed as

$$c\Phi_F = \frac{1}{K} = 0.0333$$

$$J_a = \frac{a_1 c\Phi_F^2}{R_a} = 4.7517 \cdot 10^{-6}$$

$$L_a = \frac{a_2 c\Phi_F^2}{J_a} = 0.0915$$

The parameters of the transfer function $G_a(s)$ are summarized in Table 3.1. Figure

R_a	$c\Phi_F$	J_a	L_a	K_u	K_ω
20	0.0333	$4.7517 \cdot 10^{-6}$	0.0915	12 V/255	$\frac{24T}{2\pi}$

Table 3.1: Parameters of $G_a(s)$

?? shows a comparison of the measured step response and a simulation of the identified plant model. It is readily observed that both measurement and simulation are very close. Hence, the plant model is considered as suitable for a model-based controller design. It only has to be noted that a small oscillation can be observed in the step response measurement of the DC motor. This oscillation is due to a mechanical misfit between the two connected DC motors.

3.2 APPLICATION OF POLE PLACEMENT

The plant transfer function is actually a second-order lag transfer function with one large and one very small time constant as follows.

$$G_a(s) = \frac{0.2701}{(0.081 s + 1)(0.0049 s + 1)}$$

In order to simplify the pole placement design, this transfer function is approximated by neglecting the small time constant. Hence, the transfer function used in the design is

$$G_{app}(s) = \frac{0.2701}{(0.081 s + 1)}$$

For the pole placement design, we choose two different closed-loop polynomials. In the first case, we obtain a slow closed loop without oscillations by placing all poles at $s = -5$. In the second case, we want a faster closed loop with poles at $s = -10$. In order to achieve integral control, we choose a controller transfer function with integrator according to Section 2.3.

$$C(s) = \frac{p_0 + p_1 s}{l_1 s}. \quad (3.5)$$

Using the closed-loop polynomial $Q_1(s) = (s + 5)^2 = s^2 + 10s + 25$, the first case yields the controller parameters $p_0 = 92.55$, $p_1 = -8.681$ and $l_1 = 12.35$. That is, the controller transfer function for the first case is

$$C(s) = \frac{-8.681 s + 92.55}{12.35 s}. \quad (3.6)$$

The discrete-time controller transfer function is computed for the sampling time $T = 0.05$ with the trapezoidal method in Section 2.8. The resulting z -transfer function is

$$C(z) = \frac{6.576 z - 3.475}{z - 1}. \quad (3.7)$$

Finally, the inverse z -transform as described in Section 2.8 leads to the discrete-time controller equation

$$u_k = u_{k-1} + 6.576 e_k - 3.475 e_{k-1} \quad (3.8)$$

Considering the case of $T = 0.05$, the exemplary controller code is as follows.

```
int dutycycle = 0;
```

```

unsigned int count=0; //This count used in interrupt routine
unsigned countOld =0; //Counter for the previous time
int eold=0; // old error value
int ek=0; // current error value
unsigned short uold=0; // old value of the control input
unsigned int difference;

void interrupt() // count the encoder pulses
{
    count++; // If "+" direction
    INTCON.INTF=0; // Clear flag
}
void main()
{
    TRISB = 0xff; // PORTB is input for encoder
    PORTC = 0xff; // PORTC is output for PWM
    INTCON=0b11010000; // enable RBO interrupts
    OPTION_REG=0b01000000; //Enable portB internal PULL-UP resistors
    PWM2_Init(8000); // Initialize PWM 8MHz
    PWM2_Start();

while(1)
    countOld = count;
    Delay_ms(50); // wait for 50 msec
    difference = count - countOld; // pulses counted per ms
    ek=34-difference; // error signal
    //control input computation
    dutycycle= (int)(6.576*(float)ek + (float)uold + 3.475*(float)eold);
    if (dutycycle > 255)
        dutycycle=255;
    if (dutycycle < 0)
        dutycycle = 0;

```

```

    uold = (unsigned short)dutycycle; // old control input
    eold = ek; // old error
    // set the value of the dutycycle
    PWM2_Set_Duty((unsigned short)dutycycle);
}
}

```

Applying this code on the PIC microcontroller, the following reference step response measurement in Figure 3.3 is obtained for a reference step of 52.4 rad/sec.

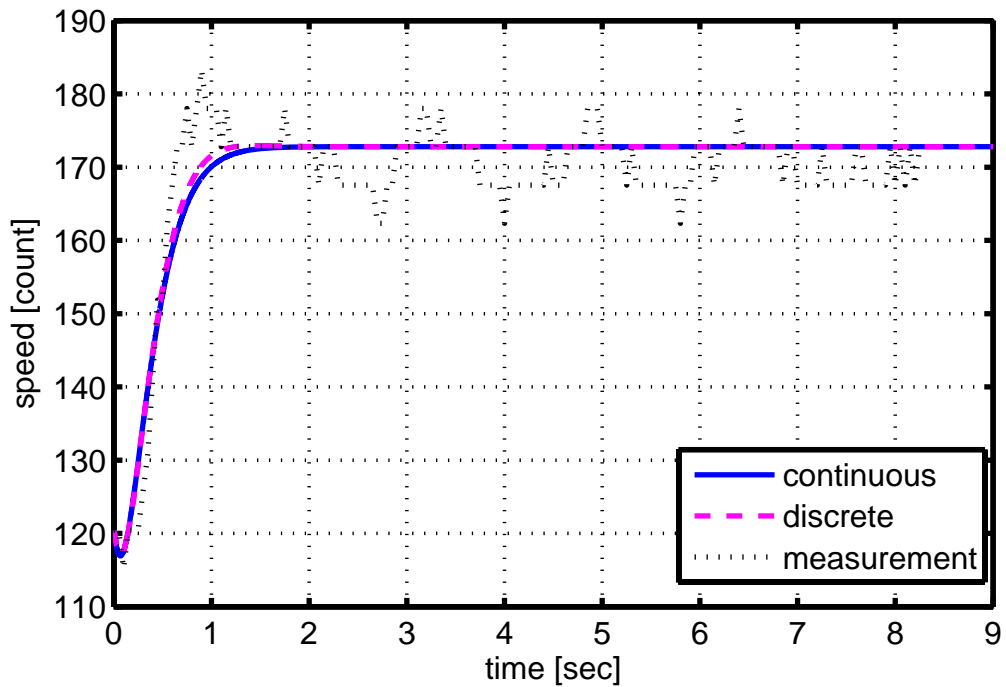


Figure 3.3: Pole placement control with slow closed-loop poles.

The figure shows the measurement from the real DC motor experiment in comparison with the simulation of the continuous time and discrete-time closed loop in Simulink. It can be seen that the measurement result is very close to the simulation. It only has to be mentioned that there is a slight overshoot in the measurement that does not appear in the simulation. This deviation is caused by the mechanical misfit as discussed in Section 3.1.

Using the closed-loop polynomial $Q_2(s) = (s + 10)^2 = s^2 + 20s + 100$, the second case yields the controller parameters $p_0 = 370.2$, $p_1 = 28.34$ and $l_1 = 12.35$. That is, the controller transfer function for the first case is

$$C(s) = \frac{28.34s + 370.2}{12.35s}. \quad (3.9)$$

The discrete-time controller transfer function is computed for two different sampling times $T = 0.05$ and $T = 0.02$ with the trapezoidal method in Section 2.8. The resulting z-transfer functions are

$$C_{0.05}(z) = \frac{12.77z - 6.885}{z - 1} \quad C_{0.02}(z) = \frac{11z - 8.65}{z - 1}. \quad (3.10)$$

Finally, the inverse z-transform as described in Section 2.8 leads to the discrete-time controller equations

For $T = 0.05$

$$u_k = u_{k-1} + 12.77e_k - 6.885e_{k-1} \quad (3.11)$$

For $T = 0.02$

$$u_k = u_{k-1} + 11e_k - 8.65e_{k-1} \quad (3.12)$$

The comparison of measurement and simulation for a reference step of 52.4 rad/sec is shown in Figure 3.4 ($T = 0.05$) and Figure 3.5 ($T = 0.02$).

Again, there is very good agreement between measurement and simulation. In addition, it has to be mentioned that there is small overshoot for a sampling time of $T = 0.05$ but no overshoot for $T = 0.02$. This difference illustrates the effect of decreasing the sampling time.

It has to be noted that another approximation of the second-order transfer function $G_a(s)$ is possible by considering the residues for the different poles of the transfer function. In this case, a partial fractional decomposition of the form

$$G_a(s) = \frac{A}{1 + 0.081s} + \frac{B}{1 + 0.0049s} =$$

is performed. The computation of A and B results in

$$G_a(s) = \frac{0.2701}{(1 + 0.081s)(1 + 0.0049s)} = \frac{0.2875}{1 + 0.081s} + \frac{-0.0174}{1 + 0.0049s}.$$

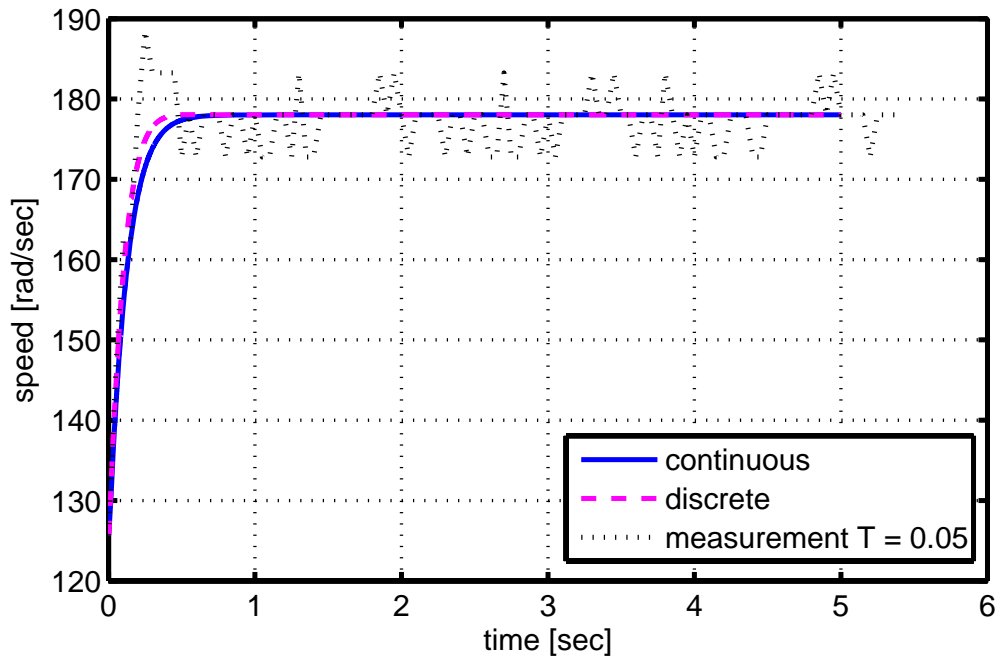


Figure 3.4: Pole placement control with fast closed-loop poles and $T = 0.05$.

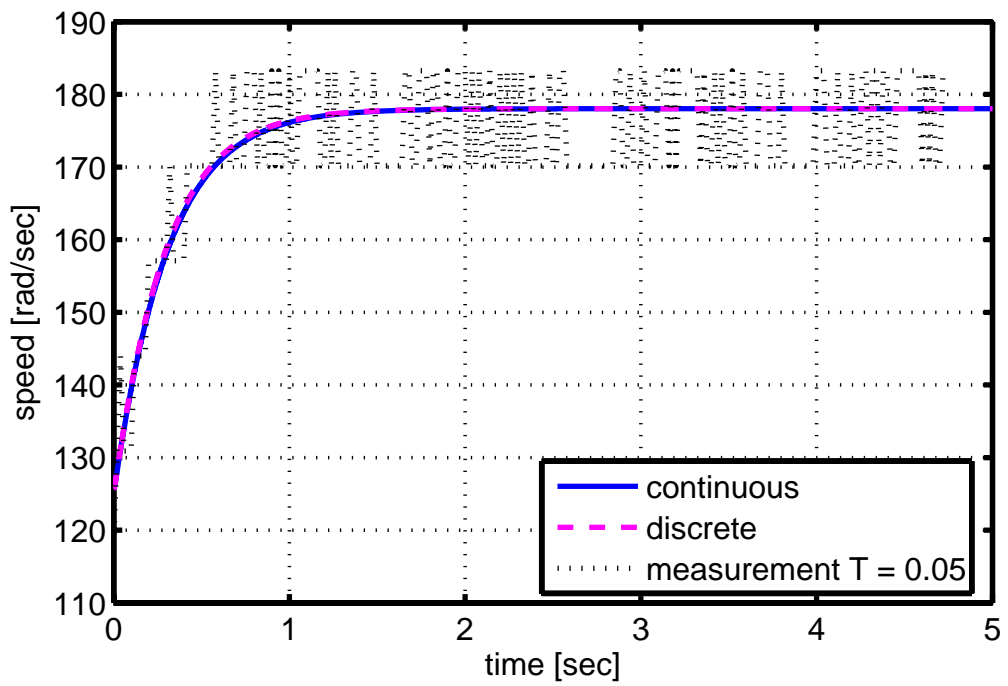


Figure 3.5: Pole placement control with fast closed-loop poles and $T = 0.02$.

In that case, $G_a(s)$ is approximated as

$$G_{app}(s) = \frac{0.2875}{1 + 0.081 s}.$$

The controller computation and control experiment for this approximation is very similar to the result obtained for the previous approximation.

3.3 APPLICATION OF ROOT LOCUS DESIGN

3.3.1 Proportional Control

We follow the root locus design method as described in Section 2.4 using the plant transfer function

$$G_a(s) = \frac{687.5}{s^2 + 218.5 s + 2545} = \frac{687.5}{(s + 12.3)(s + 206.2)}. \quad (3.13)$$

The root locus plot of $G_a(s)$ (for the case of proportional control) is shown in Figure 3.6.

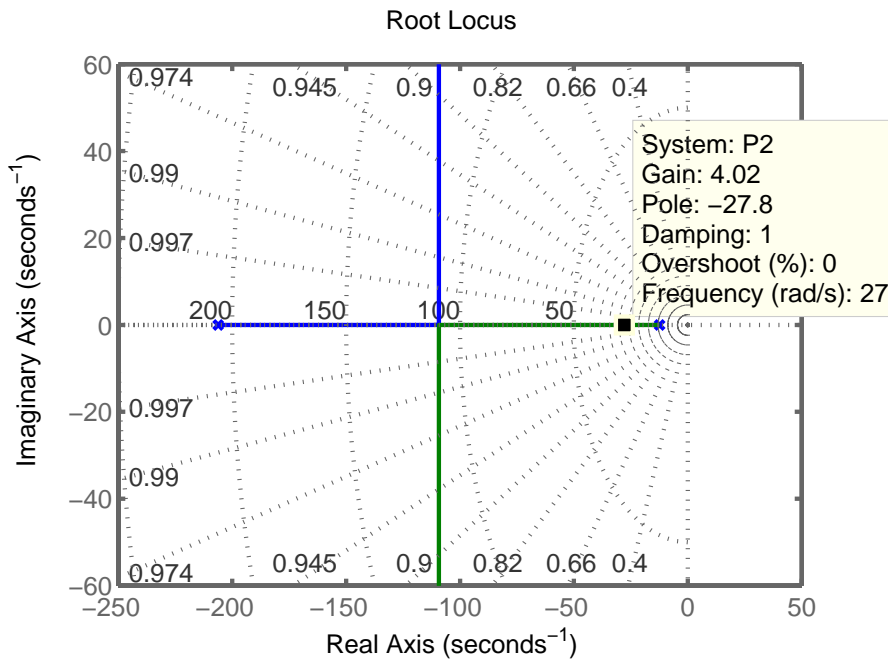


Figure 3.6: Root locus plot for proportional control.

In this experiment, we choos $K = 4$ in order to achieve a closed loop that is not too

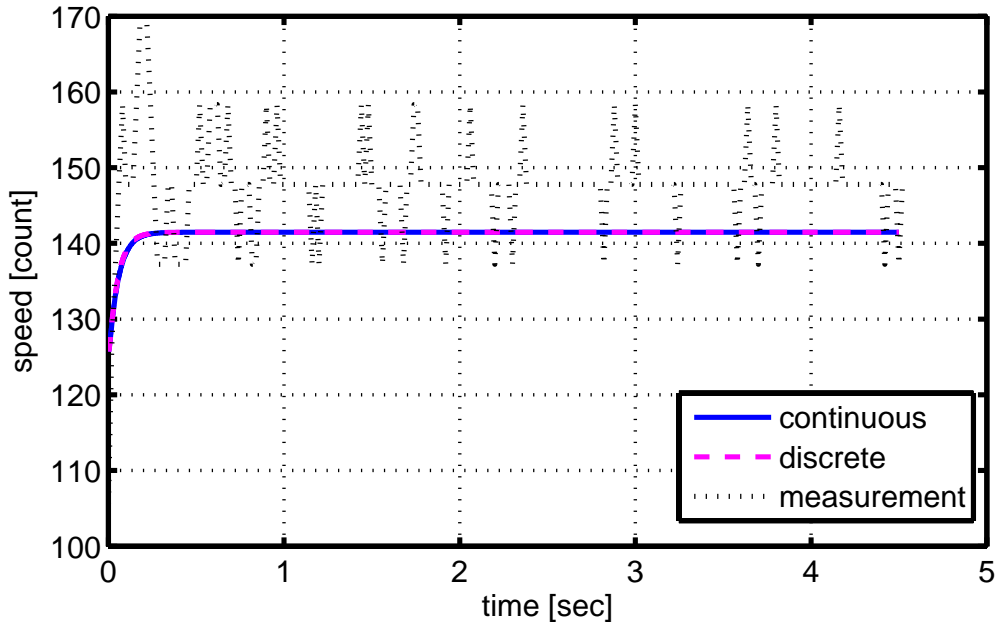


Figure 3.7: Root locus design for proportional control and $T = 0.02$.

fast for control by the PIC microcontroller. It has been observed in the experiment that faster closed loops lead to large oscillations. In the experiment, we perform a reference step from 125.7 rad/sec to 178 rad/sec and use the discrete-time control input computation

$$u_k = 120 + 4e_k. \quad (3.14)$$

Here, 120 is the duty cycle for achieving the initial speed 125.7 rad/sec. Using this control input equation, the resulting comparison for simulation and measurement and a sampling time of $T = 0.02$ is shown in Figure 3.7.

In this experiment, it can be seen that there is a deviation between the steady-state values of simulation and measurement. According to the experimental evaluation, this deviation is most likely caused by the low precision of the speed measurement for small sampling times as $T = 0.02$. Hence, it is suggested to increase the precision in future experiments by using a more precise encoder or by using a DC motor with larger speed.

3.3.2 PI Control

We use the same plant as in the case of proportional control but modify the controller structure as

$$C(s) = K_P \frac{s + 12.3}{s} \quad (3.15)$$

That is, we suggest to compensate the slow pole at 12.3. The resulting root locus plot for $G_a(s)C(s)$ is shown in Figure 3.8.

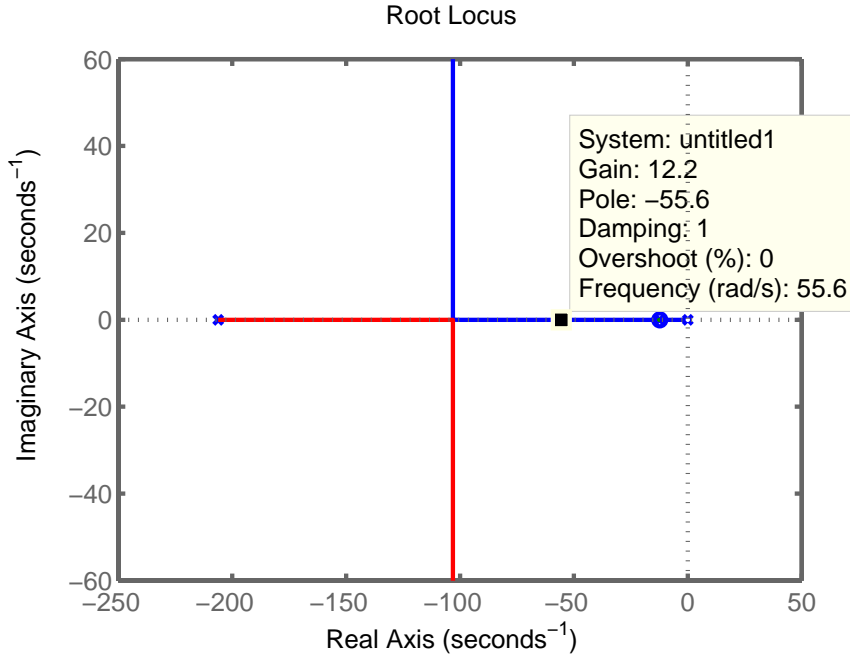


Figure 3.8: Root locus plot for PI control.

Again, in order to achieve a closed loop that is not too fast for the PIC microcontroller, we choose $K_P = 12.2$. This results in the PI controller transfer function

$$C(s) = 12.2 \frac{s + 12.3}{s}. \quad (3.16)$$

The conversion to the controller z-transfer function with the trapezoidal method for a sampling time of $T = 0.02$ leads to

$$C(z) = \frac{13.7z - 10.7}{z - 1} \quad (3.17)$$

and the discrete-time control input equation is

$$u_k = u_{k-1} + 13.7 e_k - 10.7 e_{k-1}. \quad (3.18)$$

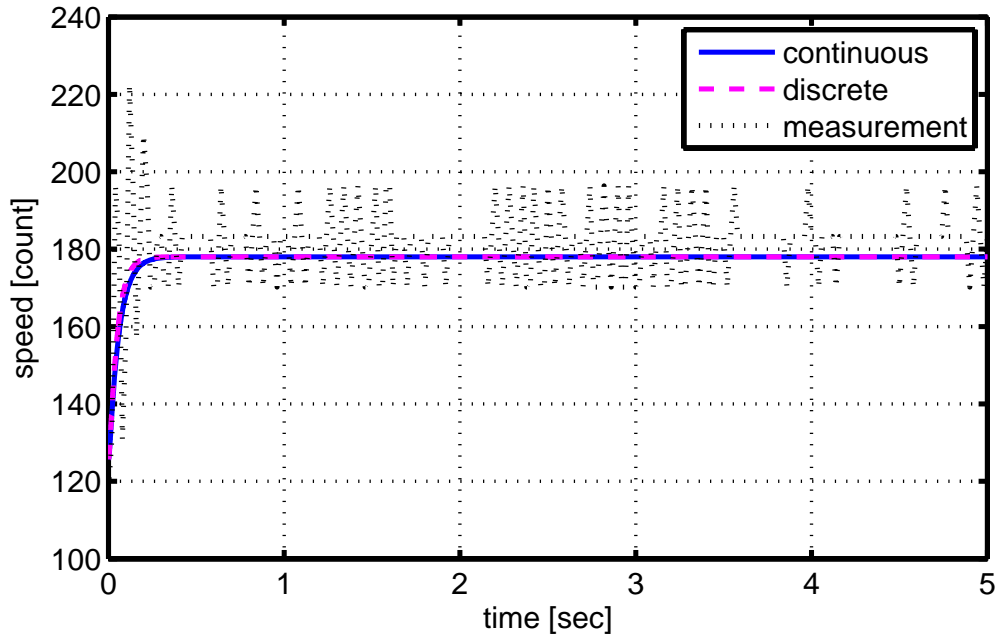


Figure 3.9: Root locus design for PI control.

Figure 3.9 shows the comparison between the reference step responses of simulation and measurement for a reference step from 125.7 rad/sec to 178 rad/sec. Again, there is a small deviation of the steady-state value and the measurement observes oscillations that are not present in the simulation. However, the dynamics of closed-loop measurement and simulation are in very good agreement.

3.4 APPLICATION OF SYMMETRICAL OPTIMUM DESIGN

We next perform the symmetrical optimum design as described in Section 2.5 using the plant transfer function

$$G_a(s) = \frac{0.2701}{(0.081s + 1)(0.0049s + 1)} \quad (3.19)$$

It is readily observed that $G_a(s)$ has a small time constant $\tau = 0.0049$ and a large time constant $T_1 = 0.081$. In addition, $K = 0.2701$ and the number of large time constants is given by $n = 1$. Using these values, the controller parameters K_P and T_P can be directly computed using the formulas in Section 2.5. They result in

$K_P = \frac{T_1}{2K\tau_4\tau} = \frac{0.081}{8 \cdot 0.0049^2 \cdot 0.2701} = 3982$ and $T_P = 4\tau = 0.0194$. That is, the controller transfer function is

$$C(s) = K_P \frac{1 + sT_P}{s} = 3982 \frac{1 + 0.0194s}{s}. \quad (3.20)$$

The computation of the controller z-transfer function is performed for the two sampling times $T = 0.01$ and $T = 0.02$ using the trapezoidal method. The result is

$$C_{0.01}(z) = \frac{154.5z - 74.89}{z - 1} \quad C_{0.02}(z) = \frac{77.27z + 2.378}{z - 1}. \quad (3.21)$$

and the discrete-time equations for the control input computation evaluate as

For $T = 0.01$

$$u_k = u_{k-1} + 154.5 e_k - 74.89 e_{k-1} \quad (3.22)$$

For $T = 0.02$

$$u_k = u_{k-1} + 77.27 e_k + 2.378 e_{k-1} \quad (3.23)$$

Figure 3.10 and 3.11 show the simulation for the two different sampling times.

It can be seen that the symmetric optimum controller results in a very fast closed loop. As a consequence, choosing a too large sampling time as $T = 0.02$ already leads to an instable feedback loop. Because of that reason and because of the lacking precision of the speed measurement it was not possible to achieve a stable closed loop for the real experiment. It is suggested to improve the precision of the speed measurement and to use a DC motor with a larger nominal speed.

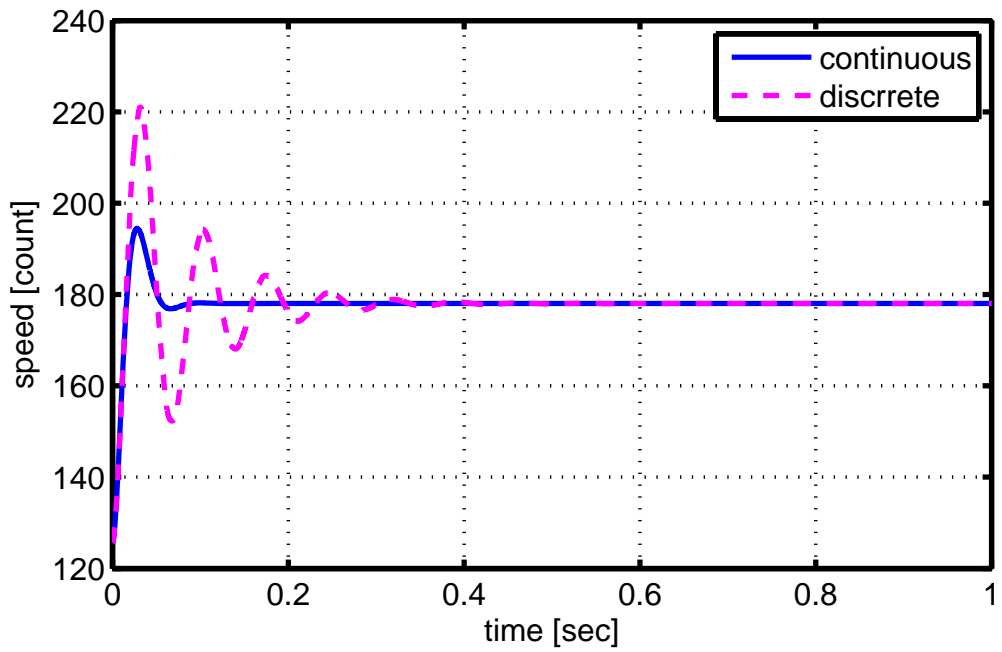


Figure 3.10: Symmetrical optimum design for $T = 0.01$.

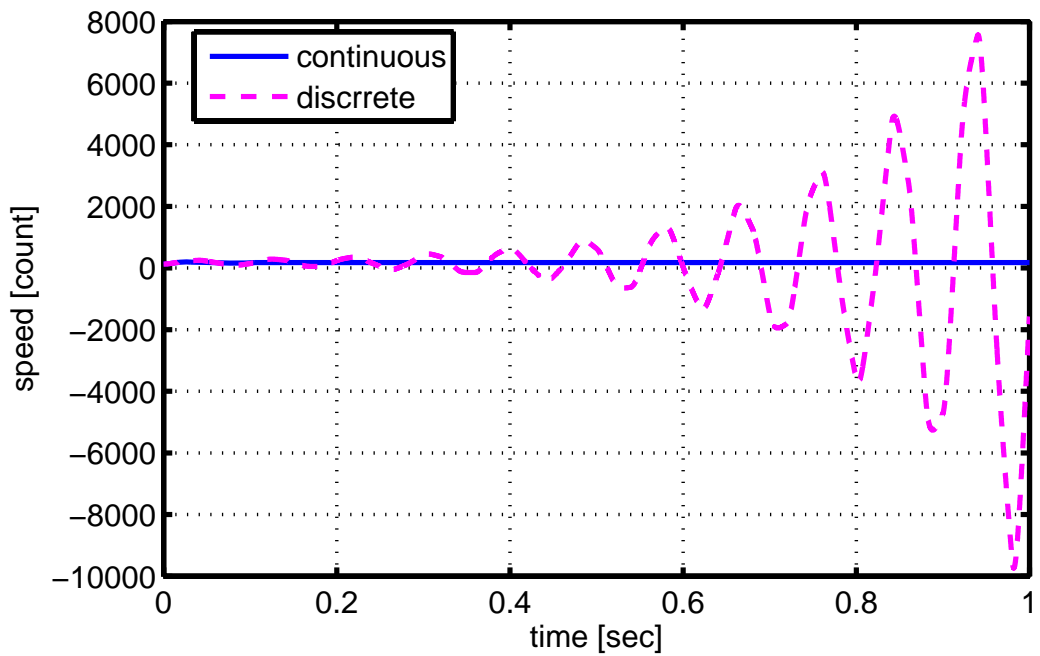


Figure 3.11: Symmetrical optimum design for $T = 0.02$.

3.5 APPLICATION OF MAGNITUDE OPTIMUM DESIGN

We finally apply the magnitude optimum design as described in Section 2.6 with the plant transfer function

$$G_a(s) = \frac{1}{3.7018 + 0.3178 s + 0.0015 s^2}. \quad (3.24)$$

The resulting PI-controller parameters for this plant are computed as Find $p_0 = a_0 \frac{a_1^2 - a_0 a_2}{a_1 a_2 - a_0 a_3} = 765.73$ and $p_1 = a_1 \frac{a_1^2 - a_0 a_2}{a_1 a_2 - a_0 a_3} - a_0 = 62.0396$. Hence the resulting controller transfer function is

$$C(s) = \frac{p_0 + p_1 s}{2 s} = \frac{62.0396 + 765.73 s}{2 s}. \quad (3.25)$$

The controller z-transfer function is computed for a sampling time of $T = 0.01$ as

$$C(z) = \frac{32.93z - 29.11}{z - 1} \quad (3.26)$$

and the control input equation is

$$u_k = u_{k-1} + 32.93 e_k - 29.11 e_{k-1} \quad (3.27)$$

The simulation of the closed-loop system for a reference step from 125.7 rad/sec to 178 rad/sec is shown in Figure 3.12. Similar to the symmetrical optimum design the closed-loop response is too fast for the PIC microcontroller and leads to an instable measurement result. The simulation shows that the steady-state value is achieved very fast.

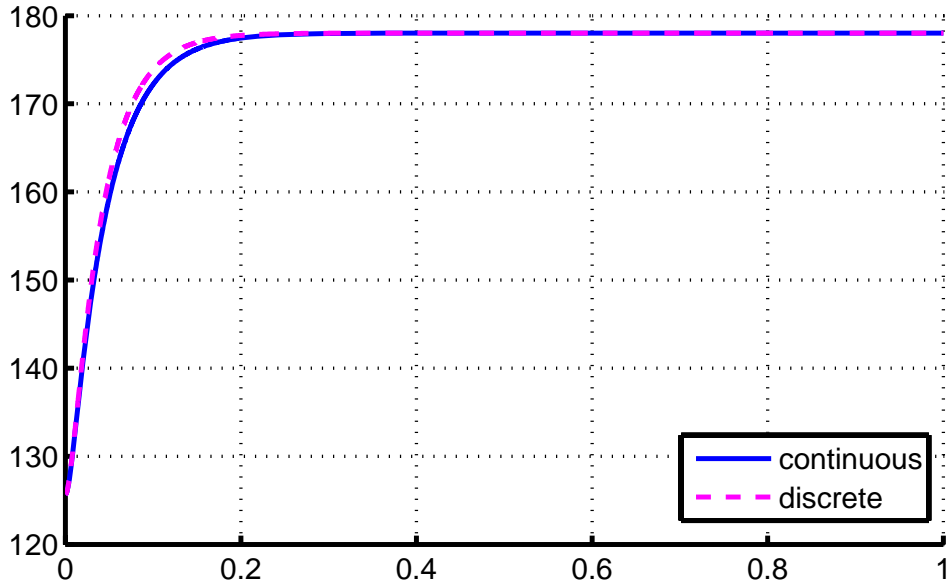


Figure 3.12: Magnitude optimum design for $T = 0.01$.

3.6 DISTURBANCE REJECTION

In this section, the effect of a load disturbance on the open-loop and closed-loop behavior of the DC motor experiment is discussed. We know from Section 2.1 that the plant has the structure as shown in Figure 3.13. According to the parameter

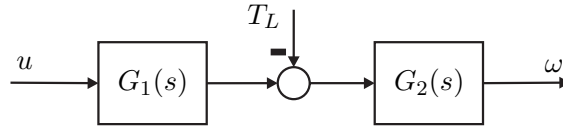


Figure 3.13: Transformed block diagram of the DC motor

identification in Section 3.1 we obtain the transfer functions

$$G_1(s) = \frac{7.828 \cdot 10^{-5}}{0.004577 s + 1}$$

and

$$G_2(s) = \frac{R_a}{c\Phi_F^2} \cdot \frac{1 + s L_a/R_a}{1 + s (J_a \cdot R_a)/c\Phi_F^2 + s^2 (L_a \cdot J_a)/c\Phi_F^2} = \frac{7.74 s + 1691}{9.629 \cdot 10^{-4} s^2 + 0.21 s + 2.451}.$$

We apply a load torque by using the second DC motor in generation mode as a

generation. To this end, a load resistor $R_L = 80 \Omega$ is connected to the second DC motor. Depending on the voltage U_g supplied by the generator, the generated power P_g is evaluated as

$$P_g = \frac{U_g^2}{R_L}. \quad (3.28)$$

Considering that this electrical power is equivalent to the mechanical power supplied $P_m = T_L \omega$ of the DC motor, it holds that

$$P_g = \frac{U_g^2}{R_L} = T_L \omega = P_m. \quad (3.29)$$

Here, T_L is the load torque and ω is the angular velocity of the motor. It is now possible to find the load torque from the previous equation. We compute

$$T_L = \frac{U_g^2}{R_L \omega}. \quad (3.30)$$

In the next experiment, we run the first DC motor at a constant speed of 230.4 rad/sec and connect the load resistor to the second DC motor. We measure a voltage of 3.9 V at the load resistor, that is, the load torque is $T_L = \frac{3.9^2 V^2}{80 \Omega 230.4 \text{ rad/sec}} = 8.25 \cdot 10^{-4} \text{ Nm}$. The following figure 3.14 shows a comparison of the disturbance measurement and simulation.

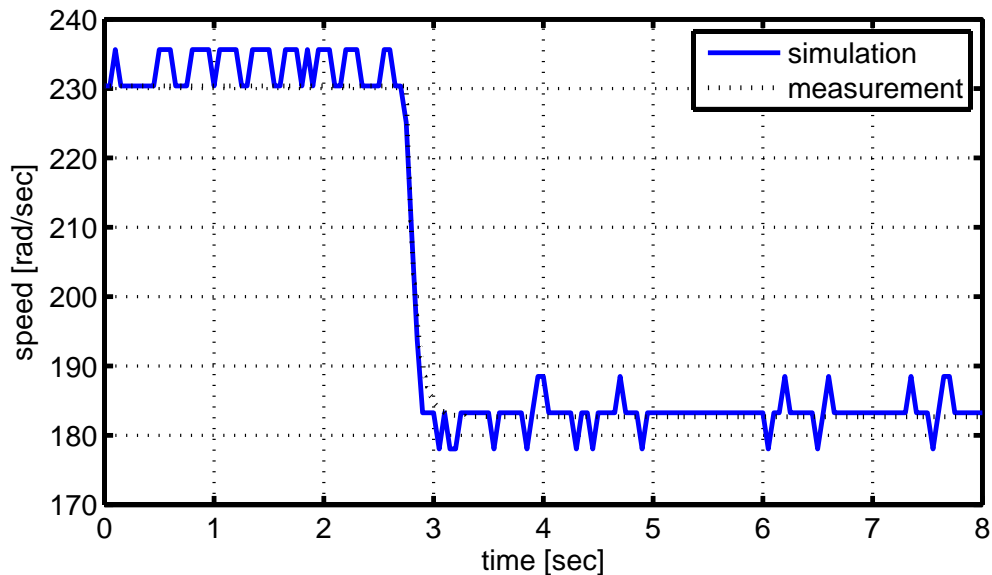


Figure 3.14: Disturbance step response measurement.

It can be seen that there is very good agreement between simulation and measurement. We next study the applicability of control in case of disturbances for the

different methods in the previous sections. We first refer to the pole placement method as studied in Section 3.2. The resulting measurement and simulation for the controller in (3.6) is shown in Figure 3.15.

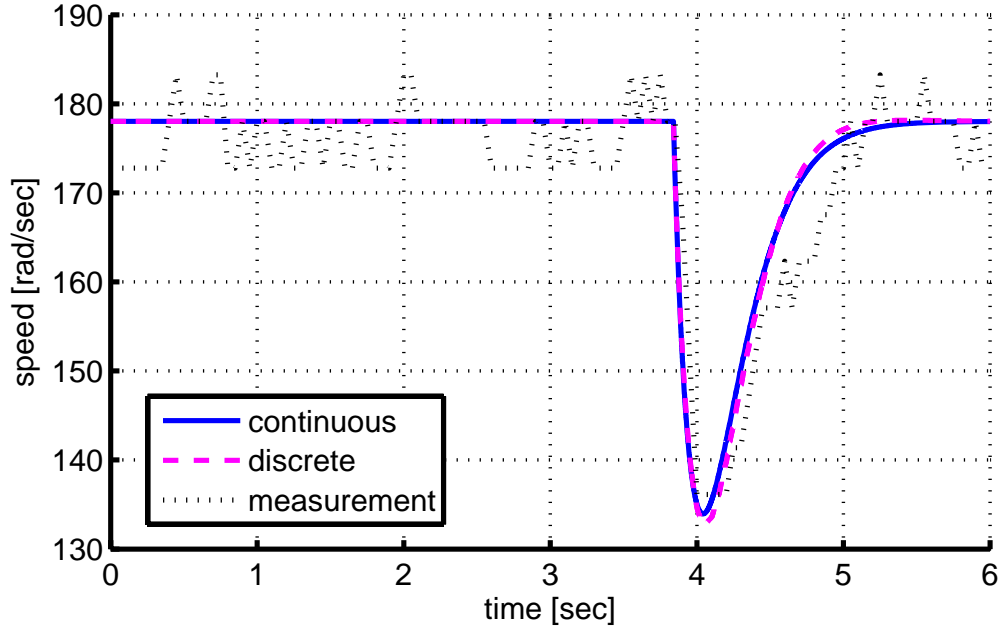


Figure 3.15: Disturbance step response measurement for pole placement control with slow poles.

In addition, Figure 3.16 shows the analogous measurement for the controller in (3.9).

3.7 APPLICATION OF DISTURBANCE FEED-FORWARD CONTROL

Finally, the concept of disturbance feedforward as described in Section 2.7 is applied. According to (2.31), the disturbance feedforward filter should be chosen as

$$F(s) = \frac{1}{G_1(s)} \frac{1}{1 + s\theta} = \frac{0.004577s + 1}{7.828 \cdot 10^{-5}} \frac{1}{1 + s\theta}, \quad (3.31)$$

whereby θ is a small time constant. Since the time constant of $G_1(s)$ is very small, the application of disturbance forward in our experimental setup only allows to use a proportional filter $F(s) = 1/7.828 \cdot 10^{-5} = 12.7$ (otherwise the time constant θ has to be chosen smaller than the possible sampling times on the PIC microcontroller).

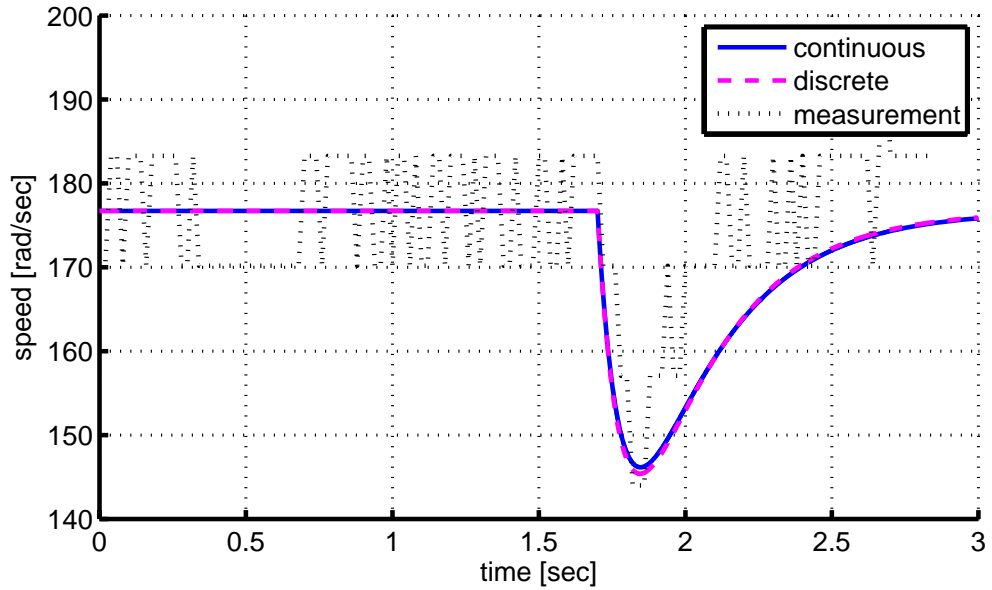


Figure 3.16: Disturbance step response measurement for pole placement control with fast poles.

That is, the filter can be realized by the simple discrete-time equation

$$\Delta u_k = 12.7 t_{l,k}. \quad (3.32)$$

Figure 3.17 shows a disturbance step response measurement. In this measurement, the pole placement controller in (3.6) is used in conjunction with the disturbance feedforward in (3.32).

In comparison with Figure 3.15, it can be seen that the effect of the disturbance is smaller if disturbance feedforward is used. However, since the lack of speed of the PIC microcontroller does not allow to use a more complicated filter transfer function than the proportional transfer function in (3.32), the effect is relatively small.

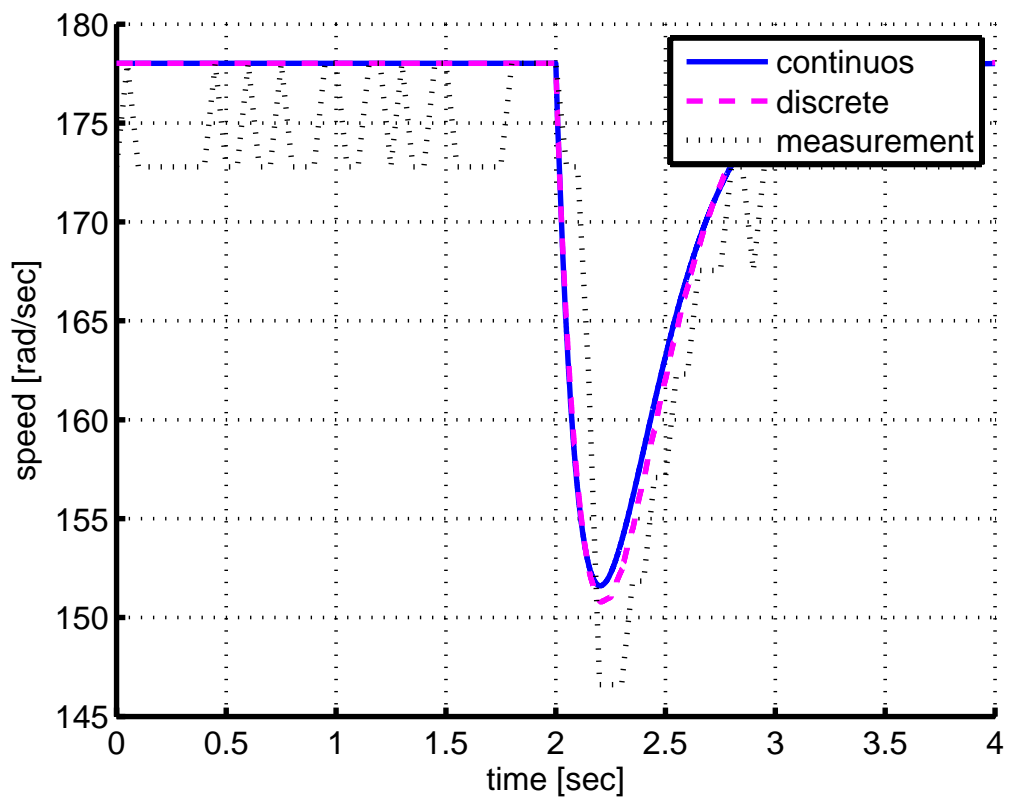


Figure 3.17: Application of disturbance feedforward.

CONCLUSIONS

As reported in the literature, the DC motor speed control experiment is very suitable for control laboratories. It allows to study basic aspects of control system design such as linear system modeling and parameter identification. Second it allows applying a large number of different controller design methods. Third, it enables the easy implementation and test of discrete-time control algorithms.

In this thesis, a DC motor control experiment for control laboratories is designed. In the practical part of the thesis, basic components of this system are described and assembled to perform feedback control. The main components are.

- PIC microcontroller.
- DC motor with encoder for speed measurement.
- Motor driver.
- Serial communication for data analysis on a PC.
- Load generator and load measurement.

In the theoretical part of the thesis, basic properties of the system are investigated and suitable control designs are evaluated. The works include.

- Linear modeling and parameter identification of the DC motor system.
- Application of various controller design methods.
- Discretization of continuous-time control algorithms for the digital controller implementation.

- Simulation and validation of control systems using Matlab/Simulink.

Using the above steps and equipment, a very good agreement of the real system measurements and the simulations in Matlab/Simulink could be concluded. However, it also has to be mentioned that one disadvantage of the current setup is the low speed of the selected DC motor. One consequence of this low speed is the low precision of the speed measurement, which has a negative effect on controller designs with fast closed loops. A similar effect is observed when choosing small sampling times for the control algorithm. In this case, the PIC microcontroller is not suitable due to its computational limitations. Hence, a necessary step for future work is the selection of DC motors with a larger nominal speed.

In summary, the following tasks remain for future work in order to further improve the DC motor experiment.

- A microcontroller with better performance might be used to achieve smaller sampling times.
- A DC motor with higher nominal speed and without gearbox might be used to improve the measurement accuracy.
- A mechanical design for properly aligning the two DC motors should be performed.

REFERENCES

- [1] Bequette, B.W. (2003) Process Control: Modeling, Design, and Simulation, USA.
- [2] Bhattacharya S. K.(2008), Control Systems Engineering, second edition .
- [3] Danaher Encoder, www.dynapar.com.pdf.
- [4] Golnaraghi, F., Kuo,B. (2009) Automatic Control Systems,9th edition, Canada.
- [5] Goodwin, G., M.S. E. (2000), Control System Design.
- [6] Han-Way Huang.(2002), An Introduction to Software and Hardware Interfacing.
- [7] Jan Axelson(2007) , Serial Port Complete: Programming and Circuits for Rs-232 and Rs-485 Links.
- [8] Kozak,L., Huba,M.(2003), Control System design, Slovak republic.
- [9] Ledin, J. (2003), Embedded Control Systems in C/C++.
- [10] M. Gopal.(2002), Control Systems: Principles and Design,second edition.
- [11] Microchip PIC 16F877A microcontroller, url: <http://www.microchip.com/wwwproducts/Devices.aspx?dDocName=en010242>.
- [12] Nisa,N.s. (2011), Nise Control Systems Engineering, 6th edition.
- [13] Ogata, K., Hall, P. (2010), Modern Control Engineering, USA.
- [14] Shah, S.L, Macgregor.J.(2004), Dynamics And Control of Process Systems, vol2, Cambridge Massachusetts, USA.
- [15] Stephen W. Fardo and Dale R.Patrick(2008),Electrical Power Systems Technology,Third Edition.
- [16] Texas Instruments (2004), L293D Quadruple Half-H drivers, datasheet, url: <http://www.ti.com/lit/ds/symlink/l293d.pdf>.
- [17] TexasInstruments(2004),MAX232,datasheet,url: <http://www.ti.com/lit/ds/symlink/max232.pdf>.
- [18] Umland, J. W., Safiuddin, M. (1990) Magnitude and Symmetric Optimum Criterion for the Design of Linear Control Systems: What Is It and How Does It Compare with the Others, IEEE Transactions on Industry Applications, vol. 26, no. 3,USA.

CURRICULUM VITAE

PERSONAL INFORMATION

Surname, Name: Kahya, Ardm.

Nationality: Iraqi (IR).

Date and Place of Birth: 1 Jan 1973 , Kirkuk.

Marital Status: Married.

Number of kids: One.

Phone/Fax: +90 5342945003.

+9647701317846

email: ardamkahi73@yahoo.com

EDUCATION

Degree	Institution	Year of Graduation
MS	Çankaya Univ. Electronic and Communication Engineering	2012
BS	College of Technology/Kirkuk Electronic Control	2003
ITE	Institute of Technical/Kirkuk	1996
High School	Kirkuk	1992

FOREIGN LANGUAGES

Arabic, English, Turkish.

HOBBIES

Football, Political, Reading, Computer, Swimming.