

Final Report

Stabilizing an Inverted Pendulum within TwinCAT3

Company supervisors' information

1st company supervisor	Nikolas Eimer
Email	N.Eimer@beckhoff.com
2nd company supervisor	Marnick Sluismans
Email	msluismans@beckhoff.nl

School supervisor's information

Name	Amin Mannani
Email	amin.mannani@han.nl

Student's information

School	HAN University of Applied Science
Student name	Suat Nguyen
Student number	593604
Email	snguyen@beckhoff.nl ss.nguyen@student.han.nl

By submitting this report, the intern certifies that this is his original work, and he has cited all the referenced materials, in the forms of texts, models, and books properly.

1. Preface

At HAN University of Applied Sciences, also known as Hogeschool van Arnhem en Nijmegen, every Industrial Power Systems student, who is following an Electronic and Electrical Engineering programme, finishes his studies with a graduation project either at a company or at the university. This report is the product of a five-month graduation project performed at BECKHOFF Automation.

This project is about stabilizing an Inverted Pendulum within TwinCAT3 which is an IDE, Integrated Development Environment, developed by BECKHOFF. Besides the company and the Project Plan report, this report describes a problem diagnosis and an analysis of an internship machine at BECKHOFF. Also in this report, you will read my journey about how I found an answer for my question:

How can I use Control System theories in the real world?

During my graduation project, which was not easy, I have received a ton of support from different people. Therefore, I would like to use this opportunity to show my appreciation to these people. Firstly, my first BECKHOFF supervisor, Nikolas Eimer, for always being willing to listen to my questions and to give advice. Secondly, my second BECKHOFF supervisor, Marnick Sluismans, for the knowledge and help with the understanding of software and hardware. He has encouraged and believed in me since the first day of my project. Next, Simon Sleeking, Geert Baars, Rob Rawlyk, and other colleagues who always helped me during the project for their time and willingness to help. Furthermore, my teacher, Amin Mannani, for his time and feedback. I feel truly lucky to have a teacher who understands and cares about his students. I am very glad to have such wonderful people around me.

Finally, I want to dedicate my thesis to my parents and my brother who have supported me from the beginning of my life.

Control System is difficult for everyone. Applying it in practice is even more difficult. Curiosity and perseverance matter.

Sincerely,

Suat Nguyen, Eindhoven, January 2022

2. Summary

The Inverted Pendulum is one of the classical control problems of an inherently unstable system. The idea is that the pendulum, fixed on a cart, stays upright in its vertical position while the cart is moving, or disturbances are applied. The system is used to demonstrate the capability of the target hardware and software platform developed by BECKHOFF. With the platform, it is possible to run controllers in real-time while connecting to Input/Output signals from the real world.

The Inverted Pendulum setup is built based on a machine mainly used in transportation, called XTS, eXtended Transport System. Due to a limitation of the setup, the pendulum starts from an upright position, making a unique setup. Compared to other work that has already been done in the past, their pendulums start from a downward position. The Inverted Pendulum setup was not working at BECKHOFF. A controller, written in PLC languages, must be implemented to keep the pendulum in its vertical position.

In order to obtain a controller for the system, first, a mathematical model of the system is analyzed, then a physical machine is investigated. How does the machine work? How can it be controlled? Besides this, an encoder, which reads an angle of the pendulum, is analyzed. When the inputs and outputs are clearly understood, a correct model corresponding to the system has been built and simulated in MATLAB/Simulink. A PID Tuner from Simulink is used to tune PID controller parameters to achieve a robust design with the desired response time.

It turned out that the traditional PID-Controller was not suitable enough for the system. In specific, the steady state error never becomes 0, and the control signal grows unboundedly. Instead of using the PID-Controller, a pole placement technique has been used. A new controller has been made and simulated. The simulation gives a good result. Based on the new controller and the Simulink model, a PLC program has been implemented.

As shown on Figure 1, the Inverted Pendulum is balanced on its own for the first time on 15-12-2021.



Figure 1: Stabilized inverted pendulum

Table of Contents

1. Preface	1
2. Summary	2
3. Version log	6
4. Introduction	7
4.1. About the company	7
4.2. About an Inverted Pendulum setup	7
5. Analysis of research context and problem	8
5.1. Balancing a stick	8
5.2. Problem statement	8
5.3. Current situation	8
5.4. The scope of the project	9
5.5. Goal and research questions	9
6. Mathematical modeling	10
6.1. Force analysis	12
6.2. Linearization	13
7. Traditional motors	14
7.1. DC motor	14
7.2. Servo motor	15
7.3. Linear motor	15
8. Inverted Pendulum setup	17
8.1. Traditional motors and the XTS system	18
8.1. Connection diagram	19
8.2. XTS machine itself	19
8.2.1. Motor module	20
8.2.2. Mover	21

8.2.3.	Guide rail	21
8.3.	Controller	22
8.4.	Controlling the mover	23
8.4.1.	Movement without programming	24
8.4.2.	Movement with programming	25
8.5.	Pendulum encoder	27
8.5.1.	Homing position of the pendulum	28
8.5.2.	Counting direction of the encoder	28
9.	Controller design	29
9.1.	Laplace transform	29
9.2.	Transfer function	29
9.3.	Parameter identification	30
9.3.1.	Travelling distance of the mover	30
9.3.2.	Mass and length of the pendulum	30
9.3.3.	Mass moment of inertia of the pendulum (I)	31
9.3.4.	Mass of the mover	31
9.3.5.	Angular displacement of the blocking angle unit	32
9.4.	Selecting a controller and simulating	33
9.4.1.	Importing data to MATLAB	34
9.4.2.	P-Controller	35
9.4.3.	PI-Controller	35
9.4.4.	PID and PD-Controller	37
9.4.5.	PI + I2-Controller	38
10.	PLC program	41
10.1.	Building a PLC program	41
10.2.	Hold and drop strategy	42

10.3.	Full motion strategy	43
11.	Conclusion and Recommendation	44
11.1.	Conclusion	44
11.2.	Recommendation	44
	Reflection	46
Appendix A.	Approved LTC statement.....	50
Appendix B.	Approved graduation assignment application form	51
Appendix C.	Final version of Project Plan	56
Appendix D.	Move functions	63
Appendix E.	Connecting the encoder with the BECKHOFF system.....	65
Appendix F.	Servo Control Technology	67
Appendix G.	MoveAbsolute UML Code.....	73
Appendix H.	Hold and drop strategy UML code	75
Appendix I.	Full motion strategy UML code	77
	References	79

3. Version log

Below is the version log of the document:

Version	Date	Author	Comments
0.01	20-09-2021	Suat Nguyen	First version
0.02	01-11-2021	Suat Nguyen	Half report
0.03	14-12-2021	Suat Nguyen	Draft of the final report
0.04	19-12-2021	Suat Nguyen	Finished the report
0.05	31-12-2021	Suat Nguyen	Updated the report based on feedback
0.06	05-01-2022	Suat Nguyen	Final version

4. Introduction

This document will present the Final Report towards the **Stabilizing an Inverted Pendulum within TwinCAT3** project by first introducing the project as a whole, and then further specifying the machine used in this project.

4.1. About the company

BECKHOFF Automation, also known as BECKHOFF, is a manufacturer of automation technology. Their products are divided into four different areas [1]:

- IPC – Industrial PC is a PLC built with standard PC hardware and software, allowing users to choose the devices that are best-suited to users' specific applications, such as selecting OS (Windows or TC/BSD which is a FreeBSD derivate), or hard drive (HDD, SSD, M2 ...). Also, interfaces can be chosen which is important for a flexible solution.
- Fieldbus I/O – Input/Output terminals. These terminals are used to make a communication between the IPC and the real world. The diverse I/O systems offer solutions for all common signals in automation technology, such as Digital/Analog Input/Output signals, power measurement, current, voltage, and communication with almost every fieldbus standard.
- Motion – stepper motor, servo motor, linear motor. Producing different types of drives in 48VDC region as well as in high voltage area 230 – 480VAC.
- Real-time control software TwinCAT to implement PLC programs, and other purposes such as HMI integration, measurement with scope view...

4.2. About an Inverted Pendulum setup

The Inverted Pendulum is a classical control problem of an unstable dynamic system. The system consists of an inverted pendulum mounted on top of a cart, which is moving on a guide rail. The pendulum will simply fall over if the cart is not moved to balance it.

The system is used to demonstrate the capability of the target hardware/software platform developed by BECKHOFF. With this platform, it is possible to run models and controllers in real-time while connecting these models with input/output signals from the real world. The demonstrator will be used at BECKHOFF for educational (external and internal) and promotional purposes.

5. Analysis of research context and problem

In this chapter, research questions and problem statements will be addressed.

5.1. Balancing a stick

During childhood, each of us had a chance to balance a stick on your finger. Simply put the stick vertically on one of our fingers and try to move our hand in such a way that the stick should not fall. When the stick is not moving or balanced, it is in a state called equilibrium. By that time, any forces on the stick are balanced by the force in the opposite direction. Observe carefully and we can see that to balance the stick, we have to move our hand quickly to a direction where the stick is falling.



Figure 2: Balancing a stick on one finger [2]

That is the most basic way of balancing a rod on one finger. In this project, the stick, or an inverted pendulum will be balanced by a PLC.

5.2. Problem statement

The problem of this project is that a control system needs to be designed to maintain an inverted pendulum in an upright position while compensating disturbances. The control system should be capable of controlling the pendulum within TwinCAT3 which is an IDE, Integrated Development Environment developed by BECKHOFF.

5.3. Current situation

The current Inverted Pendulum set-up is not yet working at BECKHOFF. The system consists of six different parts, which are not yet fully assembled as shown in Figure 3: an IPC, Industrial PC

(1), an encoder for the pendulum (2), a mover/cart (3), a guide rail and motor modules with a built-in encoder (4), a pendulum (5), and a blocking angle unit (6).

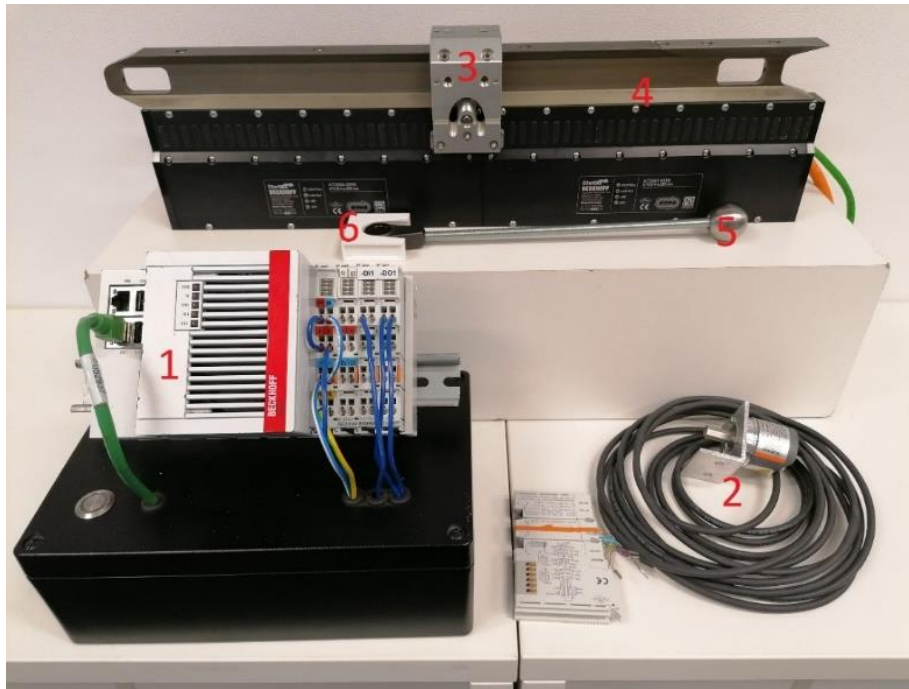


Figure 3: Inverted Pendulum set-up

5.4. The scope of the project

The scope of the project is restricted to the Inverted Pendulum set-up, the BECKHOFF hardware/software platform and two different feedback systems. What is more, the swing-down process of the pendulum is ignored due to the limitation of the hardware. That means the pendulum will start from an upward position.

5.5. Goal and research questions

The ultimate goal of the project is to have the controller for the Inverted Pendulum running in TwinCAT3. That means a PLC program, written in Structured Text (ST), must be implemented. The intern is allowed to use MATLAB/Simulink to build and verify models. After that, a code generator will be used to convert the Simulink blocks to the PLC language. If the converting process is not feasible, the intern needs to build a separate control system in TwinCAT3.

To make the Inverted Pendulum run in TwinCAT3, the following table shows research questions and sub-questions.

Tag	Question
1	How is the XTS machine built up? What is the role of each component?
1.1	How to control the mover?
2	What are the equations of motion of the system?
2.1	What are the equations for the transfer function?
2.2	What are the Simulink models corresponding to the system?
3	What kind of controller can control the pendulum? How can it be designed?
4	How to use the code generator to convert Simulink blocks to PLC languages? Is this task feasible? If not, how to write a similar controller in PLC languages?

6. Mathematical modeling

Mathematical modeling is a tool to gain a clear understanding of a system. In Figure 4 a schematic of the cart and the pendulum are given with the applicable forces. Free body diagrams of each part are illustrated clearly to visualize the applied forces, moments, and reactions.

In this system, we will consider a two-dimensional problem where the pendulum is tilted in a vertical plane, and a force F that moves the cart horizontally. The outputs of the system are the angular position of the pendulum θ and the horizontal position of the cart x .

The following process is guided by a theory book, Feedback Control of Dynamic Systems 7th Edition by Gene F. Franklin, J. David Powell, Abbas Emami-Naeini, on page 61. [3]

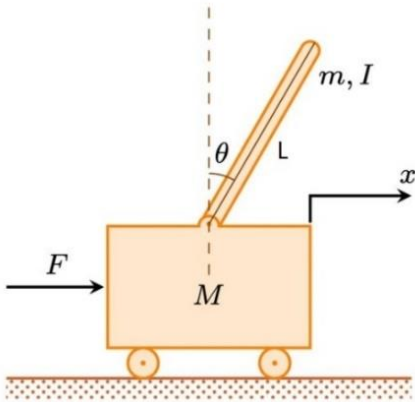


Figure 4: Free body diagram of the system [4]

Below a list of the relevant parameters and their meaning are given:

Parameter	Meaning	SI Unit
x	Cart position	mm
m (small m)	Mass of the pendulum	kg
M	Mass of the cart	kg
b	Coefficient of friction of the cart	N/m/sec
I (I in inertia)	Mass moment of inertia of the pendulum	kg.m ²
L	Total length of the pendulum	m
l (l in length)	Length to pendulum center of mass	m
θ	Angular position of the pendulum	radian
F	Force applied to the cart	N

The variable b is defined as a coefficient of friction of the cart. There is also a coefficient of friction for the pendulum, but its value compared to the cart is considerably small, which is why the friction of the pendulum is not mentioned here.

6.1. Force analysis

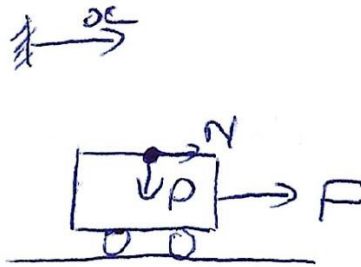


Figure 5: Free body diagram of the cart

Summing the forces in the free body diagram of the cart in the horizontal direction, we have

$$-F_{inertial} - F_{friction} + F_{reaction\ from\ the\ pendulum} + F_{applied\ to\ the\ cart} = 0$$

$$\rightarrow -Ma - bv + N + F = 0$$

$$\rightarrow F = M\ddot{x} + b\dot{x} - N$$

Equation 1

Summing the forces in the free body diagram of the cart in the vertical direction, no useful information can be found.

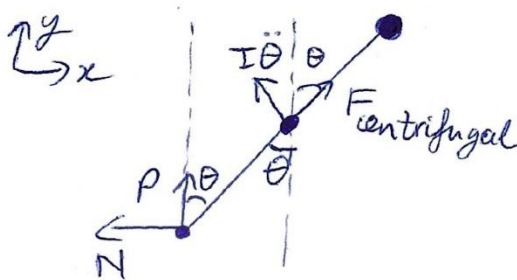


Figure 6: Free body diagram of the pendulum

Summing the forces in the free body diagram of the pendulum in the horizontal direction, we have

$$-F_{reaction\ from\ the\ cart} - F_{inertial\ from\ the\ cart} - F_{centrifugal} + F_{tangential} = 0$$

$$\rightarrow -N - ma - l\omega^2 \sin \theta + l\alpha \cos \theta = 0$$

$$\rightarrow N = -m\ddot{x} - ml\dot{\theta}^2 \sin \theta + ml\ddot{\theta} \cos \theta$$

Equation 2

Summing the forces in the free body diagram of the pendulum in the vertical direction, we have

$$F_{reaction\ from\ the\ cart} - F_{gravitational} + F_{centrifugal} + F_{tangential} = 0$$

$$\begin{aligned} \rightarrow P - mg + I\omega^2 \cos \theta + I\alpha \sin \theta &= 0 \\ \rightarrow P - mg + ml\dot{\theta}^2 \cos \theta + ml\ddot{\theta} \sin \theta &= 0 \\ \rightarrow P &= mg - ml\dot{\theta}^2 \cos \theta - ml\ddot{\theta} \sin \theta \end{aligned} \quad \text{Equation 3}$$

Substituting Equation 2 into Equation 1, we have

$$\begin{aligned} F &= M\ddot{x} + b\dot{x} - m\ddot{x} + ml\dot{\theta}^2 \sin \theta - ml\ddot{\theta} \cos \theta \\ \rightarrow F &= (M + m)\ddot{x} + b\dot{x} + ml\dot{\theta}^2 \sin \theta - ml\ddot{\theta} \cos \theta \end{aligned} \quad \text{Equation 4}$$

Summing moments about the center of mass, we have

$$\begin{aligned} I\ddot{\theta} &= Pl \sin \theta - Nl \cos \theta \\ \rightarrow I\ddot{\theta} &= (mg - ml\dot{\theta}^2 \cos \theta - ml\ddot{\theta} \sin \theta)l \sin \theta - (-m\ddot{x} - ml\dot{\theta}^2 \sin \theta + ml\ddot{\theta} \cos \theta)l \cos \theta \\ \rightarrow I\ddot{\theta} &= mgl \sin \theta - ml^2\ddot{\theta} + ml\ddot{x} \cos \theta \\ \rightarrow (I + ml^2)\ddot{\theta} - mgl \sin \theta &= ml\ddot{x} \cos \theta \end{aligned} \quad \text{Equation 5}$$

6.2. Linearization

Before a model can be made in Simulink, Equation 4 and Equation 5 have to be linearized. Assuming a small deviation θ from equilibrium, the following small angle approximations of the nonlinear functions can be used:

$$\begin{aligned} \theta &\cong 0 \\ \cos \theta &\cong 1 \\ \sin \theta &\cong \theta \\ \dot{\theta}^2 &\cong 0 \end{aligned}$$

This results in the equations of motion expressed by Equation 6 and Equation 7.

$$F = (M + m)\ddot{x} + b\dot{x} - ml\ddot{\theta} \quad \text{Equation 6}$$

$$(I + ml^2)\ddot{\theta} - mgl\theta = ml\ddot{x} \quad \text{Equation 7}$$

Before using a Laplace transformation to solve the differential equations, a physical machine must be analyzed in order to know which characteristics have influence on the equations.

7. Traditional motors

The XTS system is a combination of rotary and linear systems. In order to understand the basic principle of the XTS system, this chapter briefly describes a working principle of a DC motor, a servo motor, and a linear motor.

7.1. DC motor

A DC motor, which uses direct current, converts electrical energy to mechanical energy. The DC motor has two electrical components called stator and rotor. A current, from a power source, runs through a stator, which contains a set of coil windings, generating an electromagnetic field that lets the rotor rotate. The coil windings on the stator are turned ON and OFF in a sequence to make the rotor keep spinning. As a result, torque and speed will be formed. The output torque and speed depend on the input voltage and the design of the motor.

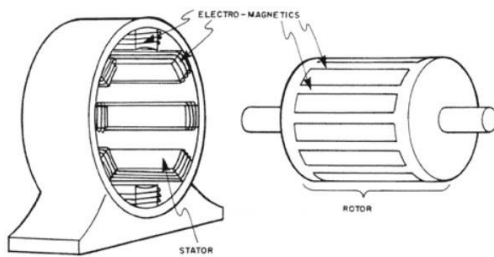


Figure 7: Electrical components of a DC motor [5]

Figure 8 shows a basic principle of how to control DC motors with a small “computer.” Two DC motors are connected to a drive which is controlled by an Arduino board. By using an Arduino IDE and a small program, making two DC motors work is easy.

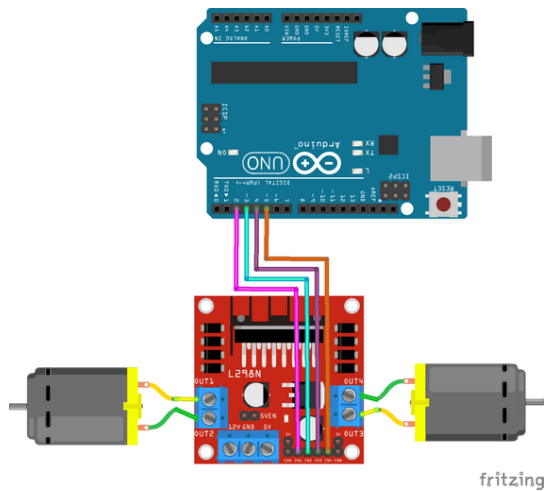


Figure 8: DC motors are driven by an Arduino board [6]

7.2. Servo motor

The concept of servo motors is quite similar to the DC motor, but the servo motor is constantly monitored to control its motion. Besides the stator and rotor, the motor uses sensors, feedback encoder and controller to create a closed-loop system, allowing accurate control of its position, torque, velocity, and acceleration.

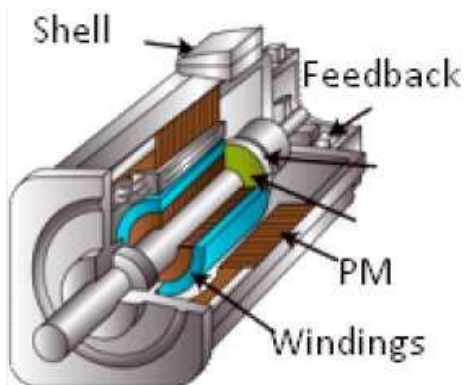


Figure 9: Servo motor components [7]

Controlling the servo motor is similar to the DC motor. The motor will be connected to a drive which is driven by a small computer. By using a sophisticated IDE and a small piece of program, the motor will work.

7.3. Linear motor

A DC motor and servo motor are commonly called rotary motor because a rotor or a shaft keeps spinning within a stator. A linear motor does not have a spinning feature. In fact, a “shaft” of the

linear motor moves back and forth along a track. The following picture is illustrating the relationship between the rotary motor and the linear motor.

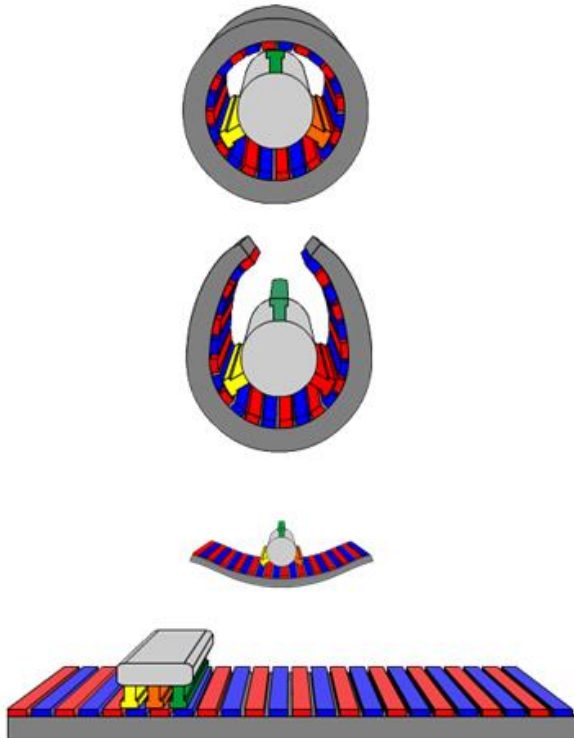


Figure 10: The stator has been "cut" and flattened out [8]

As can be seen, the linear motor has been constructed the same as the rotary motor but flattened out. A current, from a power source, runs through the flattened coils magnetizing the phases north or south, which results in an electromagnetic field. By turning the coils ON and OFF in a sequence, a carriage with its own permanent magnets can be moved in a desired direction. A load can be attached to the carriage that moves along the electromagnetic field. With this principle, the linear motor can be used in transportation.

Controlling the linear motor is the same as the rotary motor. The motor will be connected to a drive which is driven by a small computer. By using a sophisticated IDE and a small piece of program, the motor will work as desired.

8. Inverted Pendulum setup

XTS, eXtended Transport System, is a linear transport system from BECKHOFF that moves in a circle or in desired geometries that are suited for applications, for example: straight, s-shape, square, rectangle, as shown in Figure 11. The machine can have one (or several) wireless “movers” that can be moved highly dynamically up to a velocity of 4 m/s. The “movers” can accelerate, position, and brake. They can group themselves, avoid collisions and queue. [9]

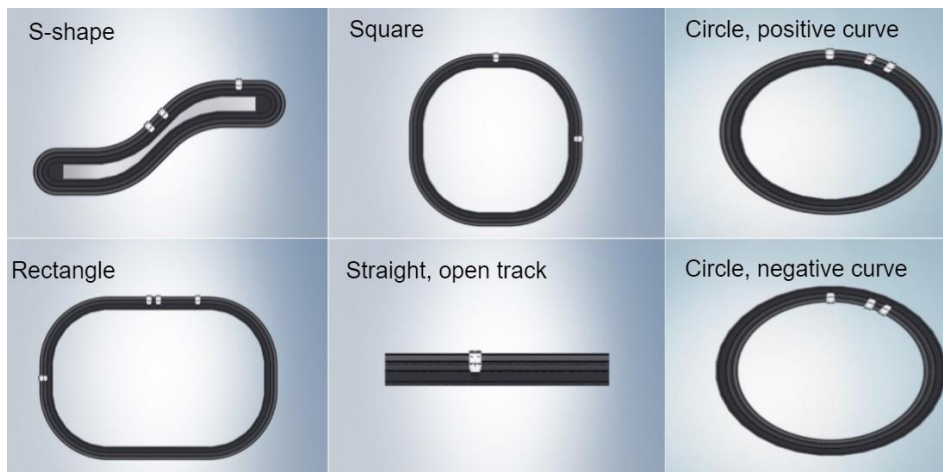


Figure 11: Flexible track layouts [9]

Figure 12 is a representation of a complete XTS system. The system consists of movers, guide rail, motor modules and a control system. Communication between the XTS system and a PC is done by the EtherCAT protocol [10]. In the graduation assignment, straight modules and one mover will be used, as shown in Figure 13.

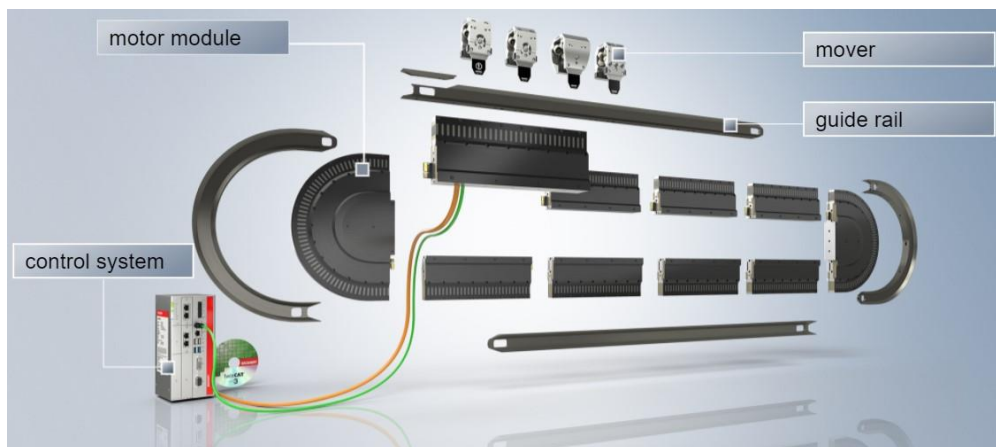


Figure 12: A complete XTS system [9]

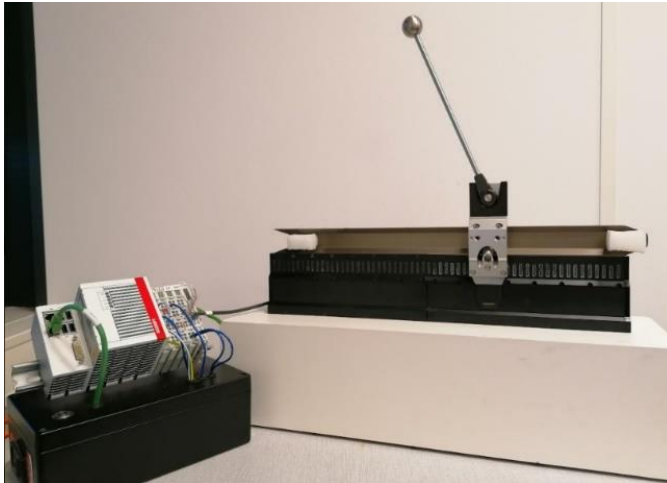





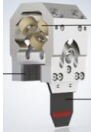


Figure 13: XTS module used in the project

8.1. Traditional motors and the XTS system

The following table shows a connection between traditional motors and the XTS system.

Feature	Traditional motors		XTS system
	Rotary motor	Linear motor	
Stator	 [11]	 [8]	 [9] Called Motor module
Rotor/Carriage	 [12]	 [8]	 [9] Called Mover
Drive	L298N...		Integrated inside the motor module
Controller	Arduino Uno, Raspberry Pi...		BECKHOFF IPC CX5140
IDE	Arduino, Atmel studio, PyCharm...		TwinCAT3.1

Technically speaking, a giant XTS system can be interpreted as a small servo motor, but it has been developed in such a way that it can be used in transportation. The XTS system uses a servo technology to drive the mover. That means a current, from a power source, runs through each motor module creating an electromagnetic field. By turning the coils ON and OFF in a certain sequence, the mover with its own permanent magnets can be moved in a desired direction. A built-in feedback system is monitoring its motion.

8.1. Connection diagram

The following diagram shows different components connected electrically, and how data is exchanged.

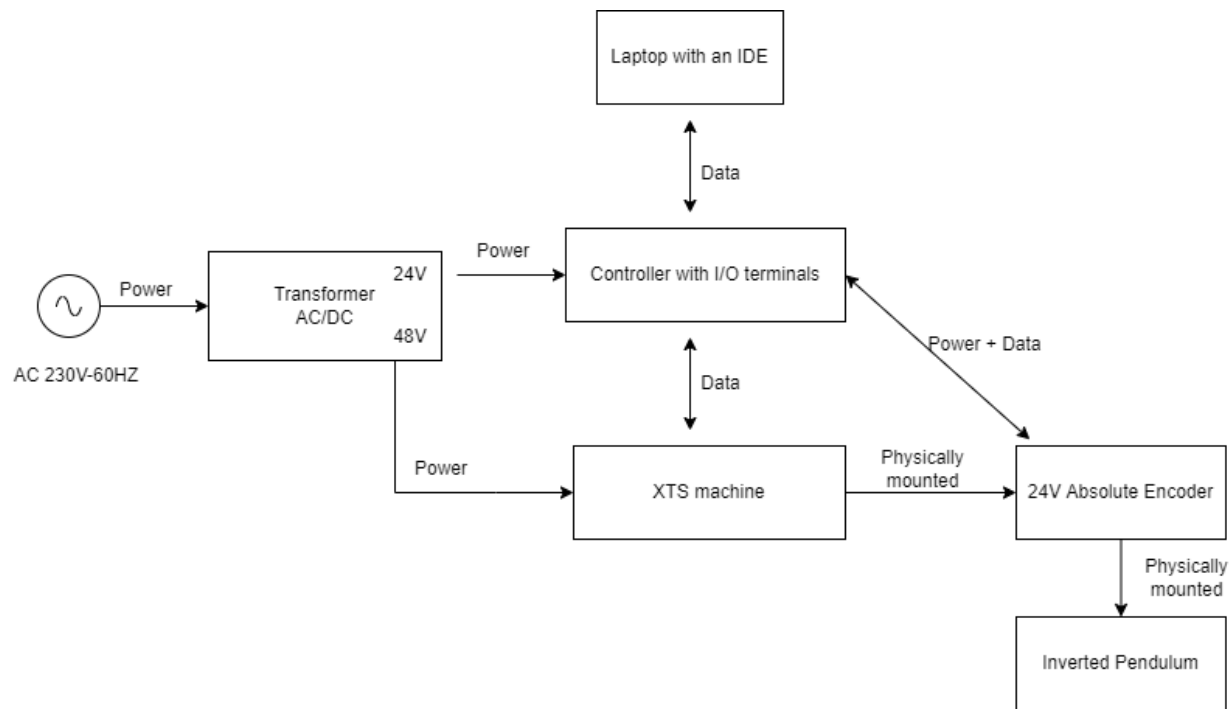


Figure 14: Connection diagram

8.2. XTS machine itself

The XTS itself has three physical components: motor modules, a mover, and a guide rail. These components will be explained in the following chapters to get a better understanding of the working system.

8.2.1. Motor module

The XTS motor module can be interpreted as the stator of DC motors.

The system consists of individual motor modules that can be combined to form a complete “roundabout.” There are 2 types of motor module used in the system: with and without supply cables and data cable. The length of each motor module is 250 mm, in total the system in the project will be 500 mm in length.

The article-name for the motor module with supply cables and data cables is: AT2001-0250

The article-name for the motor module without supply cables and data cables is: AT2000-0250



Figure 15: Motor module with supply and data cable (left), motor module without supply cables (right) [9]

The motor module consists of a coil package to generate a magnetic field for the mover. EtherCAT and power for control at 24V DC, and power for the motor coil at 48V DC, are passing from module to module via the PCB between them. For power electronics, the motor current is controlled directly on the motor module, and temperature is internally monitored to prevent overload.

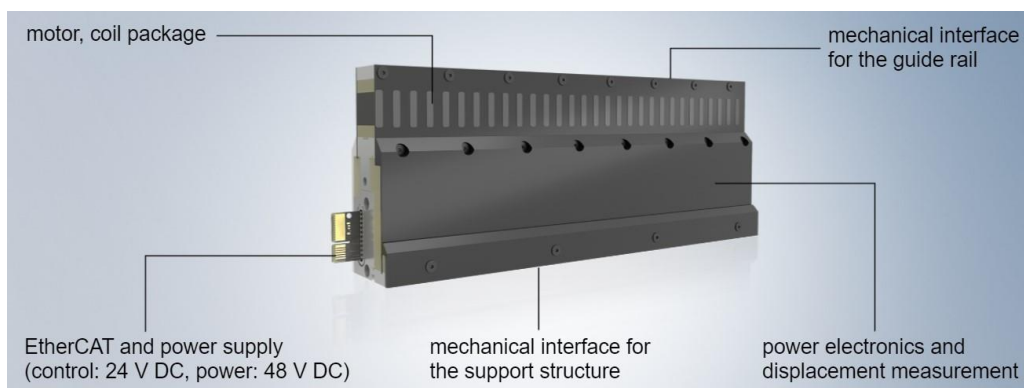


Figure 16: Motor module characteristics [9]

8.2.2. Mover

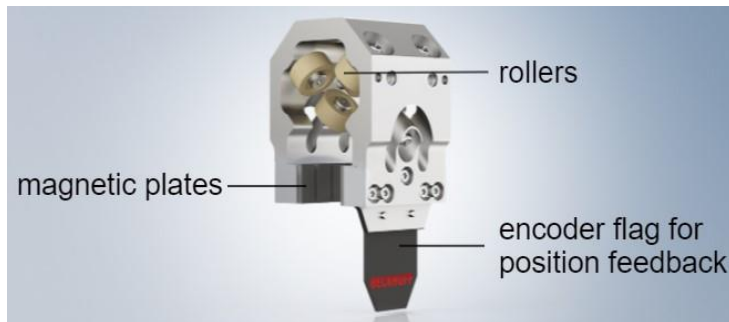


Figure 17: Mover [9]

The XTS mover can be interpreted as the shaft of DC motors.

The article-name for the XTS mover used in this project is AT9011-0050-0550. There is a variety of different movers based on different requirements like speed, weight, lifetime etc.

The XTS mover with a width of 50 mm has 6 guide rollers, made of Teflon which a good combination for low friction and reasonable lifetime of the rollers, and mounted permanent magnets. As mentioned above, a magnetic field, which is created by the motor modules, is used to propel the mover. The position feedback is generated within the motor module. The system uses non-contact position detection feedback utilizing an encoder flag attached to the mover.

The mover is entirely passive and purely mechanical. There are no sliding contacts or cables attached to the mover. Individual energizing of coils of the motor module generates a travelling magnetic field, which moves the magnetic plates of the mover along with it.

The lifetime of the rollers depends on the payload, speeds, surroundings, and application.

8.2.3. Guide rail

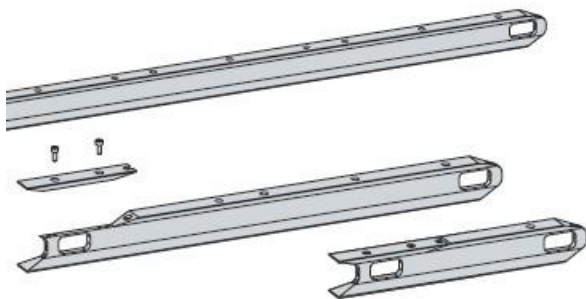


Figure 18: Guide rail [13]

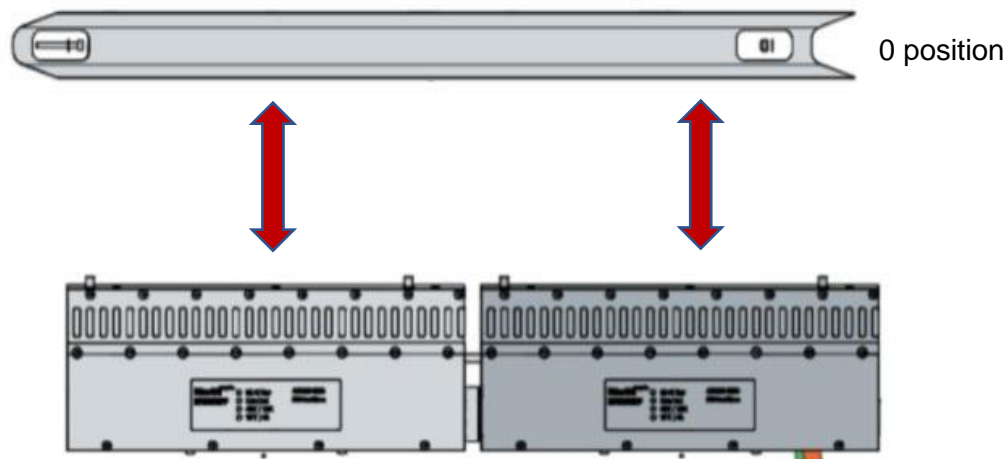


Figure 19: Simple assembly of the guide rail via mechanical interface on the motor module [13]

The article-name for the guide rail used in this project is: Guide rail with lock, 500mm in length.

In addition to the motor modules, a rail system is needed for the mover to travel smoothly and easily. The guide rail is designed intentionally like an arrow. The encoder counts in the direction of the guide rail. The 0 position is at the tail of the arrow unless defined otherwise in the parameters.

8.3. Controller

An Arduino board or a Raspberry Pi is a tiny computer that is used to explore the electronic world, for example, in schools. By using a sophisticated IDE running on a personal computer, a piece of code can be transferred from a user computer to the dedicated board via a communication protocol.

At BECKHOFF, these types of computers are called IPC's, Industrial PC's, the sophisticated IDE is called TwinCAT Engineering, and EtherCAT is the communication protocol.

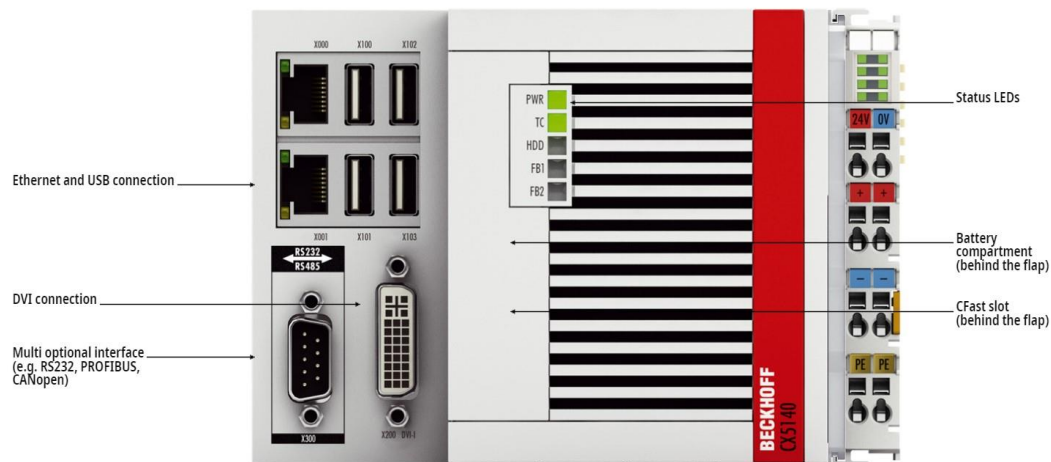


Figure 20: CX5140 | Embedded PC with Intel Atom® processor [14]

The CX5140 is an embedded PC with an Intel Atom quad-core processor. Its operating system can be Windows 7, Windows 10, Windows CE or TC/BSD. Depending on the TwinCAT runtime environment, the CX5140 can be used for implementing PLC or PLC/motion control projects with or without visualization.

Unlike the Arduino board or the Raspberry Pi, the CX5140 itself does not have I/O terminals integrated. The I/O terminals can be physically plugged in, or taken out depending on what users need, allowing more freedom of selecting components and building a control cabinet.

The following pictures show how the system can be extended.

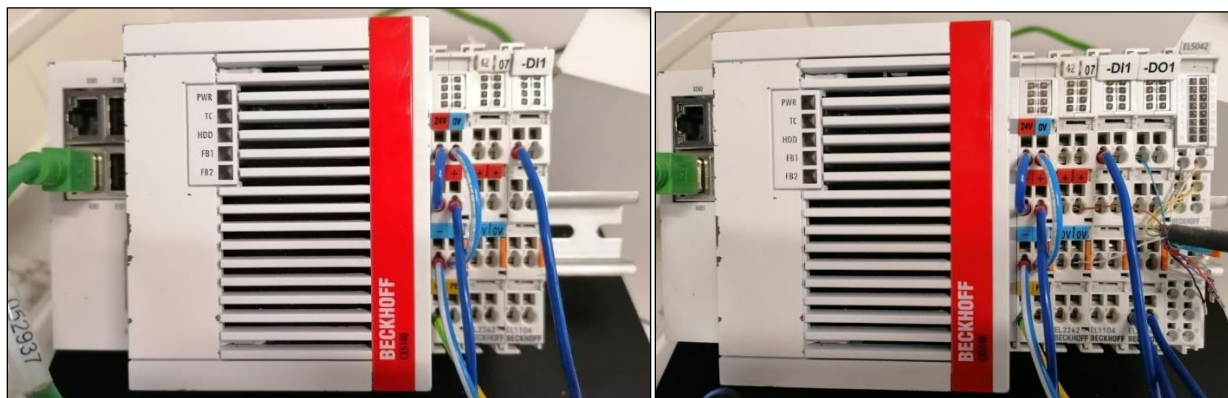


Figure 21: Expandable system. Left: 2 terminals are added. Right: 4 terminals are added

8.4. Controlling the mover

There are two possible ways to control the mover: with and without programming ("manual" moving). Both solutions can be done within TwinCAT Engineering.

Before controlling the XTS mover, a tuning process needs to be taken into consideration. More details can be found in Appendix F.

8.4.1. Movement without programming

By using the NC Axis, Numerical Control, in TwinCAT, the mover can be controlled manually. This method is used for testing and tuning purposes.

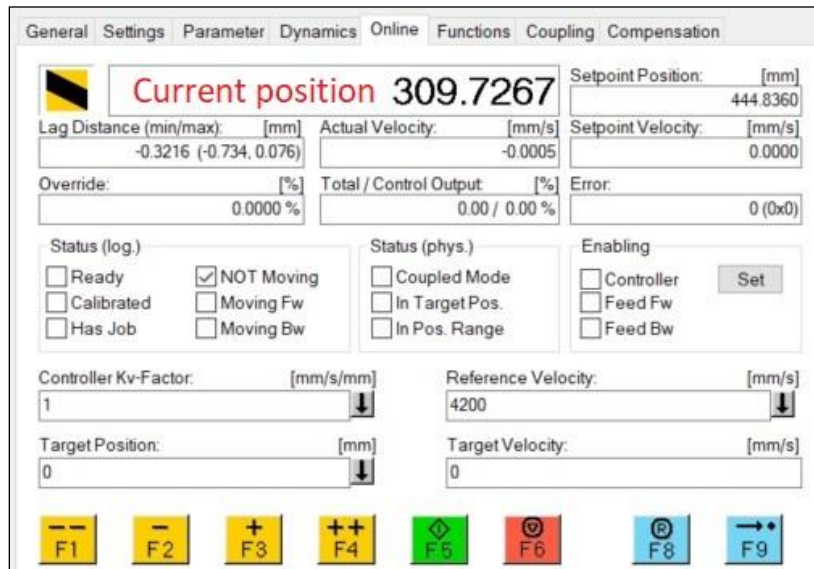


Figure 22: System operation

To move manually, first set the controller enable by meaning of pressing the Set button. There are 8 buttons to controller a mover.

- F1: The mover will travel instantly with a velocity of 600mm/s in the negative direction.
- F2: The mover will travel instantly with a velocity of 100mm/s in the negative direction.
- F3: The mover will travel instantly with a velocity of 600mm/s in the positive direction.
- F4: The mover will travel instantly with a velocity of 100mm/s in the positive direction.
- F5: Start movement. For example: a target position of 500mm, and target velocity is 300mm/s. As soon as the F5 button is pressed, the mover will move to the target position with the target velocity.
- F6: Stop movement
- F8: Reset any error
- F9: Start the homing sequence

8.4.2. Movement with programming

Function blocks from the TC2_MC2 library, developed by BECKHOFF, can be used for controlling motors from the PLC. For example, a MC_MoveAbsolute function is used to start positioning to an absolute target position and monitors the axis movement over the entire travel path.

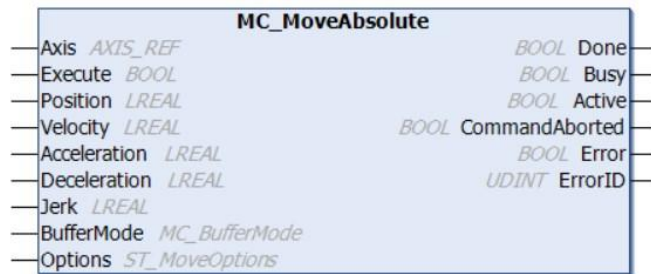


Figure 23: Inputs and Outputs of the MC_MoveAbsolute function [15]

MC_MoveVelocity starts a continuous movement with specified velocity and direction. Position input can be ignored. The NC itself will calculate the setpoint for position and acceleration for the next cycle.

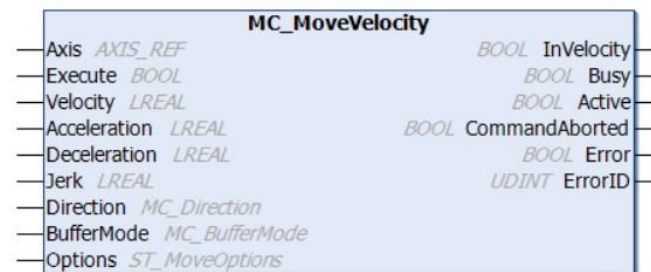


Figure 24: Inputs and Outputs of the MC_MoveVelocity function [16]

In order to use these move functions (or other function in the TC2_MC2 library) and to have a desired sequence, a state machine has to be built. In the following example, the state machine has been designed with the MC_MoveAbsolute function. In this state machine, a mover will move forwards and backwards.

Besides the MC_MoveAbsolute function itself, the MC_Power function and the MC_Reset function must be used as well.

- MC_Power: An axis (the mover) will be enabled and powered.
- MC_Reset: Reset any error during movement.

Figure 25 shows a state diagram for the move-process of an axis.

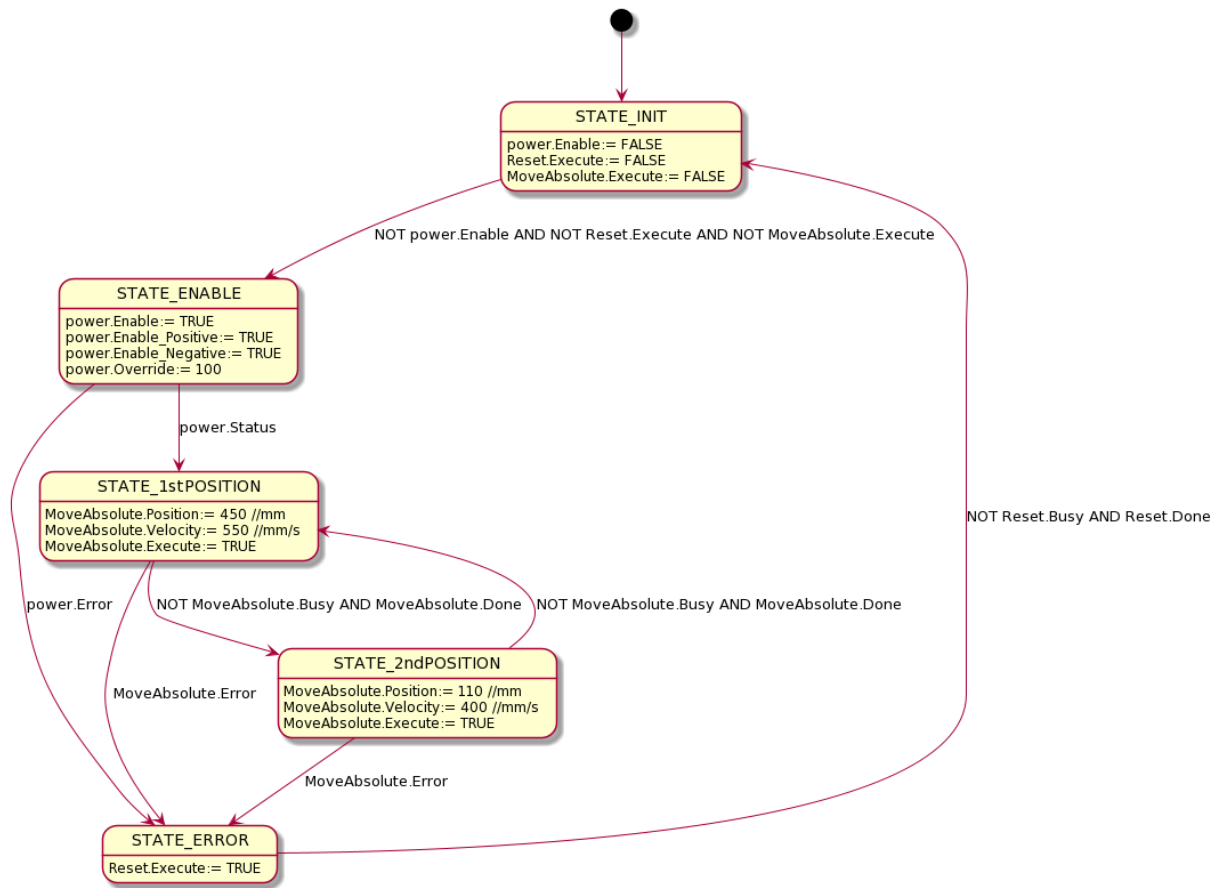


Figure 25: State Machine for the MC_MoveAbsolute function. Full UML code can be found in Appendix G

There are many more functions that can be used for different purposes. Their descriptions and functionalities can be found in the information system.

Appendix D will explain in depth which move functions are suitable for the project.

8.5. Pendulum encoder



Figure 26: Pendulum encoder [17] and mounting units

An encoder is used to measure the angular position of the pendulum. The encoder is attached to the mover through mounting units, as shown in Figure 26. The pendulum is mounted directly to the rotational shaft of the encoder. By measuring the distance or displacement of the pendulum in a rotary motion, the encoder converts rotational movements into electrical signals.

The following table shows the technical data of the encoder.

Company	Kubler
Order code	8.F3653.3543.C412
Interface protocol	BiSS-C
Supply voltage	10-30V DC
Resolution	14-bit
Encoder type	Absolute encoder, single turn

Table 1: Technical data of the encoder [17]

BiSS stands for Bidirectional Serial Sensor Interface. It is a communication protocol used to transmit data between a transmitter and a receiver. In the project, the transmitter is the encoder of the pendulum, and the receiver is the BECKHOFF IPC. BiSS can be configured as uni-directional or bi-directional communication. The uni-directional communication means data can only be transmitted in one way which is from the transmitter to the receiver. It is not possible for the receiver to send configuration data to the transmitter. On the other hand, for the bi-directional

communication, data can be exchanged between the receiver and the transmitter at once without interrupting sensor data transmission. [18]

With the absolute single turn encoder, a coded position value is assigned to each angular position to record the absolute position. Even after the encoder has been disconnected and connected again, the position is still available. The encoder has a resolution of 14-bit single turn, that means it will count from 0 to 16383 ($2^{14} - 1$) over one rotation and then starts from 0 again.

$$\text{Angular position} = \frac{\text{Output value from the encoder}}{16384} * 2\pi \text{ (or } * 360^\circ) \quad \text{Equation 8}$$

The connection wiring is shown in Appendix E.

8.5.1. Homing position of the pendulum

In this project, a homing position of the pendulum is defined when the pendulum stays naturally to the left, as shown in Figure 28. According to the datasheet of the encoder, there is one pin used to set a zero position, called SET input. As soon as the pin is set to HIGH, the homing position will be set to 0.

SET input	
Input	active HIGH
Input type	comparator
Signal level (+V = power supply)	HIGH min. 60 % of +V, max: +V LOW max. 30 % of +V
Input current	< 0.5 mA
Min. pulse duration (SET)	10 ms
Input delay	1 ms
New position data readable after	1 ms
Internal processing time	200 ms
<p>The encoder can be set to zero at any position by means of a HIGH signal on the SET input. Other preset values can be factory-programmed. The SET input has a signal processing time of approx. 1 ms, after which the new position data can be read via SSI or BiSS. Once the SET function has been triggered, the encoder requires an internal processing time of typ. 200 ms; during this time the power supply must not be switched off.</p> <p>The SET function should be carried out whilst the encoder is at rest.</p> <p>If this input is not used, it should be connected to 0 V (Encoder ground GND) in order to avoid interferences.</p>	

Figure 27: Homing position function

8.5.2. Counting direction of the encoder

In this project, the counter value, or the output value from the encoder, will be increased at clockwise rotation, and vice versa. The counter value will be decreased at counterclockwise rotation.



Figure 28: Counting direction

According to the datasheet, it is possible to change the counting direction from counterclockwise (CCW) to clockwise (CW), and vice versa. However, this function will be not used. The positive counting direction is clockwise.

9. Controller design

To stabilize the inverted pendulum, in this chapter, a controller will be chosen and simulated in MATLAB/Simulink. Its result will be explained in more detail.

9.1. Laplace transform

As explained in Appendix D, velocity is one of the inputs that can be used to control the mover, so Equation 7 needs to be adjusted.

$$(I + ml^2)\ddot{\theta} - mgl\theta = ml\ddot{x}$$

$$\rightarrow (I + ml^2)\ddot{\theta} - mgl\theta = ml\dot{v}$$

Taking the Laplace transformation of the linearized equations of motion, we get

$$(I + ml^2)s^2\Theta - mgl\Theta = mlsV$$

$$\rightarrow ((I + ml^2)s^2 - mgl)\Theta = mlsV \quad \text{Equation 9}$$

9.2. Transfer function

From Equation 9 can be obtained:

$$\frac{\Theta}{V} = \frac{mls}{(I + ml^2)s^2 - mgl} \quad \text{Equation 10}$$

By making the denominator of Equation 10 equal to 0, we can see that there is no free “s”s, so the type of system is 0.

In addition, from the transfer function, we get

- Zero at $s = 0$
- Poles at $s = \pm \sqrt{\frac{mgl}{I+ml^2}}$

There is one pole located in the right half plane, so the system is unstable in open loop.

9.3. Parameter identification

9.3.1. Travelling distance of the mover

Due to safety purposes, there are two plastic pieces added to the guide rail to prevent the mover from travelling out of the system. Because of that, the travelling distance of the mover is shortened. With the help of TwinCAT and the encoder flag on the mover, the travelling distance can be precisely measured, as shown in Figure 29.

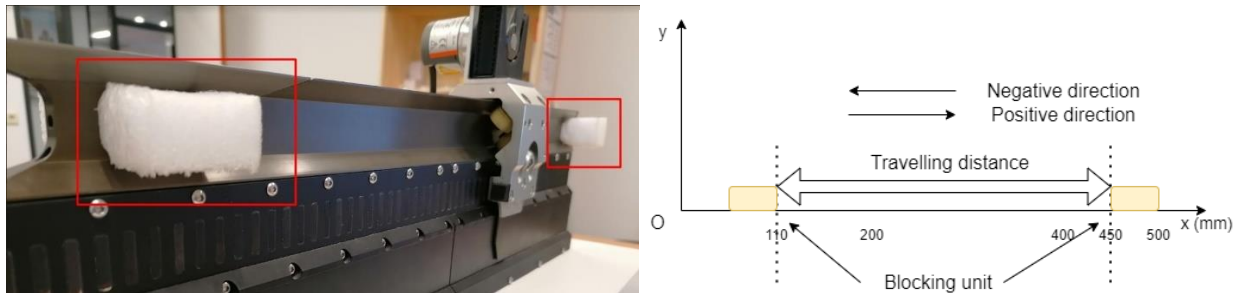


Figure 29: Two blocking plastic pieces and the coordinate system

9.3.2. Mass and length of the pendulum

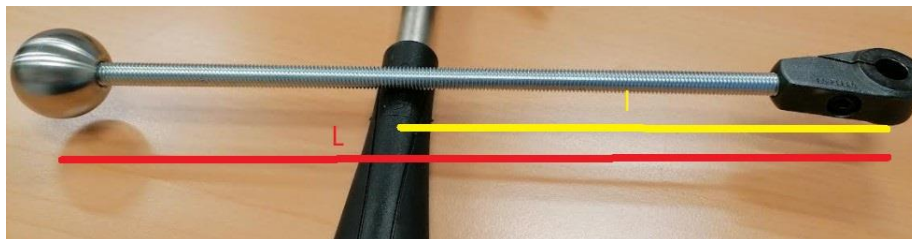


Figure 30: Pendulum being balanced around its center of mass

The mass of the pendulum, including a mounting base and a ball, is determined with the help of a scale $m = 0.254(\text{kg})$. A ruler was used to identify the length of the pendulum $L = 0.275(\text{m})$

The length to pendulum center of mass, $\ell = 0.16(\text{m})$

Feature	Value
m	0.254(kg)
L (total length of the pendulum)	0.275(m)
l (length to pendulum center of mass)	0.16m

9.3.3. Mass moment of inertia of the pendulum (I)

The moment of inertia can be easily calculated with the formula for I below. In this formula, it is assumed that the system weight is at the center of mass of the pendulum.

$$I = \frac{ml^2}{3} = \frac{0.254 \times 0.16^2}{3} = 2.167 \times 10^{-3} (kg.m^2)$$

9.3.4. Mass of the mover



Figure 31: Mover and mounting units

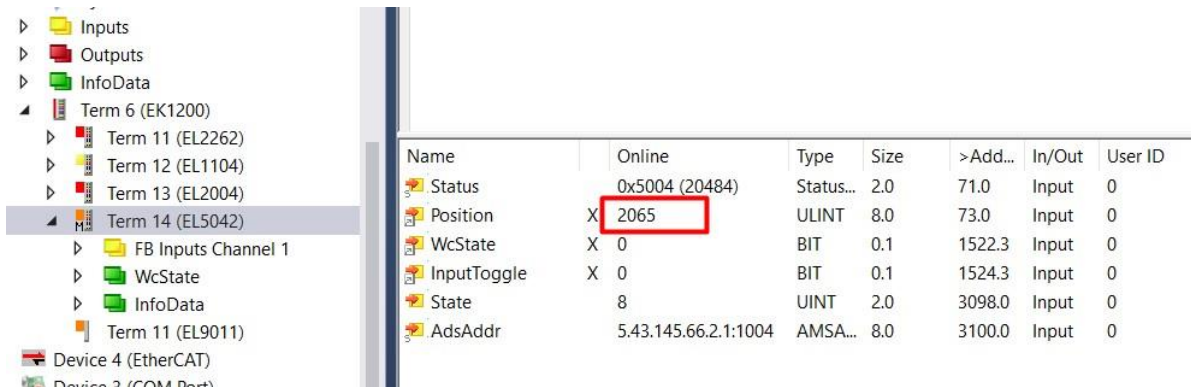
The mass of mover consists of the mover itself, the L unit, the encoder, and the blocking angle unit.

Name	Mass (kg)
Mover	0.41
L unit + Encoder	0.181
Blocking angle unit + 2 bolts	0.018
Total	0.609

9.3.5. Angular displacement of the blocking angle unit

Because of the short travelling distance, a blocking-angle was designed, so the pendulum will not turn towards the ground. With the help of TwinCAT, the angular position can be calculated by the following equation.

$$Angle = \frac{Reading\ value}{16384} * 2\pi \text{ (or } * 360^\circ)$$

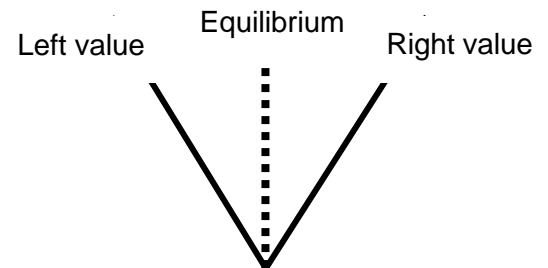


Name	Online	Type	Size	>Add...	In/Out	User ID
Status	0x5004 (20484)	Status...	2.0	71.0	Input	0
Position	2065	ULINT	8.0	73.0	Input	0
WcState	0	BIT	0.1	1522.3	Input	0
InputToggle	0	BIT	0.1	1524.3	Input	0
State	8	UINT	2.0	3098.0	Input	0
AdsAddr	5.43.145.66.2.1:1004	AMSA...	8.0	3100.0	Input	0

Figure 32: Output value from the encoder shown in TwinCAT (the red box)

Measuring procedure:

- Move the pendulum by hand to the left side of the blocking angle unit. The SET pin of the encoder must be set HIGH to reset the homing position. Record the reading value from the encoder, in this case, 0.
- Move the pendulum by hand to the right side of the blocking angle unit and record the reading value from the encoder, in this case, 2065.



$$\begin{aligned}
 Angle &= \frac{RightValue - LeftValue}{16384} * 2\pi \text{ (or } * 360^\circ) \\
 &= \frac{2065 - 0}{16384} * 2\pi \text{ (or } * 360^\circ) \\
 &= 0.252\pi \text{ (or } 45.373535^\circ)
 \end{aligned}$$

Since we want the pendulum to stabilize around the equilibrium point, the angular position of the equilibrium point is



Figure 33: Angular displacement

$$\frac{0.252\pi}{2} = 0.126\pi \left(\text{or } \frac{45.373535}{2} = 22.686767^\circ \right)$$

The above-mentioned value is also the setpoint value for the controller.

Also, using the earth gravity, the setpoint value can be found easily, as shown in Figure 34.

$$\begin{aligned} \text{Angle} &= \frac{\text{Reading value}}{16384} * 2\pi \text{ (or } * 360^\circ) \\ &= \frac{1033}{16384} * 2\pi \text{ (or } * 360^\circ) = 0.126\pi \text{ (or } 22.6978^\circ) \end{aligned}$$



Figure 34: Checking the setpoint value with the earth gravity

9.4. Selecting a controller and simulating

In order to stabilize the inverted pendulum in its upright position, a feedback control system can be used, which monitors the pendulum's angle and moves the mover sideways when the pendulum start to fall over.

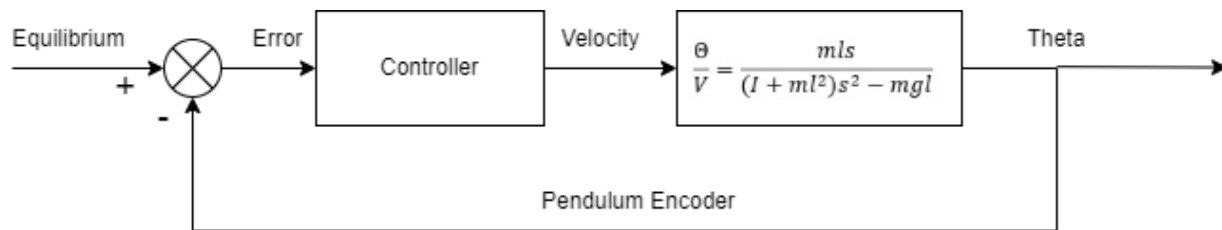


Figure 35: A closed-loop system

A PID controller has been selected first because of its simplicity. A PID controller controls a process through three parameters: Proportional (P), Integral (I), and Derivative (D). These parameters can be weighted, or tuned, to adjust their effect on the process.

$$u(t) = K_p \text{error}(t) + K_i \int_0^t \text{error}(t) dt + K_d \frac{d}{dt} \text{error}(t)$$

A PID controller works by calculating the error between the output of the Process and a given desired value. With this error, an output is calculated based on equation above.

- **Proportional:** the actual error is multiplied by a gain of K_p , which means that the bigger the error, the bigger the output. Ideally, this constant should be as large as possible to get

a quick response but increasing the gain too much can cause oscillation and overshooting in output values.

- **Integral:** the error is integrated over time and multiplied by K_i , making it possible to correct the constant error that the proportional part can have. The integration is often approximated as the sum of errors in each time step. The I-Control will increase the type of system by one, meaning one pole will be added to the system.
- **Derivative:** the error is derived with respect to time in order to get the error change rate, and then multiplied by K_d . The derivative action is sensitive to noise, which could cause overshooting in output values.

Using a PID tuner in Simulink, controller gains can be found automatically.

9.4.1. Importing data to MATLAB

The first step is importing data and the transfer function to MATLAB.

<pre>mp = 0.254 % mass of the pendulum (kg) l = 0.16 % length to pendulum center of mass (m) I = 2.167*10^-3 % mass moment of inertia of the pendulum g = 9.81 % gravity (m/s^2) A = mp*l B = (I+mp*l^2) C = -mp*g*l G_theta_velocity = tf([A,0],[B,0,C])</pre>	<pre>mp = 0.2540 l = 0.1600 I = 0.0022 g = 9.8100 A = 0.0406 B = 0.0087 C = -0.3987 G_theta_velocity = 0.04064 s ----- 0.008669 s^2 - 0.3987 Continuous-time transfer function.</pre>
---	--

Figure 36: Importing data to MATLAB

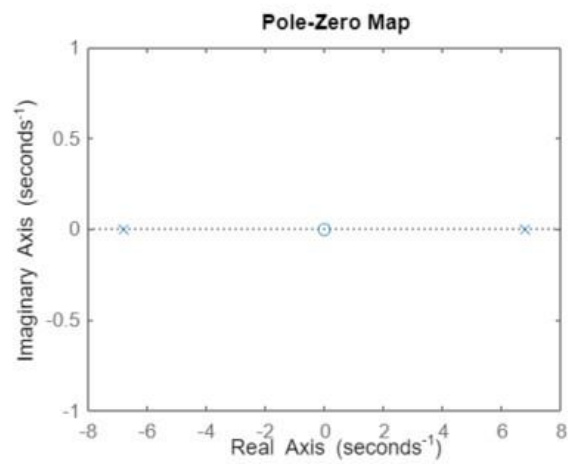


Figure 37: 1 zero and 2 poles

9.4.2. P-Controller

At first, a P-Controller is selected.

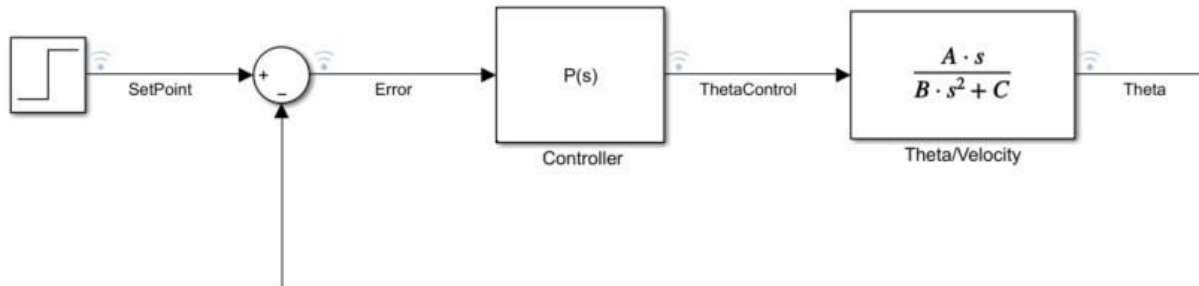


Figure 38: P-Controller

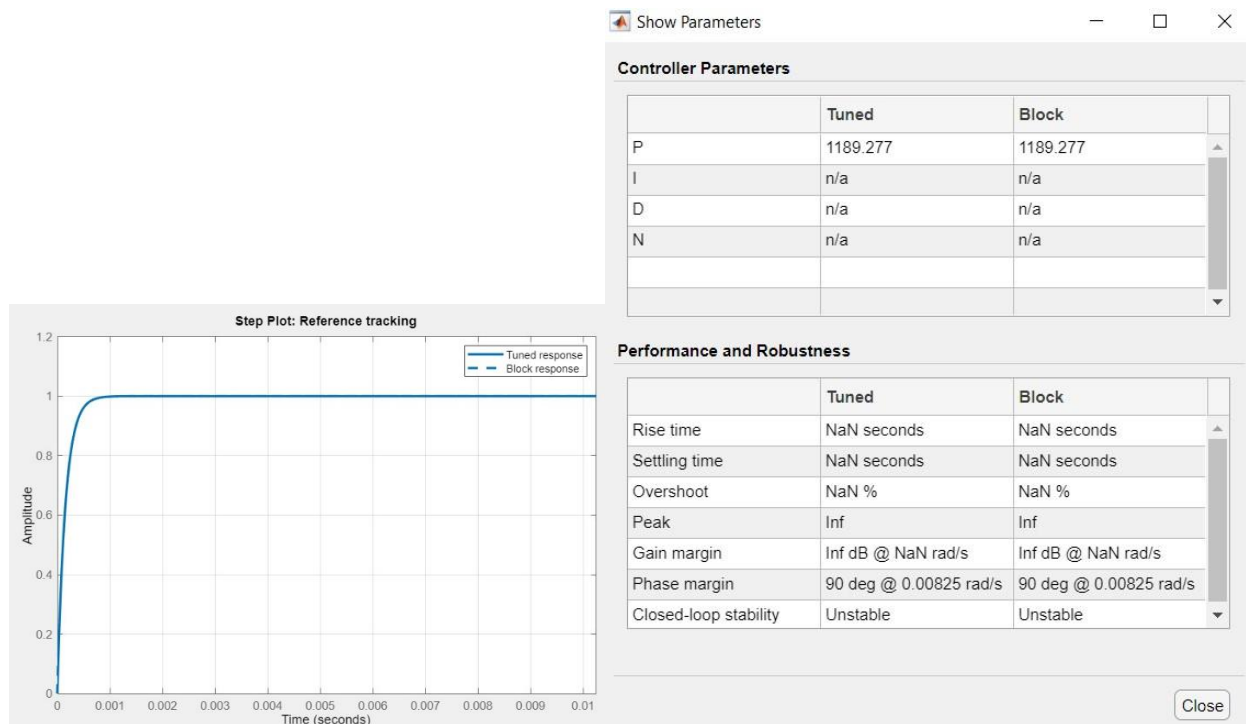


Figure 39: P-Controller results

As shown in Figure 39, the steady state error becomes 0, but on the Show Parameters window, it says that the closed-loop system is unstable. That means the P-Control is not suitable for the system.

9.4.3. PI-Controller

Next, a PI-Controller is selected.

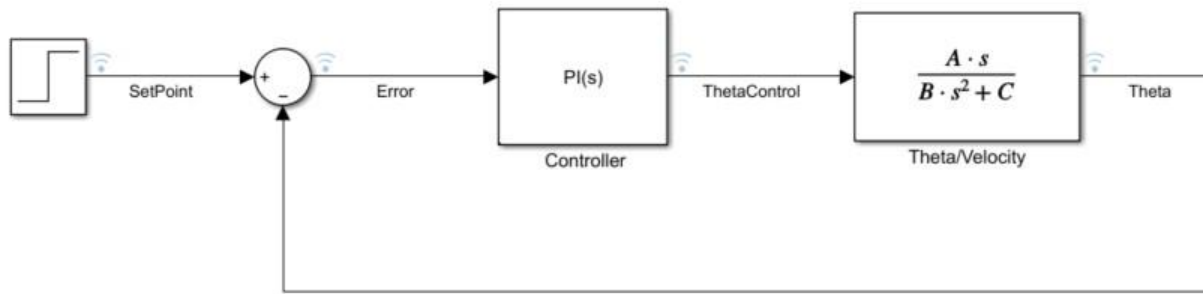


Figure 40: PI-Controller

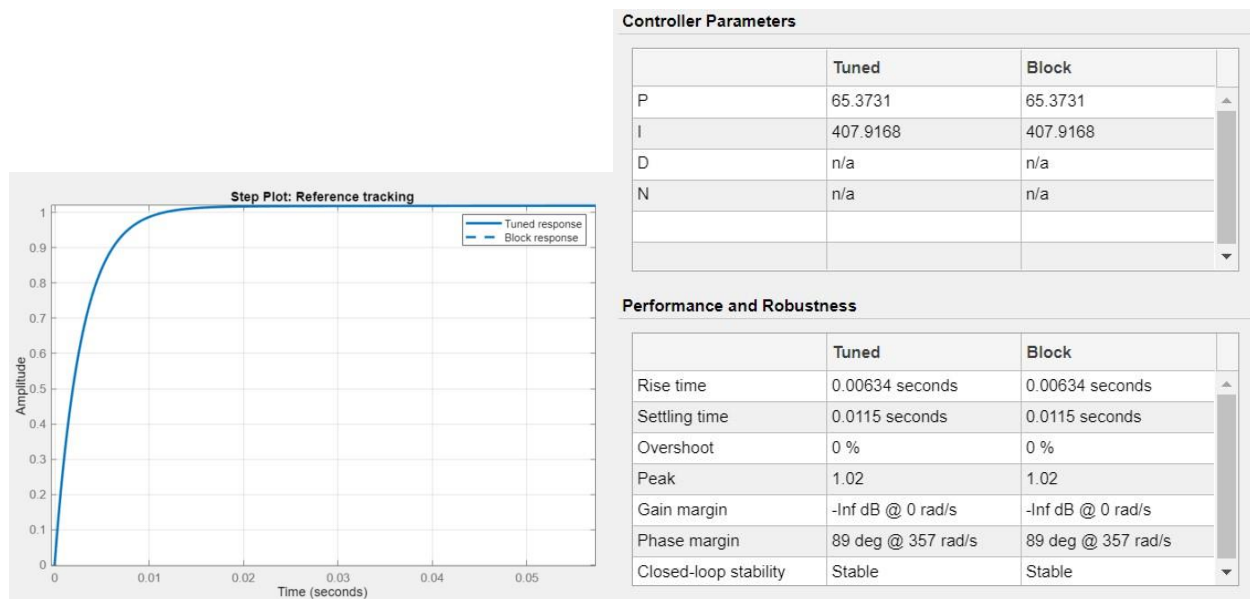


Figure 41: PI-Controller results

As can be seen from Figure 41, the steady state error does not become 0. In fact, there is a 1.02 – 1 = 0.02 difference, and the PID tuner says the closed-loop system is stable.

A simulation has to be run to check the results in the data inspector.

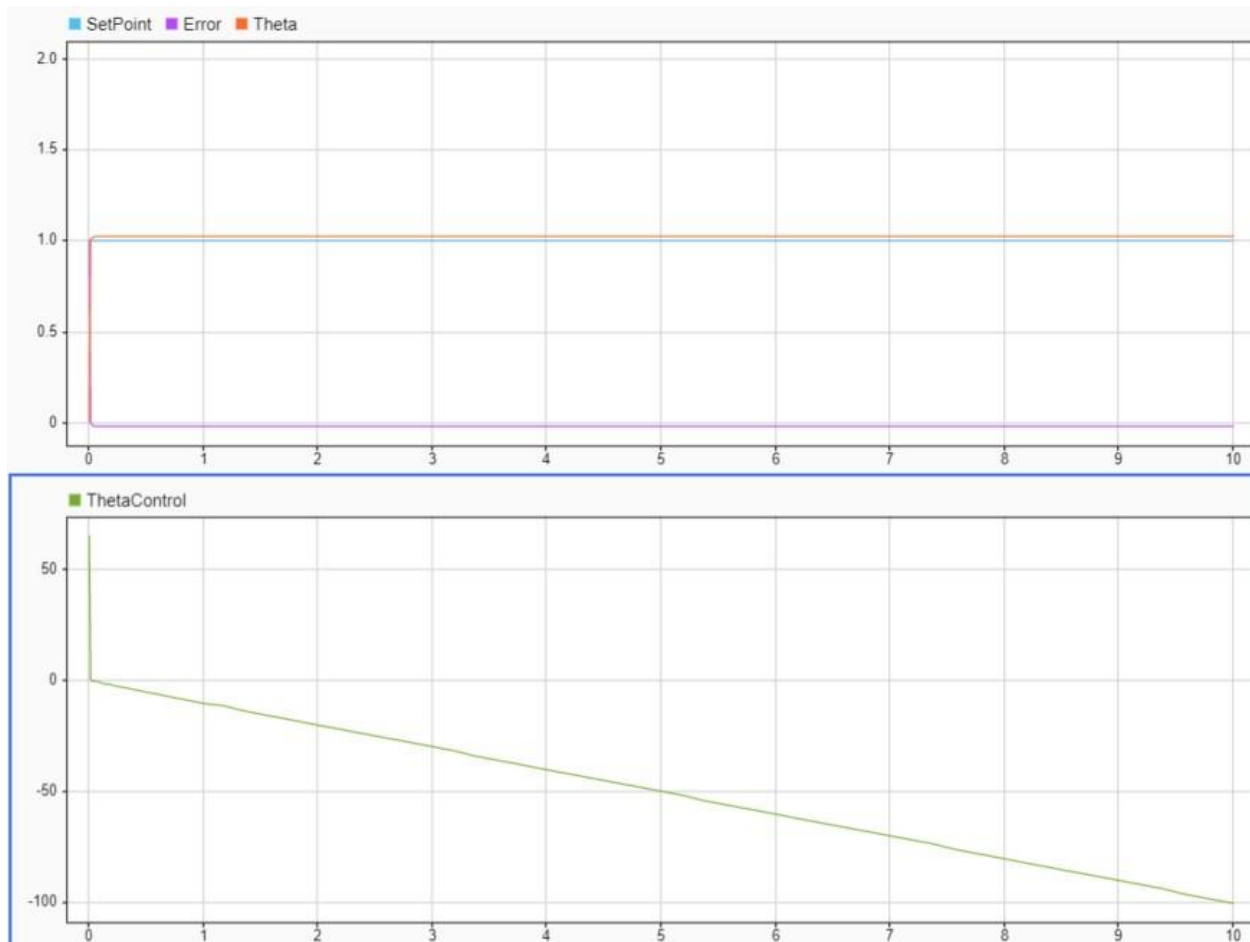


Figure 42: Data inspector for the PI-Controller

On the first graph of Figure 42, we can see that the output, the orange line, remains constant, and it does follow the input with a 0.02 difference, as explained above.

On the second graph of Figure 42, the green line, which is the output of the PI-controller, which is also the control signal, shrinks unboundedly as time increases toward infinity.

So, the PI-Controller is not suitable for the system.

9.4.4. PID and PD-Controller

Both PD and PID-Controller give the same result as the PI-Controller. That means the steady state error becomes constant with a 0.02 difference and the control signal shrinks unboundedly as time increases toward infinity.

So, the PD and PID-Controller are not suitable for the system.

9.4.5. PI + I2-Controller

This controller is built based on a pole-placement method. Pole-placement method is to set the desired pole location and to move the pole location of the system to that desired pole location to get the desired system response. In specific, the desired poles will be placed in the left half plane where the system becomes stable.

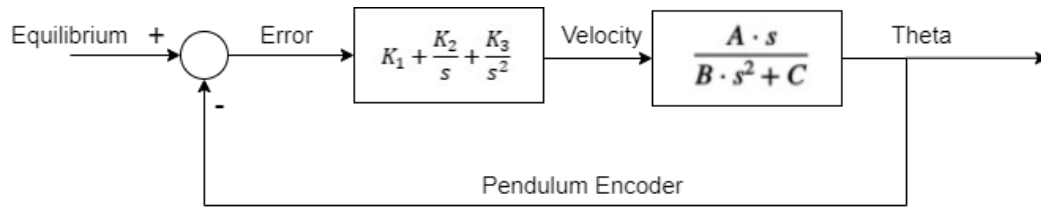


Figure 43: PI-I2 Controller

For the PI+I2 controller, K_1 can be represented as P-Control. $\frac{K_2}{s}$ can be represented as I-Control. $\frac{K_3}{s^2}$ can be represented as double I-Control.

A transfer function of the system will be

$$TF = \frac{\frac{K_1 s^2 + K_2 s + K_3}{s^2} \times \frac{As}{Bs^2 + C}}{1 + \frac{K_1 s^2 + K_2 s + K_3}{s^2} \times \frac{As}{Bs^2 + C}} = \frac{(K_1 s^2 + K_2 s + K_3)A}{s(Bs^2 + C) + (K_1 s^2 + K_2 s + K_3)A}$$

Since we are interested in pole locations, the denominator of the transfer function must be equal to 0.

$$\begin{aligned} s(Bs^2 + C) + (K_1 s^2 + K_2 s + K_3)A &= 0 \\ \rightarrow Bs^3 + K_1 As^2 + (K_2 A + C)s + K_3 A &= 0 \\ \equiv (s - \alpha)(s - \beta)(s - \gamma) &= (s^2 - (\alpha + \beta)s + \alpha\beta)(s - \gamma) \\ &= s^3 - (\alpha + \beta + \gamma)s^2 + (\alpha\beta + \alpha\gamma + \beta\gamma)s - \alpha\beta\gamma \\ \rightarrow \begin{cases} B = 1 \\ K_1 A = -(\alpha + \beta + \gamma) \\ K_2 A + C = \alpha\beta + \alpha\gamma + \beta\gamma \\ K_3 A = -\alpha\beta\gamma \end{cases} &\rightarrow \begin{cases} K_1 = \frac{-(\alpha + \beta + \gamma)}{A} \\ K_2 = \frac{\alpha\beta + \alpha\gamma + \beta\gamma - C}{A} \\ K_3 = \frac{-\alpha\beta\gamma}{A} \end{cases} \end{aligned}$$

Alpha, Beta and Gamma are the desired pole locations. They must be located in the left half plane. Based on these values, the three parameters of the controller K_1 , K_2 and K_3 can be calculated.

```
mp = 0.254 % mass of the pendulum (kg)
l = 0.16 % length to pendulum center of mass (m)
I = 2.167*10^-3 % mass moment of inertia of the pendulum (kg*m^2)
g = 9.81 % gravity (m/s^2)

% desired pole locations, they must be negative
alpha = -1
beta = -2
gamma = -6

A = mp*l
B = (I+mp*l^2)
C = -mp*g*l

G_theta_velocity = tf([A,0],[B,0,C])
```

```
mp = 0.2540
l = 0.1600
I = 0.0022
g = 9.8100
```

```
alpha = -1
beta = -2
gamma = -6
```

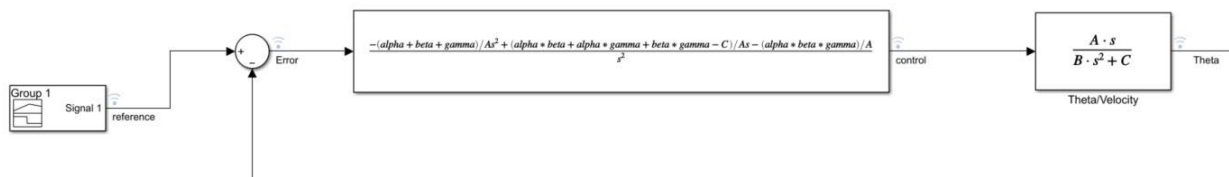
```
A = 0.0406
B = 0.0087
C = -0.3987
```

```
G_theta_velocity =
```

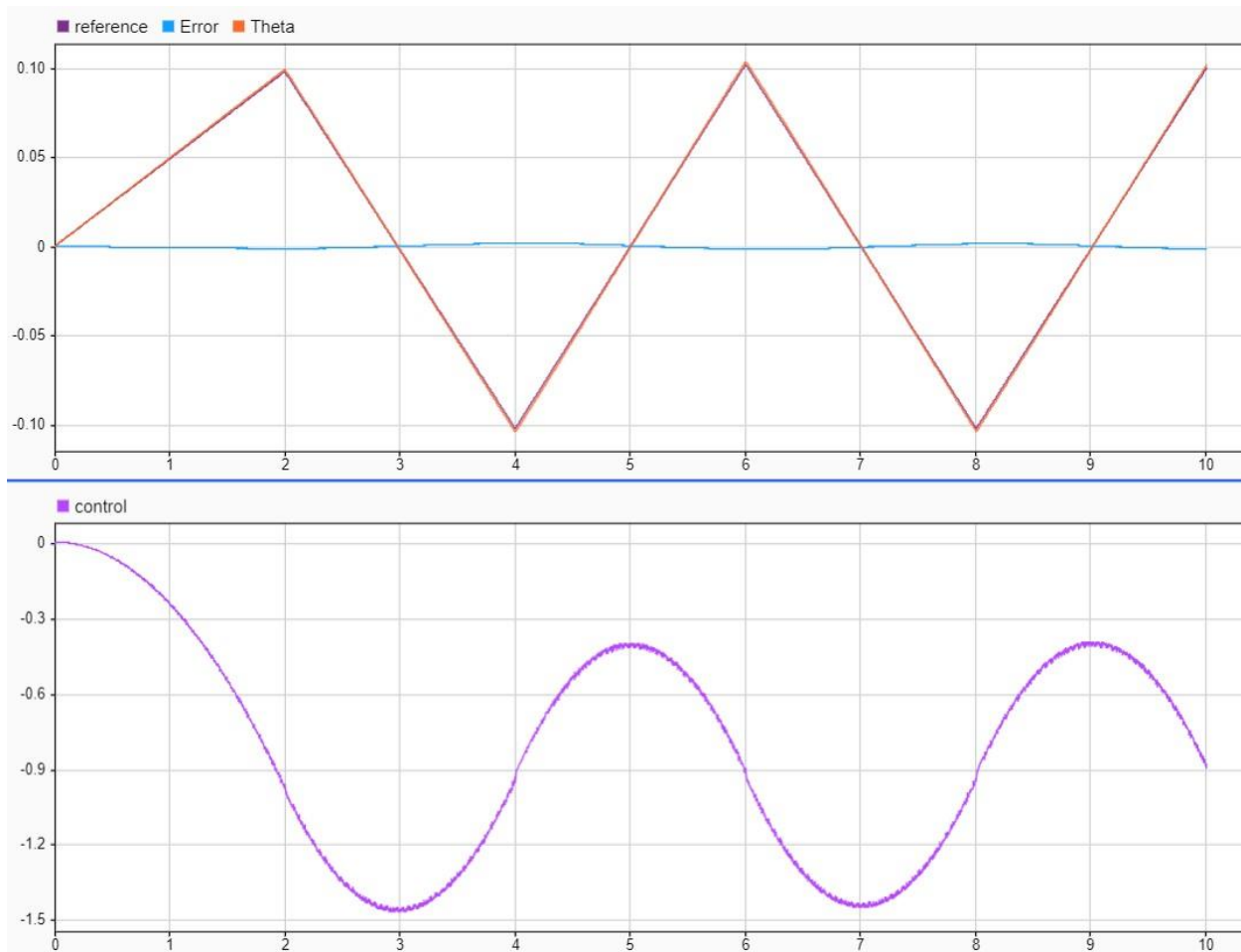
```
0.04064 s
-----
0.008669 s^2 - 0.3987
```

Continuous-time transfer function.

Figure 44: Selecting Alpha, Beta and Gamma



Check the Data Inspector.



On the first graph, we can see that the error value is nearly 0 as time increases, and the output is following the input nicely. On the second graph, the control output remains bounded. Therefore, with this controller, the system will be stable around the equilibrium.

With the three selected pole locations, controller gains can be found easily.

$$\begin{cases} K_1 = \frac{-(\alpha + \beta + \gamma)}{A} \\ K_2 = \frac{\alpha\beta + \alpha\gamma + \beta\gamma - C}{A} \\ K_3 = \frac{-\alpha\beta\gamma}{A} \end{cases} = \begin{cases} K_1 = \frac{-(-1 - 2 - 6)}{0.0406} = 221.67488 \\ K_2 = \frac{2 + 6 + 12 + 0.3987}{0.0406} = 502.431 \\ K_3 = \frac{1 * 2 * 6}{0.0406} = 295.5665 \end{cases}$$

10. PLC program

In this chapter, strategies and state machines will be addressed.

Since the main goal of the project is to make the pendulum stay at its upright position, there are two strategies to implement a PLC program: hold and drop strategy, and full motion strategy.

- Hold and drop strategy: the pendulum must be moved manually to its equilibrium range, Figure 45, and then a PLC program will be executed instantly to balance the pendulum.
- Full motion strategy: a PLC program has to move the Mover in such a way that the pendulum can enter the equilibrium range, and then balance the pendulum.

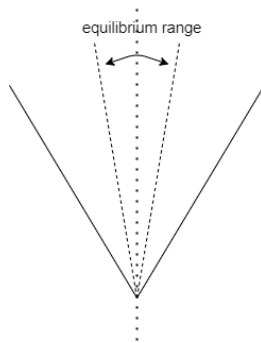


Figure 45: Equilibrium range

10.1. Building a PLC program

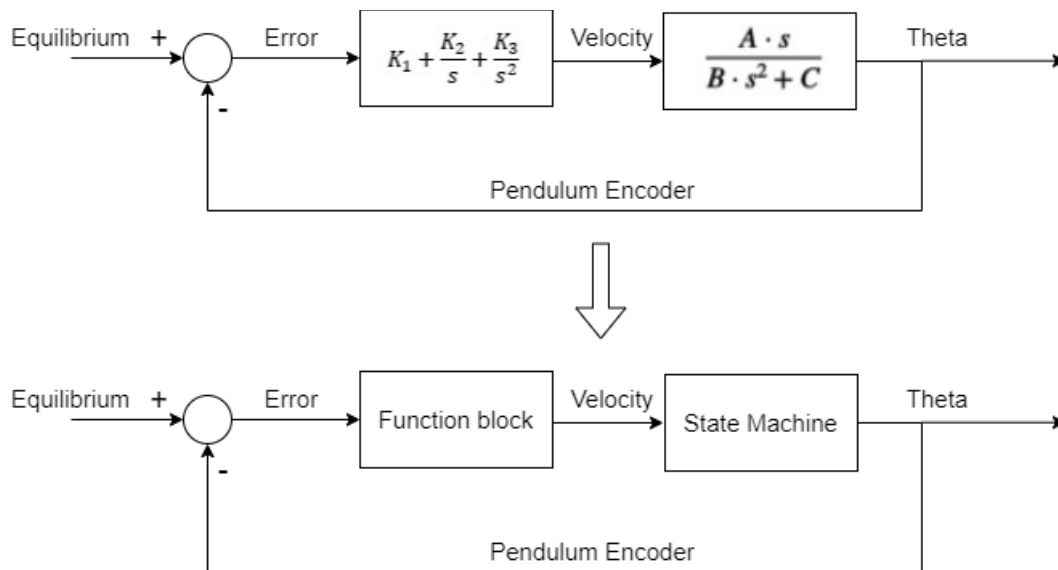


Figure 46: Translating from the Simulink model to the real application

Since the correct model and the controller have been found, we now can design a PLC program based on Figure 46.

A state diagram is a tool used to describe the behavior of systems. It helps showing the flow of a process and needed conditions to switch states. This effort is quickly compensated by faster development of the PLC program. In other words, the total development time is shorter if one expends a little time in planning the program structure. More importantly, the number of bugs in the code that must be detected and corrected will also be smaller.

Unified Modeling Language (UML) is used to make the state diagram.

10.2. Hold and drop strategy

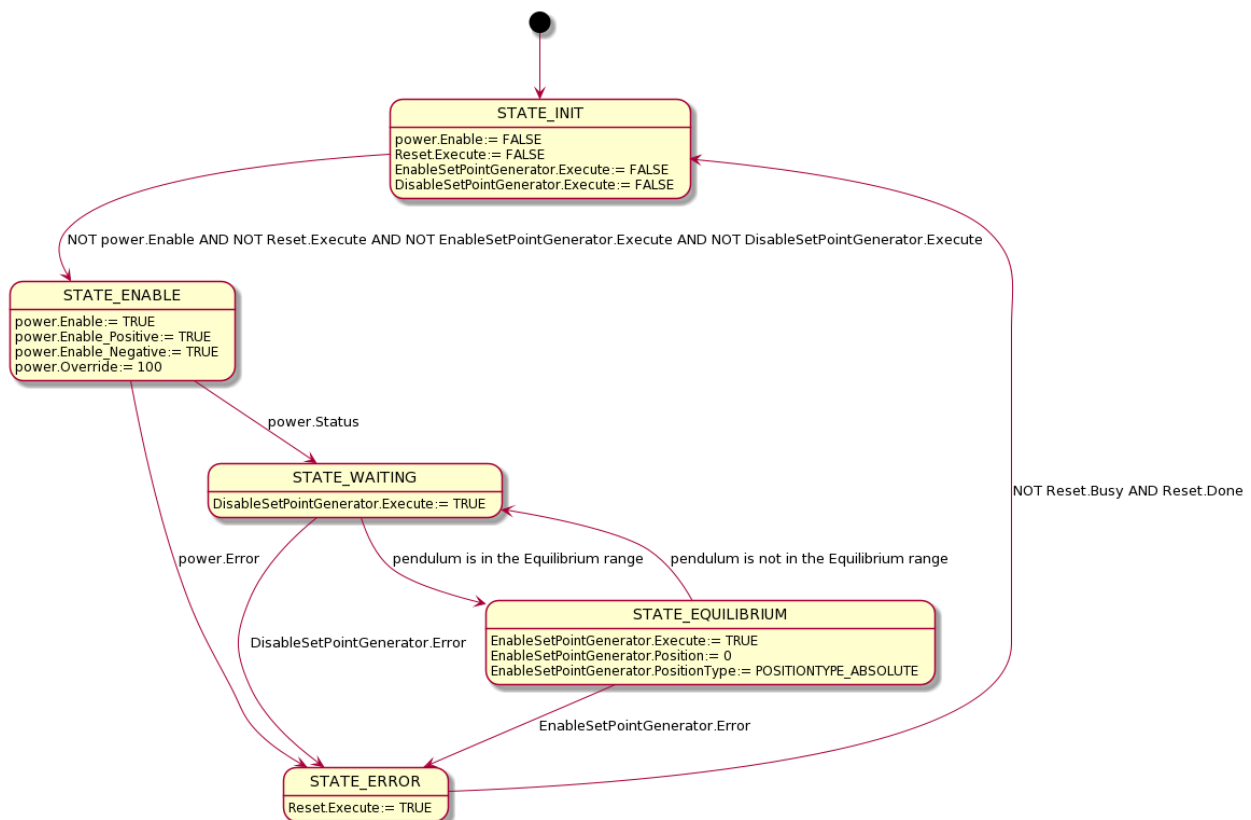


Figure 47: Hold and drop strategy

The full UML code can be found in Appendix H.

10.3. Full motion strategy

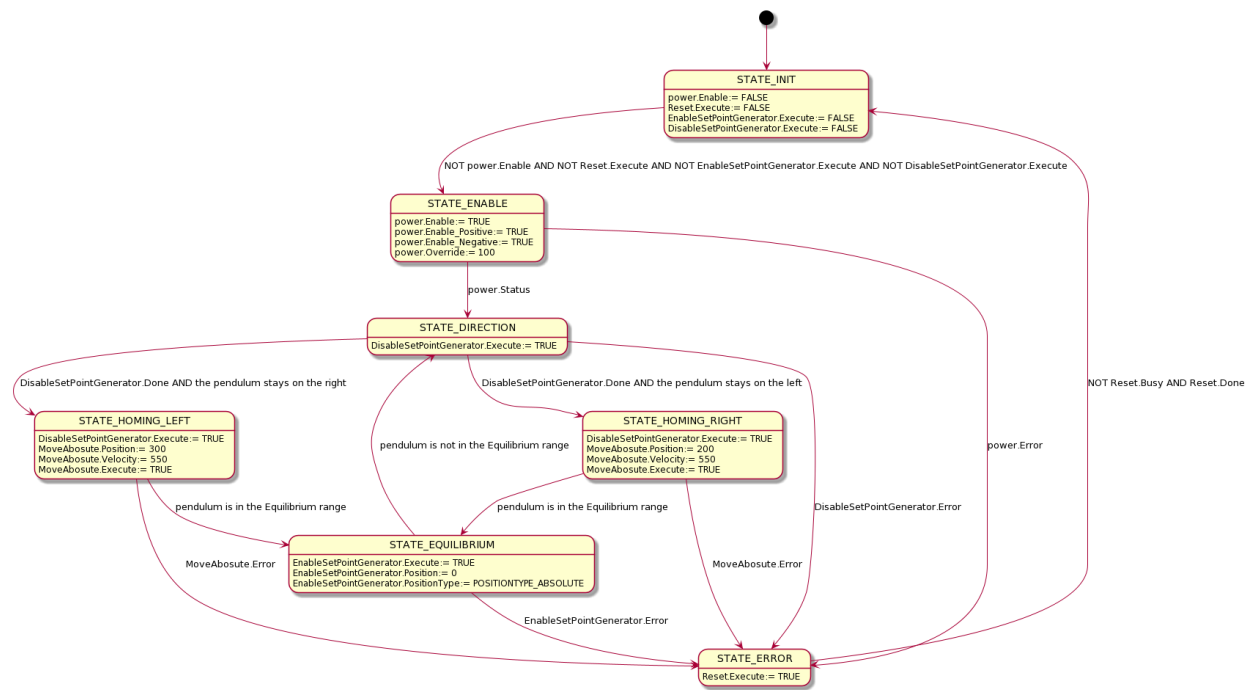


Figure 48: Full motion strategy

The full UML code can be found in Appendix I.

11. Conclusion and Recommendation

In the final chapter of the report, the main conclusion as well as possible recommendation will be discussed. The research questions will be concluded.

11.1. Conclusion

Tag	Research questions	Where has it been answered?
1	How is the XTS machine built? What is the role of each component?	Chapter 8.2
1.1	How to control the mover?	Chapter 8.4
2	What are the equations of motion of the system?	Chapter 6
2.1	What are the equations for the transfer function?	Chapter 9.2
2.2	What are the Simulink models corresponding to the system?	Chapter 9.4
3	What kind of controller can control the pendulum? How can it be designed?	Chapter 9.4.5
4	How to use the code generator to convert Simulink blocks to PLC languages? Is this task feasible? If not, how to write a similar controller in PLC languages?	<p>The Simulink PLC Coder does not generate a PLC code from the Simulink model.</p> <p>The intern has already implemented a similar controller in PLC languages.</p>

11.2. Recommendation

It would be great to have a gyroscope sensor to define the setpoint value for the system. During the testing phase, I found out the mover keeps moving to the right while the pendulum stays at upright position. I re-checked the function block that calculates the setpoint value and Equation 8. Everything seems to be correct. Apparently, the issue does not come from the PLC program nor the equation, it is from the box, holding the XTS machine. The box itself has a small slope.

The temporary solution is modifying manually the setpoint value a bit to the left. In the future, the permanent solution will be setpoint autotuning.

Reflection

When control system lectures were introduced to me in my second year, I found it pretty difficult to understand the topic and its usage in the “real world.” I could not find any practical connection between the theory and the real world. It was way too abstract. After two courses of studying the control system, nearly one year, I did not know how to use control system theories in the real world. I only knew how to build simple blocks in MATLAB/Simulink, but for the real world, I had no idea.

At HAN, every semester, students have to do a project either at a company or at school. Every semester, I went to different companies to look for a project that had control systems involved. Every semester, I got rejected because control system projects were meant for final year students.

When I knew I would be doing a graduation soon, I started looking for a company that had control system projects or PLC related projects. Many companies offered a lot of interesting projects, but the one I wanted was not found yet. One day, I got a response that hooked me immediately. BECKHOFF would like to have an Inverted Pendulum running with a XTS machine.

With the knowledge by that time, I barely knew what control system was. The simplest controller, PID, I did not know how to use it. I am aware that the graduation project is very different compared to other semester’s projects. If I fail the graduation assignment, I have to do another 5-month assignment. Despite the lack of knowledge, I made a decision that I am still really proud of it: Let’s fail successfully.

In September, when I officially worked on the machine, I realized the machine itself was more complex and overwhelming than expected. The final goal was even more overwhelming. I had a ton of questions and I even had doubts about the questions themselves whether they were the right questions. There was a steep learning curve and a large number of subtasks to do.

One of the first problems that I encountered was an unpleasant sound coming from the machine. When the pendulum was mounted firmly on the mover for the first time, the machine got powered and then a loud sustained metal-on-metal noise suddenly happened. I had no choice but turned off the machine immediately. I thought I broke the costly machine already.

Talking to colleagues, I knew that the machine must be tuned in order to find its stabilization. I received a lot of documentations and e-learning videos to do the tuning procedure. After three days of trial and error, I could not reduce the sound. One of the support members checked my

laptop and did the tuning for me. By looking at the way how he solved it, I realized many steps were hidden and were not documented, or it is assumed that users already know about it. XTS machine is an advanced transport system from BECKHOFF. The machine is sold with a training included. Unfortunately, the XTS training takes place physically in Germany. Due to Corona, I could not attend the training.

Remembering how that colleague did the tuning, and doing it again, I found out a connection between the control system theory and the machine. Basically, the machine has an internal closed-loop system to control itself, and that closed-loop system has been tuned already. Any disturbance applied to the machine will change the behavior of the system. In this case, the pendulum, the disturbance, messed up the closed-loop system. At school, I often saw Figure 49, but I was not sure how disturbance looked like in the real world. Now I can easily see it.

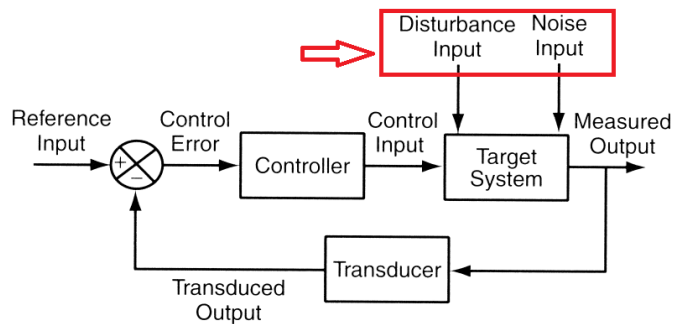


Figure 49: Closed-loop control system [19]

Furthermore, repeating the tuning procedure, I saw myself following a sequence that I could not interpret clearly. One month later, in October, Marnick encouraged me to build a controller for the pendulum, I recognized that the XTS internal controller was built based on Ziegler-Nichols method. Turn off all controller gain parameters except the P-parameter. Tune the Kp value until the ultimate gain is found. Adjust the Kp value and turn on the I-parameter. Keep tuning until the oscillation period is found.

That date, I understood how to use the simplest PID controller in practice. That date, I developed a new skill. Also, that date, I recognized distinctly that I had to deal with two different closed-loop systems: one was for the XTS machine itself, and another one was for the pendulum which was the one I had to build.

Third week of October, I built a PLC program based on the Ziegler-Nichols method, and with some help from Marnick, the pendulum was able to balance on its own for a short moment. It was one of the remarkable moments for both of us. I thought the Ziegler-Nichols method was the right way

to find a controller. I just needed to improve my PLC program a bit, and then I could see the final goal. I was so excited.

However, I kept trying and trying, different state machines were designed, I still could not make the pendulum balance for a long period. As soon as the pendulum entered the equilibrium range, aggressive movements happened, messing up the entire balancing state. I got stuck.

In November, I confronted my difficulties with Amin. He understood the struggles and explained control system in the way that I could understand it. However, the special “lectures” were different compared to the time I studied at school. It was way too easy to visualize in my head. I could see a connection between Simulink models and my PLC program. I knew exactly where I could implement the Simulink parameters in my PLC program. It was just a strange feeling.

The reason why I got the strange feeling was from the experience that I worked on the physical machine. By trying to build different PLC programs to make the pendulum move and seeing its physical movement, I had also built up my intuitive understanding of the system. But it was not organized in a neat way. Listening to Amin’s instruction, I could find a way to link them together.

By that month, I had a chance to talk to an XTS expert. He understood the struggle I had been through, and what I was after. The main issue of my program is that the pre-defined move function is not suitable for my application. He wanted me to build my own move function using “External SetPoint Generator” functions.

That task of developing-my-own-move-function was not easy because there was no documentation, no example code. I returned to the trial-and-error method of iterative programming. Put that function in TwinCAT, change some parameters, run it, and see its result. After nearly four weeks of trial and error, I had my own move function running. By that time, I realized the underlying concept of using the External SetPoint Generator functions. It is building a tracking trajectory which is used to effect desired trajectories of a target device. Another achievement was developing.

The second week of December, I found out a critical error that brought me down deeply, the communication between the encoder and the BECKHOFF system was not set correctly. Instead of getting a constant signal, I received a fluctuating signal. After hours of diagnostics, the problem solved itself by changing just one parameter. But I did not feel delighted at all, I felt hopeless. The deadline of submitting the final report was nearby, but my report was not halfway done yet. Month

after month trying and trying, I could not make the pendulum stay upright. Another error just solved. What errors were still hidden? I was getting lost in anxiety.

The next day, I got an unexpected call from Amin. That talk was longer than usual. I will not forget that talk.

On Wednesday morning, 15-12-2021, around 9 o'clock, I triggered one of the parameters in my PLC program for testing, and then I noticed that the pendulum was able to balance on its own for a few seconds. The duration was short, but I saw it. The balancing of the pendulum was smooth. That brief moment linked many uncertain things together inside my head. Some states and parameters were implemented incorrectly, and then.

It finally worked.

Stabilizing an Inverted Pendulum within TwinCAT is one of the most difficult challenges that I have been through. My journey had a lot of ups and downs. Sometimes, it went deeply down. Without encouraging and supporting from other people, I might not have been able to make it work.

It was a fantastic journey.

Appendix A. Approved LTC statement

L T C – Statement
ENGINEERING - BACHELOR
Graduation Internship

HAN_UNIVERSITY
OF APPLIED SCIENCES

Student Name: Suat Nguyen Student Nr.: 593604

Course: Industrial Power Systems Variant: BA Type: Full Part Time

Study Contract Status:

- Propeduse / Both Basic Modules is/are completed: Yes / No
- Minor Semester / Module is completed: Yes / No
- All other Major Semesters / Main Modules are completed: Yes / No
- not completed Major Semesters / Main Modules can be completed during the graduation internship term (give all details below in Remarks): NotAp / Yes / No

Preparatory activities (for all BA variants):

- Graduation information session was attended: Yes / No
- Orientation on preferred professional task / project: Yes / No

Preparatory activities (only for the BA Full-time variants of ME and IDE):

- Profile portfolio updated for graduation purposes: Yes / No

The student has met all the conditions to start the part of the curriculum indicated below, to be scheduled for the indicated duration, expressed in number of consecutive Periods:

Curriculum part: Graduation Internship Semester / Module

Start Date / Period: 30-08-2021 Duration of term (Periods): 1 2 / 3 / 4 / 5

Remarks:

.....

S6 needs to be completed before graduation.

.....

.....

.....

.....


Learning Team Coach (LTC): Name (LTC): Francesco Ursino

Date: 27-04-2021 Signature (LTC): *Francesco Ursino*

Appendix B. Approved graduation assignment application form

Ruud Elsinghorst

Digitally signed by Ruud Elsinghorst
DN: OU=Academie Engineering en Automotive, O=Hogeschool van Arnhem en Nijmegen, CN=Ruud Elsinghorst, E=ruud.elsinghorst@han.nl
Reason: Akkoord met resultaat
Location: Arnhem
Date: 2021-05-17 15:35:19
Font Reader Version: 10.0.0

Graduation Internship Assignment Application Form					
AEA - ENGINEERING - BACHELOR					
Student name: (including initials)	Suat Nguyen		Stud.no.:	593604	
e-mail address:	ss.nguyen@student.han.nl		Phone:		
Address (street, number, post code, place, country):	Heggerankstraat 10, 6833 DE, Arnhem, Netherlands				
Permission from LTC?	<input checked="" type="radio"/> Yes <input type="radio"/> No	Name LTC:	Francesco Ursino		
You have all 120 ECTS for Major/Main Modules? (i.e. excluding Propedeuse/Basic Modules and Minor)	Yes <input checked="" type="radio"/> No <input type="radio"/>	if Not, nmbr. of ECTS is:	90		
Degree course: (incl. variant)	Industrial Power Systems		Full Time <input checked="" type="radio"/> / Part Time <input type="radio"/>		
If in Part Time: are 3 or more days/week available?	Yes / No / not appl.		if Not, no. of days/wk is:	1 / 2	
Internship term starts in: (academic year)	Start Date: (yyyy/mm/dd)	Start with Period:	Planned number of Periods:		
2021 - 2022	30-08-2020	<input checked="" type="radio"/> P1 / P2 / P3 / P4	1 <input checked="" type="radio"/> 2 / 3 / 4		

1. Company / organization details	
Company / organization name:	BECKHOFF
Company / organization website:	https://www.beckhoff.com/
Department (incl. group):	
Mailing address:	Dillenburgstraat 5, 5652 AM Eindhoven
if different: visiting address:	
Company Supervisor name (incl. initials):	Nikolas Eimer (N.Eimer)
Function / job title (incl. degree):	Location manager (MBA&Eng.)
e-mail address:	N.Eimer@beckhoff.com
Phone number:	+31 (0)40 303 2650

Note: expand available space for the text below in Word as needed, but keep it short and to the point.

2. About the company (or organization)
2.1. What kind of company is it and in which field and part of the field of interest relevant to the assignment is it mainly active? What is the size of the company (in terms of people, locations, capital, impact on national or global market)?
<p>Beckhoff, also known as Beckhoff Automation, is a manufacturer of automation technology. Their products are divided into 4 different areas: IPC – industrial PC, also known as PLC; I/O – Input/Output card; motion – stepper motor, servo motor, linear motor; and Automation – an IDE to implement a PLC program, integrated into Microsoft Visual Studio environment.</p> <p>The assignment is about developing a PLC program to control an inverted pendulum attached to a servo motor. The assignment will cover 2 areas: motion – control the servo motor, and Automation – implement a PLC program.</p>

<p>Headquarter: Verl, Gütersloh district, Germany. In the Netherlands, there are 3 branches located in Haarlem, Eindhoven and Enschede. Me, as an intern, will be working in Eindhoven. The company employs around 4300 people worldwide, including 1200 engineers.</p> <p>Impact on national and global market: EtherCAT, one of the industrial protocol communications, standardized in IEC 61158, was originally developed by Beckhoff. This protocol is mainly used for automation application which requires short cycle times (less than 100 µs per cycle) in order to have a real-time machine. Also, an existing network such as CAN bus, Profibus can be integrated into the EtherCAT environment, reducing further investment costs.</p>
<p>2.2. Who in the company will guide and supervise you with this assignment? What is the expertise / field and the level of education of this business coach / company supervisor?</p>
<p>1. Nikolas (Location manager/PS sales for XTS). He is an expert in XTS machine. He will guide me on how to use the machine in the first place, for example: how to connect components together, how to turn ON the machine... General information can be answered by him.</p> <p>2. Marnick (Technical). He will support me in PLC programming and control system.</p>
<p>2.3. What expertise is present in the company or with those involved in the project, with which you can work together to successfully complete the project?</p>
<p>The XTS machine itself was internally developed by Beckhoff, so all documents related to the machine are available at the company.</p>

<p>3. About the assignment</p>
<p>3.1. What is the problem (or issue) that needs to be solved?</p>
<p>At Beckhoff, there is one machine called XTS - eXtended Transport System. The machine consists of 2 major parts: a track and (a) moving servo motor(s). The servo motor is called a mover.</p> <p>A pendulum is attached to a moving servo motor, moving on a track. When the motor, controlled by PLC, is moving, the pendulum will be swinging. At a moment in time, when the pendulum gets enough acceleration, it will be swinging up. There will be one moment that the pendulum will stay upright.</p> <p>Problem that needs to be solved: implement a control system algorithm, written in TwinCAT3, to control the inverted pendulum. The control loop should also react accordingly to external influences like "shoving/touching" the pendulum.</p> <p>TwinCAT3, developed by Beckhoff, is an IDE to implement a PLC program. It supports PLC languages and motion control.</p>
<p>3.2. Why is there a problem and why is a solution urgently needed? Also briefly outline the context of the problem.</p>

<p>Firstly, MATLAB/Simulink is a well-known software for creating a control system. However, MATLAB/Simulink is a third-party software which doesn't have any package to support other software. In specific, there is no option to export a Simulink model to TwinCAT3.</p> <p>Secondly, they would like to have a showcase from their linear track system that shows the responsiveness and fast control loops. An inverted pendulum is a perfect example for this purpose. Also, solution made by the intern will be used for training purposes.</p>
<p>3.3. Who are the stakeholders? How are they involved in the project and what is their interest in it? TIP: don't forget to think about yourself and the school!</p>
<p>There are 3 stakeholders: company, school and me.</p> <p>For the company, they would like to see how I manage to get the inverted pendulum working in TwinCAT3 environment. Working in Simulink is also great, but the final goal is to have a model working in TwinCAT3. Also educate young engineers, like me, and excite them about automation technology. Probably also find a new employee if the project goes well.</p> <p>For school, they would like to see how I apply learned knowledge from the previous years to this project and how I deal with difficulties when something doesn't go as expected.</p> <p>For me, as an intern, this is an opportunity for me to learn and work in the real world and apply what I have learned so far. I also will be responsible to keep track of everything, and inform other people at the right moment.</p>
<p>3.4. What are the main requirements for the solution? In other words: what result does the internship client (and / or the customer) expect? Formulate the process related and substantive objectives for the project.</p>
<p>There are several requirements that need to be done.</p> <p>Firstly, get familiar with the machine and control the servo motor without the inverted pendulum attached to the machine.</p> <p>Secondly, attach the pendulum, and try to implement a control system in MATLAB/Simulink.</p> <p>Next, which is the main goal, implement the control system in TwinCAT 3, or in another word, write a control system algorithm in PLC languages. Structured text, ladder diagram or function block diagram can be used.</p> <p>When the main goal has been reached, there are some minor tasks that can be made:</p> <ul style="list-style-type: none"> • Develop an HMI program for the machine. • Find the minimal power consumption (or minimal timing) from the moment when the pendulum is free to the inverted moment.
<p>3.5. What is a catchy title, possibly including a more explanatory subtitle, for your graduation work, that can be used on the cover page of your graduation report, suitable for publication and announcements?</p>
<p>Title: Generating a control-loop for an inverted pendulum within TwinCAT 3</p>

4. About the orientation on the approach to the problem (or issue):
As a student you must be able to solve a problem in a structured way, which means that the substantiation of your approach and the explanation of your choices are important. During the course and possibly your minor you have already acquired a lot of knowledge and skills: how will a successful completion of this assignment demonstrate this?
4.1. What is the appropriate formulation of the problem definition for this graduation assignment?
Designing a control system algorithm, written in PLC languages, to control an inverted pendulum.
4.2. How do you think you can successfully solve the problem of the client and which learned knowledge and skills can you use for this (and with which accompanying research proposal)?
<p>At this moment when I am writing this report, it is hard to say the final goal would be achievable or not. If my planning schedule is correct, I have around 2-3 months to work physically on the machine. September, hand in a Plan of Approach. October, hand in half of the Research proposal report. December, 2 weeks holidays. January 3rd, hand in the final report.</p> <p>Despite having a limited time, there are some tasks that I know will be achieved, such as: write a PLC program to control the servo motor with and without the pendulum. As soon as I have the control system working in Simulink, it is just a matter of time to translate the Simulink model to TwinCAT3.</p> <p>For the Research proposal report, I would like to research about some following topics:</p> <ul style="list-style-type: none"> • How is the XTS machine built? • How to control the servo motor? • How to make the pendulum inverted? <p>The Research report will be handed in 2 times. For the first time, a working control system might not be there yet. For the second time, it will be handed in with the final report.</p>
5. About yourself
5.1. Why did you choose this assignment?
<p>In semester 5, I did my internship at an industrial company where a Beckhoff PLC was used. I got familiar with the Beckhoff system and its environment, so I am totally confident in making the first few steps of this project.</p> <p>Secondly, the inverted pendulum took my attention. In semester 3 and 6, I learned fundamentals of control system and built a control system in Simulink. I would say I have knowledge that I need for the assignment. Creating a control system for the machine might not be that easy, but I will try my best.</p>
5.2. How does this assignment and this company suit you as a starting professional?

Industrial Power Systems is one of the EEE courses that I am following. After nearly 4 years studying at the HAN and after doing my first internship, I have found my ambitions in industrial companies where automation application is used. I love physical activity and have a strong belief in entering a field that I am not familiar with.

This assignment at Beckhoff, I find it the best possible fit for my career goals. No doubt I can learn a lot in such an environment, improve on my skills, and step by step progress in my career.

Submit the completed form by e-mail to your graduation internship coordinator.

Warning: incomplete forms will not be processed!

Note 1: indicate number of attachments: ____ and total number of pages: ____ pages added, excluding the LTC statement form, see next point.

Note 2: Attach the approved and signed LTC statement as a separate PDF file.

Note 3: Replace "Student Name" through "YYMMDD" and the like in the file name to display the current and individual version of the completed form you are submitting. Increase the version number (starting from "01") for each renewed version you submit. ("EN" in the file name indicates the choice in Alluris, that you choose English as the course language. Leave it as it is if it is true).

Appendix C. Final version of Project Plan

Problem statement

The Inverted Pendulum is a classical control problem of an unstable dynamic system. The system consists of an inverted pendulum mounted on top of the mover, which is moving on the guide rail. The pendulum will simply fall over if the mover is not moved to balance it.

The system is used to demonstrate the target hardware/software platform developed by BECKHOFF. With this platform, it is possible to run models and controllers in real-time while connecting these models with input/output signals from the real world. The demonstrator will be used at BECKHOFF for promotional and educational purposes.

The current Inverted Pendulum demonstrator is not yet working at BECKHOFF. The system consists of six different parts, which are not yet fully assembled as shown in Figure 2: an IPC (1), an encoder for the pendulum (2), a mover (3), a guide rail/motor module with a built-in encoder (4), a pendulum (5), and a blocking angle unit (6).

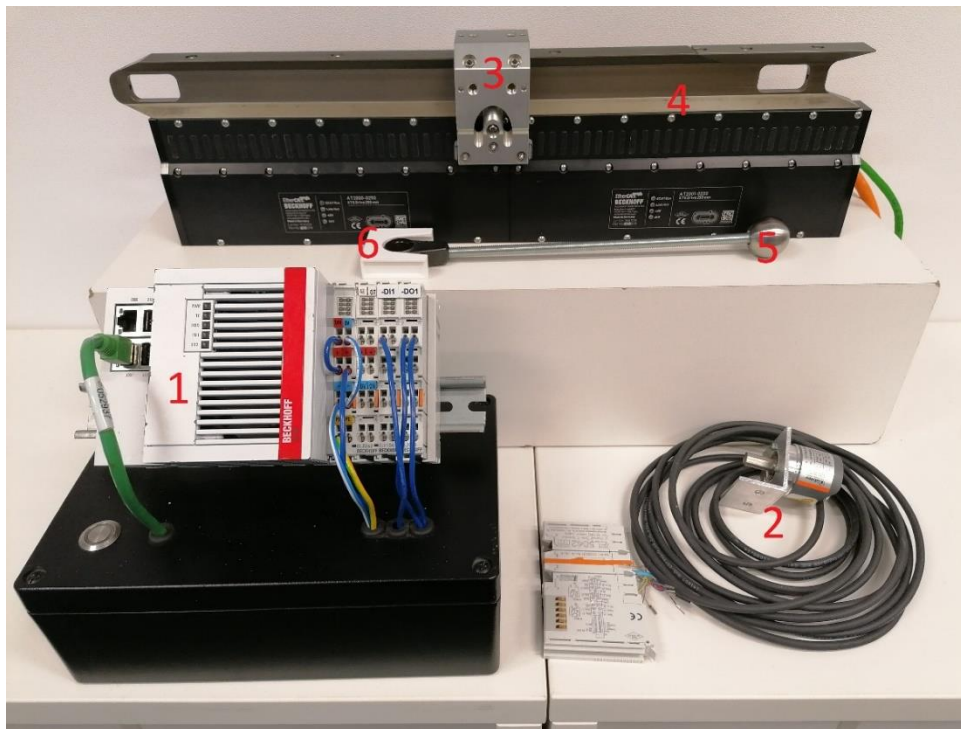


Figure 50: Inverted Pendulum set-up

Figure 51 shows how to assemble the pendulum with the mover. The intern needs to find out what screws are suitable to mount the pendulum to the mover.

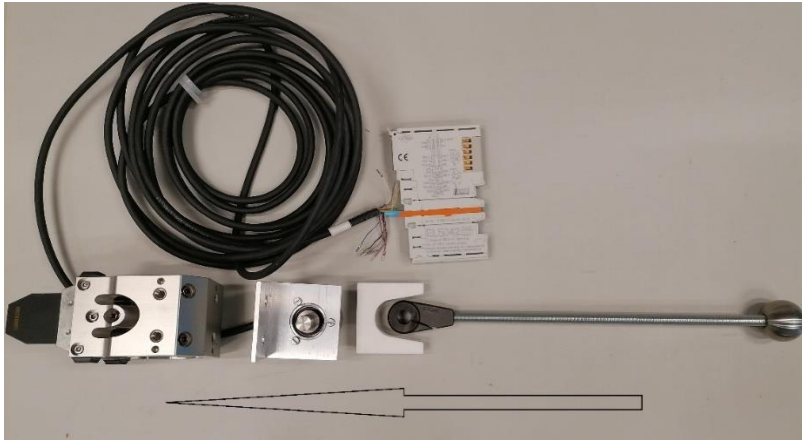


Figure 51: Correct way to assemble the pendulum to the mover

A basic schematic of the system is shown in Figure 52. A controller has to be designed to keep the pendulum upwards in its vertical position. Therefore, the deviation of the pendulum from its upright position and the position of the mover are significant measurements. The encoder is used to measure the position (or angle) of the pendulum. The built-in encoder in the guide rail itself, mentioned above, is used to measure the position of the mover.

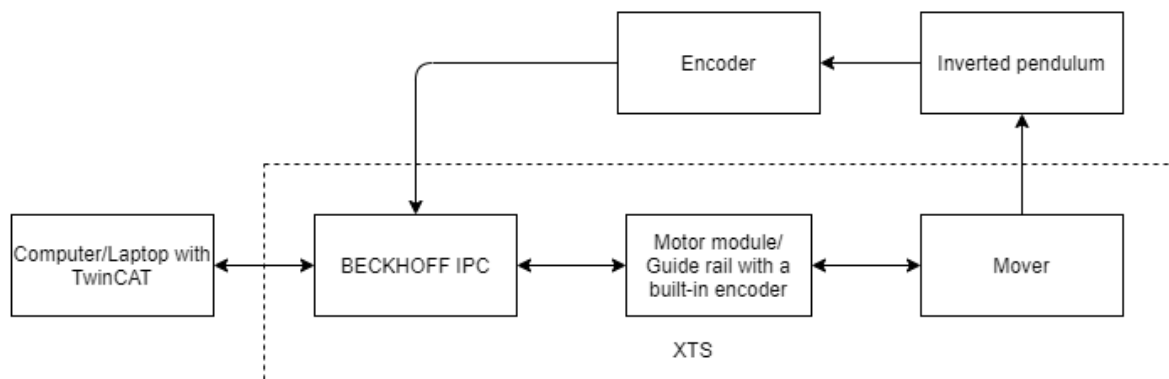


Figure 52: sketch of the system

The scope of the project

The scope of the project is restricted to the Inverted Pendulum set-up, the hardware/software platform and two different feedback systems. What is more, the swing-down process of the pendulum is ignored due the limitation of the hardware. That means the pendulum will start in a vertical upward position.

What must be done?

The ultimate goal of the project is to have the controller for the Inverted Pendulum running in TwinCAT3. That means a PLC program, written in Structured Text, must be implemented. The intern is allowed to use MATLAB/Simulink to build and verify models. After that, a code generator will be used to convert the Simulink blocks to the PLC language. If the converting process is not feasible, the intern needs to build a separate control system in TwinCAT3.

To make the inverted pendulum run in TwinCAT3, the following table shows research questions and sub-questions.

Tag	Question
1	How is the XTS machine built? What is the role of each component?
1.1	How to control the mover?
2	What are the equations of motion of the system?
2.1	What are the equations for the transfer function?
2.2	What are the Simulink models corresponding to the system?
3	What kind of controller can control the pendulum? How can it be designed?
4	How to use the code generator to convert Simulink blocks to PLC languages? Is this task feasible? If not, a similar controller must be separately built in PLC languages.

What could be done?

Human Machine Interface (HMI) is a dashboard that enables users to engage and interact with machines. For this project, it would be great if the intern can use TwinCAT HMI to implement the HMI program.

Planning and schedule

A global planning will be shown in Table 2: Global planning. For the time being, the intern will join several training courses (and/or customer visits, customer meetings) provided by BECKHOFF. These unscheduled events might change the global planning. The intern undertakes to inform all people involved in the project about this in good time.

Every Monday, at 11.00, the intern will attend a group meeting, via Microsoft Teams, with BECKHOFF support team to present what he did in the previous week, what he will do next, and ask for support if needed.

Month	Week	Activities	Deliverables	Notes
August + September	35	Get to know colleagues, TwinCAT, laptop, offices, etc. Collect all information related to the XTS assignment. Start writing the Project Plan report.		
September	36	TwinCAT training (3 days) provided by BECKHOFF.	Finish the Project Plan report. (Deadline 10-09-2021) Hand in the first biweekly report.	
	37	Motion control theory provided by BECKHOFF. Start writing a Research report.		
	38	PLC motion in TwinCAT.	Hand in the biweekly report.	The first meeting between school, company, and intern, via Microsoft Teams. Date and time will be informed in good time.
September + October	39	XTS – first step Design a model.		
October	40	Design a model.	Hand in the biweekly report.	
	41	Design a model.		
	42	Design a model.	Hand in the biweekly report.	
	43		Submit (partial) Research report v1 to school. (Deadline 29-10-2021)	

November	44	Combining everything. Start writing the final report.	Hand in the biweekly report.	
	45	Combining everything. Start writing the final report.		
	46	Combining everything. Start writing the final report. TwinCAT training in Dutch as language practice.	Hand in the biweekly report.	
	47			
November + December	48		Submit the draft final document to school, and the Research report v2 if needed. (Deadline 26-11-2021)	
December	49	Continue the final report. Write a manual guide for BECKHOFF.		
	50	Continue the final report. Write a manual guide for BECKHOFF.	Hand in the last biweekly report.	
	51	Christmas and new year holidays		
	52			
January 2022	01		Submit the final document to school. (Deadline 07-01-2022)	
	02	Team presentation with BECKHOFF via Microsoft Teams		Date and time will be informed in good time.

	03	Final presentation with school via Microsoft Teams		The presentation will be held via Microsoft Teams.
	04			

Table 2: Global planning

Explanation:

Biweekly reports – The deadline is every 2 weeks.

The supervising lecturer must be capable of providing adequate supervision. For this reason, it is important that the intern keeps him up to date during the graduation project. The intern does so by writing a report every 2 weeks. These reports should provide clarity on:

- How things are going in general (work relationships, contact with the company supervisor, etc.).
- The progress of the graduation assignment.

Biweekly reports give the supervising lecturer a better idea of the work level and allow him to step in if the intern is veering off course.

Research report – The deadline of version 1 will be on **29-10-2021**, version 2 will be on **(26-11-2021)** (if needed).

Final report – The deadline of the draft report will be on **26-11-2021**, final report will be on **07-01-2022**.

The final report consists of a main document with appendices. The research report will be one of the parts of the final report.

The goal of the draft report is to give both the lecturer and the intern a good idea of the progress.

Project organization

During the internship, the intern will spend five days or 40 hours per week on the assignment. Communication between the company supervisor and the intern is important, so in this case, it will be mainly oral. Due to Covid restriction, some colleagues will be working from home, an online meeting via Microsoft Teams will take place.

Mr. Amin Mannani, the school supervisor, will help the intern as well. To keep the school supervisor up-to-date with the internship, the intern needs to send biweekly reports.

Working hours: from 8:15 – 8:30 to 16:30. Rest hours: 12.30 – 13.00

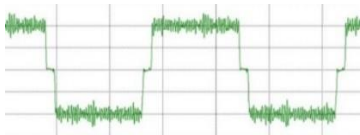
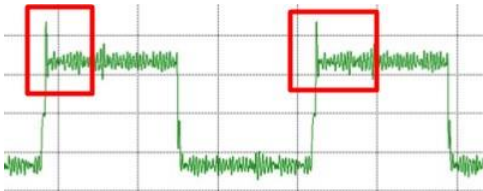
The entire team involved in the project are listed in the table below.

	Name	Role	Email
Company	Nikolas Eimer	Location manager	N.Eimer@beckhoff.com
	Marnick Sluismans	Support Engineer	msluismans@beckhoff.nl
School	Amin Mannani	School supervisor	amin.mannani@han.nl
Intern	Suat Nguyen	Electrical student	snguyen@beckhoff.nl
			ss.nguyen@student.han.nl

Table 3: group's information

Appendix D. Move functions

One of the essential things that I have noticed during the time working on the machine is the Move function. BECKHOFF library has a lot of Move functions which can be used for different purposes, but which Move function is suitable for this task? With the help of many BECKHOFF colleagues, I am able to distinguish between BECKHOFF standard Move functions and BECKHOFF custom Move function, in this case the External Setpoint Generator function.

Feature	MoveVelocity function	External Setpoint Generator function
Motion profile and complexity	<p>A pre-made trajectory is already integrated in the Move function itself.</p> <p>User-friendly and very easy to use.</p>	<p>Users have to build a partial (or full) trajectory from scratch.</p> <p>Need to understand what is happening inside a black box, and how a signal goes from A to B.</p>
Overshoot	<p>Velocity does not make a big jump at the beginning.</p> 	<p>Velocity is pretty high at the beginning because a full motion profile has not been built completely. The mover has to travel a short distance but starting from 0 velocity. That means its acceleration and its derivative, jerk, go to infinite.</p> 
Fast response	<p>There are some delays between a commanded velocity and an actual velocity because the PLC itself needs to have some time to calculate the entire motion profile before sending to the mover.</p>	<p>Instant update.</p> <p>The mover will move based on the motion profile pre-defined by users.</p>

Is the pendulum able to swing?	Yes, the pendulum is able to swing, but aggressive movement will happen around the equilibrium point.	The current function itself cannot make the pendulum swing because it has a high velocity at the beginning, messing up the entire path. The function itself should not start from 0 velocity or a full motion profile has to be designed.
--------------------------------	---	---

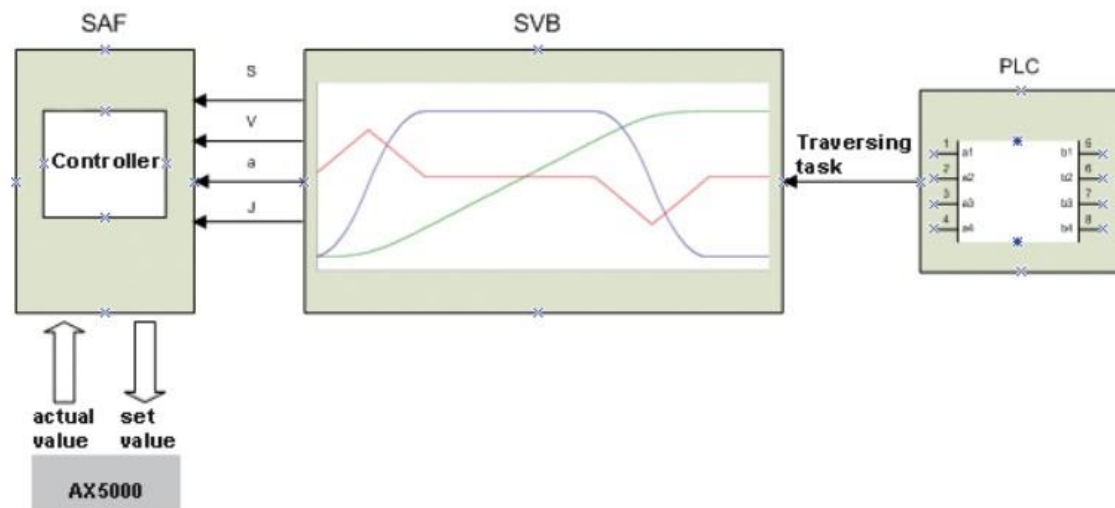


Figure 53: Signals travelling from the PLC to the actual drive [20]

In Figure 53, there are four different blocks corresponding to different tasks. AX5000 is taken as an example.

- PLC block: this is a place where users type a PLC program.
- SVB block: this is an internal system where a trajectory, max. velocity, acceleration, deceleration, and jerk are designed automatically. That means the system itself will try to calculate what value should be suitable for the next cycle.
 - o When the External SetPoint Generator function is used; the SVB block is disabled, allowing users to design their own desired trajectory.
 - o However, when the SVB block is disabled, there is no safety function enabled. Besides designing the Move function, users have to design their own safety functions, otherwise, damage will be unpredictable.
- SAF block: this block will directly send all calculated parameters to a drive.
- AX5000: A drive receives the calculated parameters and sends them to a physical Axis.

Appendix E. Connecting the encoder with the BECKHOFF system

Encoder			BECKHOFF terminals		
Abbreviation	Color	Function	EL5042	EL2004	EL2262
WH	White	Power supply ground			Pin 7
BN	Brown	Power supply +V DC			Pin 6
GN	Green	C+, clock signal	Pin 2		
YE	Yellow	C-, clock signal	Pin 10		
GY	Gray	D+, data signal	Pin 1		
PK	Pink	D-, data signal	Pin 9		
BU	Blue	Set input		Pin 1	
RD	Red	Direction input			
BK	Black	Incremental output channel A (cosine+)			
VT	Violet	Incremental output channel A (cosine-)			
GY-PK	Gray-Pink	Incremental output channel B (sine+)			
RD-BU	Red-Blue	Incremental output channel B (sine-)			

The SET input pin means the encoder can be set to zero at any position by means of a HIGH signal.

After wiring, some parameters need to be adjusted in the **CoE - Online** tab in TwinCAT, as shown below.

The screenshot shows the TwinCAT interface with the **CoE - Online** tab selected. The left sidebar shows the project structure, with **Term 5 (EL5042)** selected under the **I/O** section. The main area displays a list of parameters for the selected device. Several parameters are highlighted with red boxes:

- 8008:0 FB BiSS Settings Ch.1** (Flags: RW, Value: > 17 <)
- 8008:0 FB BiSS Settings Ch.1** (Flags: RW, Value: > 24 <)
- 8008:0 Invert Feedback Direction** (Flags: RW, Value: FALSE)
- 8008:0 Disable Status Bits** (Flags: RW, Value: FALSE)
- 8008:0 CRC Invert** (Flags: RW, Value: TRUE)
- 8008:0 CRC Polynomial** (Flags: RW, Value: 0x00000000 (0))
- 8008:0 Supply Voltage** (Flags: RW, Value: 5.0 V (50), Unit: 0.1 V)
- 8008:0 Clock Frequency** (Flags: RW, Value: 5 MHz (1))
- 8008:0 Coding** (Flags: RW, Value: Dual Code (0))
- 8008:0 Multiturn [Bit]** (Flags: RW, Value: 0x00 (0))
- 8008:0 Singleturn [Bit]** (Flags: RW, Value: 0x0E (14))
- 8008:0 Offset LSB Bits [Bit]** (Flags: RW, Value: 0x00 (0))
- 8008:0 Mode** (Flags: RW, Value: BiSS-C (0))

Check communication in the **Diag History** tab.

The screenshot shows the **Diag History** tab in the TwinCAT interface. The **Update History** button is highlighted with a red box. The table below shows the communication logs:

Type	Fla...	Timestamp	Message
Info	N	10/27/2021 16:53:3...	(0x1304) Encoder initialization successfully, channel: 0001
Info	N	10/27/2021 15:02:4...	(0x1303) Encoder Supply ok
Info	N	10/27/2021 15:02:4...	(0x0002) Communication established

There is no red error message which means the terminal is ready to deploy.

Appendix F. Servo Control Technology

Since the XTS is a closed-loop servo system, the motion control will command and change both the present position and present velocity to the updated target position and target velocity, and target acceleration profile.

The XTS system as delivered is configured for an empty mover, which means there is no payload on top of the mover. Any payload added to the mover will affect how the mover behaves. The mass, the stiffness, and the location of the center of gravity of the load have a drastic influence on the behavior of the mover.

As soon as a load is added to the mover, and the mover is enabled, the mover will begin to oscillate and make a very loud unpleasant noise. If the mover keeps oscillating, there is no further steps that can be achieved, for example: the mover cannot move left or right with a simple command. Therefore, oscillation must be eliminated, and updating the motion profile must be done.

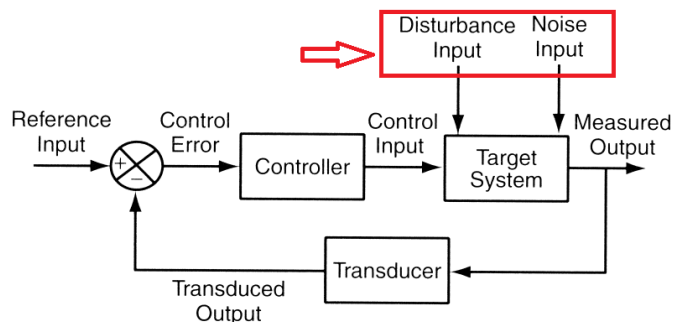


Figure 54: Closed-loop control system [16]

By using the Tuning Assist, implemented in TwinCAT, the mass, the stiffness, the location of the center of gravity of the load and the mover will be updated in the motion control profile. According to the Tuning documentation, depending on the construction of the load and natural resonant frequencies it may be the case that it is not possible to tune the load mover combined. The load must be redesigned.

Servo Control Technology

The XTS Soft Drive is equivalent to a Servo Drive, but it has been developed for the XTS system. For the XTS, the Soft Drive calculates the current command and gives that current (force) to the XTS Driver to commutate. A typical Servo Drive Configuration has the following setup.

$$\text{Error} = \text{Value}_{\text{Setpoint}} - \text{Value}_{\text{Actual}}$$

Actual motor values might not be equal to setpoint values because error always exists in the system. It is possible to reduce the error in the system by tuning properly. As a result, this will reduce the error in the shortest possible time and improve system stability, and to give a more predictable response.

The Controller calculates the motion control profile and provides target positions to the Servo Drive which calculates and turns resulting numbers to a current. The current goes to a motor, flows through coils and provides a torque which moves the mover. The motor contains an encoder that the Drive can read and turns that back to the Drive. The Drive performs the following separate but related tasks.

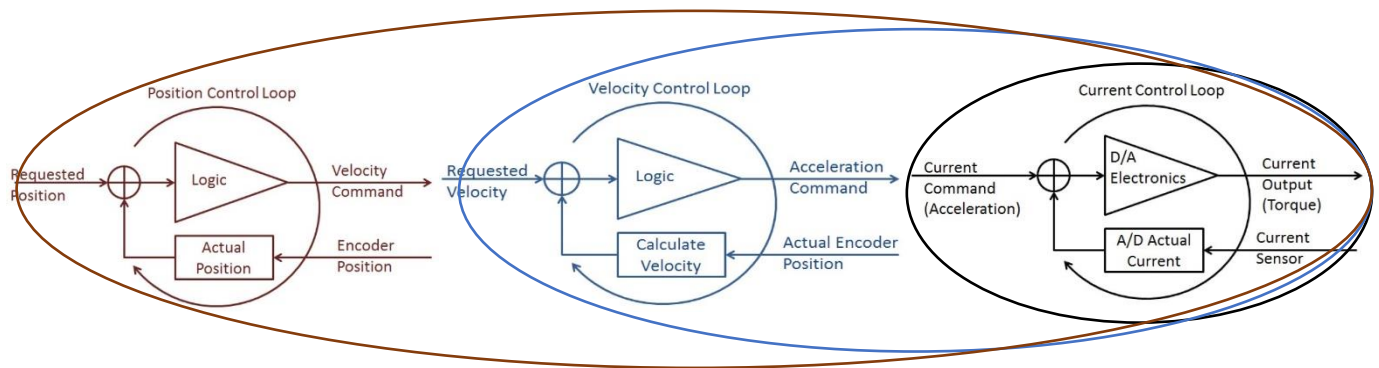


Figure 55: Cascade control [21]

Cascade control has many layers in which the outer layer controls the inner layer. As can be seen in Figure 55, the output of the position loop drives the input of the velocity loop which its output drives the input of the current loop.

The inner loops always have a higher update rate compared to the outer loops to ensure the outer loop sends the right commands to the inner loop. That means the current control loop has the highest update rate, and the position control loop has the slowest update rate.

Current/Torque control loop

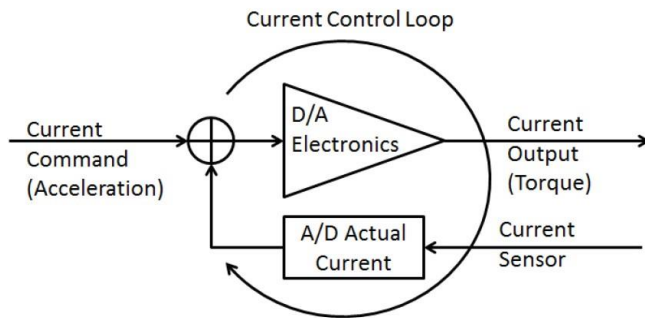


Figure 56: Current/Torque control loop [21]

The Drive ensures the amount of the current flowing through the coil matches with the amount of current requested. Constant current flow is equal to constant torque. The Drive also must ensure that the current control loop work well. If the current/torque/force cannot be controlled accurately, nothing else will function

Velocity control loop

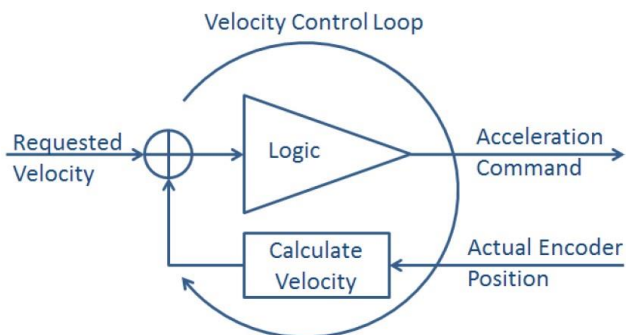


Figure 57: Velocity control loop [21]

As the name implies, the next loop is responsible for the velocity of the system. The first integral of acceleration and the first derivative of position. This control loop is generally not calculated as often as the current loop because it takes time for the acceleration to be converted in velocity (integration), and position to be converted in velocity (derivative).

The velocity control loop takes a requested velocity and based on the actual encoder position; an actual velocity is calculated. After that, a new acceleration command is generated to speed the mover up or slow it down. To ensure the velocity control loop work properly, the current control loop must work properly.

Position control loop

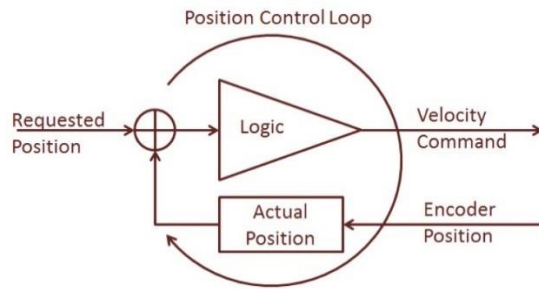


Figure 58: Position control loop [21]

The final loop is the position control loop. This loop works the same as the velocity loop. The loop makes use of the position as feedback and outputs a new commanded velocity. Typically, this loop runs at the slowest update rate. If the velocity control loop doesn't work correctly, the position control loop has no chance to control the position.

Tuning procedure

According to the Tuning documentation, the current control loop is performed within the motor module. This loop is optimized and does not need to be tuned nor be accessible to end-users. Some parameters exist in TwinCAT that should never be changed.

The Soft Drive is located under each individual Axis, and it is made of several parts, as shown in Figure 59.

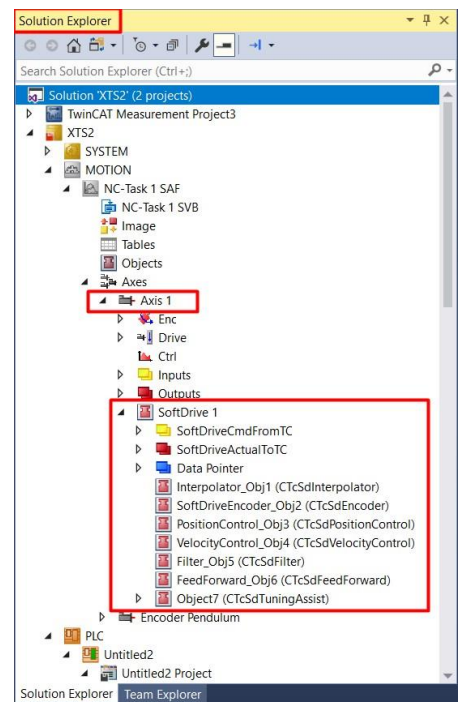


Figure 59: Soft Drive

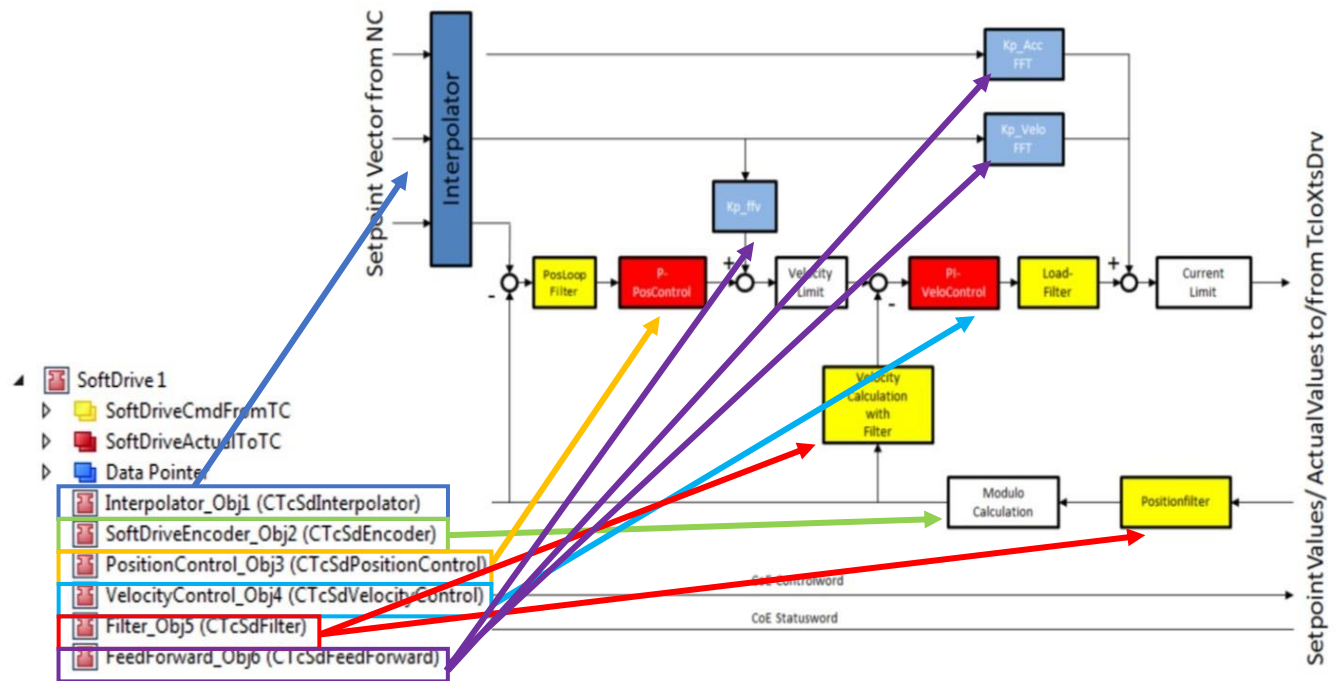


Figure 60: Cascading Gain diagram, and how the gain structure is laid out

The Position Control Loop allows users to change Proportional Gain of the system.

The Velocity Control Loop allows users to adjust Velocity Proportional Gain as well as the Integral time constant.

The Filter Object will help to reduce the resonant frequencies which may cause Axis oscillation.

The Feed Forward object will adjust the acceleration Feedforward Proportional Gain.

In the following lines, I will describe shortly the way of working in order to tune the control loops:

The order in which to tune the control loops is the following:

- Disable the Position and Velocity Control Loop.
- Enable the Tuning Assist object.
- Eliminate resonant frequencies. The goal of eliminating resonant frequencies is to get rid of an unpleasant sound, and select filters needed for the system.
 - o The use of the Tuning Assist can determine the filters and filter settings needed much more precisely than trial and error.
 - o Changing these filters until the mover settles nicely without significant oscillations. As a result, an unpleasant sound will be reduced significantly.
- Disable the Tuning Assist object.

- Re-enable and tune the Velocity Control Loop. The goal of tuning the Velocity Control Loop is to get the mover to respond as quickly and accurately as possible to new velocity commands. The Velocity Command should change as quickly as possible.
 - First, set all Integral parameters to zero. With T_n set to zero, the system is no longer looking at the error over time.
 - Adjust the Velocity K_p so that it brings the velocity to 85-90% of the commanded velocity.
 - Then, adjust the Velocity K_i , or integral time constant, so that the system has the velocity overshoot by about 5-10% and settle quickly. The Velocity K_p might be increased or decreased slightly to give a final velocity tuning.
- Re-enable and tune the Position Control Loop. With the Velocity Control Loop tuned, the Position Control Loop can be adjusted. If the Velocity Control Loop is not well tuned, it is impossible to tune the Position Control Loop.
 - The feedforward values will be adjusted here.
 - The Position K_p can be adjusted so that the following error is as small as possible through the move but stop before getting more noise or oscillation.

After doing all steps above, the mass, the stiffness, the location of the center of gravity of the load and the mover have been updated in the motion control profile. The system is ready to be controlled by NC and PLC.

Appendix G. MoveAbsolute UML Code

@startuml

'----- MC_MOVEABSOLUTE@BECKHOFF

hide empty description

[*] --> STATE_INIT

STATE_INIT --> STATE_ENABLE: NOT power.Enable AND NOT Reset.Execute AND NOT MoveAbsolute.Execute

STATE_INIT: power.Enable:= FALSE

STATE_INIT: Reset.Execute:= FALSE

STATE_INIT: MoveAbsolute.Execute:= FALSE

STATE_ENABLE --> STATE_ERROR: power.Error

STATE_ENABLE --> STATE_1stPOSITION: power.Status

STATE_ENABLE: power.Enable:= TRUE

STATE_ENABLE: power.Enable_Positive:= TRUE

STATE_ENABLE: power.Enable_Negative:= TRUE

STATE_ENABLE: power.Override:= 100

STATE_1stPOSITION --> STATE_ERROR: MoveAbsolute.Error

STATE_1stPOSITION --> STATE_2ndPOSITION: NOT MoveAbsolute.Busy AND MoveAbsolute.Done

STATE_1stPOSITION: MoveAbsolute.Position:= 450 //mm

STATE_1stPOSITION: MoveAbsolute.Velocity:= 550 //mm/s

STATE_1stPOSITION: MoveAbsolute.Execute:= TRUE

STATE_2ndPOSITION --> STATE_1stPOSITION: NOT MoveAbsolute.Busy AND MoveAbsolute.Done

STATE_2ndPOSITION --> STATE_ERROR: MoveAbsolute.Error

STATE_2ndPOSITION: MoveAbsolute.Position:= 110 //mm

STATE_2ndPOSITION: MoveAbsolute.Velocity:= 400 //mm/s

STATE_2ndPOSITION: MoveAbsolute.Execute:= TRUE

STATE_ERROR --> STATE_INIT: NOT Reset.Busy AND Reset.Done

STATE_ERROR: Reset.Execute:= TRUE

@enduml

Appendix H. Hold and drop strategy UML code

@startuml

'----- HoldAndDrop@BECKHOFF

hide empty description

[*] --> STATE_INIT

STATE_INIT --> STATE_ENABLE: NOT power.Enable AND NOT Reset.Execute AND NOT EnableSetPointGenerator.Execute AND NOT DisableSetPointGenerator.Execute

STATE_INIT: power.Enable:= FALSE

STATE_INIT: Reset.Execute:= FALSE

STATE_INIT: EnableSetPointGenerator.Execute:= FALSE

STATE_INIT: DisableSetPointGenerator.Execute:= FALSE

STATE_ENABLE --> STATE_ERROR: power.Error

STATE_ENABLE --> STATE_WAITING: power.Status

STATE_ENABLE: power.Enable:= TRUE

STATE_ENABLE: power.Enable_Positive:= TRUE

STATE_ENABLE: power.Enable_Negative:= TRUE

STATE_ENABLE: power.Override:= 100

STATE_WAITING --> STATE_EQUILIBRIUM: pendulum is in the Equilibrium range

STATE_WAITING --> STATE_ERROR: DisableSetPointGenerator.Error

STATE_WAITING: DisableSetPointGenerator.Execute:= TRUE

STATE_EQUILIBRIUM --> STATE_WAITING: pendulum is not in the Equilibrium range

STATE_EQUILIBRIUM --> STATE_ERROR: EnableSetPointGenerator.Error

STATE_EQUILIBRIUM: EnableSetPointGenerator.Execute:= TRUE

STATE_EQUILIBRIUM: EnableSetPointGenerator.Position:= 0

STATE_EQUILIBRIUM: EnableSetPointGenerator.PositionType:=
POSITIONTYPE_ABSOLUTE

STATE_ERROR --> STATE_INIT: NOT Reset.Busy AND Reset.Done

STATE_ERROR: Reset.Execute:= TRUE

@enduml

Appendix I. Full motion strategy UML code

@startuml

'----- FullMotion@BECKHOFF

hide empty description

[*] --> STATE_INIT

STATE_INIT --> STATE_ENABLE: NOT power.Enable AND NOT Reset.Execute AND NOT EnableSetPointGenerator.Execute AND NOT DisableSetPointGenerator.Execute

STATE_INIT: power.Enable:= FALSE

STATE_INIT: Reset.Execute:= FALSE

STATE_INIT: EnableSetPointGenerator.Execute:= FALSE

STATE_INIT: DisableSetPointGenerator.Execute:= FALSE

STATE_ENABLE --> STATE_ERROR: power.Error

STATE_ENABLE --> STATE_DIRECTION: power.Status

STATE_ENABLE: power.Enable:= TRUE

STATE_ENABLE: power.Enable_Positive:= TRUE

STATE_ENABLE: power.Enable_Negative:= TRUE

STATE_ENABLE: power.Override:= 100

STATE_DIRECTION --> STATE_HOMING_LEFT: DisableSetPointGenerator.Done AND the pendulum stays on the right

STATE_DIRECTION --> STATE_HOMING_RIGHT: DisableSetPointGenerator.Done AND the pendulum stays on the left

STATE_DIRECTION --> STATE_ERROR: DisableSetPointGenerator.Error

STATE_DIRECTION: DisableSetPointGenerator.Execute:= TRUE

STATE_HOMING_LEFT --> STATE_EQUILIBRIUM: pendulum is in the Equilibrium range

STATE_HOMING_LEFT --> STATE_ERROR: MoveAbsolute.Error

STATE_HOMING_LEFT: DisableSetPointGenerator.Execute:= TRUE

STATE_HOMING_LEFT: MoveAbsolute.Position:= 300

STATE_HOMING_LEFT: MoveAbsolute.Velocity:= 550

STATE_HOMING_LEFT: MoveAbsolute.Execute:= TRUE

STATE_HOMING_RIGHT --> STATE_EQUILIBRIUM: pendulum is in the Equilibrium range

STATE_HOMING_RIGHT --> STATE_ERROR: MoveAbsolute.Error

STATE_HOMING_RIGHT: DisableSetPointGenerator.Execute:= TRUE

STATE_HOMING_RIGHT: MoveAbsolute.Position:= 200

STATE_HOMING_RIGHT: MoveAbsolute.Velocity:= 550

STATE_HOMING_RIGHT: MoveAbsolute.Execute:= TRUE

STATE_EQUILIBRIUM --> STATE_DIRECTION: pendulum is not in the Equilibrium range

STATE_EQUILIBRIUM --> STATE_ERROR: EnableSetPointGenerator.Error

STATE_EQUILIBRIUM: EnableSetPointGenerator.Execute:= TRUE

STATE_EQUILIBRIUM: EnableSetPointGenerator.Position:= 0

STATE_EQUILIBRIUM: EnableSetPointGenerator.PositionType:= POSITIONTYPE_ABSOLUTE

STATE_ERROR --> STATE_INIT: NOT Reset.Busy AND Reset.Done

STATE_ERROR: Reset.Execute:= TRUE

@enduml

References

- [1] "beckhoffinfo," [Online]. Available: <https://www.beckhoff.com/nl-nl/>.
- [2] "acrome.net," [Online]. Available: <https://acrome.net/post/what-is-a-linear-inverted-pendulum>. [Accessed 2021].
- [3] "pearson.com," [Online]. Available: <https://www.pearson.com/uk/educators/higher-education-educators/program/Franklin-Feedback-Control-of-Dynamic-Systems-Global-Edition-7th-Edition/PGM1069431.html>.
- [4] "latexdraw.com," [Online]. Available: <https://latexdraw.com/free-body-diagram-of-an-inverted-pendulum-in-tikz/>.
- [5] "galco.com," [Online]. Available: <https://www.galco.com/comp/prod/moto-ac.htm>.
- [6] "create.arduino.cc," [Online]. Available: <https://create.arduino.cc/projecthub/ryanchan/how-to-use-the-l298n-motor-driver-b124c5>.
- [7] "forum-cnc.pl," [Online]. Available: <http://forum-cnc.pl/index.php?action=printpage;topic=1197.0>.
- [8] "wikimedia.org," [Online]. Available: <https://upload.wikimedia.org/wikipedia/commons/thumb/f/f6/Linearmotorprinzip.png/330px-Linearmotorprinzip.png>.
- [9] "beckhoffus.com," [Online]. Available: <https://learn.beckhoffus.com/enrollments/96608648>.
- [10] "EtherCAT – the Ethernet Fieldbus," [Online]. Available: https://www.ethercat.org/pdf/english/ETG_Brochure_EN.pdf.
- [11] "marposs.com," [Online]. Available: <https://www.marposs.com/eng/application/stator>.
- [12] "dragonwinch.com," [Online]. Available: <https://www.dragonwinch.com/en/rotor,84,465.html>.

- [13] "beckhoff.com," [Online]. Available:
https://download.beckhoff.com/download/document/motion/xts_ba_en.pdf.
- [14] "cx5140," [Online]. Available: <https://www.beckhoff.com/en-en/products/ipc/embedded-pcs/cx5100-intel-atom/cx5140.html>.
- [15] "MC_MoveAbsolute," [Online]. Available:
https://infosys.beckhoff.com/content/1033/tcplclib_tc2_mc2/70094731.html .
- [16] "MC_MoveVelocity," [Online]. Available:
https://infosys.beckhoff.com/content/1033/tcplclib_tc2_mc2/70102411.html?id=590573568675090408 .
- [17] "kubler.com," [Online]. Available:
<https://www.kuebler.com/us/products/measurement/encoders/product-finder/product-details/F3653>.
- [18] "BiSS and SSI - An Overview," [Online]. Available:
<https://www.celeramotion.com/zettlex/support/technical-papers/biss-and-ssi-an-overview/>.
- [19] "Basic introduction to feedback control," [Online]. Available:
<https://www.eecs.umich.edu/courses/eecs373.w05/lecture/control.html>.
- [20] "AX5000 | Digital Compact Servo Drive," [Online]. Available:
https://infosys.beckhoff.com/english.php?content=../content/1033/ax5000_system_doku_hw2/647266187.html&id=1824986000795241014.
- [21] "Servo Control Technology," [Online]. Available:
https://infosys.beckhoff.com/content/1033/xts_soft_drive/3526933643.html .
- [22] "PLC Libraries Motion," [Online]. Available:
https://infosys.beckhoff.com/content/1033/tcplclib_tc2_mc2/index.html?id=4786081293094367280 .
- [23] "EtherCAT," [Online]. Available: https://ethercat.org/en/why_use_ethercat.htm.

- [24] "TF4100 | TwinCAT 3 Controller Toolbox," [Online]. Available:
<https://www.beckhoff.com/en-en/products/automation/twincat/tfxxx-twincat-3-functions/tf4xxx-tc3-controller/tf4100.html>.
- [25] "FB_CTRL_PID," [Online]. Available:
https://infosys.beckhoff.com/content/1033/tf4100_tc3_controller_toolbox/245435787.html?id=3129912176963096789 .
- [26] "Tuning a PID regulator," [Online]. Available: https://isd-soft.com/nl/tech_blog/tuning-pid-regulator/.
- [27] "twincat," [Online]. Available: <https://www.beckhoff.com/en-en/products/automation/twincat/>.