External measurements and analysis of slew motion of the crane

Ismail Mohamed Hussein

Supervisor

Michael Ruderman

University of Agder, 2019

Faculty of Technology and Science

Department of Engineering

Preface

This report presents the method and results of a Master thesis given by the University of Agder within the mechatronics study. The thesis was part of the final work, which constitutes 30 out of 120 study points necessary to receive a master degree. This research originally stemmed from my passion for modeling and system identification of mechatronic systems.

In trust, I could not have achieved my current level of success without the active support of my mother, who support me with love. Further, I will want to thank my supervisor form University of Agder, Associate Professor Michael Ruderman, and the mechatronic lab engineers. The help and advice they gave me throughout the research process were crucial.

Ismail Mohamed Hussein

Abstract

Cranes and other large machines with hydraulic cylinder are the first choices in various heavy duty application. The step of modeling and controlling states like torque, force, and motion control feature difficulties due to nonlinearities on the system. This thesis research describes the simplified and full-order model of a HMF-2020-K4 crane slew motion joint. The description covers full dynamic analysis which can be utilized in control design. The simplified model neglects the direction control valve, orifice, and continuity equations in theoretical assumptions. The simplified model is second-order in terms of mathematical modeling and system identification through a black box method. The full-order takes into account the direction control valve, both orifice, and continuity equation and the mechanical part of the crane. The resulted model is of a fifth-order. The nonlinear parts for both models are identified and compensated in term of dead-zone. Furthermore, the experimental results incorporate internal and external measurements through various sensing devices. This thesis also presents different types of results like the simulation of both models and sub-models of the full-order system where the simulation result are verified with the exact measurement of the system.

Contents

List of Figures

List of Tables

Chapter 1
Introduction

Cranes and other large scale machines with multiple degrees of freedom feature complex multibody dynamics with a coupled motion of the actuated joints and links. This master thesis focuses on investigating and analyzing the slewing actuator of the crane in the Mechatronics laboratory at the university in Agder. The slewing actuator is the most used, crucially important and complex to operate [1, 2]. Therefore analyzing and examining the system dynamics is essential to develop the model and design control. The main task is to investigate and model slew joint by using the external 3D measurements of reference points on the moving structure and combine with internal data for improvements. Based on the literature review [3] hydraulic cylinder model can be presented as full- or reduce order model. In this thesis, both models will be reviewed, and the equations explained. Furthermore, identification methods like a black box, gray box, and white box are utilized based on the literature [4].

1.1 Task description

The requirements and tasks given for this thesis were:

- Mechatronic design of complex experiments and system instrumentation
- Extension of the available real-time interfaces, programming and data processing.
- Signals analysis, modeling, identification of system behavior, reverse engineering.
- Design of the estimation algorithms, experimental evaluation, systematic hypothesis tests.

1.1.1 Problem Statement

The slew joint that rotates various links relative to the base and that is most important for the overall performance feature complexity. The build-up of this joint includes rack-and-pinion. This kind of transmission of the slew joint, which is coupled with multiple kinematic chains acts as causes of uncertainties. The undesired behavior occurs even when the accurate dynamic model of the actuators are obtained and controlled. The difficulties in controlling the actuators are due to complex nonlinear behavior in the hydraulic cylinder, which needs exploitation in the modeling and identification of the whole system.

1.2 The Objective of Research Work

This research aim is to investigate and analyze the slew motion of the crane and identify and model the dynamics of the system. Therefore, obtaining the exact rotation of the crane is vital for modeling and identification. To accomplish this, two types of motion measurement are used; internal and external. The internal is a high-resolution position sensor installed on the crane, and external is 3D measurement sensor called C-track. The addition of external data is to increase the knowledge of the slew motion where C-track is implemented to track the precise rotation of the crane in real time. Also, the external measurements obtain can be combined or validate the internal process data of the crane.

Chapter 2

Theoretical Part

A foundation for every claim that is made during the report should be based upon scientific theories and mathematical equations. In this chapter the main theories used will be explained to introduce the reader to the science behind the report. Most theories are based upon information learned through the mechatronics education at UiA unless otherwise specified.

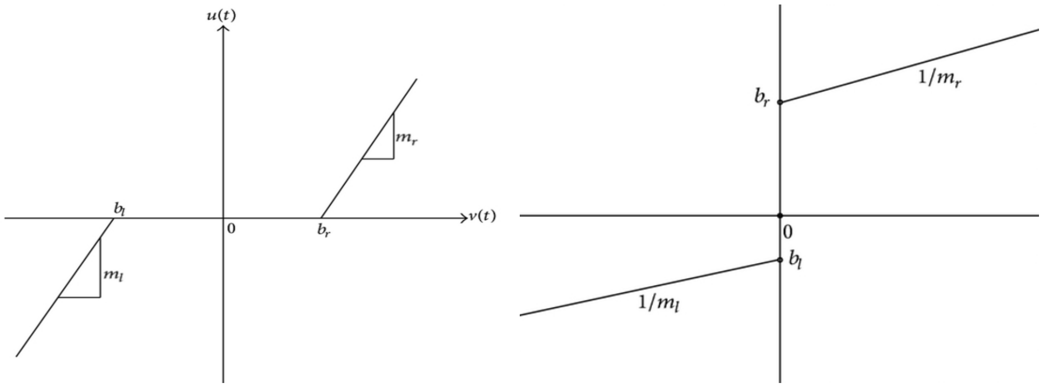
2.1 Butterworth Lowpass Filter

A low-pass filter is a useful application when it comes to removing unwanted high-frequency signal in the application like industrial process or communication. To remove the noise from a signal and leave only the low-frequency requires to have a low-pass filter that is competent to reject high frequency. The Butterworth low-pass filter can be designed in a different order where the higher the order, the better response of the filter becomes. This can be done by arranging the low-pass filter in series.

$$G(j\omega) = \frac{1}{\sqrt{1 + \epsilon^2 (\frac{j\omega}{j\omega_c})^{2n}}} \tag{2.1}$$

2.2 Deadband

A general class of industrial motion control systems has the structure of the dynamical system, usually of the Lagrangian form, preceded by some nonlinearities in the actuator, either dead-zone or saturation[5]. This involves actuators in cranes or hydraulic systems. The challenge of these systems come when they are to be controlled and where the demand for accuracy is essential. To solve the problem of nonlinearity is often assumed that the inverse of the dead-zone function can compensate dead-zone. Further, it is also assumed that dead-zone function of a system can be identified.



(a) Dead zone

(b) Inverse dead zone

Figure 2.1: Dead zone vs. Inverse dead zone [6]

The characteristic of a dead-zone in the figure can be expressed as piecewise function in equation 2.2

$$DZ(x_{out}) = \begin{cases} m_r(x_{out} - b_r) & x_{out} \geq b_r \\ 0 & b_l < x_{out} < b_r \\ m_r(x_{out} - b_l) & x_{out} \leq b_r \end{cases} \tag{2.2}$$

and the inverse dead-zone in equation 2.3

$$DZ^{-1}(x_{in}) = \begin{cases} \frac{1}{m_r}x_{in} + b_r & x_{in} > 0 \\ 0 & x_{in} = 0 \\ \frac{1}{m_l}x_{in} + b_l & x_{in} < 0 \end{cases} \tag{2.3}$$

where b_l and b_r are the breakpoints and m_r,m_l are the constant slope of the dead-zone. The cancellation of nonlinearity occurs when the inverse dead-zone is implemented; this will result in the system overcome the dead-zone and restoring the system output.

$$y = (ax_{in} + b) \cdot DZ^{-1} \cdot DZ = ax_{out} + b \tag{2.4}$$

The inverse dead-zone function 2.3 it is a jump approximate this mean it has a sharp input into a system. The more appropriate inverse function is known as sigmoidal where it has a continuous signal the other is hyperbolic tangent.

$$y = 0.5 \cdot D \cdot \tanh(S \cdot x) + x \tag{2.5}$$

2.3 System Identification

System identification is a method that gives opportunities to generate mathematical models of dynamic systems. This method is very useful when measurements of input/output signals of the system are available. There are three steps involved in developing a mathematical model from data.

- The data
- A set of candidate mode
- A rule by which candidate model can be assigned using data

The data sampling should be recorded where it can gather more information from the experiment. The elements that influence the experiments like nonlinearity should be taken into consideration. The model candidate should be evaluated with previous knowledge; one should able to combine the properties of the model when the model is to be estimated [7]. When estimating one should utilize the previous knowledge of the physical system when selecting the method to model or recognize the system. The approach to identifying the model can be structured as color-code; the are three types of color-code[4].

- white-box model; In this situation, it is possible to generate a mathematical model from the knowledge of the physical law of the system. When this method is used one should, therefore, be able to create an accurate model of the plant.
- Grey-box model; In this situation, some physical knowledge of the system is available, but the unknown parameters should be determined form data estimation.

- Black-box model; In this situation, no physical information of the system is available.

In any identification method, the fundamental factor when it comes to identifying the system is the data. The data can be in different types of format like time domain input-output data, frequency response, etc. Frequency response function data collection is a procedure where any signal like a sine-wave is used as input to excites the system and output is measured. The input signal can be expressed as

$$u = U \sin(\omega t) \tag{2.6}$$

and the output can be expressed as

$$y = Y \sin(\omega t + \phi) \tag{2.7}$$

When $u(t)$ is applied to a system, the output y(t) will have different magnitude and a shift phase respect to the system. .The relationship between phase shift and a delay time of sine-wave can be described as

$$\phi = -\frac{2\pi}{T_p} \cdot \delta t \tag{2.8}$$

2.4 State-space

State-space modeling is a useful method when describing complex dynamic system of the n-th order. To reduce a 2nd order system, which is what will be described here, to a 1st order system, the dynamic state variables, x_1 and x_2, can be introduced. The state space design is one of the dominant aspects in control theory and modeling. The advantage of State-space is that it works directly with state variables in the form of the first order differential equation. The conversion of higher order to first order differential equation is often referred to as state space representation. For instance, a second order system can be represented in the form of state space as shown in equation 2.9 to 2.11.

$$\ddot{x} + \dot{x} = u \tag{2.9}$$

$$\dot{x}_1 = x_2 \tag{2.10}$$

$$\dot{x}_1 = \ddot{x}_1 \tag{2.11}$$

Where the input is u. These equations can be expressed in the state variable form:

$$\dot{\mathbf{x}} = A\mathbf{x} + \mathbf{B}u \tag{2.12}$$

$$y = \mathbf{C}\mathbf{x} + \mathbf{D}u \tag{2.13}$$

Where the output is y, the system matrix is A, the input matrix is B, the output matrix is C and the feedforward matrix is D. In the case where the system does not have a direct feedforward, D is a zero matrix.

2.5 Estimator

Estimator, also known as the observer, is a vital part in state-space control design. In some cases, the sensor that is needed may be expensive or impossible to place, for example a sensor in systems like a rocket or nuclear power plant. An estimator becomes useful in these scenarios. Estimating requires the full order of the plant dynamics. The representation of the model is the same as in equation 2.12.

$$\dot{\hat{\mathbf{x}}} = A\hat{\mathbf{x}} + \mathbf{B}u \tag{2.14}$$

The difference is that the state-variable $\hat{\mathbf{x}}$ is the estimated of the actual state $\mathbf{x}$. If the matrices A and B are known, the estimator can be constructed and have satisfactory result if the initial condition of the state $\hat{\mathbf{x}}(0)$ and $\mathbf{x(0)}$ are correct. The information of the initial term is critical because both models will be run in parallel. In the real world, the plant is the actual physical system where the model of the observer is a mathematical equation (2.14).

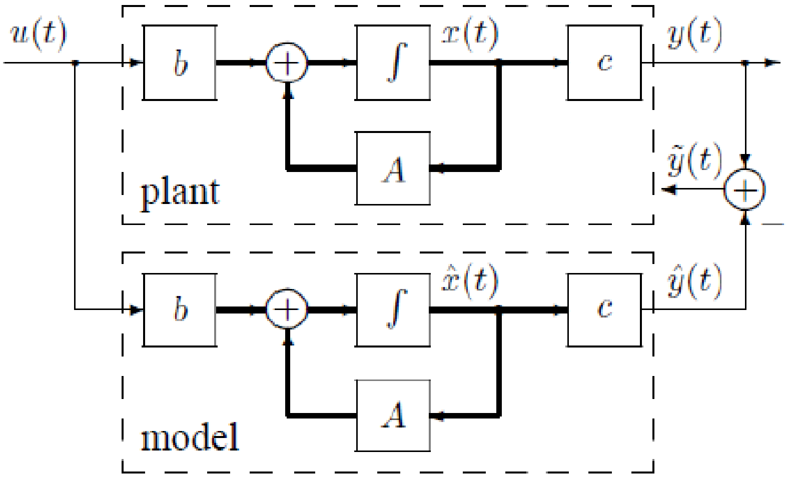


Figure 2.2: Close loop estimator

The plant starting position should be provided to the estimator model for it to track the true state. Otherwise, if the estimation of the initial condition is wrong or has some deviation, the observer model error will slowly converge to zero or diverge from the actual state. This depends on the estimation of the initial condition. To influence the estimator to converge to the actual state, the feedback is introduced to the model. The feedback signal is the difference between the actual state and the estimated which is known as an error.

$$\dot{\hat{\mathbf{x}}} = A\hat{\mathbf{x}} + \mathbf{B}u + \mathbf{L}(y - \boldsymbol{C}\hat{\boldsymbol{x}}) \tag{2.15}$$

Equation 2.14 can then be expanded to equation 2.15 since a new input signal was added to the model. The L term is a proportional gain that is chosen to make sure the error converges to zero. The error dynamic of the model can be obtained by manipulating equations 2.15 and 2.12.

$$\dot{\hat{\mathbf{e}}} = (\boldsymbol{A} - \boldsymbol{LC})\hat{e} \tag{2.16}$$

The equation is supposed to converge rapidly to zero. The choice of L that determines equation 2.16 has to have stable and reasonably fast eigenvalues. This will cause the observer error to remain zero. However, if the matrices A, B, and C are not identical to the actual plant, the zero error will be impossible to achieve by equation 2.16. Nevertheless, it may still be possible to find $\boldsymbol{L}$ such that the difference from the actual state will be stable and have a small error value [8].

Chapter 3

Experimental Plant

In this chapter, plant information is given for each hardware and software used in this research. It is essential to gather knowledge of the system and gain valuable information on the existing system. The crane used in this thesis was a HMF-2020-K4 loader crane located in the mechatronic laboratory at the University in Agder, shown in figure 3.1. The crane as a whole consists of several parts like a valve, hydraulic system, and electrical connections used in sensors and communication to extranal Real-Time Machine. The crane also consists of 5 joints, where only the slew joint will be analyzed, and the functionality and known variables will be exposed. Further, workstation and Real-time interface of the system will be explained in depth.

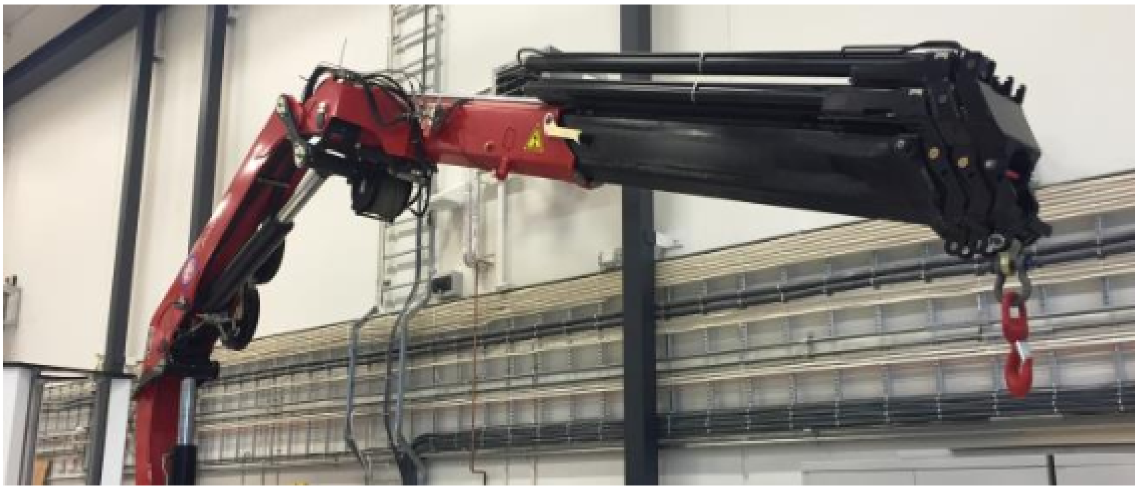
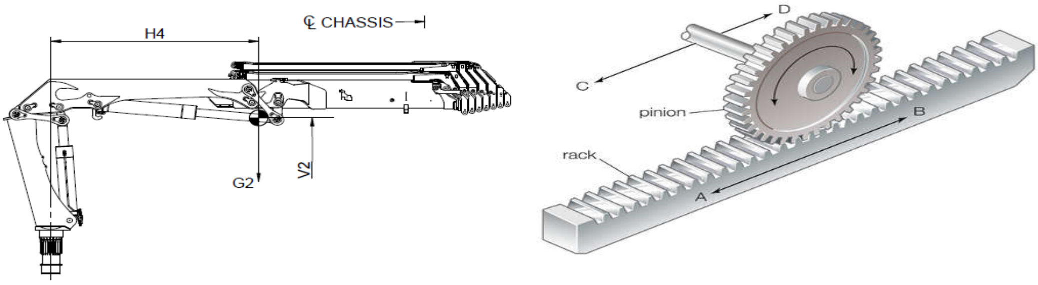


Figure 3.1: HMF-2020-K4 loader crane

3.1 Rack and Pinion

The slew motion joint on the crane is driven by rack and pinion. The rack and pinion are composed of a helical gear where the pinion is fixed at the base of the crane, and the rack helical gear is on a hydraulic piston. The rotation of the crane occurs when the rack is being driven linearly which causes the pinion to rotate the crane.



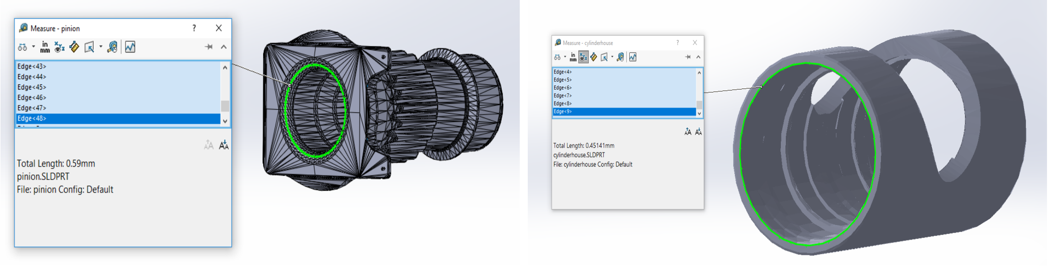
(a) Pinion of the crane

(b) rack and pinion

[9]

Figure 3.2: Linear to rotational motion [9]

The crane consists of four-cylinder which control different joint in the system. One of the cylinders linear motion is converted to rotation motion by the used of pinion connected to the cylinder piston.



Edge<43>
Edge<44>
Edge<45>
Edge<46>
Edge<47>
Edge<48>

Total Length: 0.59mm
pinion.SLDPRT
File: pinion Config: Default

Edge<4>
Edge<5>
Edge<6>
Edge<7>
Edge<8>
Edge<9>

Total Length: 0.45141mm
cylinderhouse.SLDPRT
File: cylinderhouse Config: Default

(a) Pinion of the crane

(b) Cylinder house

Figure 3.3: Rack and pinion

A VRML CAD model of the crane was used to find the dimensions of the rack and pinion. The parameters need for modeling a the pinion radius and the area of the cylinder. The area of the cylinder and pinion radius are given in the table 3.1.

Slewing:

Part	**Radius**	**Area mm**
Pinion	100[mm]	...
Cylinder	73[mm]	8419.3[mm]

Table 3.1: Dimensions of pinion and Cylinder

The hydraulic system of the slew motion consists of several hydraulic units like an oversentral valve, filter, and Hydraulic power unit(HPU). The cylinder used in generating the force is a double acting cylinder which has a port at both ends of the boring side. The rod of the cylinder is not subjected to hydraulic fluid since the piston is used as a rack.

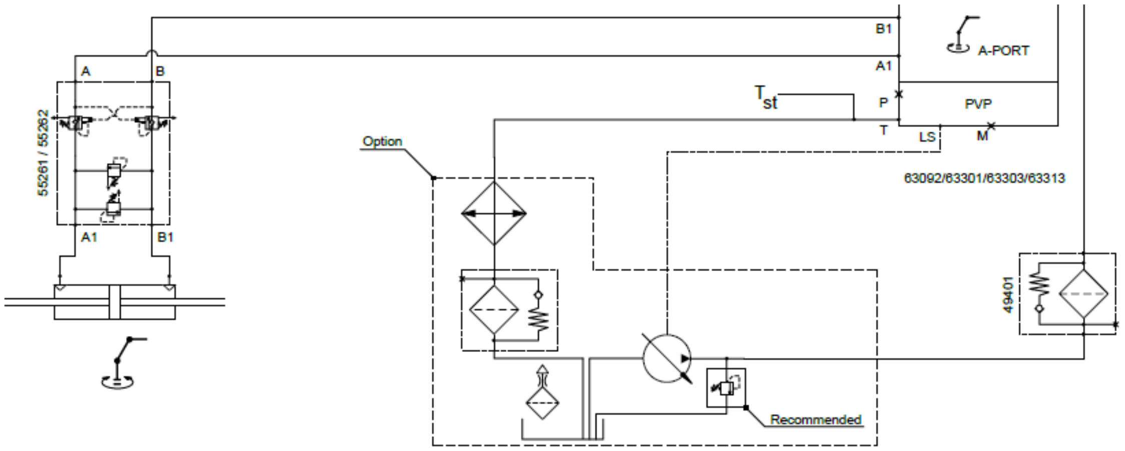


Figure 3.4: Hydraulic System

3.2 Valve

Valve

Abbreviation	Description
PLC	Programmable Logical Circuit
P_p	Pilot Pressure
PVE	Proportional Valve Electric actuator
PVB	Proportional Valve Basic module - valve slice
PVG	Proportional multi-section Valve Group
PVM	Proportional Valve Manual control with handle
SP	PVE with Spool Position feedback
U_{DC}	Power supply Direct Current
U_S	Steering voltage for the PVE control

Table 3.2: List of Abbreviations

The valve installed in the crane to controlled the actuates was SAUER DANFOSS PVE- Series 4. The type of valve is PVES(Proportional Valve Electric actuator) where the term S stands for a variant with 0-2 hysteresis. The PVE can be controlled manually, joystick or PLC. The PVG comes in stack up to 12 individual, but in the crane 6 PVE was installed to control different actuates in the crane. The PVE was attached in PVB which are grouped to become PVG.

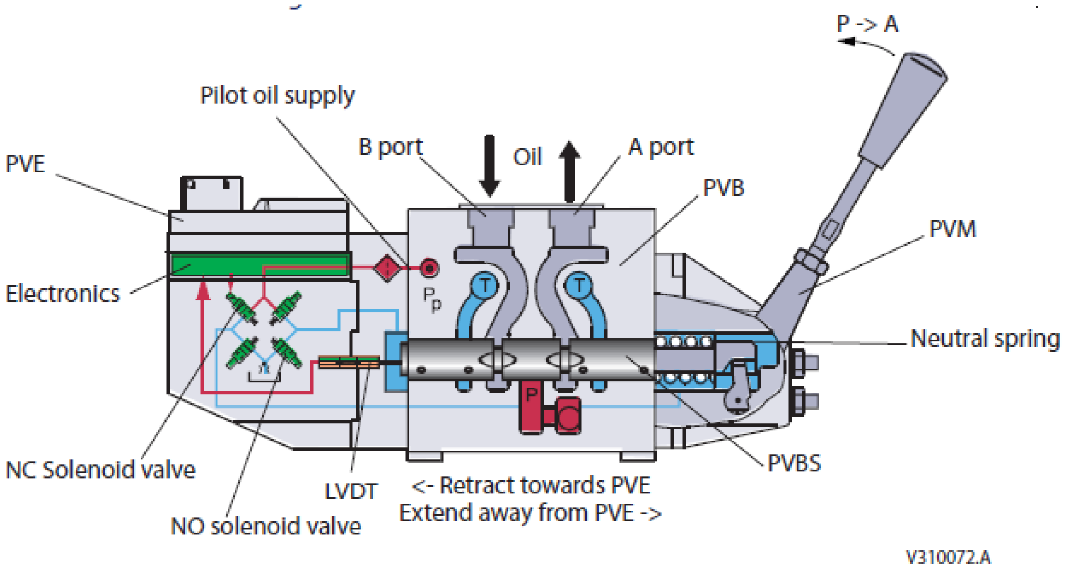
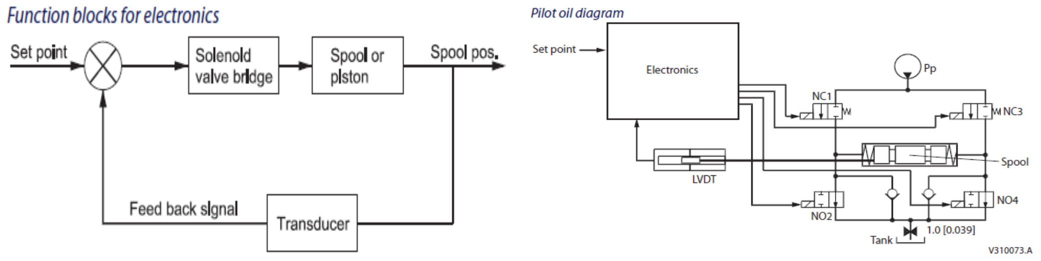


Figure 3.5: Valve Sections

There are two methods of controlling the oil flow out of the port (A or B). The fist is PVE controlling the spool using the pilot oil pressure and a manual handle(PVM) which is a mechanical interface with the spool. The functionality of the PVE depends on the embedded hydraulic, mechanical and electrical condition given by the application. The hydraulic in the valve moves the spool which allows the flow to a specified workflow in (Port A or B). The hydraulic circuit in the valve uses a solenoid which controll the pilot pressure on the spool ends.



(a) PVE Control System

(b) Valve Hyraulic System

Figure 3.6: Electronic and Hydraulic System of the Valve

The electronic system in the valve is used to control the spool position. The PVE features a feedback control and transducer which makes possible to measure spool movements. The set-point U_s is the reference value which commands the spool to the desired position. The reference value U_s determine the pressure (P_p) which causes the spool to moves. The supply voltage of the valve is 24 volt, and the spool position set-point value in voltage is U_sp. The range in spool voltage is 0.5v to 4.5V which represent the spool travel 0-7mm from both port A and B. The spool position voltage U_sp is directly related to the feedback signal that is measured from the valve U_s.

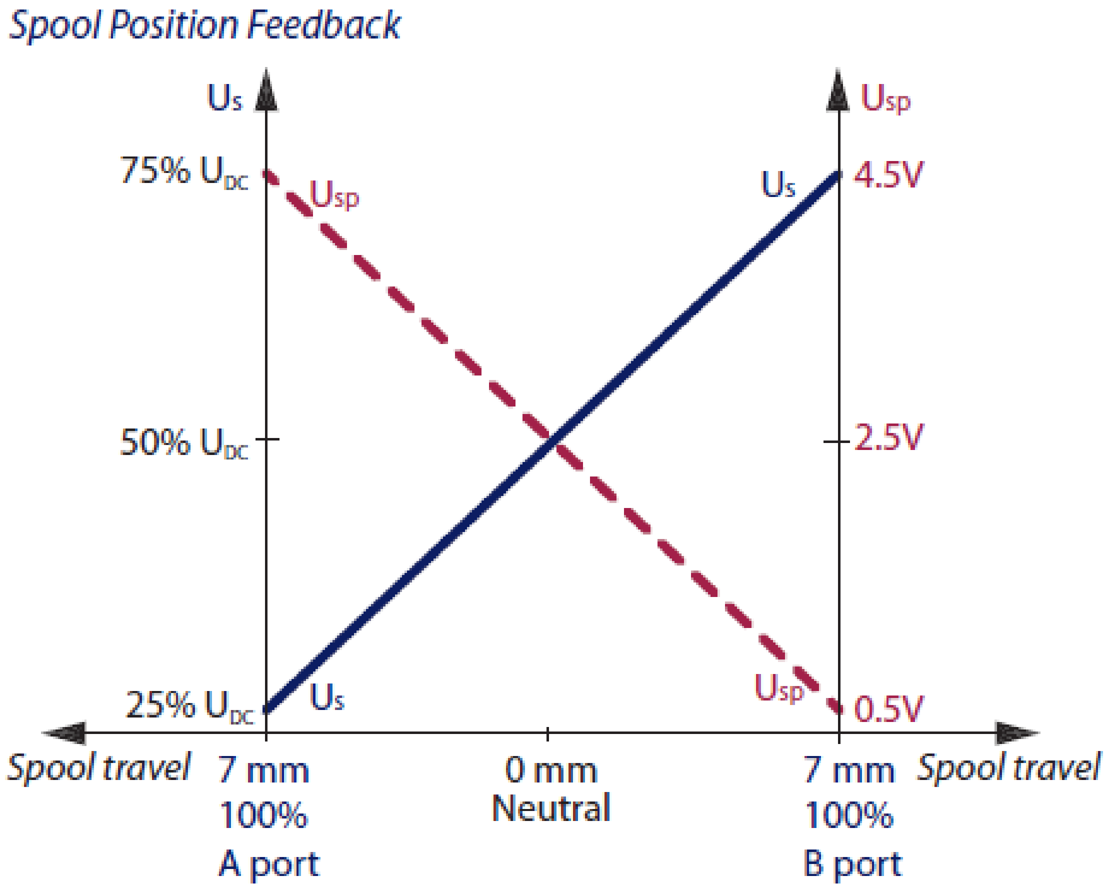


Figure 3.7: The valve characteristic

The open valve characteristic in figure3.7 shows the 2.5 V as the neutral position.The valve open from port A to port B is linear by 2V is each position.

3.3 Sensors

The sensor used in this research is separated into an internal and external sensor. The are two types of internal sensors the first sensor type is the sensor of the slew motion and second is the pressure of the cylinder. The external sensor is a sensor of the slew motion of the system which is used to verify the internal sensor. The internal sensor used in measuring the slew motion of the crane is a $ASMWS17KT$ position sensor. The sensor utilizes a potentiometer to measure a distance where the output data is voltage in a range 0-10V.

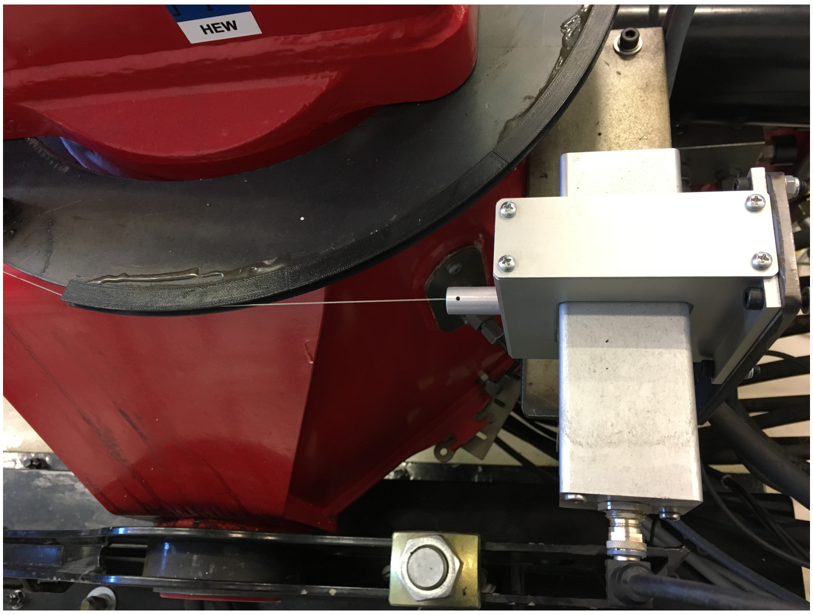


Figure 3.8: ASM Position Sensor

The sensor uses a circular disk to measure the angular of the slew motion, and maximum measuring length is 15000mm. Pressure is measured by $ParkerSCP01$ pressure sensor. The range of the sensor is 0-400 bar, and the output signal is 0-10V.

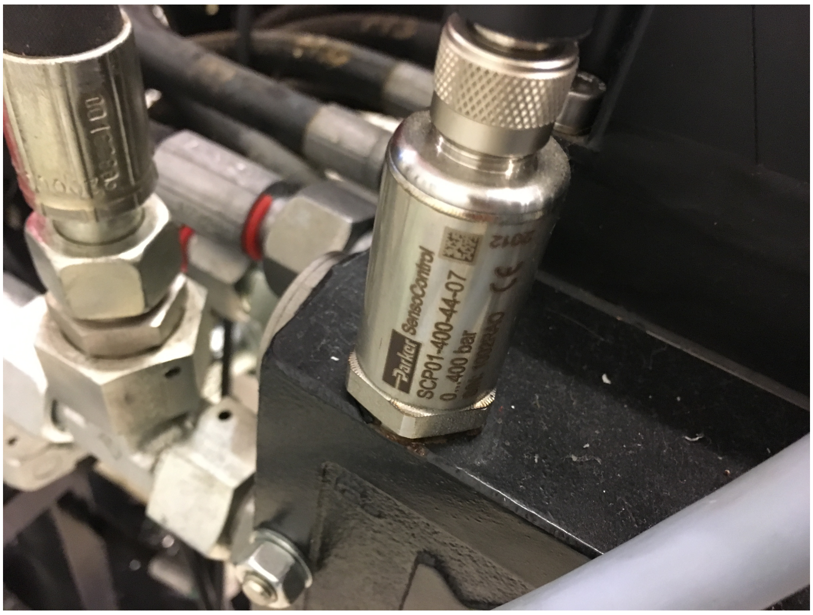
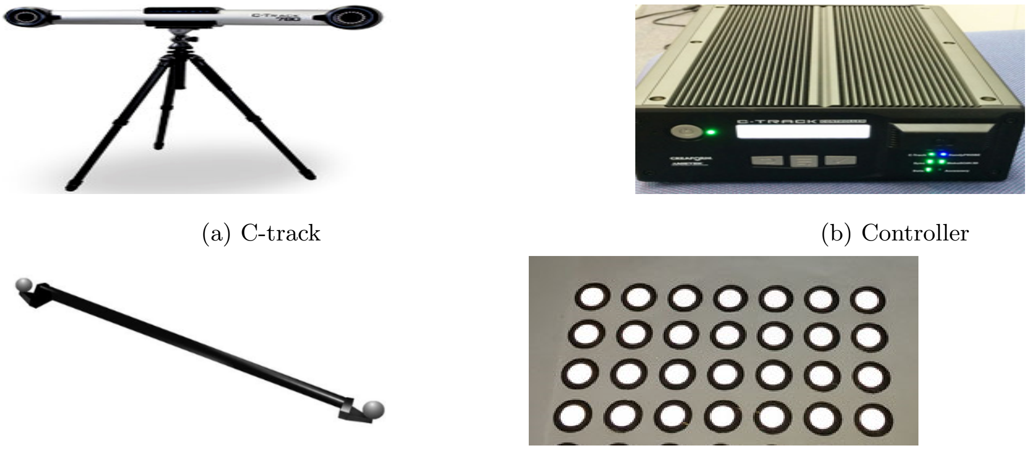


Figure 3.9: Parker SCP01 pressure sensor

Three pressure sensors were used in this research. The sensor in the figure 3.9 was installed in the inlet valve of the supply pressure from the HPU, and the others were installed at the bore side of the cylinder.

3.4 C-Track

The external sensor used to validate the slew motion of the crane is a type of laser tracker. The C-track provided an accurate measurement of rotation of the crane. The C-track uses the same functionality as other lasers, and it consists of two sensing camera which is sensing the reflectors which are attached in the crane. The C-track system consists of several parts which make possible to track the slew motion of the crane. Those parts are reflectors, a controller, and calibrate bar.



(a) C-track

(b) Controller

(c) Calibrate bar

(d) Reflector

Figure 3.10: C-track parts

3.5 System Instrumentation

In this section, all instrumentation and setup of the workstation will be explained. The explanation will cover the connection of the electric diagram of the crane system and how the Compact Reconfiguration Input Output (Compact Rio) is utilized briefly. Furthermore, the signal processing, scaling, and the communication of Real-time interface will be explained.

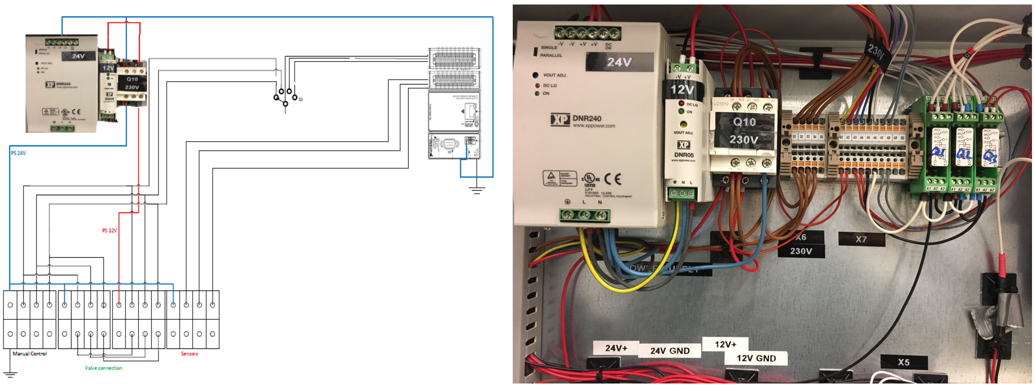
3.5.1 Electric Connection Diagram

The electronic part serves as the complex mechatronic part of the system where it links the mechanical and communication part together. The connection objective was to control the valve and acquire the sensor value installed on the crane. The connection was made in such a way that the crane can be controlled manually and by a computer. The components used are

Name	Model
AC-DC Power supply	DNR240
AC-DC Power supply	DNR05
Magnetic Contact	LC1D12
Switches	....
CompactRIO	cRio9075
Analog Input	NI 9205
Analog Output	NI 9264

Table 3.3: Components

The electrification of the crane system was divided into two subsystems in terms of voltage. The component $DNR240$ convert the alternating voltage from the main power supply to 24 direct voltage where $DNR05$ converts to $12V$. The $24V$ voltage section is the main part of the electrification system where the $12V$ was used to fulfill the $24V$ demand since the analog output of component $NI9264$ is in the range of ± 10. The valve input control voltage is in the range of 25% to 75% of the supply 24V which the analog output of the (Compact Rio) CRio can't meet. The solution was to connect the $12V$ as a natural ground to the analog output.



(a) Electric draw scheme

(b) Electrical Control Box

Figure 3.11: Electric Connection Diagram

Both power supplies a connected to a contactor $LC1D12$ where the main functionality is to cut power supply to the sensor and the valve when the switch is activated. The electronic installation cabinet consists of three switches star/stop, emergency stop and auto/manual switch. The CRio is always on when the main power from the utility switches on.

3.5.2 CompactRIO

The development of the CRio application was necessary for system instrumentation and communication to an external computer. The development of CRio application is embedded with a Labview program where a Labview VI was created to accomplish the task. The Labview application is equipped with RIO scan interface which makes easy to access the Compact Rio series modules in the Labview VI. The Crio is programmable in two ways graphically, and Field programmable gateways(FPGA) which are silicon chips in the Crio. The graphically programming part was used for scaling sensor value, and the FPGA was only dedicated for programming filters in the (input and output) I/O modules. The following software was used to develop the Crio application.

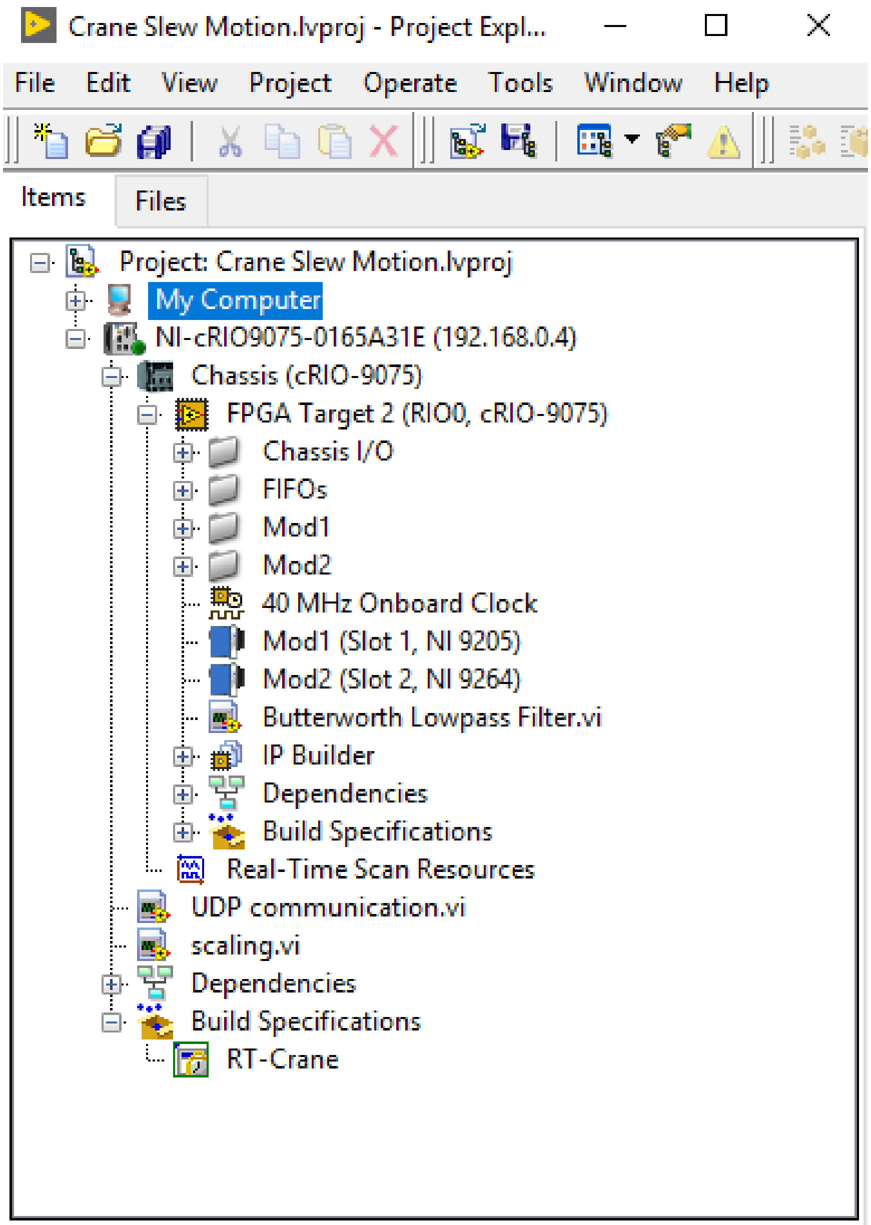
- NI Labview 2018.
- NI Labview Real Time Module.
- NI myRio Tools kit.
- NI CompactRio.
- NI FPGA Module.
- NI FPGA Xillinx Compilation Tools.

The following hardware was used to develop the Crio application.

- cRIO 9075.
- NI 9205.
- NI 9264.

3.5.3 LabVIEW

The hardware configuration and software installation on the controller had to be completed before development of the Labview VI. National instrument Measurement & Automation Explorer(NI MAX) was used for configuration and installation of the cRio. NI MAX was used in IP configuration where static IP, and subnet mask was assigned to the controller. Scan interface mode was utilized on the Labview programme which will allow using C Series modules and easy access to the FPGA.



- Project: Crane Slew Motion.lvproj
  - My Computer
  - NI-cRIO9075-0165A31E (192.168.0.4)
    - Chassis (cRIO-9075)
      - FPGA Target 2 (RIO0, cRIO-9075)
        - Chassis I/O
        - FIFOs
        - Mod1
        - Mod2
        - 40 MHz Onboard Clock
        - Mod1 (Slot 1, NI 9205)
        - Mod2 (Slot 2, NI 9264)
        - Butterworth Lowpass Filter.vi
        - IP Builder
        - Dependencies
        - Build Specifications
      - Real-Time Scan Resources
    - UDP communication.vi
    - scaling.vi
    - Dependencies
    - Build Specifications
      - RT-Crane

Figure 3.12: LabVIEW project with Modules

The LabVIEW crane project in figure 3.12 consists of three (Virtual Instruments)VI. The main

VI is the UDP *Communication* which contains the other two VI inside it. The programming of the VIs was separated into two systems. The first programming was done on a real-time processor and the second on the FPGA. The first two Vis which are for communication and scaling of the sensor values were programmed on $NIcRIO - 9075$ real-time processor. The last Vi is for acquiring data from the sensor and filter the data was programmed on the FPGA.

The communication VI utilizes the (User Datagram Protocol)UDP which provides the mechanism for interacting data packets from one recipient to another. UPD does not guarantee the arrival of the data to the destination. Moreover, the data packets sent may not arrive in the same order they were sent if the data are more substantial. The advantages of using UDP is that it does not need to acknowledge the send and received data, retransmitting of the loose packets, and it less delay in sending data packets. In the main Vi in figure 3.13 the UDP block for opening of the communication requires the local port where UDP socket is created. The send and received block requires both communication port and (Internet Protocol)IP address.

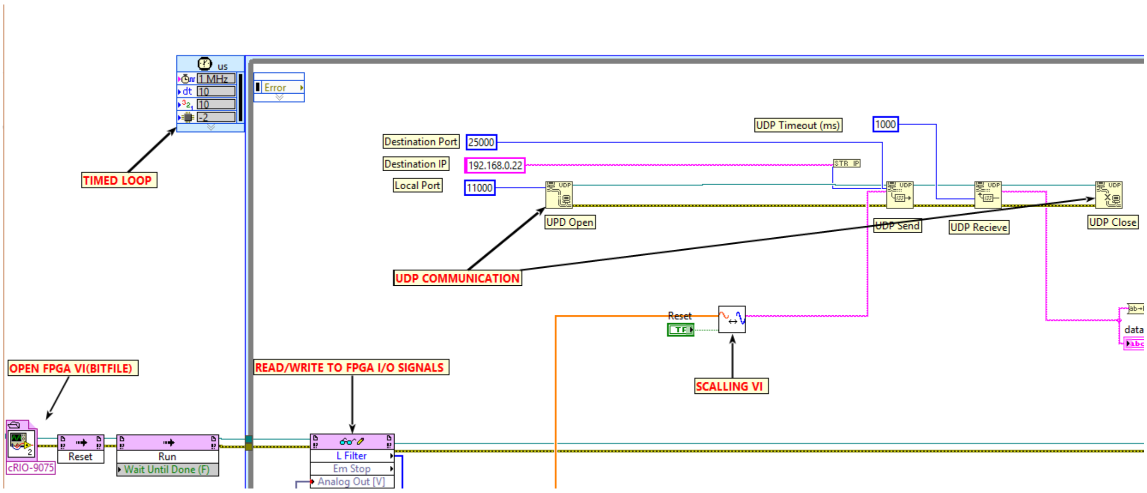
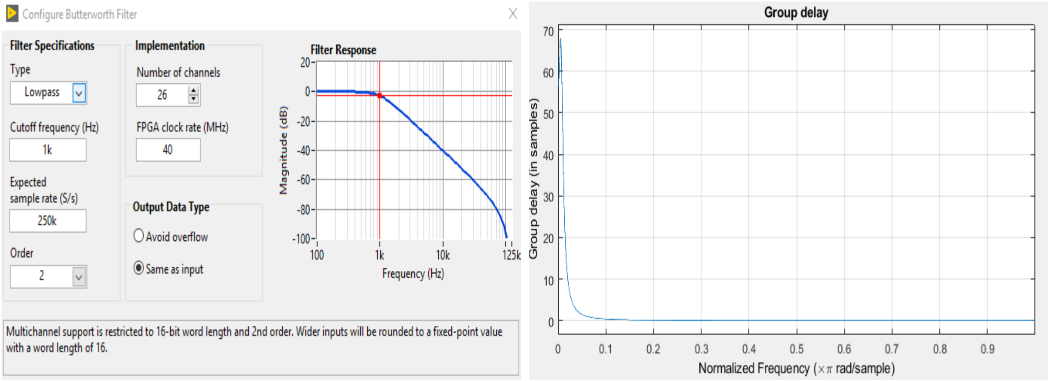


Figure 3.13: Main VI

The joinery of the data packet sent from cRio to a receiver contains an IP package what so-called datagram which holds information of the data and a header that shows the destination and the source address. The main functionality of the IP is to transport data from one source to another and UDP is sending the data packet that arrived in its destination port. The acquisition of the data from the sensor was made by programming the FPGA. The FPGA is located under cRIO chassis where a VI was created which is only dedicated for FPGA programming. The VI includes an I/O node which accesses the data from the sensor and a to low pass filter.



Configure Butterworth Filter

Filter Specifications

Type: Lowpass

Cutoff frequency (Hz): 1k

Expected sample rate (S/s): 250k

Order: 2

Implementation

Number of channels: 26

FPGA clock rate (MHz): 40

Output Data Type

○ Avoid overflow

● Same as input

Multichannel support is restricted to 16-bit word length and 2nd order. Wider inputs will be rounded to a fixed-point value with a word length of 16.

(a) Butterworth filter information

(b) Sample delay of the filter

Figure 3.14: Butterworth filter Second order

The filter type which was implemented is a Butterworth second-order lowpass filter with a cutoff frequency of $1kHz$. The normalized frequency of the filter is

$$\epsilon = \frac{2 \cdot F_c}{F_s} = 0.008 \tag{3.1}$$

The FPGA VI was called into the main VI by using the bitfiles generated from the compile. The compiling process of the FPGA consists of three systems.

- Development computer
- Compile serve
- Compile work

where the development computer holds the VI which has the programme, the compile server takes jobs from the computer to available compile work which consists of Xilinx compilation tools installed for FPGA which generates the bitfiles. When the compile work is finished it sends the bitfiles back to the computer.

The raw data from the FPGA Vi was scaled to engineering values. The scaling VI takes in raw voltage data and scaled them to typical engineering values.

$$Value = \frac{MaxEngineeringValue}{MaxmeasuredVoltage} \cdot measuredvoltage \tag{3.2}$$

3.5.4 VXelements

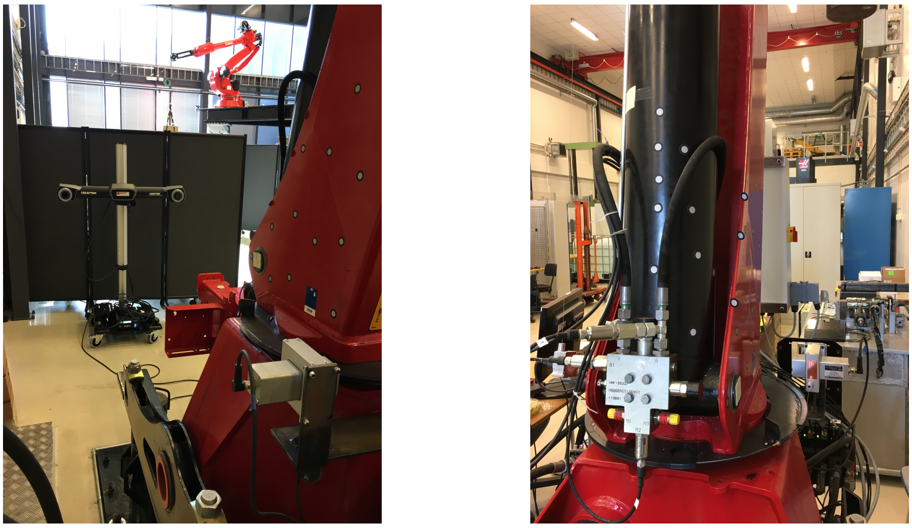
The configuration of the C-track and models set up for measuring the slew motion of the crane was done by using Vxelements. The Vxelements is a Creaform software which enables to interact with the C-track. Several steps had to be taken to measure the rotation of the crane, the following step had to be done

- calibration of C-track
- Model creation

The calibration of the C-track was essential to get accurate measurements of the rotation. The following steps and criteria have to be satisfied.

- The C-track has to worm up
- During calibration, the environment temperature should not varies ±2°
- During work session, the environment temperature should not varies ±10°
- C-track should not be facing intense light because it can cause noise or inaccuracy measurements.

After the calibration and satisfactory of the accuracy the model to be tracked was created. The model is a set of markers attached to the crane, for a model to be classified as a model; it must consist of a minimum of four makers mark as a group.

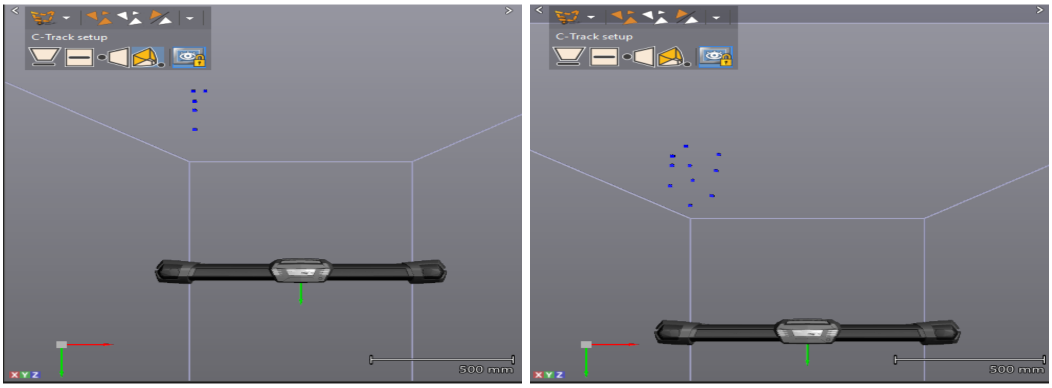


(a) C-Track and Crane with markers

(b) Markers first Model

Figure 3.15: C-Track and Markers

The markers were attached to the crane according to the positive rotation of the crane and the location of the C-track. The estimated tracking range due to the places where the markers were attached is 90 degrees. In Vxelement software, models had to be created where a set of markers more than four is selected as a group.



(a) Model1

(b) model2

Figure 3.16: Vxelemets Markers Selection

In figure 3.16a and 3.16b markers were selected and classified as a model. The first model had five markers and the second had ten. Furthermore, a reference frame was created on the middle point of the crane where the models were selected and at the C-track midpoint. By using this method, the rotation of the crane with respect to the C-track frame can be tracked.

C-track can only track models when it can see at least four markers which was a challenge due to the cylindrical shape of the crane base and the markers placed position. In this case, the C-track can only track one model at a time, to overcome this issue a sequence was created where a relationship between the model was added.

The models created had the range where the C-track can detect. The range is given in degrees where it corresponds to the rotation of y-axis.

C-Track models:

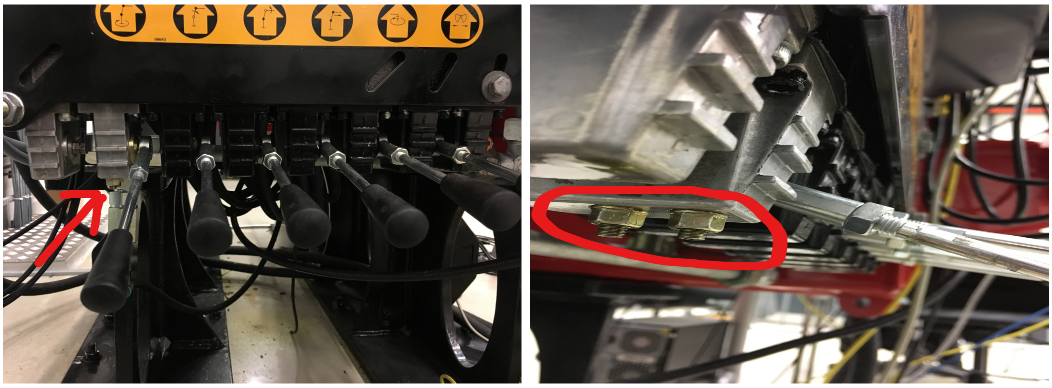
Model	Min	Max
Model 1	0	12°
Model 2	0	58°

Table 3.4: Models range

The total tracking range of the C-track is 70° degrees

3.5.5 Valve Scaling

The valve controlling the slew motion of the crane is manually restricted due to HMS safety in the lab. This was done manually by installing two bolts to limits the PVM and the spool to move ordinarily. The limitation changed the natural position of the valve as shown in figure 3.17a and cause the feedback voltage signal to differ from the manual data-sheet.



(a) Natural position

(b) Restriction

Figure 3.17: SAUER DANFOSS Valve

The restriction reduces the spool travel in the valve which causes the flow to be diminished in port A and B. The valve characteristic was used to determine the work area of the valve by recreating a new plot similar as the unrestricted plot in figure 3.7. The characteristic of the valve in figure 3.18 was found by examining the valve limitation. This was done by increasing the set-point voltage Us linearly in the range ±6V and sample the feedback signal from the spool position. The range of the voltage 6V is determined to be 75% U_{DC} maximum and 25% U_{DC} minimum of the supply voltage.

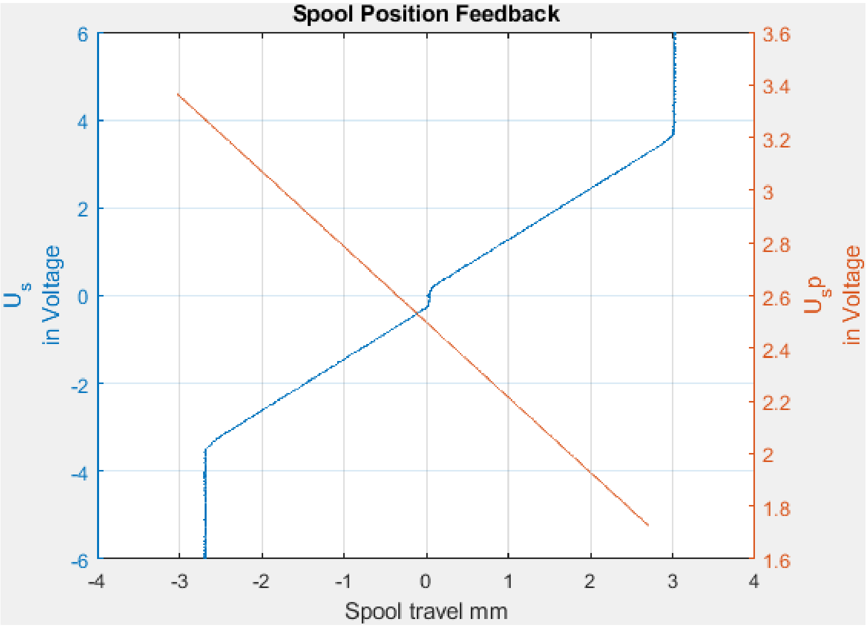


Figure 3.18: Valve characteristic form measured data

The response in figure 3.18 shows the valve has nonlinear at the start point which causes the open of the valve to be nonsymmetric. The asymmetric behavior causes the scaling of the valve in ±1 input value to be challenging; this is due to the end position of both port (A and B) are not equal.The maximum open area for the port (A and B) was determined by figure 3.18 and its corresponding input voltage.

Valve[1]:

Port	Input Value Voltage	Area mm	Output voltage
Port A	6.0[V]	3.0275[mm]	3.365[V]
Port B	-6.0[V]	-2.6880[mm]	1.7320[V]

Table 3.5: Behaviour of figure 3.18

Figure 3.18 confirm that the valve is indeed restricted; the input voltage was not causing any change in spool travel beyond the maximum value given in the table 3.5. The valve was linearized by determining the start and end of the deadzone value, the equation 2.2 to was used to compensate the deadzone.The nonlinearity value of the valve was determined by examining the start position of the spool travel of the valve in figure 3.18. The start value of the deadzone for port A was 0.15 V and port B was -0.24 V were used to linearize the valve.

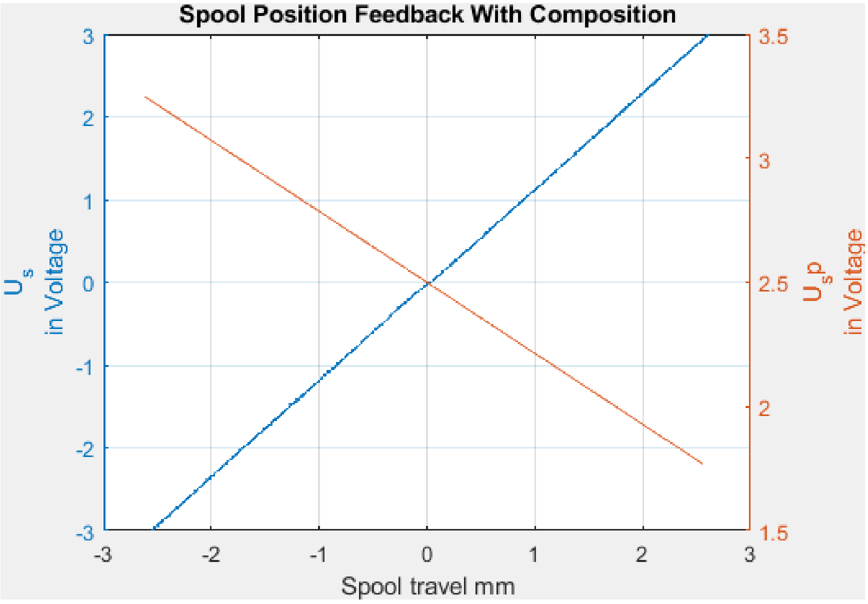


Figure 3.19: The valve characteristic linearized(measured data)

The linearization will ensure that the open valve area was equally distributed in port (A and B) corresponding to the input value. The result of the linearized valve in figure 3.19 guarantees symmetric in the open valve area in port A and B compare to the input signal.The new working area and the maximum input signal a represented in table 3.6

Valve[1]:

Port	**Input Value Voltage**	**Area mm**	**Output voltage**
Port A	3[V]	2.6[mm]	3.2480[V]
Port B	-3[V]	-2.6[mm]	1.7690[V]

Table 3.6: Behaviour of figure 3.19

The input voltage was determined to be ±3V to give a linear response of the valve. The scaling of input of ±1 was now possible which indicate the valve open in percentage in each direction.

3.6 Workstation-Real time interface

The workstation was set up in the Mechatronics laboratory at UiA for analyze and identify the crane dynamics. The crane operation range has been extended. The crane has three rotary joints, and one telescopic the first join is the slew of the crane, and the other moves the crane in XY. The telescope and winch are not included in the interface.
The hardware used in the setup is the crane.

- cRIO cabinet which includes all electronics devices explained in previous sections.
- C-Track with an additional computer for data exchange.
- HMF crane
- Real-Time Target machine(SpeedGoat) with an extra screen connected to the Target.
- A Host Window Computer with Matlab Simulink installed.

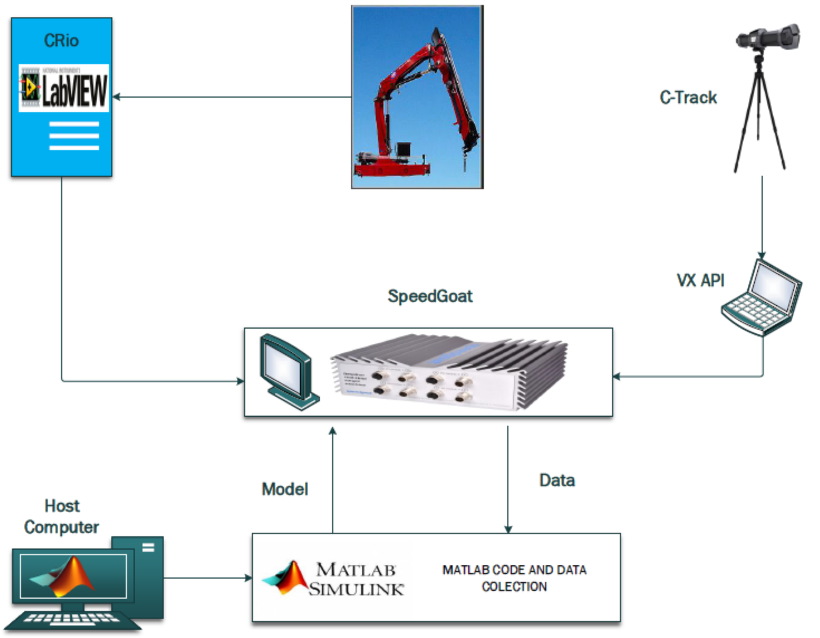


Figure 3.20: Setup Overview

The cRIO is programmed to send all activated sensor value to the real-time target machine(SpeedGoat) and receiving valve commands. The communication between SpeedGoat and cRIO is through an

Ethernet cable where UPD was utilized. The cRIO send the internal data of the crane and the external which is the slew of the crane is send by C-tack.
The C-track is connected to a computer which has Vxelement installed. Also, the Vxelemets has an extended add-on program Vx-Track which enable to configure model and logging of the data but not for data exchange interface. The data exchange between C-track and Target(Speedgoat) need an Application programming interface(API). The Vxelements API kit is a collection of .NET class which acts as a programming interface between Vxelements and the outside world. Furthermore, the API gives opportunities to control all Vxelements functionalities like retrieving data produced by Vxelemets modules like Vx-Track which produced the slew angular created in section 3.5.4. The VxAPI is a Visual Basic .NET which enables the user to utilize the UPD socket for communication with Target machine(Speedgoat).
The Speedgoat target machine is optimized and integrated with Simulink Real-Time and HDL coder. A Host computer running window application which has Simulink installed was used in the setup. The Speedgoat has a wide range of I/O which includes Ethernet.

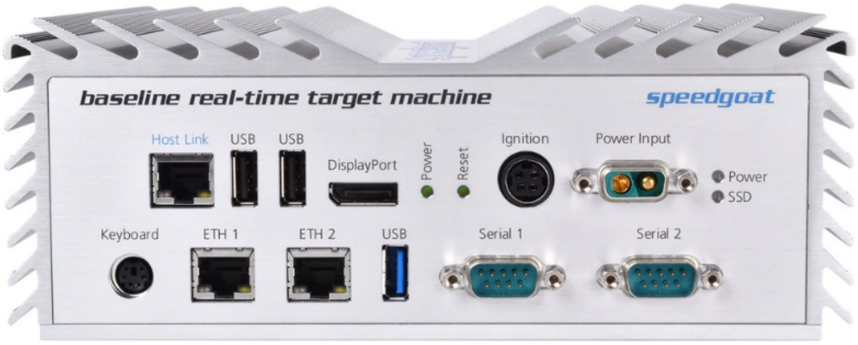


Figure 3.21: Speedgoat I/O modules

The Host computer was connected to host link, and the monitor screen was to display port. The cRIO was connected to ETH 1 and the computer with VxAPI to ETH 2.

Network configuration

The workstation is configured in a way that all connected hardware are in the same network.

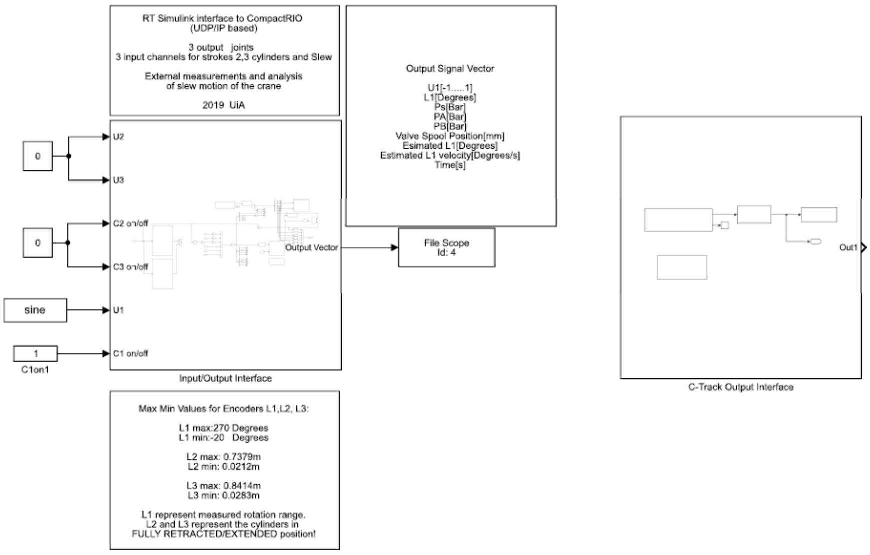
Network:

Hardware	**IP-address**	**Subnet mask**
Host Computer	192.168.7.1	255.255.255.0
SpeedGoat Host	192.168.7.2	255.255.255.0
SpeedGoat ETH1	192.168.0.22	255.255.255.0
SpeedGoat ETH2	192.168.0.23	255.255.255.0
cRIO	192.168.0.4	255.255.255.0
Vx Computer	xxx.xxx.xxx.xxx	255.255.255.0

Table 3.7: Network configuration

Matlab Simulink Real-Time Model

Matlab model was developed to interface with the crane and C-track. The model consists of two primary subsystem model for internal and external data of the crane.



RT Simulink interface to CompactRIO
(UDP/IP based)

3 output joints
3 input channels for strokes 2,3 cylinders and Slew

External measurements and analysis
of slew motion of the crane

2019 UiA

Output Signal Vector

U1[-1.....1]
L1[Degrees]
Ps[Bar]
PA[Bar]
PB[Bar]
Valve Spool Position[mm]
Esimated L1[Degrees]
Estimated L1 velocity[Degrees/s]
Time[s]

Max Min Values for Encoders L1,L2, L3:

L1 max:270 Degrees
L1 min:-20 Degrees

L2 max: 0.7379m
L2 min: 0.0212m

L3 max: 0.8414m
L3 min: 0.0283m

L1 represent measured rotation range.
L2 and L3 represent the cylinders in
FULLY RETRACTED/EXTENDED position!

Figure 3.22: Simulink Model

NB The communication between SpeedGoat and C-track failed due to license.B.1

The inputs to the cRIO interface model are three valves $U1, U2, U3$ with value range ±1 and its corresponding safety feature for each valve. The $U_{on/off}$ active and deactivates the valve ports where the user must give value.

- Valve ON: $C_{on/off} = 1$
- Valve OFF: $C_{on/off} = 0$

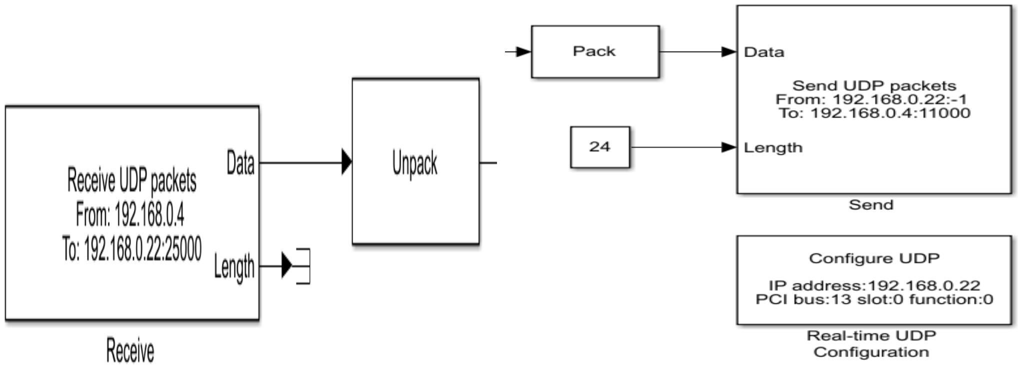
The output of the model is given as a vector with a total of nine available variables. The variables are sampled under experiment and saved in Matlab workspace. Furthermore, the sampling time of the system is $1kHz$.The selected measured parameters for this research are

Output vectot:

Name	**Min**	**Max**
Slew valve[U1]	-1	1
Slew	$-20°$	$270°$
HPU SP	0[Bar]	170[Bar]
PA	0[Bar]	400[Bar] sensor
PB	0[Bar]	400[Bar] sensor
Valve spool position[V]	-3[mm]	3[mm]
Slew Estimate	$-20°$	$270°$
Slew Velocity Estimate	$-2.5°/s$	2.5°/s
Time	Start[s]	End[s] sensor

Table 3.8: Measured output

The measured signals are received from a Matlab UDP receiver which gets a total of 7 IP packages from cRIO with a total length of 28 bytes. The received packages are unpacked from bytes to a signal vector. The Matlab sends data to cRIO through a Matlab UPD send block which requires the data to be packed before sending. The length of send data is six which corresponds to a total of 24 bytes.



(a) Matlab receive Block

(b) Matlab send and RT UDP Configuration Block

Figure 3.23: Matlab RT Communication Blocks

The real-time UPD configuration is done by activating Ethernet port 1 for communication with cRIO in figure 3.23b by assigning the IP address, subnet mask, and the ETH port.

HPU

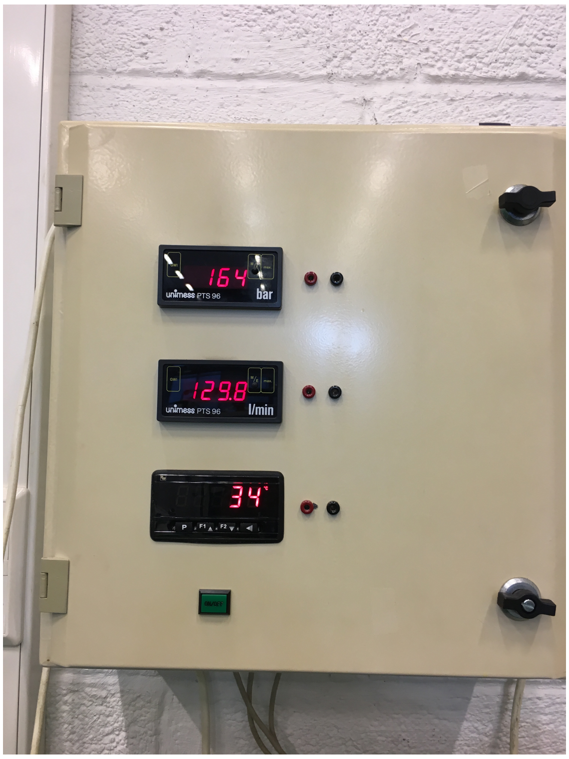


Figure 3.24: HPU setup

The hydraulic power unit has to be set as following 3.24 and the main valve to 55 $liter/min$ to have consecutive results dunging each experiment.

Chapter 4

Modelling

This chapter looks deeper into the steps utilized to solve the tasks in hand concerning the theories explained in section 2. The main task was divided into two chapters where the instrumentation and workspace interface was described genuinely in section 3. In this chapter, we will be reviewing two types of modeling the slew actuator of the crane. The first part of the variant is what so-called the Black-box where the hole slewing model was simplified and some reasonable assumptions were taken into account when identifying the simplified model. The second part was the full order of the model where the Gray-box principles was utilized.

4.1 Simplified Model

The first developed model was a simplified one where several assumptions were taken to reduce the order and complexity of the system. The simplification was on the hydraulic valve which was neglected. The part of the system that was taken into consideration was the mechanical part where the input of the system was the force and output is the slew motion.

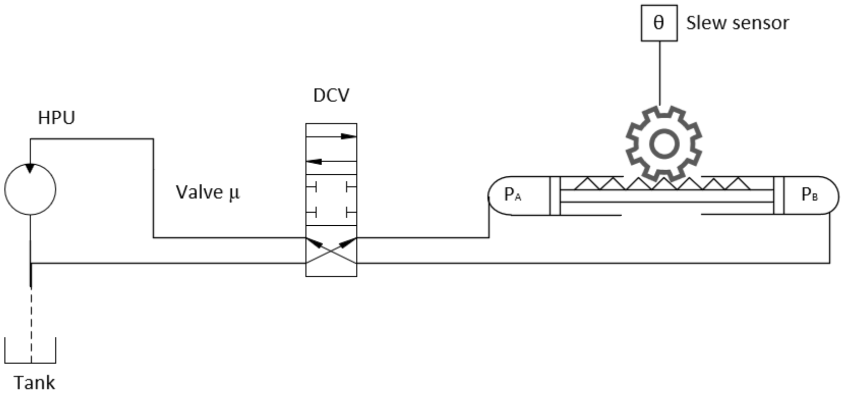


Figure 4.1: Slewing actuator block diagram

Figure 4.1 describes the functionality of the hydraulic and the mechanical part. The HPU provides the fluid to the hydraulic cylinder which is controlled by the valve. The valve parameter u controllers the direction of the rack. Furthermore, if $P_A > P_B$ the rack will move in the right direction and force F_A will rotate the crane in an anti-clockwise direction. If $P_A = P_B$ the will be no movement and if $P_B > P_A$ the rotation will be in a clockwise direction. The torque τ that rotates the pinion can be determined by the information of the cylinder, pinion, and pressure differential.

The equation of motion for the mechanical part of the crane can be described as

$$J\ddot{\theta} = \tau_{total} \tag{4.1}$$

The parameter J is the inertia of the system, and θ is the angular position of the crane. The τ_{total} is the sum of all torque acting on the system where it can express as

$$\tau_{total} = \tau_{hydraulic} - \tau_{friction} - \tau_{load} \tag{4.2}$$

The $\tau_{hydraulic}$ is the torque produced by the hydraulic cylinder which causes the motion on the crane, and it can be formed as

$$\tau_{hydraulic} = F_{hydraulic} \cdot r \tag{4.3}$$

where the force is

$$F_{hydraulic} = (P_A - P_B) \cdot A \tag{4.4}$$

The first assumption was to consider the $\tau_{fiction}$ is depended on the τ_{load} by doing so, the load on the system can be negligent.Therefore the friction torque can be formulated as

$$\tau_{friction} = b \cdot \dot{\theta} + \tau_{coulomb} \cdot sign(\dot{\theta}) \tag{4.5}$$

Equation 4.5 the parameter b is the viscous friction, and the $\tau_{coulomb}$ is the represent the Coulomb friction part. Furthermore, the $\tau_{coulomb} \cdot sign(\dot{\theta})$ can be considered as the nonlinear part of the system. The overall model can be expressed as

$$J\ddot{\theta} = \tau_{hydraulic} - \tau_{friction} \tag{4.6}$$

As explained early the input of the system is the valve input u which supplies the fluid into the cylinder and generates the torque. So it is reasonable to assume that the input torque $\tau_{hyraulic}$ is directly related to input u. Furthermore, the nonlinear part produced by the Coulomb friction can be compensated by determining the dead-zone of the system. Therefore if the dead-zone function is found and inverse function is applied into the model the estimated simplified model can be expressed as

$$\widehat{J} \cdot \ddot{\theta} = u_{valveopen} - \widehat{b} \cdot \dot{\theta} \tag{4.7}$$

The transfer function of the estimated system can be represented as

$$\widehat{G(s)} = \frac{\theta(s)}{U(s)} = \frac{1}{s(J \cdot s + b)} \tag{4.8}$$

4.1.1 Deadband Compensation

The deadzone of slewing motion was identified by providing fluid into the cylinder and measured the output response. The input signal which corresponds to valve opening was scaled into ±1 and was increased linear from 0 to 1 and 0 to -1. The output response was observed to be non-linear as explain in section 2.2 that the relationship between input and output is always not linear.

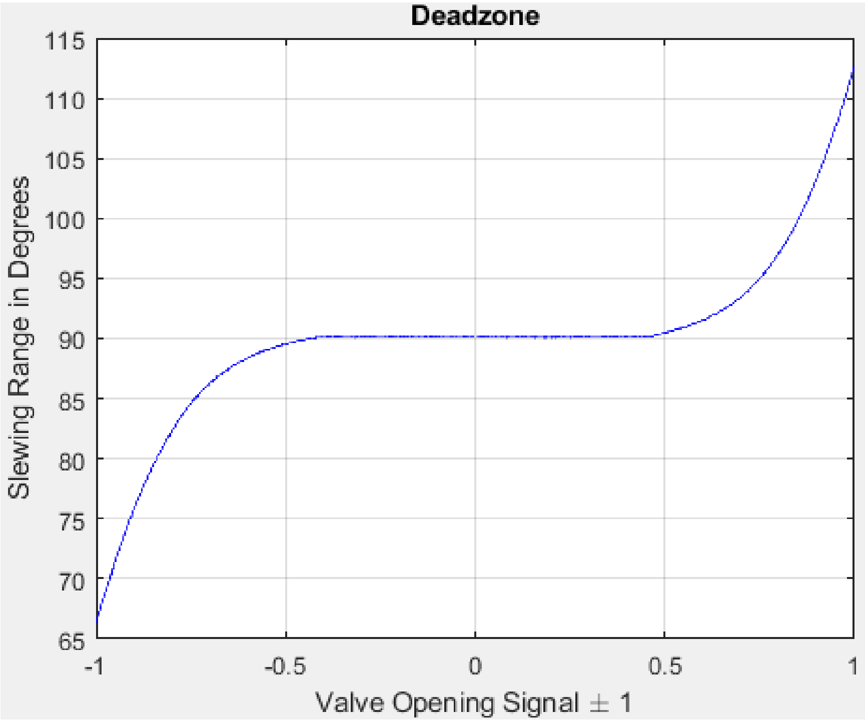


Figure 4.2: Measured data of Nonsymmetric deadzone nonlinearity

The result of deadzone in figure 4.2 indicates the function was nonlinear, the points of discontinuity were nonsymmetric and the response shows the behavior was sigmoidal and not a basic jump function. The function of the deadzone was assumed to be invertible where equation 2.5 was used to determine the inverse of the deadzone.

discontinuity parameters:

Actuator	d_+	d_-
Valve	45%	40%

Table 4.1: discontinuity parameters form figure 4.2

The discontinuity parameters were determined by observing figure4.2 where the input was starting affecting the system. The goal was to used the value in the equation 4.9 which was built upon equation 2.3. The parameter d_+ was the start of the deadzone in port A and d_- for the port B, the parameter U_{in} was the input value which corresponds to the valve opening signal, and S was a constant value which determines the sharpness of the input values to the system.

$$DZ^{-1}(u_{in}) = \begin{cases} \tanh(u_{in}S)d_+ + U_{in} & U_{in} > 0 \\ 0 & U_{in} = 0 \\ \tanh(u_{in}S)d_- + U_{in} & U_{in} < 0 \end{cases} \tag{4.9}$$

In order to make the system linear the inverse of the deadzone had to be determined. The implementation of the inverse function will ensure the cancelation of the deadzone and the system to be linear. The inverse function was created by equation4.9 where the goal here was to generate similar function as figure4.3.

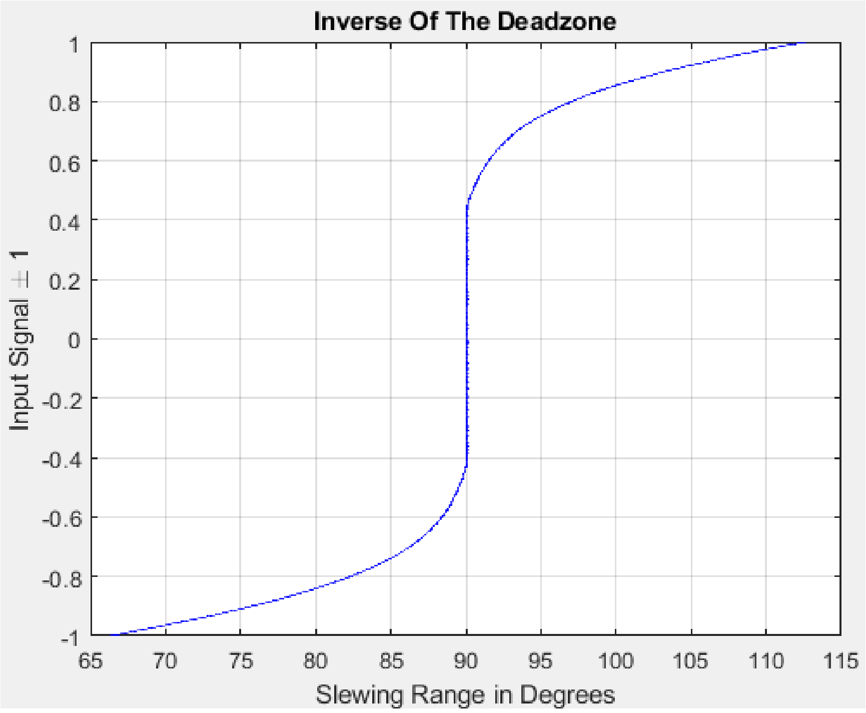
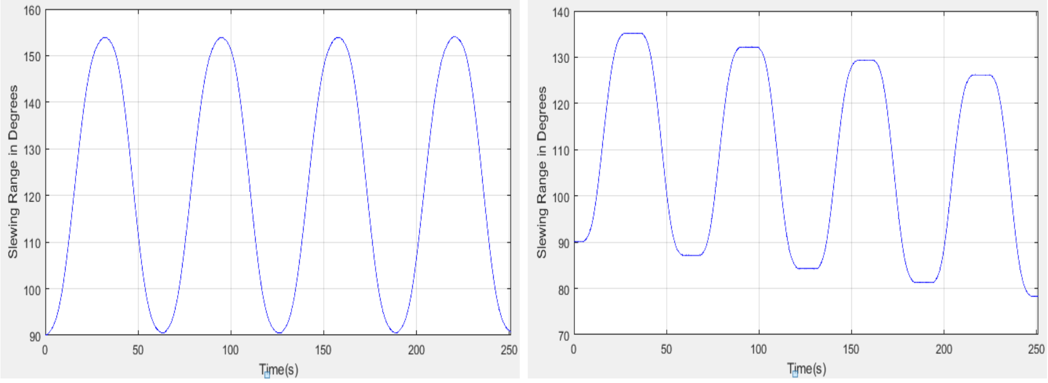


Figure 4.3: Measured data of Deadzone inverse

Figure4.3 represent the inverse deadzone of the system; if the equation 2.3 will generate similar input to the system then this will guarantee liner behavior in the output of the system.



(a) Undistorted Output Signal

(b) Nonlinearly distorted

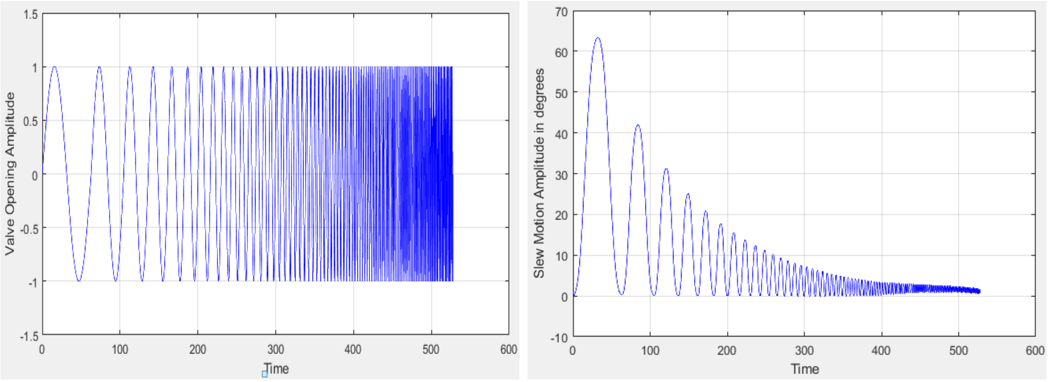
Figure 4.4: Compensated and Uncompensated Output Signal

4.1.2 System Identification

Identification of the system model is essential in analyzing system behaviors or control and observer design. The goal of identifying the system was to get a mathematical model which will be used in interpreting the slew motion of the crane and the designing of the estimator. The are three steps in developing a mathematical model of the dynamic system which has to be followed precisely to identifying a model of the system as described in section 2.3. The steps for identifying the system was chosen in favor of utilizing Matlab system identification toolbox.

The first step was to determine the type of data collection. In this case, the frequency response function(FRF) was chosen to represent the data because it has more information about the system. To collect FRF data, system excitation had to be performed with different kinds of signal types, frequency, and amplitude. The experiment was done under careful consideration of the system safety.

The FRF test was done at UiA mechatronic lab, where the procedure of FRF data collection was followed as explained in section 2.3. The mentioned method has certain drawbacks when it comes to measuring the amplitude and the phase of the FRF. Those disadvantages are mainly noise or distortion of the output signal which is used to calculate the FRF data; those issues such as drifting of the output signal and non-linearity of the crane were neutralized in section 4.1.1 . The FRF was performed by using the internal data of the crane, the slew motion of the crane signal comes from an encoder where the signal was captured by cRio and sent to Speedgoat. The FRF requires two signal, input and output signal. The input signal was the amplitude of the valve opening ±1, and the output signal was the response of the slew motion of the crane in degrees. The FRF was conducted on the system with the frequency range of [0.1...5] in rad/s, and the step size between each test iteration was chosen to be 0.05 rad/s. The phenomena of a frequency sweep was used where a Matlab code was made to generate a sine wave in the given frequency range. The input of the system was the generate sine/square wave which is the valve opening of the system, and the output response was the slew motion were both are presented in figure 4.5



(a) Valve Input Signal

(b) Output Signal

Figure 4.5: Input and Output signal

The FRF contains tree data types which are the magnitude, phase, and frequency. The calculation of the magnitude and phase were done by a Matlab code, where the data were logged for each frequency. The calculation of magnitude in dB can be obtained by equation 4.10

$$M = 20log(A) \tag{4.10}$$

The measured Amplitude was A for each frequency of the FRA. The test was performed when the initial position of the slew angle of the crane was zero. This will ensure the right magnitude of the system at the starting frequency.The phase lag of the system between the input and the output signal was calculated by using equation 4.11

$$\phi = -\frac{360^\circ}{T_p} \cdot \delta t \tag{4.11}$$

A Matlab code was created where both equation 4.10 and 4.11 was implemented. The code was able to determine the amplitude A and able calculate efficiently the δt which is the time delay between the input and output signal and the period of the input signal is T_p. Figure 4.6 represents the first frequency and how the parameters were determined.

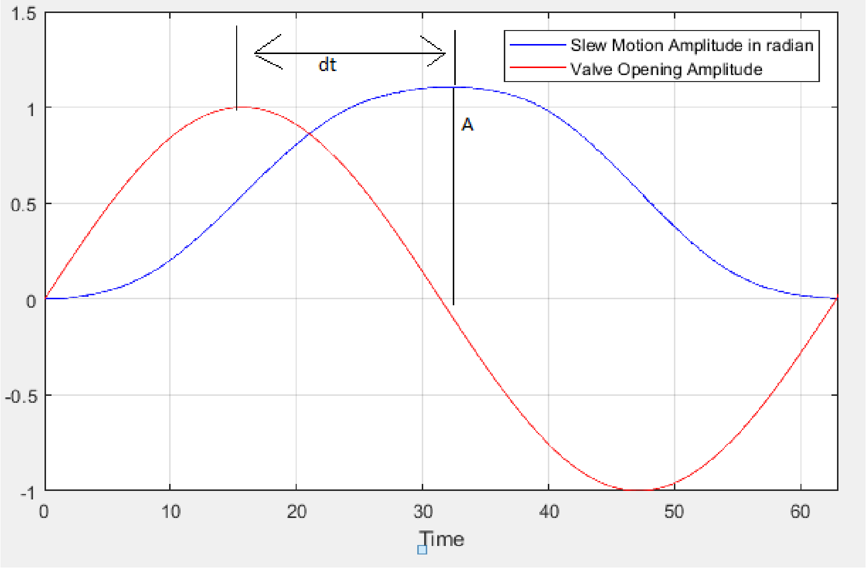
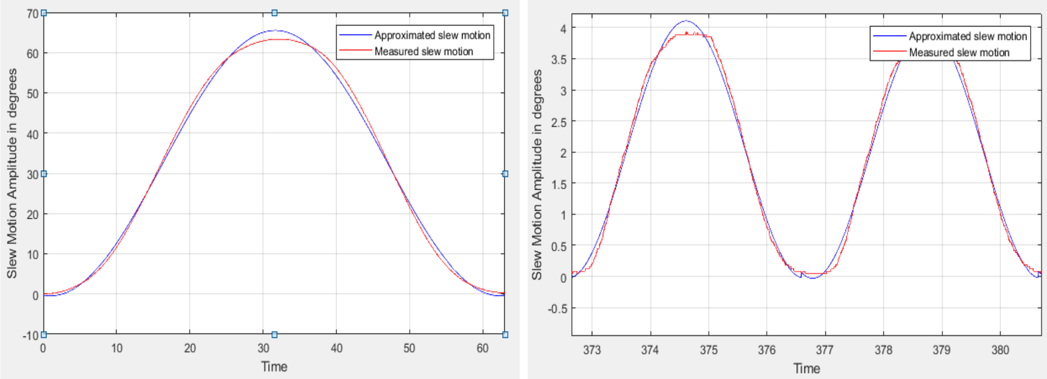


Figure 4.6: Frequency response test parameters calculation

The FRF parameters where calculated for every test frequency, in total were they 100 samples of data. The calculation of the phase requires data that was not corrupted with noise, and this is due to the high sensitivity of the calculation of the phase. To overcome the drawback of the noise an estimation algorithm of the measured slew motion response has been implemented. The output of the system can be approximated by the equation 4.12

$$\widehat{y} = \frac{y_{max} + y_{min}}{2} + \frac{y_{max} - y_{min}}{2}\sin(\omega t + \phi) \tag{4.12}$$

The approximated equation 4.12 contains four parameters, y_{max} and y_{min} is the maximum and minimum amplitude of the slew motion. The parameter ω is the frequency, and ϕ is the phase lag of the slew motion. Those parameters are unknown and had to be identified for each test frequency such that the approximated equation will fit the real output response. An optimization code in Matlab was implemented to find those four parameters where the error between the measured slew motion y and the approximated $\widehat{y}$ was minimized.



(a) Low frequency

(b) High frequency

Figure 4.7: Low frequency and High frequency

The error was minimized until the approximated, and the measured signals were almost identical. The Matlab code utilizes the optimization toolbox where the Matlab command $Isqcurvefit$ was used where it solve nonlinear curve fitting form given data. Figure 4.7 displays the measure, and approximate slew motion of the crane where the *blue* was the approximate and *red* was measured data. The obtained data from the equation4.12 are collected in a matrix B for each iteration of the FRF.

$$\mathbf{B} = \begin{bmatrix} y_{max} & y_{min} & \omega & \phi \\ . & . & . & . \\ . & . & . & . \\ y_{max} & y_{min} & \omega & \phi \end{bmatrix} \tag{4.13}$$

The data in the matrix B was used to generate the approximate slew motion signal for each frequency test. The parameters from the matrix B were applied into the equation 4.12 and used to find the FRF data. The obtained data from FRF were stored in three separate vectors equation 4.14in the frequency range [0.1....5].

$$\vec{\mathbf{A}} = \begin{bmatrix} A \\ .. \\ A \end{bmatrix}, \vec{\phi} = \begin{bmatrix} \phi \\ .. \\ \phi \end{bmatrix} and, \vec{\omega} = \begin{bmatrix} 0.1 \\ .. \\ 5 \end{bmatrix} \tag{4.14}$$

The vector A was the maximum value Amplitude of the slew motion in every tested frequency, ϕ is the phase in degrees and ω is the frequency in rad/s.The result of the FRF in equation 4.14 a plotted in figure 4.8, and the magnitude of the bode plot was given in dB.

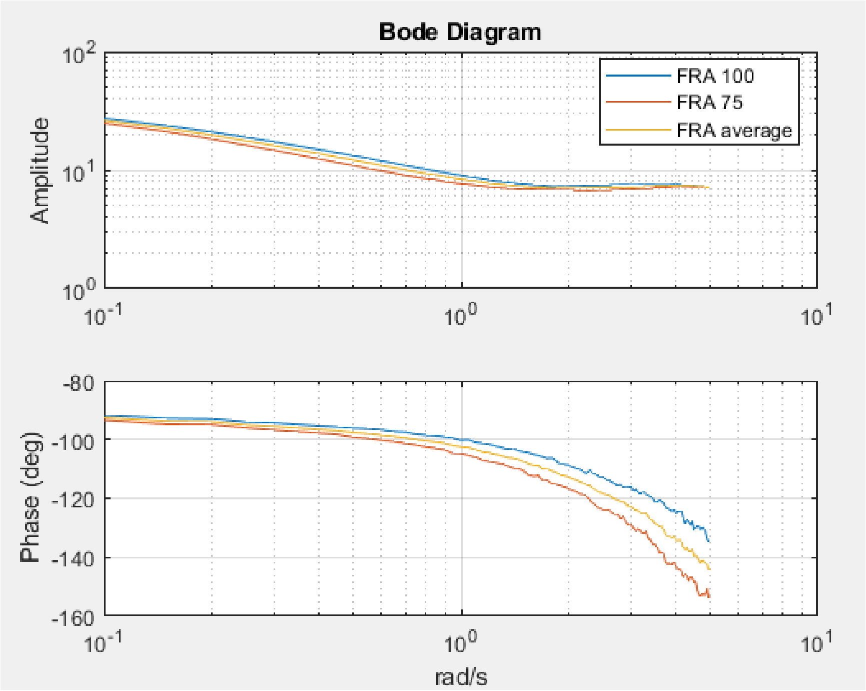


Figure 4.8: Frequency response diagram

The estimation of the mathematical model of the crane system was done by importing FRF data into the MATLAB graphical user interface (GUI) for system identification toolbox. The GUI was used to import the data in the frequency domain and the estimation of the model. They a different variant of the model that can be estimated in the GUI, the transfer function model was preferred to represent the system. They were several steps taken to utilize the estimation prosses in the GUI. The summary of the GUI and how it was implemented is shown below.

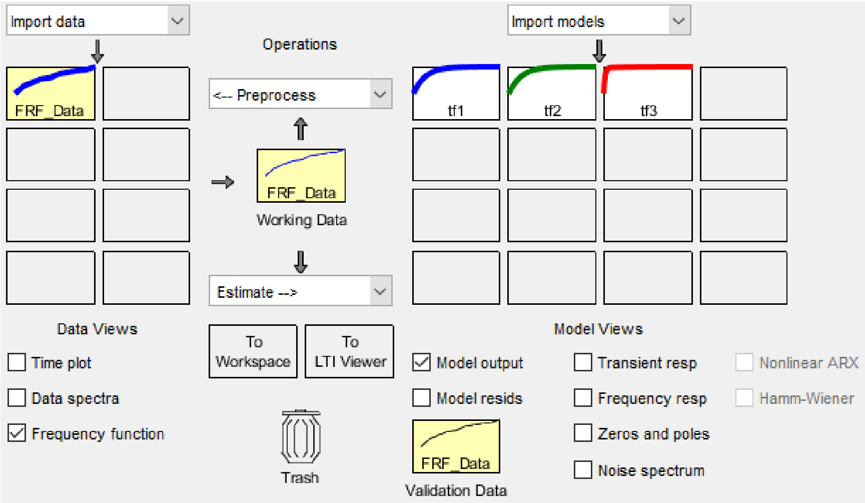


Figure 4.9: System Identification Toolbox

- The data type was selected as a frequency domain data, where the three FRF parameters were imported. The workspace variable was the three vectors in equation 4.14 which were Amplitude, ϕ , and ω.

- The imported data was given the name FRF_Data where it contains information of the frequency type and the sample rate of the data.

- The estimation of the transfer function model was done when the FRF_Data data was in the working data in the GUI. Three types of transfer function models were estimated with different orders, the estimation was done by selecting numbers of poles and zero. The results of each estimation were validated by comparing with the import data.

Name	Pole	Zero	Best Fits
TF1	2	0	98.16
TF2	3	0	68.90
TF3	3	1	99.1

Table 4.2: Transfer function model4.9

The candidate model from table 4.2 was selected based on how good the model fit the experimental data, and the overall behavior response of the model compare to the real system. The transfer function model $Tf3$ was selected since it fulfills all the criteria.

$$G(s) = \frac{\phi(s)}{V_{open}(s)} = \frac{10}{s(s+4.4)} \tag{4.15}$$

The transfer function obtained in equation 4.15 represent the crane model of the system.V_{open} is the *valveopening* which is the input of the system, and θ is the output of the system in degrees.

4.2 Full order model

The second model developed model will consider the dynamics part of the direction control valve, and mechanical component. The full order model consists of several subsystems form the valve input signal to the measured slew motion of the crane.

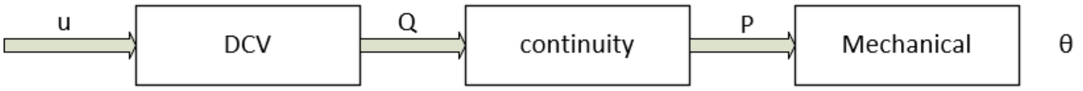


Figure 4.10: Full order model of the system

The whole system consists of three subsystems. The first is the directional control valve, the relationship between flow and pressure, and the mechanical system. The direction control valve has a control system as shown in figure 3.6a where the embedded electronics in valve control the spool position. The characteristic of a closed loop valve has a second-order transfer function [3] which can be expressed as

$$G(s) = \frac{V(s)}{u(s)} = \frac{\omega_0^2}{s^2 + 2\zeta\omega_0 s + \omega_0^2} \tag{4.16}$$

Where the input signal of the DCV is u and the output is the position of the spool. The valve is made with a closed center spool which means that the valve is closed when the spool is at the central position as shown in figure 3.7. The spool that is controlling the flow characteristic is subjected to nonlinearity which usually occurs in the central area. This leads to no flow at a low magnitude input value where the orifice remains closed in both directions. Therefore the identification of nonlinearity dead-zone will be base on the equation 4.9, and the saturation of spool displacement was taken into account. The dead zone of DCV was identified by measuring the spool position corresponding to the input value which was scaled into ±1 and was linearly increased in both opening directions.

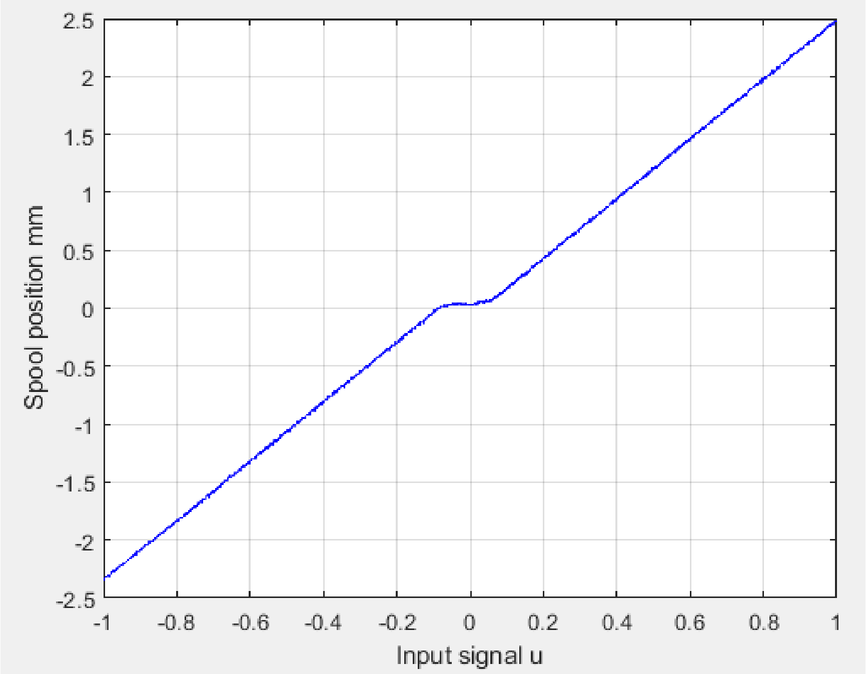


Figure 4.11: DCV dead-Zone from [measured data]

The points of discontinuity were found based on figure 4.11 which also corresponds to the input voltage found in section 4.1.1. The maximum value of the spool position was chosen to be 2.35mm to ensure the same opening in both directions.

The second subsystem model is the relationship between flow and pressure, which is governed by the orifice equations.

$$Q_A(u) = \begin{cases} C_d A_{DCV}\sqrt{\frac{2}{\rho}(P_s - P_A)} & u > 0 \\ C_d A_{DCV}\sqrt{\frac{2}{\rho}(P_A - P_T)} & u < 0 \\ 0 & s = 0 \end{cases} \tag{4.17}$$

$$Q_B(u) = \begin{cases} -C_d A_{DCV}\sqrt{\frac{2}{\rho}(P_B - P_T)} & u > 0 \\ -C_d A_{DCV}\sqrt{\frac{2}{\rho}(P_s - P_B)} & u < 0 \\ 0 & s = 0 \end{cases} \tag{4.18}$$

Q_A is the flow rate from port A to the cylinder side A, and Q_B is the flow rate from port B into side B of the cylinder as illustrated in figure4.10. The parameter C_d is the discharge coefficient,A is the area, and ρ is the density of the oil where u determine the flow direction. The calculation of pressure drop is determined as the absolute value in the orifice equations. The pressure differential equations inside cylinder chamber A, and B are expressed as

$$\dot{P_A} = \frac{\beta}{V_L + V_A} \cdot (Q_A - A_c \cdot \dot{x} - L_A) \tag{4.19}$$

$$\dot{P_B} = \frac{\beta}{V_L + V_B} \cdot (Q_B - A_c \cdot \dot{x} - L_B) \tag{4.20}$$

Where parameter β is the bulk model which is a constant value. A_c is bore areas of both side since the cylinder is a double bore configure as shown in figure 4.10. The total volume is the sum of the hydraulic line V_L , and the volume inside cylinder chamber V_A, and V_B.The parameters L_A, and L_B are leakage coefficients in the cylinder chambers which also add pressure drop into the system. The pressure difference in the system can be expressed as

$$P_L = P_A - P_B \tag{4.21}$$

Adding equation4.17 ,4.18 together can form one orifice equation which describes the average load flow

$$Q_L = -C_d A_{DCV} \sqrt{\frac{2}{\rho}(P_s - P_L)} \tag{4.22}$$

Equation 4.22 is nonlinear due to the square root in the equation. The linearization of the orifice equation can be done by introducing two gains coefficient, flow, and pressure gains.

$$\widehat{Q_L} = K_{flow} - K_{Pressure} \cdot P_L \tag{4.23}$$

The equation 4.23 can be further simplified by introducing gain K which is time variant depended on load pressure on the equation 4.22.

$$\widehat{Q_L} = K \cdot P_L \tag{4.24}$$

The load pressure differential equation can be otain by combining equation 4.19, 4.20,and 4.22

$$\dot{P_L} = \frac{\beta}{V_{Total}} \cdot (Q_L - A_{average} \cdot \dot{x} - L_C \cdot P_L) \tag{4.25}$$

The transfer function of the relationship between the flow and pressure can be represented as

$$G(s) = \frac{P_L(s)}{Q_L(s)} = \frac{1}{\frac{V_t}{\beta} \cdot s + L_c} \tag{4.26}$$

The last subsystem which is the mechanical part of the cylinder motion which is later transformed to slew motion of the crane is expressed similarly as equation 4.6.

$$J\ddot{\theta} = P_L \cdot A \cdot r - \tau_{friction} \tag{4.27}$$

and transfer function of the system can be represented as

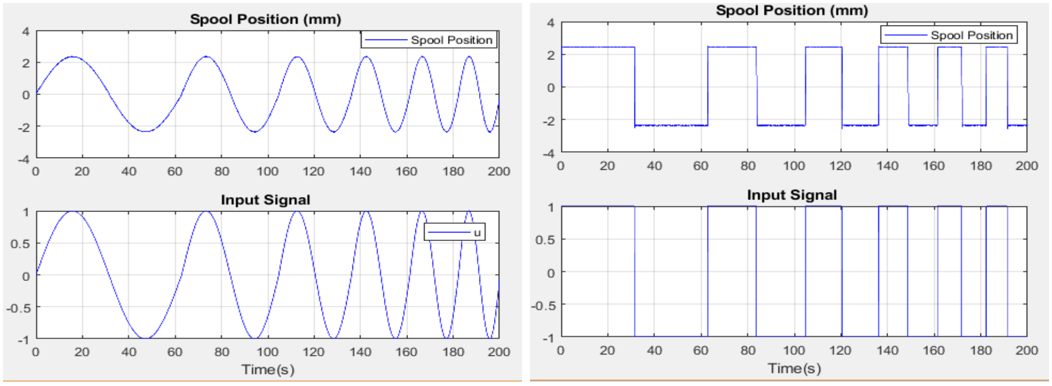
$$G(s) = \frac{\theta(s)}{T_h(s)} = \frac{1}{s(J \cdot s + b)} \tag{4.28}$$

The input torque is the measured pressure difference in the cylinder multiply by transmission constant of the pinion , and the cylinder bore area. The nonlinearity is compensated similarly as the simplified model.

4.2.1 System Identification

The identification of the full order model was carried out similarly as in section 4.1 where several steps had to be followed to identify each of those three models. The FRF was carried out for all three models with different amplitudes, and signals types. The involvement of various kinds of harmonic excitation was to get a more reliable system in total. Matlab system identification toolbox was used for identifying the parameters of all tree model presented in section 4.2. The FRF parameters calculation was limited to simple harmonic oscillation signals like a sine wave, where complex harmonic excitation signals like triangular pulse and square pulse were imported to Matlab as time-domain data for identification of the model.

The input signal for the identification of DCV was the valve opening with the maximum amplitude of 1± with a different variant of frequency, and type of oscillation.



(a) Sine wave sweep

(b) Square wave sweep

Figure 4.12: Input and output measured signal

The output response of the system was spool displacement in mm. The FRA was carried out on the system with the frequency range of [0.1....70] with a step size of 0.05 rad/s. The signal processing of the measuring data is the same as in section 4.1.2 where a Matlab code for a fit curve was used. The results of FRF with a different type of amplitude in percentage are given in figure 4.13.

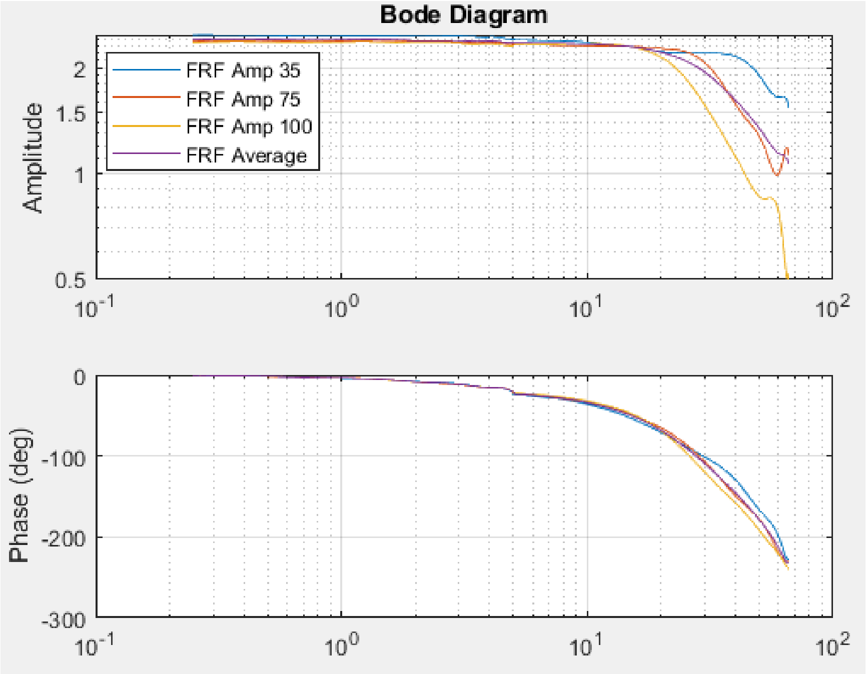


Figure 4.13: Frequency response diagram [DCV]

Three types of the data with different amplitude from 100% to 35% of valve opening signal was used, and all three tests were combined to form one average model of the DCV. The FRF data were imported to Matlab system identification toolbox for estimation of the modal parameters based on the DCV characteristic in equation 4.16.

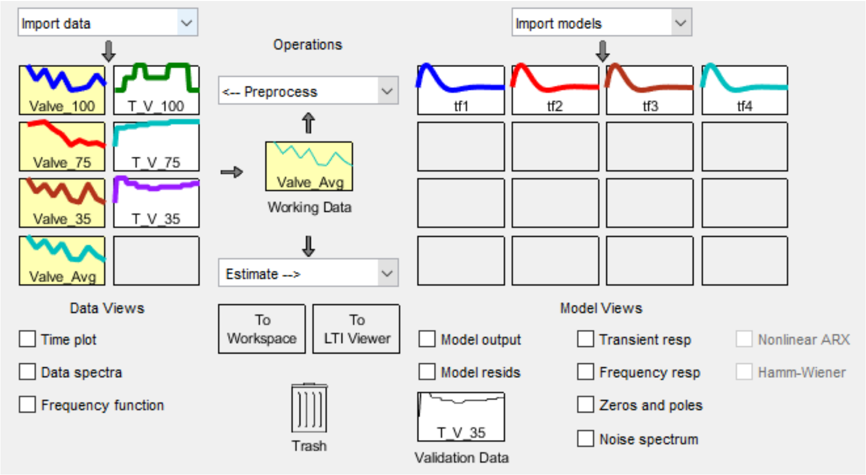


Figure 4.14: System Identification Toolbox [DCV]

Four transfer functions were estimated based on the imported data, where each of those three functions corresponds to the test data, and the four function contains all the test FRF combined. The equations that determine the parameters are

$$G(s) = \frac{V(s)}{u(s)} = \frac{K_1}{s^2 + K_2 2\zeta\omega_0 s + \omega_0^2} \tag{4.29}$$

$$K = \frac{K_1}{\omega^2} \tag{4.30}$$

$$\zeta = \frac{K_2}{2 \cdot \omega} \tag{4.31}$$

Name	K	ω	ζ	**Best Fits**
TF1	2.3048	23.19	0.55	80.37
TF2	2.3432	25.60	0.52	82.11
TF3	2.4678	27.64	0.55	76.91
TF4	2.3747	25.23	0.54	85.35

Table 4.3: DCV Transfer function parameters4.9

The identification of the subsystem which consists of orifice and the continuity equation are identified by Matlab toolbox. The identification of the type of model was influenced by equation 4.22, and 4.25.

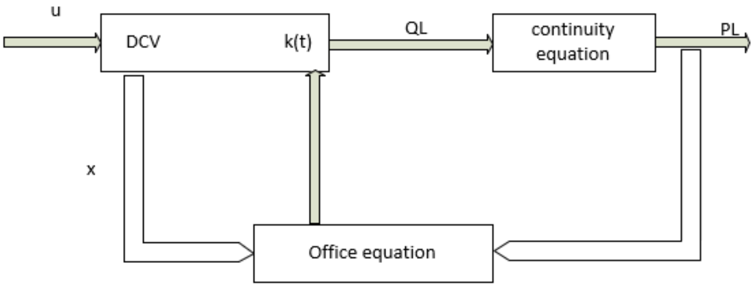


Figure 4.15: Flow and continuity subsystem

The input of the estimated model is Q_L, and the output is the load pressure. The more accurate way to calculate the load flow is equation 4.22, where the inputs were spool position, load pressure, and the supplying pressure P_S. The different method of determining the flow is to multiplying with the factor K which varies with time which depends on the flow characteristic. Due to the limitation on the sensor measuring the supplying pressure P_S on the cylinder a simplification was done to determine the load flow by utilizing only the spool position flow characteristic.

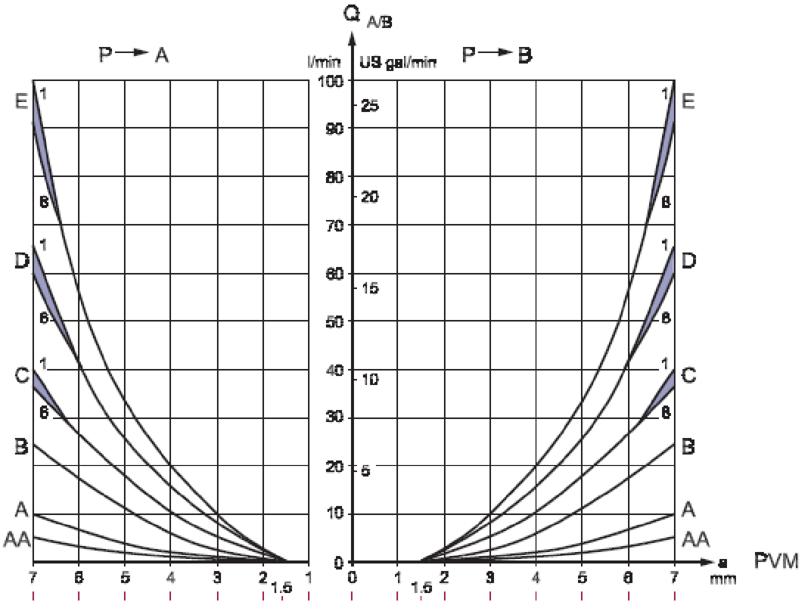


Figure 4.16: Spool position flow characteristic

The maximum flow can be calculated

$$Q_{max} = V\frac{m}{s} \cdot A[mm^2] \cdot \frac{60}{10^3} \tag{4.32}$$

where the maximum rack velocity is found to be $V_{max} = 5.7mm/s$, and maximum flow $Q_{max} = 2.84L/min$, so the assumption is the spool position is proposinal to the flow for small spool position. The measured data form FRF tests were imported into Matlab in the time domain where the input data was spool position, and output was the pressure difference.

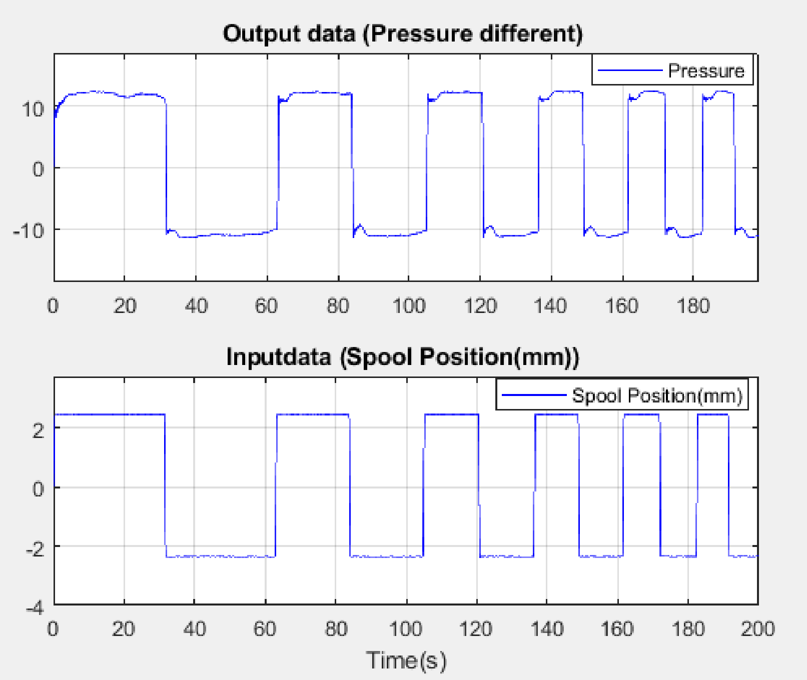


Figure 4.17: Input and output measurement data of pressure

The pressure value was noise and was filted with low pass filter. The filter introduced delay on the signal and was compensated by passing the input with the same filter. Four types of data test from experimental measurements were used to identify the system where the order of estimation is as equation 4.26

Name	K	**T**	**Best Fits**
TF1	4.638	0.07	80.6
TF2	5.709	0.10	83.8
TF3	5.709	0.18	78.9
TF4	5.28	0.11	85.5

Table 4.4: Estimated continuity-Subsystem Transfer function parameters4.9

$$G(s) = \frac{P_L(s)}{Q_L(s)} = \frac{K}{T \cdot s + 1} \tag{4.33}$$

The sub-mechanical system was identified similarly as the other models. The input of the system is torque which is calculated by multiplying the pressure difference with the cross-section of the mechanical model.

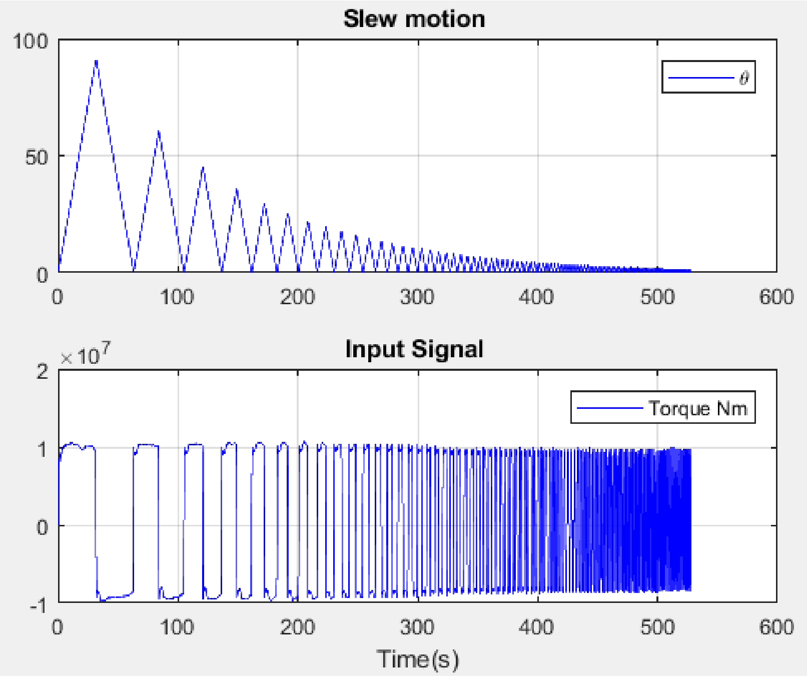


Figure 4.18: Input and output measurement of torque and rotation

The output is the rotation of the crane in *degrees*, the frequency range [0.1....5] with a step size

$0.05 rad/s$.The obtained transfer function from estimation

$$G(s) = \frac{\theta(s)}{T_h(s)} = \frac{7.5e-6}{s(s+39.4)} \tag{4.34}$$

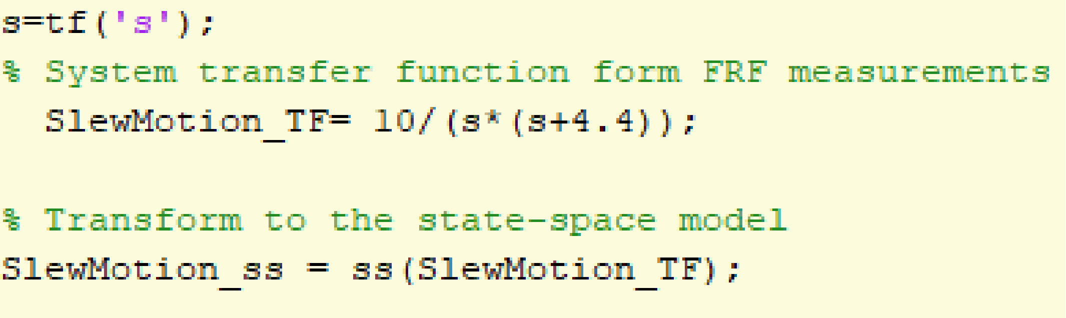
4.3 State-Space Design

The obtained transfer function in equation 4.15 was converted into first-order differential equations which described the dynamics of the cylinders. A Matlab code was created to transform the acquired transfer function to a state-space model, where the code will then generate the state of the cylinders in the form:

$$\dot{\mathbf{x}} = A\mathbf{x} + \mathbf{B}u \tag{4.35}$$

$$y = \mathbf{C}\mathbf{x} \tag{4.36}$$

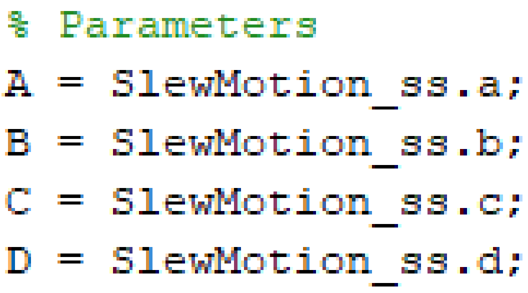
The script used to accomplish this was:



```
s=tf('s');
% System transfer function form FRF measurements
  SlewMotion_TF= 10/(s*(s+4.4));

% Transform to the state-space model
SlewMotion_ss = ss(SlewMotion_TF);
```

Figure 4.19: Matlab script from TF to SS

The first line in the code represents the transfer function of the slew motion, and the last line was the model transformed to the state space model.



```
% Parameters
A = SlewMotion_ss.a;
B = SlewMotion_ss.b;
C = SlewMotion_ss.c;
D = SlewMotion_ss.d;
```

Figure 4.20: Matlab script Matrix A B and C

The plant in state-space is obtained by taking matrix $\boldsymbol{A}$, $\boldsymbol{B}$, and $\boldsymbol{C}$ and converting them to the form of equation 2.12 and 2.13 as shown in the following equations:

$$\dot{\mathbf{x}} = \begin{bmatrix} -4.4 & 0 \\ 1.0000 & 0 \end{bmatrix} \mathbf{x} + \begin{bmatrix} 4 \\ 0 \end{bmatrix} u \tag{4.37}$$

$$\mathbf{y} = \begin{bmatrix} 0 \\ 2.50 \end{bmatrix} \mathbf{x} \tag{4.38}$$

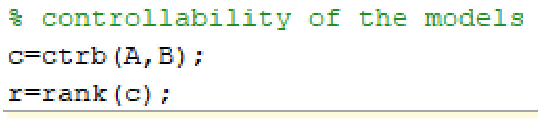
The model of the system is written in control canonical form as shown in the equations above. There are also different ways to represent the transfer functions such as partial-fraction expansion which will result in the state-space matrices to be in modal canonical form. The poles of the model will then appear in the diagonal element of the matrix A.

4.3.1 Controllability and observability

It is essential to analyze the obtained model if the model is controllable before introducing the feedback. If we take a look at the input matrix $\boldsymbol{B}$ for both equations they have only one input in the state $\boldsymbol{x_1}$ and the other state input is zero. In the system matrix $\boldsymbol{A}$ in equation 4.37 and, the states of the first row of both $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$ are connected. The contribution of the input in the first state will make the another state variable to be controlled and the whole system to be controllable. The controllability of the models depends only on the $\boldsymbol{A}$ and $\boldsymbol{B}$ matrices as formulated in equation 4.37. Another way to check the controllability of the system is to use a mathematical formula as shown in equation 4.39.

$$\mathbf{y} = \begin{bmatrix} B & AB... & A^{n-1}B \end{bmatrix} \tag{4.39}$$

If the results of the matrix $\boldsymbol{C}$ is non-singular, where it has full column rank and a linearly dependent column, the corresponding $\boldsymbol{A}$ and $\boldsymbol{B}$ matrices are said to be controllable. The calculation of the controllability of the $\boldsymbol{C}$ matrix is done in the Matlab script shown below in figure 4.21.



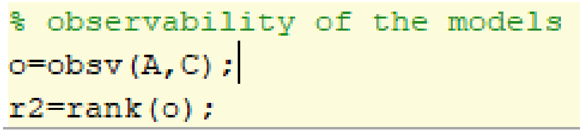
```
% controllability of the models
c=ctrb(A,B);
r=rank(c);
```

Figure 4.21: Matlab script from controllability

The use of the rank command in Matlab will give out an estimation of the number of linearly independant rows or columns of a matrix. If, and only if the rank is of magnitude n, then the system is controllable. Observability is in many ways the same as controllability. It has the ability to gather information of all state variables of the system when only one measured variable of the system is available. It is necessary to check if the system is observable as some systems can be physically disconnected from the output, so it is crucial to notice this before designing the observer. The observability matrix $\boldsymbol{O}$ can be introduced, written as in equation 4.40.

$$\mathbf{O} = \begin{bmatrix} C \\ CA \\ . \\ . \\ CA^{n-1} \end{bmatrix} \tag{4.40}$$

This matrix must have independent columns.

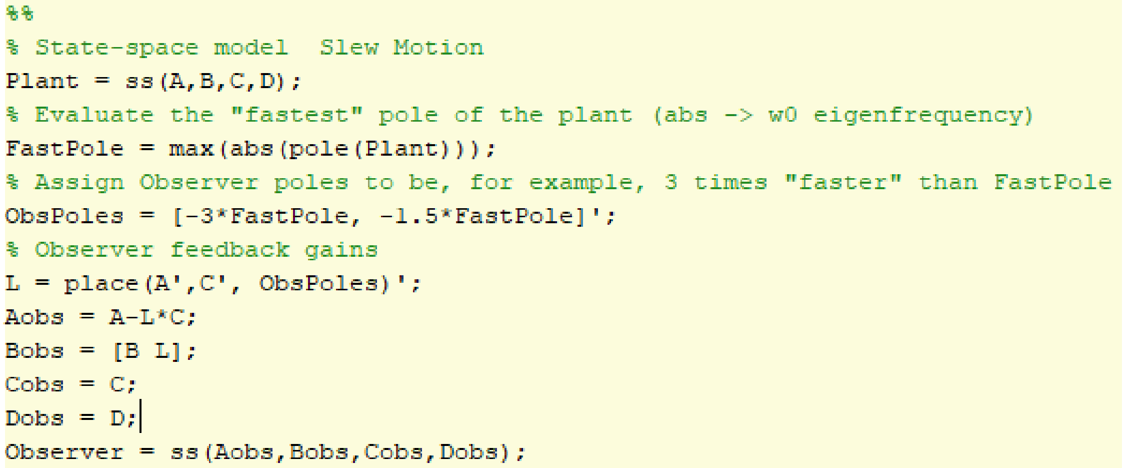


```
% observability of the models
o=obsv(A,C);
r2=rank(o);
```

Figure 4.22: Matlab script from observability

4.3.2 Estimator Design

The obtained state space in the section 4.3 is used to design the estimator. The first step is to find the gains of the estimator. A Matlab script was developed to address this issue as shown in figure 4.23.



```
%%
% State-space model  Slew Motion
Plant = ss(A,B,C,D);
% Evaluate the "fastest" pole of the plant (abs -> w0 eigenfrequency)
FastPole = max(abs(pole(Plant)));
% Assign Observer poles to be, for example, 3 times "faster" than FastPole
ObsPoles = [-3*FastPole, -1.5*FastPole]';
% Observer feedback gains
L = place(A',C', ObsPoles)';
Aobs = A-L*C;
Bobs = [B L];
Cobs = C;
Dobs = D;
Observer = ss(Aobs,Bobs,Cobs,Dobs);
```

Figure 4.23: Matlab script from Estimator

The pole of the estimator needs to be faster than the plant. There is a feature in Matlab used to find the most responsive pole of the given model. In the code in figure 4.23, it is shown how the pole is located. The pole is then used to assign the observer pole. The fastest pole is used as a start point where it is multiplied with some factor to sustain faster response as needed in the estimator.

4.4 Model Simulation

The simulation of the simplified and full order model were introduced in Matlab script for simulation. The simulation of the model was important to understand system behavior and validate the observer model. The observer model was simulated before implemented on the real-time target. The script is used to simulate the models in different scenarios to grasp the knowledge of the system before starting to analyze measured sensors parameters and the estimated state of the system.

4.4.1 State-Space Model

The open-loop of the model was implemented as shown in figure 4.24, where the simulation is in continuous time. Matlab command *Isim* was used for simulation of the obtained models with arbitrary inputs. The model nonlinearity was not introduced into the simulation since the nonlinearity of the real system was identify and compensated. The input of the model was a sinusoidal signal with different frequency and the amplitude of one.

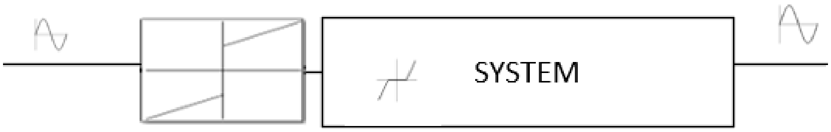
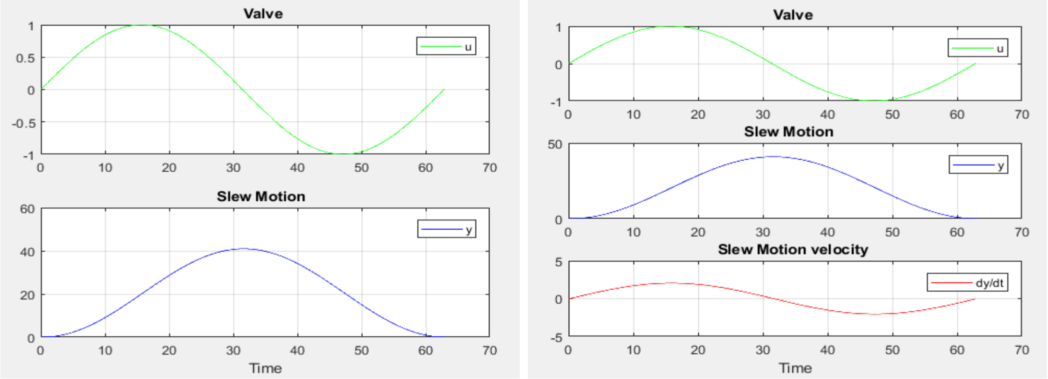


Figure 4.24: Open Loop State-Space Model

A Matlab script was developed where the state space model and the designed observer was implemented. The output of the slew motion model was in degrees where the observer has two output the slew motion and the velocity of the motion. The observer has two input the first one is the input of the system which is the valve opening and the second is slew of the motion. The result of the simulation in figure 4.25a was intriguing because of the way the cylinder velocity was following the input signal. The input of the system had the amplitude of 1 and the frequency of 0.1 rad/s.



(a) Open Loop Response of the plant

(b) Open Loop Response of the Observer

Figure 4.25: Plant and Observer

The results in figure 4.25a was from the identified plant and the observer. The observer poles were made faster than the plant model to ensure a faster response from the estimator.The poles of observer and the plant are represented in figure 4.26.

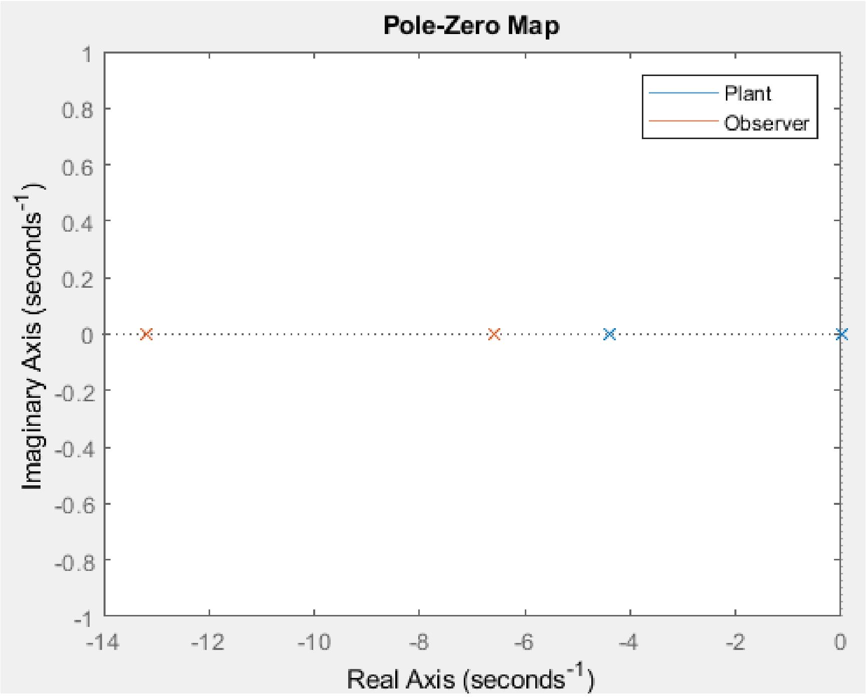
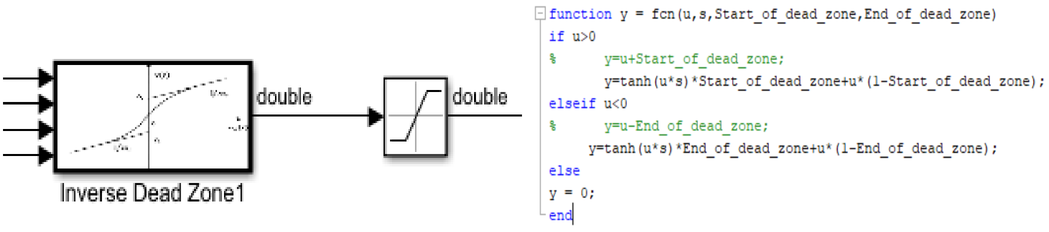


Figure 4.26: Pole and Zero of plant and Observer

The pole map in figure 4.26 confirmed that the observer pole lied in the far most left which suggests that the observer is in fact stable and is faster than the plant. The poles of the estimator are three times faster than the plants fastest pole. Having confirmed the theory in chapter 2, the results of the estimator was represented in figure 4.25. The initial condition of the plant and the observer were zero.

4.5 Simulink Real-Time

The implementation of the models like the dead zone and observer were implemented in the Simulink model designed in chapter 4. All the models that a program in the Simulink were deployed in a dedicated target machine (Speedgoat). The target machine was connected to the crane system, and testing of the model occurred. The identified inverse dead zone was programmed in Matlab and converted into a Simulink block.



```
function y = fcn(u,s,Start_of_dead_zone,End_of_dead_zone)
if u>0
%       y=u+Start_of_dead_zone;
        y=tanh(u*s)*Start_of_dead_zone+u*(1-Start_of_dead_zone);
elseif u<0
%       y=u-End_of_dead_zone;
      y=tanh(u*s)*End_of_dead_zone+u*(1-End_of_dead_zone);
else
y = 0;
end
```

(a) Simulink Block (b) Matlab Inverse Dead Zone code

Figure 4.27: Inverse Dead Zone implementation

Equation 4.9 was used to design code in figure 4.27. The output of the block was the input signal of the valve. The observer designed in section 4.3.2 was programmed in Simulink where equation 2.15 was used.

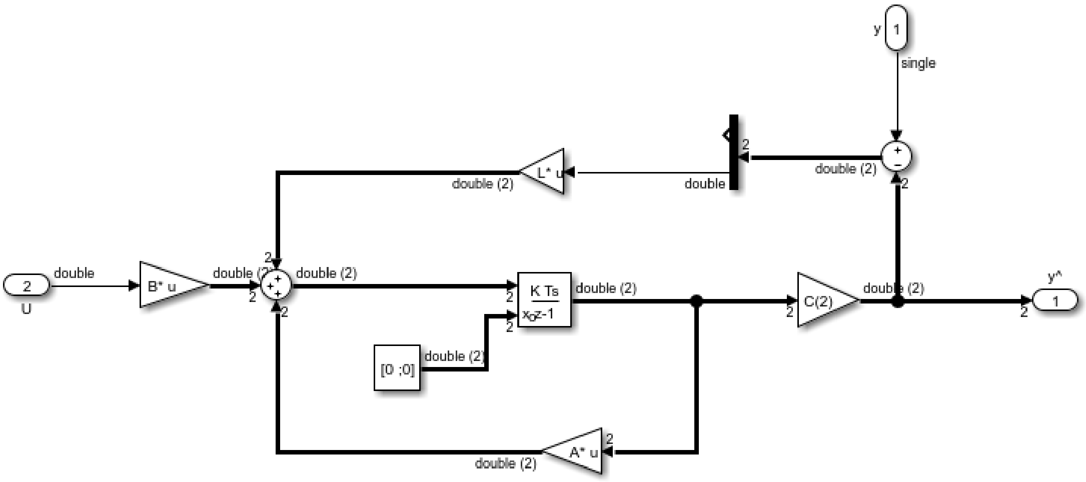


Figure 4.28: Observer Model

The Simulink subsystem in figure 4.28 represent the observer of the system. The estimator scheme has two inputs were U is the valve open of the real system, and y is the measured value of the slew motion of the crane in degrees. The output is the estimated slew motion and the velocity of the slew motion. The initial condition of the observer was set to be zero for both two states.

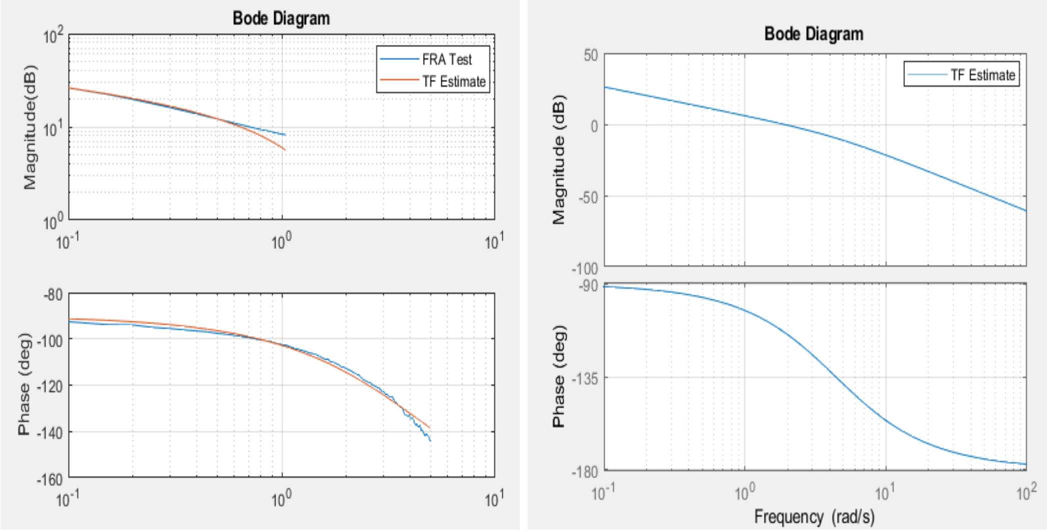
Chapter 5

Analysis of System and Results

In this chapter, we will discuss and present the results and dynamic properties of the simplified model and the full order model. Most of the results are from real experimental measurements data from the physical system, which will be given according to the steps used to model and identify the entire system. Furthermore, the quality of both model will be investigated and compared the results. The difference in all identified estimated model of simplified and all sub-models of the full order model will be demonstrated in terms of the frequency response function. All analysis and the results of the models are carried out when assumed that system have been linearized in chapter 4. Moreover, the external measurement of the slew motion will be combined with an internal measurement for validation.

5.1 Simplified model

In this section, the results of the reduced model will be presented, and the assumptions taken when simplifying the model will be discussed. For the analysis of will consider the states $[\theta,\dot{\theta}]$ to expose more of the dynamic characteristic of the hydraulic cylinder and the mechanical parts. The FRF data form the experimental test of the simplified model is compared with the estimated transfer function obtained from Matlab system identification toolbox. The transfer function of the simplified model is of 4.15.



(a) Measured FRF and Estimated model

(b) Estimated TF Bode

Figure 5.1: Bode plot for reduced model

The bode diagram in figure5.1a show the first low frequencies$[0.1..5 rad/s]$ of both measured and estimated model. The estimated transfer function can be split into two parts the pole at the origin with gain and the pole with real value at the left side of the s-plane. At the frequency range $[0.1..\infty rad/s]$ the first part contributes to the system with $-20dB/dec$ and-90 phase at the start. The second part is the time constant and form the frequency range$[4.4..\infty rad/s]$ it contributes with $-3dB/dec$ at $\omega = 4.4rad/s$ and for $\omega > 4.4rad/s$ it contribute with $-20dB/dec$ and -90 phase. The obtained transfer function of the simplified model is

$$G(s) = \frac{\phi(s)}{V_{open}(s)} = \frac{10}{s(s+4.4)} \tag{5.1}$$

The closed-loop dynamic corresponds to the estimated transfer function in equation 4.15 can be examined by looking at the feedback gain. The gain K is the feedback gain of the system.

$$G(s) = \frac{G(s)}{1 + K \cdot G(s)} \tag{5.2}$$

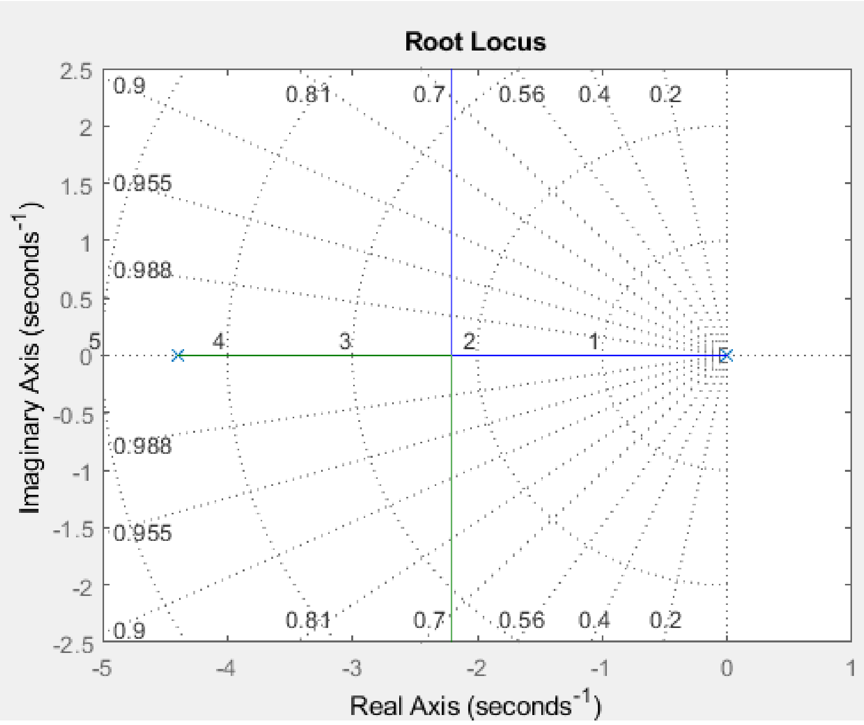


Figure 5.2: Root locus for open loop

The root locus plot in figure 4.15 explains which gain K make the system poles conjugate complex, which causes the system to oscillate or real pole. The system overall is stable since all the pole placed by any positive K will be located on the left side of the s-plane.

The verification of the estimated transfer function was done by open the valve with different types of signal and frequency range.

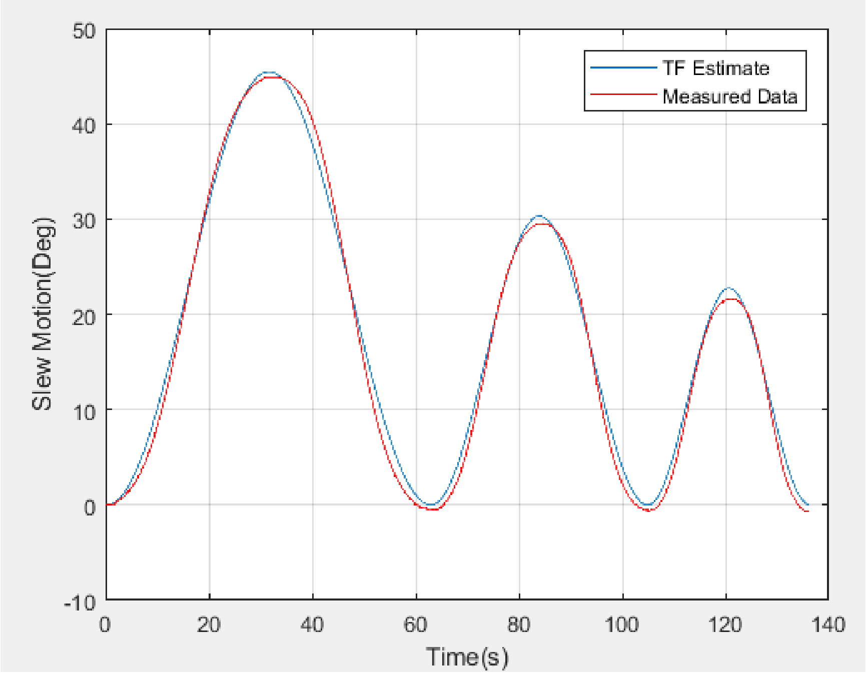


Figure 5.3: Comparison of Estimate TF and Measured Data with Sine wave Input

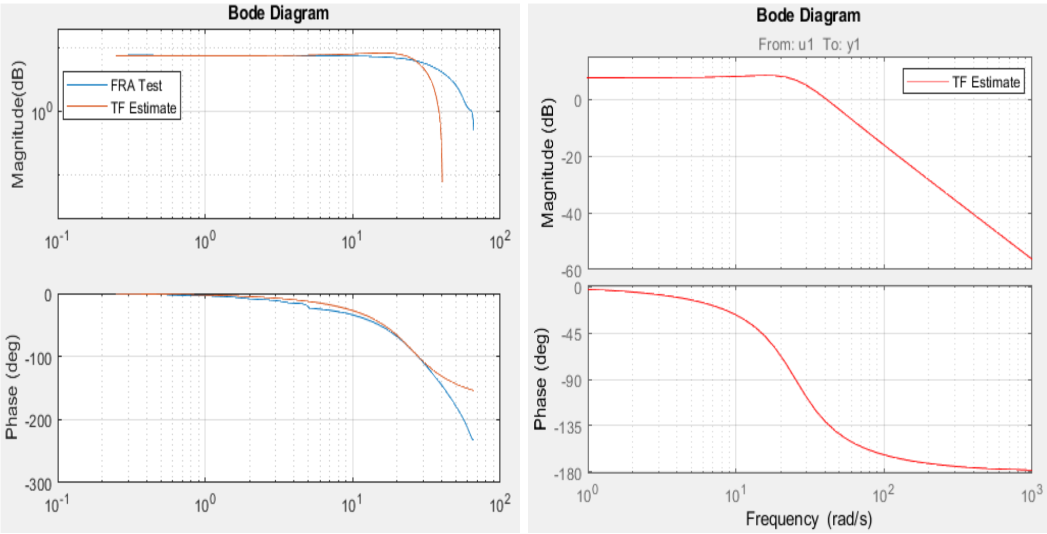
5.2 Full-order-model

The results of the full order model will be presented in this section. The assumptions and the simplification of the equation will be discussed. For the analysis, we will consider all the state of the model$[V, T, \theta]$.

5.2.1 DCV

The FRF data form the experimental test of DCV is compared with the estimated transfer function from Matlab system identification toolbox.The estimated transfer function is

$$G(s) = \frac{V(s)}{u(s)} = 2.375 \cdot \frac{25.23^2}{s^2 + 2 \cdot 0.545 \cdot 25.23s + 25.23^2} \tag{5.3}$$



(a) Measured FRF and Estimated model

(b) Estimated TF Bode

Figure 5.4: Bode plot for DCV model

The bode diagram in figure 5.4a show an equal magnitude and phase response for frequency $[0.2....\omega_n]$. The natural frequency ω_n for system both is around $20 rad/s$. The characteristic equation of the DCV can be analyzed when examing the frequency and step response of the system. By this way, we can validate the parameters given 5.3.

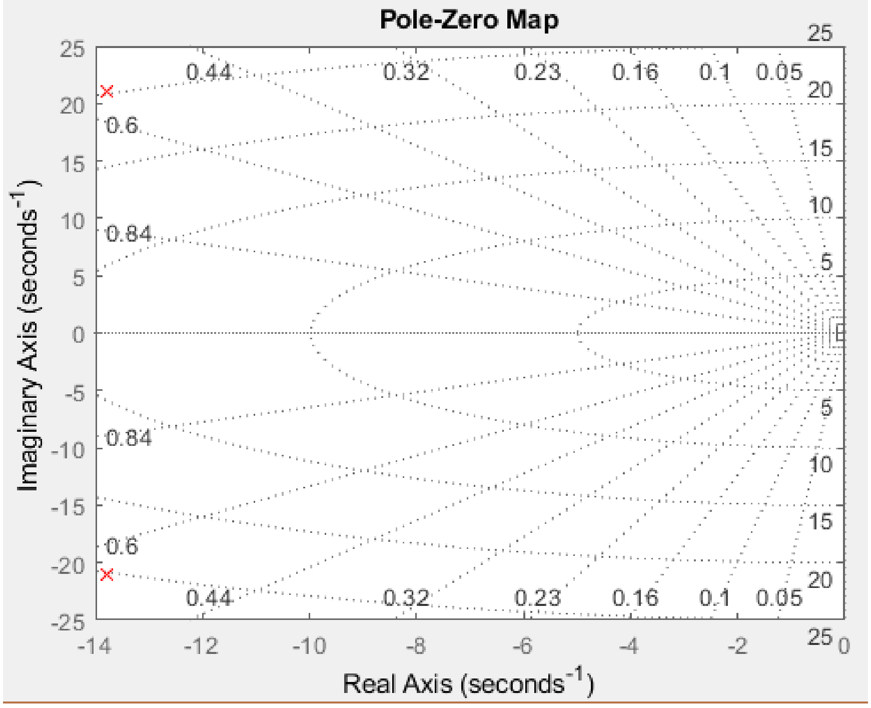
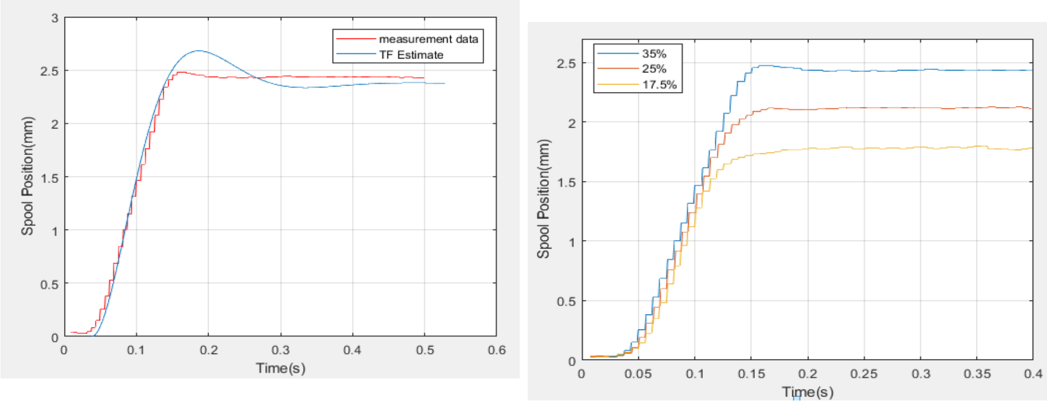


Figure 5.5: Pole-Zero map of DVC with ζ trajectories

The pole plot in figure 5.5 indicate the pole are complex-conjugate located in the side of the s-plane with a damping ratio $\zeta = 0.54$. Furthermore, the DCV can be interpreted as underdamped since the damping ration is on the interval $0 < \zeta < 1$ with two complex conjugate roots.
We can further verify the characteristic of the DCV by looking at the step response of measurement data and the estimated transfer function.

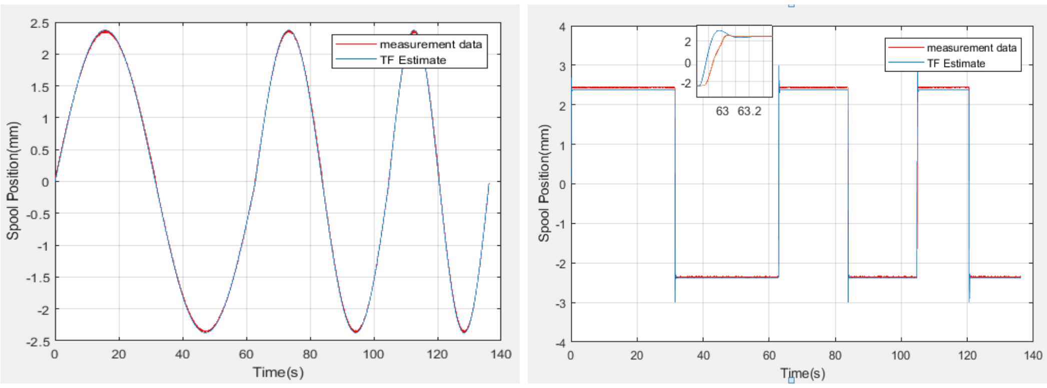


(a) characteristic of the step response of the under-damped process

(b) Step response from max to min

Figure 5.6: Step responses of DCV

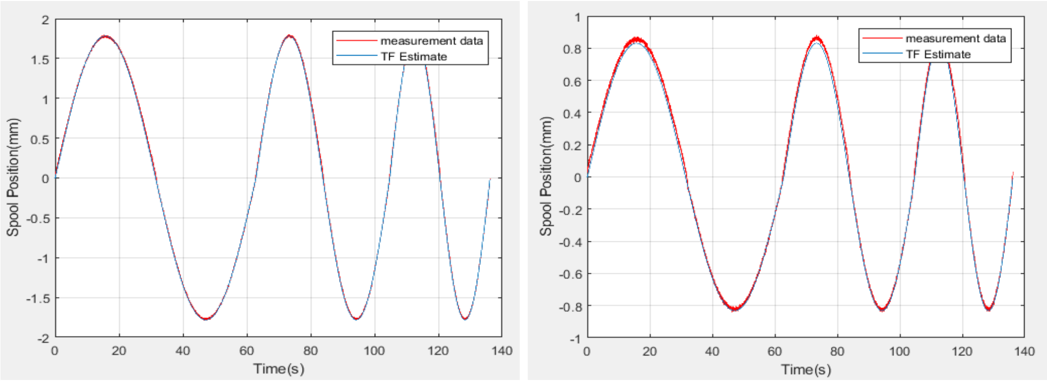
The result of the step response in figure 5.6a of the measured and simulated estimated model indicates that the DCV is more damped than the estimated model. The reason can be the nonlinearity part in DCV and that the valve is not fully open, maximum open is $7mm$ since the valve is restricted we only operate 35%. Figure 5.6b can explain that for lower operating range of the DCV amplitudes, the valve is well damped.
The verification of the estimated transfer function compared with different types of the input signals and amplitudes are presented below.



(a) Sine wave[0.1..0.3] Input 100%

(b) Square wave[0.1..0.3] Input 100%

Figure 5.7: Comparison of Estimate TF and Measured Data with Sine and Square wave Input



(a) Sine wave[0.1..0.3] Input 75%

(b) Sine wave[0.1..0.3] Input 35%

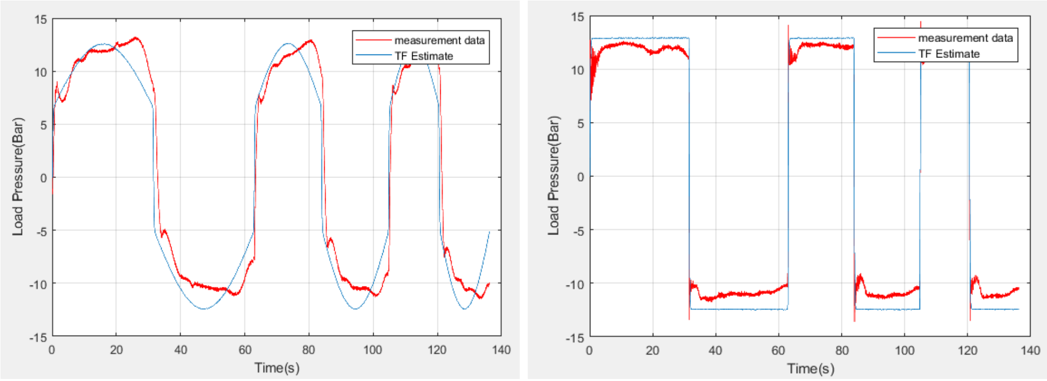
Figure 5.8: Comparison of Estimate TF and Measured Data with Sine wave Input

5.2.2 Continuity Equation

The estimated model with the load flow as input to load pressure was done by utilizing Matlab identification toolbox. The estimated transfer function obtained was

$$G(s) = \frac{P_L(s)}{Q_L(s)} = \frac{5.28}{0.11 \cdot s + 1} \tag{5.4}$$

The result of the estimated model was verified with the measured pressure difference of the hydraulic cylinder.



(a) Sine wave Flow Input Q_L 100%

(b) Square wave Flow Input Q_L 100%

Figure 5.9: Comparison of Estimate TF and Measured Data

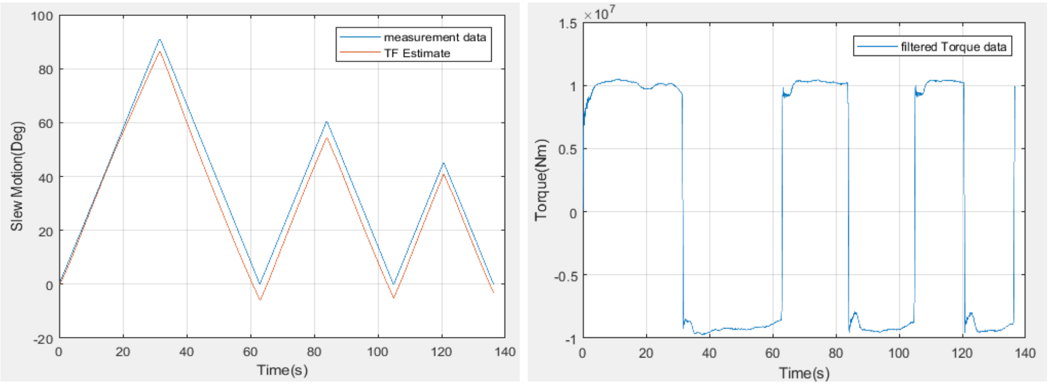
The estimated transfer function in equation 5.4 is utmost uncertainty due to the estimation taken in chapter 4. The lack of the supply pressure P_s sensor of the cylinder made it impossible to determine the factor K explained in figure 4.15. The reasonable solution was to use the flow characteristic of the valve and find one particular gain, but since the valve functions under 35%, and when examining figure4.16 was reasonable to assume that the flow and spool position of the valve is proportional to each other despite the fact this is not a correct assumption.

5.2.3 Mechanical Model

The estimate mechanical model with the torque load as input and output is the slew motion of the crane in degrees. The input value of the model is pressure load, which was multiplied with the cylinder and pinion cross-sections. The estimated transfer function of the model was

$$G(s) = \frac{\theta(s)}{T_h(s)} = \frac{1.0e-05}{s(s+39.4)} \tag{5.5}$$

The result of the estimated model was verified with the measured data the system.



(a) Slew Motion

(b) filtered Torque Input

Figure 5.10: Comparison of Estimate TF and Measured Data

According to [3], the hydraulic cylinder dynamic behavior of the system between the load flow and the velocity of the rack can be represented as

$$G(s) = \frac{\dot{x}(s)}{Q_L(s)} = \frac{\overline{A}^{-1}}{\omega_{mh}^{-2}s^2 + 2\zeta\omega_{mh}^{-1} + 1} \tag{5.6}$$

The cylinder eigenfrequency and damping is given by

$$\omega_{mh} = 2\overline{A}\sqrt{\frac{E}{V_{tm}}} \tag{5.7}$$

$$\zeta mh = \frac{\sigma}{4\overline{A}}\sqrt{\frac{V_{tm}}{E_m}} \tag{5.8}$$

The parameter $\overline{A}$ is effective cylinder area, σ damping coefficient,E bulk modulus, and V is the volume. The transfer function $H(s)$ was obtained by combining equation5.5,5.4, and converting from rotation to linear motion

$$H(s) = \frac{\dot{x}(s)}{Q_L(s)} = \frac{1.667}{\frac{s^2}{317.1} + 0.14s + 1} \tag{5.9}$$

The cylinder eigenfrequency is

$$\omega_{mh} = 17.8rad/s \tag{5.10}$$

and the damping ratio is

$$\zeta mh = 0.071 \tag{5.11}$$

The bandwidth of the cylinder increases with cylinder efficient area $\overline{A}$ meanwhile the damping decreases if $\overline{A}$ increases. The eigenfrequency of the DCV is large than the hydraulic cylinder where the thumb rule is $\omega_{DVC} > 3 * \omega_{mh}$

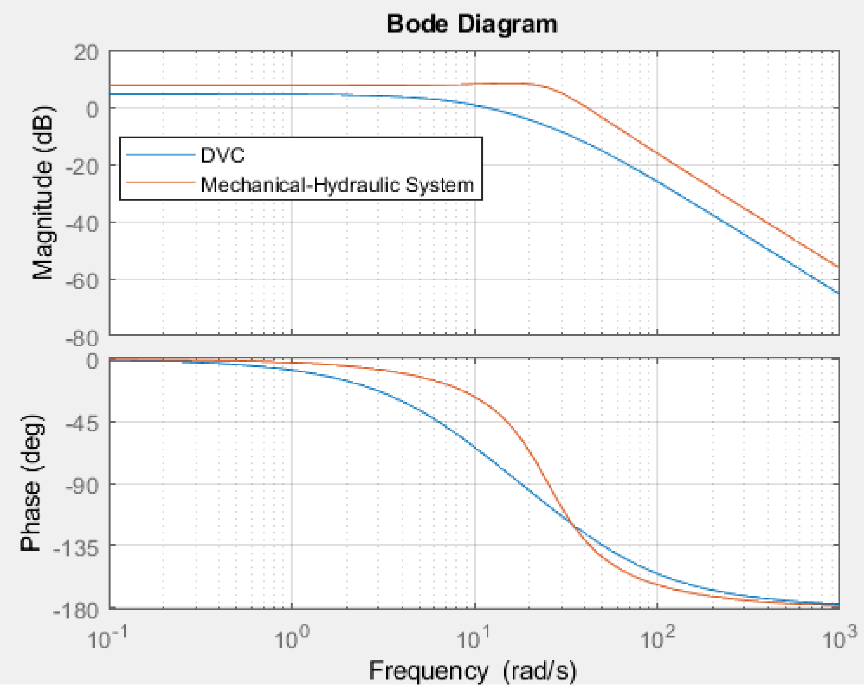


Figure 5.11: Bode diagram of DCV and Hydraulic Cylinder

The total transfer function from DCV to the hydraulic mechanical part is

$$G(s) = \frac{\theta(s)}{V_{Open}(s)} = \frac{4.582e05}{s(s^4 + 71.5s^3 + 2164s^2 + 3.673e04s + 2.02e05)} \tag{5.12}$$

The estimated full order model was verified by simulating the model and compare it with the measured model.

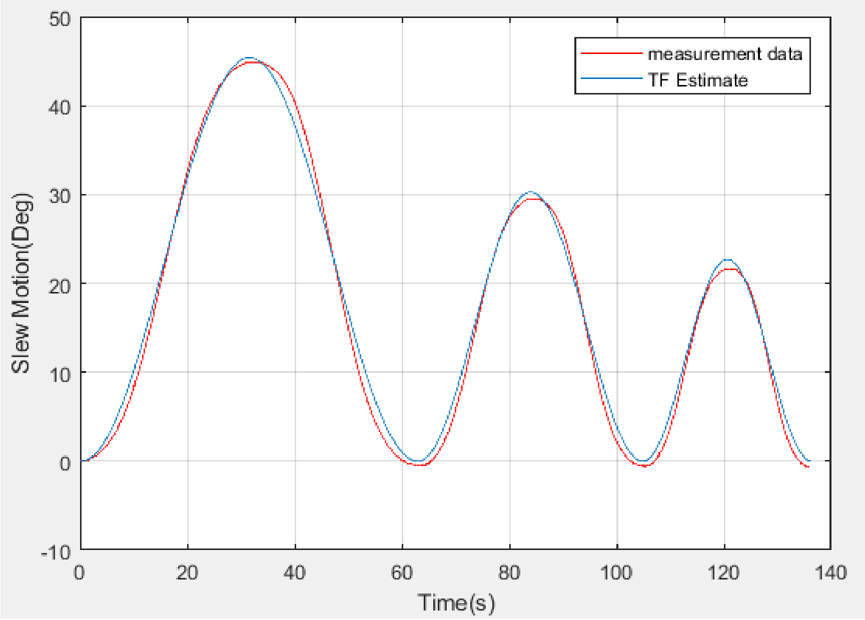


Figure 5.12: Comparison of Estimate TF and Measured Data with Sine wave Input

5.3 C-track Validation

The simplified model and the full order model will be compared in terms of frequency response and time-domain. Further, both model will be compared with the external measurements from C-track.

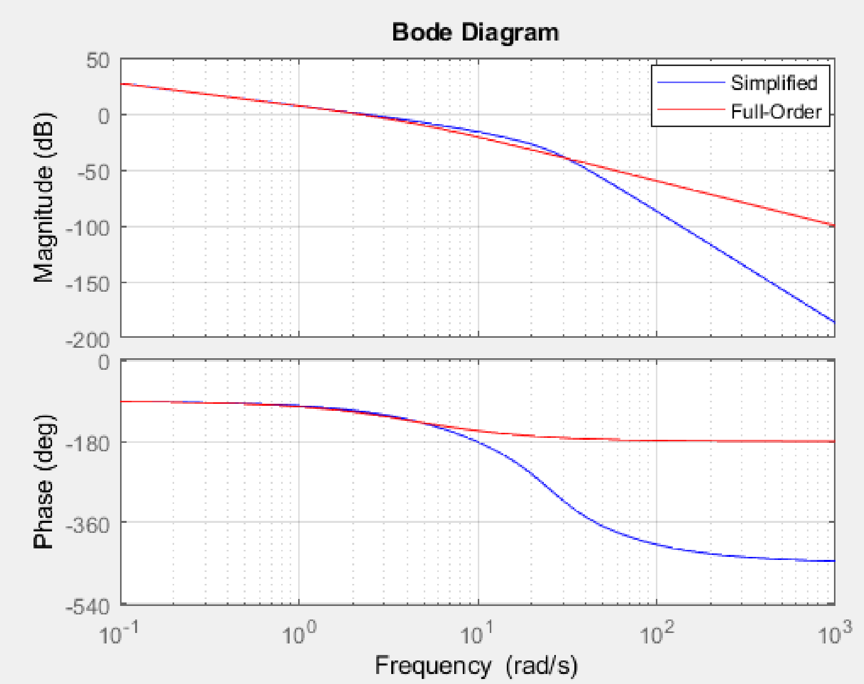
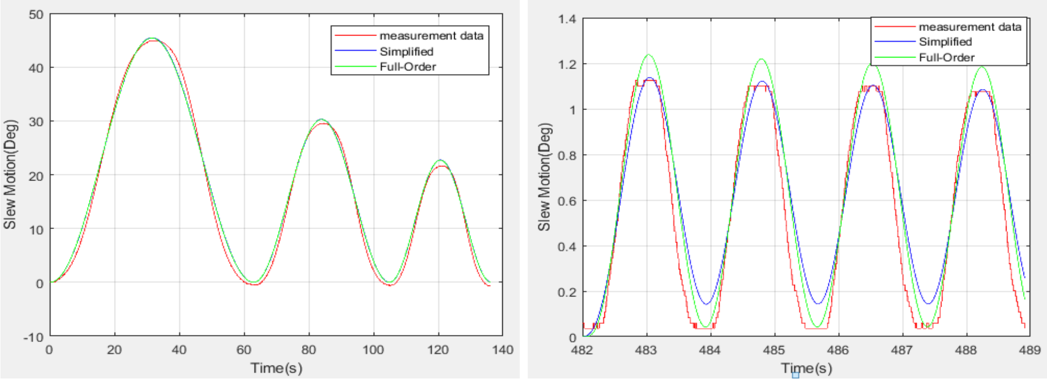


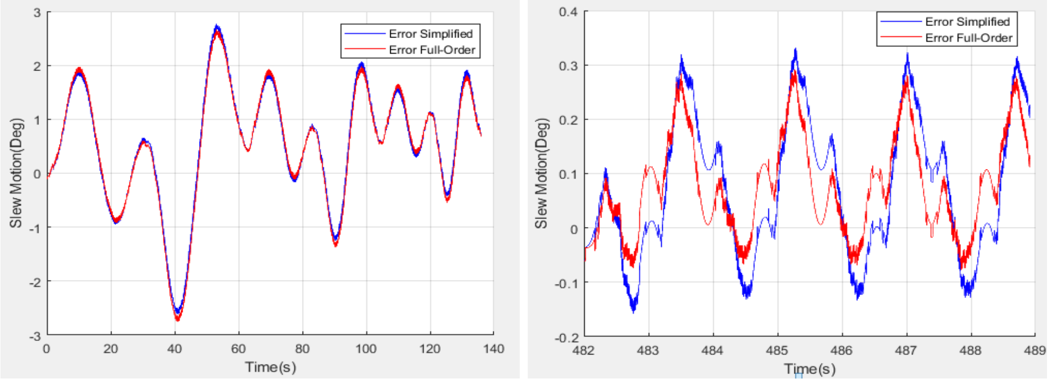
Figure 5.13: Bode diagram of Simplified and Full-Order Model

The frequency response of both model in figure 5.13 shows that the amplitude of both models are identical for low frequency and diverge at high frequency.



(a) Low frequency Input[0.1...0.2rad/s] (b) High frequency Input[3.5...3.7rad/s]

Figure 5.14: Comparison of Simplified,Full-Order Model and Measured Data

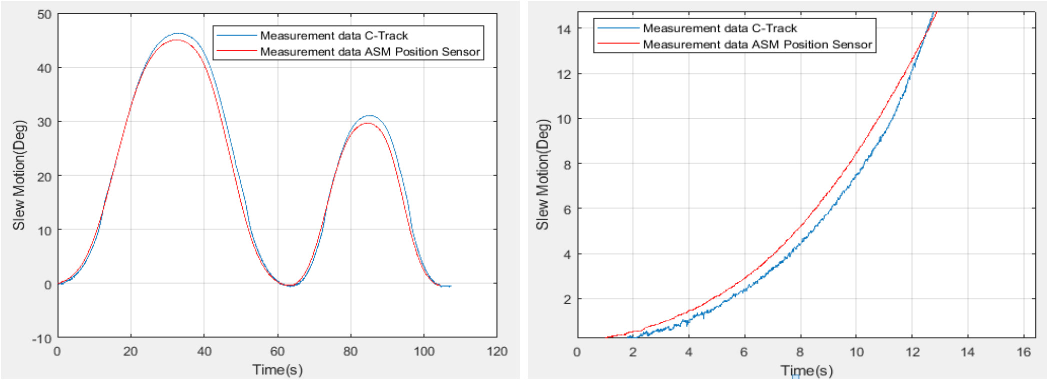


(a) Low frequency Input[0.1...0.2rad/s]

(b) High frequency Input[3.5...3.7rad/s]

Figure 5.15: Error between the estimated models and the measured data

The internal data fromASM position sensor programmed in cRio and the external data from C-track were compared for validation. The C-track data has a sample rate of $80Hz$, and the position sensor has $1000Hz$. The manual synchronizing of those two data types was imminent due to different acquisition sources which are not synced together.



(a) C-Track and Sensor position measured data

(b) C-Track first Model

Figure 5.16: Comparison between External and Internal data

The result in figure 5.16 has some deviation in the total amplitude of the system and phase shift due to synchronization. Further, it can be observed in figure 5.16b form $0..to..12°$ for each period similar behavior in deviation. The contrast in the interval $0..to..12°$ is due to two different models which were created in Vxelemts. The model created in figure3.16a has a different shape compared to the model in figure 3.16b which both are influenced to how and where the markers are placed on the crane.

Chapter 6
Conclusion

System instrumentation of the crane has been achieved by investigating the electrical connection and programmed cRio to extracting internal sensor signals. Labview was fully utilized in scaling the internal signals into related engineering value and exported to the Real-Time target machine(Speedgoat) via UDP-ethernet connection. The external measurements of the crane slew motion were achieved by utilizing C-track from Creaform, and measurement segment models were developed in Vxelement. The existing Matlab Simulink model of the Real-Time interface was extended upon the extension of the signals. Further, the signals received from cRio were processed, sorted, and exported to Matlab workspace.

Simplified and full-order models from valve to hydraulic, mechanical part have been model mathematical and expressed in term of the transfer function. The assumptions taken are sufficiently explained in terms of theory. Furthermore, the nonlinear elements of the whole system are identified and compensated with a hyperbolic tangent, which was sufficient to perform a frequency response function for the system, despite that DCV and the hydraulic system has some unknown nonlinearity.

Estimation algorithms were designed for models, and signal process. Matlab System Identification Tool-Box was utilized in identifying the transfer functions. The reduced model was converted to a state-space model to design observer, which states $\theta, \dot{\theta}$ were generated. Both models have been analyzed regarding frequency response and equation governing the model exposed in connection with relation between the systems. The reduction from 5th order to 2nd has been simulated in terms of open-loop, and the result explained. The results obtained was also compared with the measured data. Additionally, the position sensor was verified with an external measurement.In general, both models were sufficient in terms of frequency response. In figure 5.15 indicate that the full order model was better than the simplified for high-frequency inputs. comprehensive both model can be used in designing slew motion control and dynamic analysis.

Bibliography

[1] Szabolcs Fodor. Carlos Vazquez. and Leonid Freidovich. “Automation of slewing motions for forestry cranes”. In: 6 (2015), pp. 1–6.

[2] Pedro M.Uwe Mettin.Simon Westerberg. and Anton S. Shiriaev. “Modeling and Control of Hydraulic Rotary Actuators used in Forestry Cranes”. In: 6 (2009), pp. 1–6.

[3] Michael Ruderman. “Full- and Reduced-order Model of Hydraulic Cylinder for Motion Control”. In: 6 (2019), pp. 1–6.

[4] Lennart Ljung. “Balck-box form Input-output Measurements”. In: 4 (2001), pp. 1–5.

[5] Rastko R. Selmic and Frank L. Lewis. “Deadzone Compensation in Motion Control Systems Using Neural Networks”. In: 12 (2000), pp. 1–12.

[6] Et Al. José Rubio. *Proportional Derivative Control with Inverse Dead-Zone for Pendulum Systems.* 2013. URL: `https://www.hindawi.com/journals/mpe/2013/173051/`.

[7] Lennart Ljung. “System Identification Theory for the User”. In: *Theory for the User* 10 (1997), pp. 1–11.

[8] Ismail Mohamed Hussein. “Observation and Motion Control of Hydraulic Loader Crane”. In: 4 (2018), pp. 12–16.

[9] 2013. URL: `https://www.illustrationsource.com/stock/image/506900/diagram-of-a-rack-and-pinion/?results_per_page=1`.

Appendix A
Project Description

Master Mekatronikk, Våren 2019

External measurements and analysis of slew motion of the crane

Background and problem statement

The cranes, equally as other large-scale machines with multiple DOFs (degrees of freedom), feature complex multibody dynamics with a coupled motion of the actuated joints and links. Modern 3D scanning technologies allow for external measurements during the system operation and provide accurate reference data in Cartesian space. Nevertheless, systematic analysis, modeling, and identification of the relative motion of structural elements of machines are required.

Standard loader crane HMF2020, available in Mechatronic Lab at UiA, has a serial kinematics with three rotary and one telescopic joints. The first slew joint (see figure below) rotates the overall column relative to the base and is of the utmost importance for the overall operation. The rack-and-pinion type transmission of the slew joint and multiple structural couplings of the kinematic chain act as causes for large uncertainties, even when the actuators are accurately controlled.

The objectives of the Master project are in investigation of the slew motion of the crane by using the external 3D measurements of reference points on the moving structure. A system-oriented modeling approach should deliver a basis for the signal processing and estimator design of the slew motion. External measurements can be combined with internal process data of the crane for an improved analysis and identification of the system behavior. Available real-time interfaces should be extended.

Main tasks can be listed as but not limited to:

- Mechatronic design of complex experiments and system instrumentation;
- Extension of the available real-time interfaces, programming and data processing;
- Signals analysis, modeling, identification of system behavior, reverse engineering
- Design of the estimation algorithms, experimental evaluation, systematic hypothesis tests.

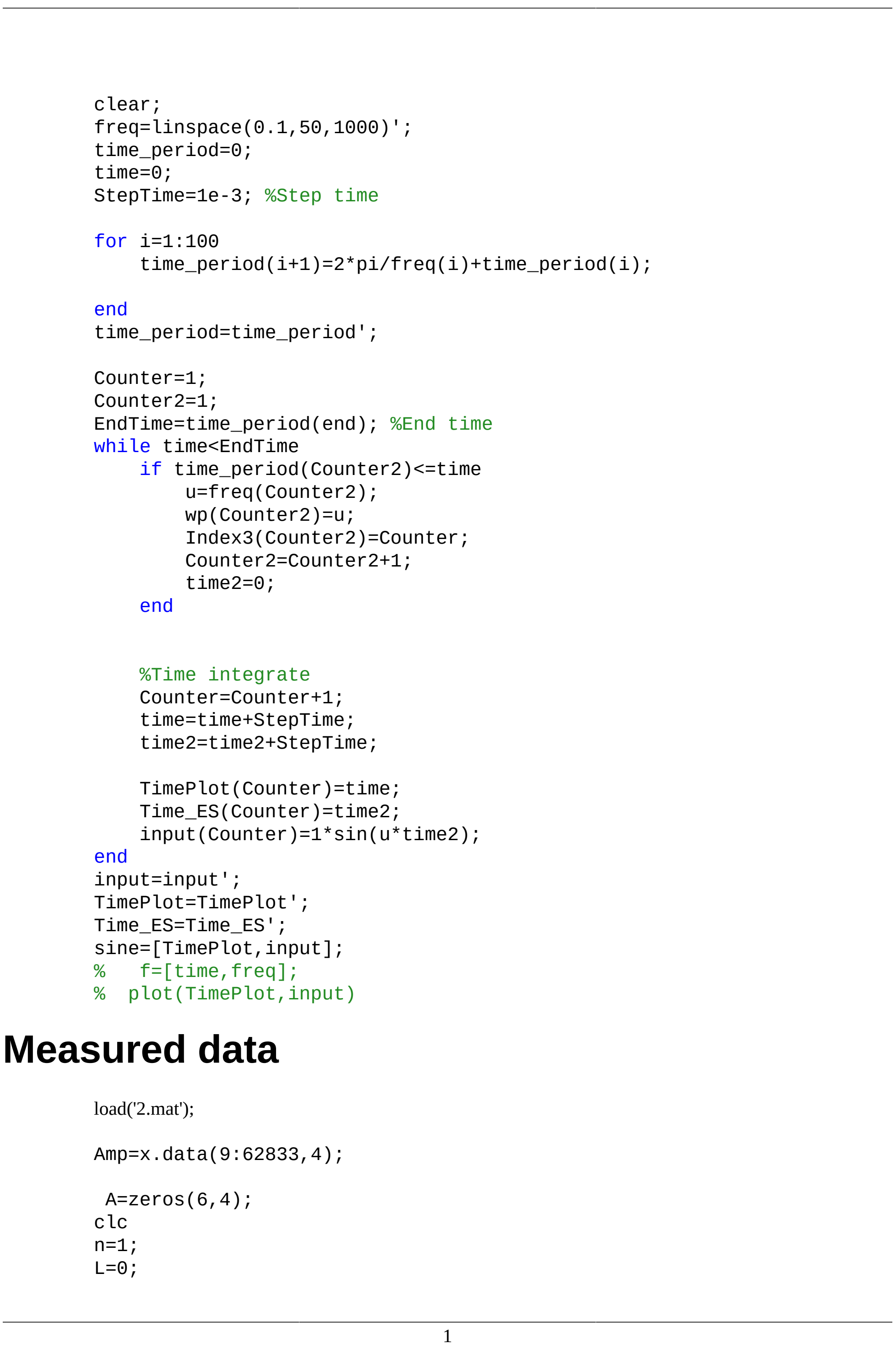
Additional information

The Master project is best suitable for a group of 2 students and will be offered under co-supervision of the Mechatronics Innovation Lab (MIL).

Contact persons: Michael Ruderman (UiA), Morten Kollerup Bak (MIL)

Appendix A

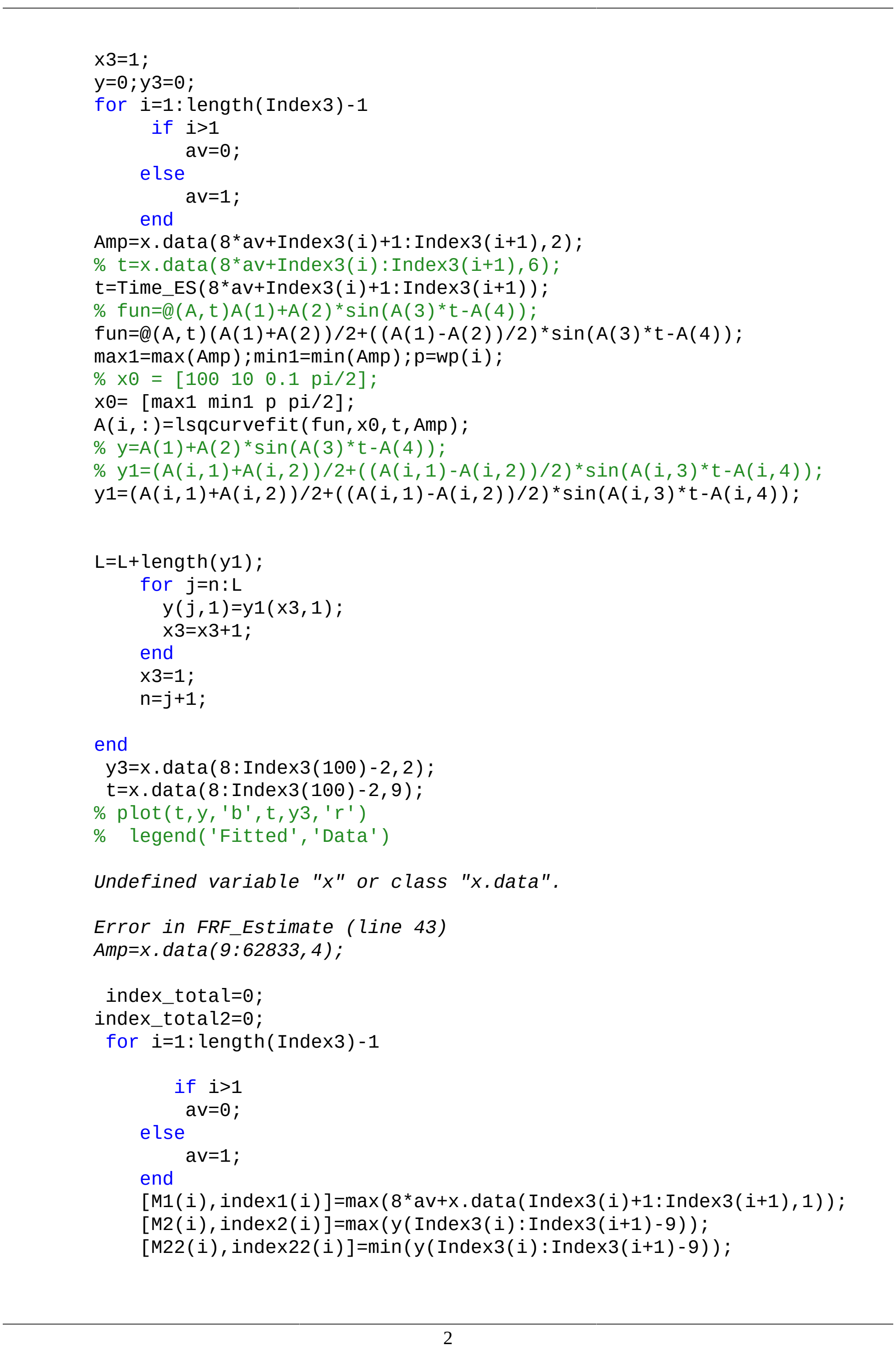
Estimation algorithms



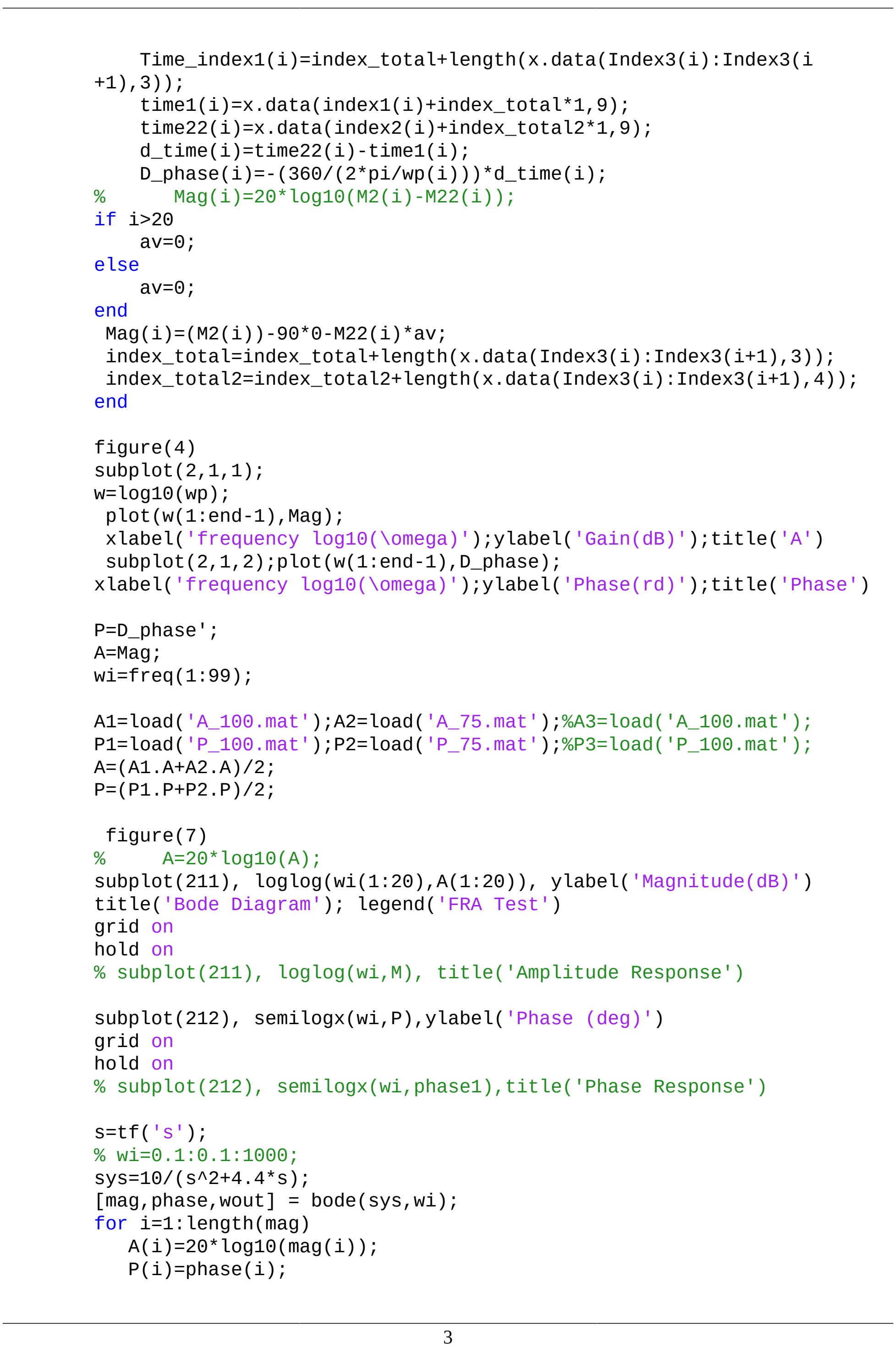
```
clear;
freq=linspace(0.1,50,1000)';
time_period=0;
time=0;
StepTime=1e-3; %Step time

for i=1:100
    time_period(i+1)=2*pi/freq(i)+time_period(i);

end
time_period=time_period';

Counter=1;
Counter2=1;
EndTime=time_period(end); %End time
while time<EndTime
    if time_period(Counter2)<=time
        u=freq(Counter2);
        wp(Counter2)=u;
        Index3(Counter2)=Counter;
        Counter2=Counter2+1;
        time2=0;
    end


    %Time integrate
    Counter=Counter+1;
    time=time+StepTime;
    time2=time2+StepTime;

    TimePlot(Counter)=time;
    Time_ES(Counter)=time2;
    input(Counter)=1*sin(u*time2);
end
input=input';
TimePlot=TimePlot';
Time_ES=Time_ES';
sine=[TimePlot,input];
%   f=[time,freq];
%  plot(TimePlot,input)
```

Measured data

```
load('2.mat');

Amp=x.data(9:62833,4);

 A=zeros(6,4);
clc
n=1;
L=0;
```



```
x3=1;
y=0;y3=0;
for i=1:length(Index3)-1
     if i>1
        av=0;
    else
        av=1;
    end
Amp=x.data(8*av+Index3(i)+1:Index3(i+1),2);
% t=x.data(8*av+Index3(i):Index3(i+1),6);
t=Time_ES(8*av+Index3(i)+1:Index3(i+1));
% fun=@(A,t)A(1)+A(2)*sin(A(3)*t-A(4));
fun=@(A,t)(A(1)+A(2))/2+((A(1)-A(2))/2)*sin(A(3)*t-A(4));
max1=max(Amp);min1=min(Amp);p=wp(i);
% x0 = [100 10 0.1 pi/2];
x0= [max1 min1 p pi/2];
A(i,:)=lsqcurvefit(fun,x0,t,Amp);
% y=A(1)+A(2)*sin(A(3)*t-A(4));
% y1=(A(i,1)+A(i,2))/2+((A(i,1)-A(i,2))/2)*sin(A(i,3)*t-A(i,4));
y1=(A(i,1)+A(i,2))/2+((A(i,1)-A(i,2))/2)*sin(A(i,3)*t-A(i,4));


L=L+length(y1);
    for j=n:L
      y(j,1)=y1(x3,1);
      x3=x3+1;
    end
    x3=1;
    n=j+1;

end
 y3=x.data(8:Index3(100)-2,2);
 t=x.data(8:Index3(100)-2,9);
% plot(t,y,'b',t,y3,'r')
%  legend('Fitted','Data')

Undefined variable "x" or class "x.data".

Error in FRF_Estimate (line 43)
Amp=x.data(9:62833,4);

 index_total=0;
index_total2=0;
 for i=1:length(Index3)-1

       if i>1
        av=0;
    else
        av=1;
    end
    [M1(i),index1(i)]=max(8*av+x.data(Index3(i)+1:Index3(i+1),1));
    [M2(i),index2(i)]=max(y(Index3(i):Index3(i+1)-9));
    [M22(i),index22(i)]=min(y(Index3(i):Index3(i+1)-9));
```



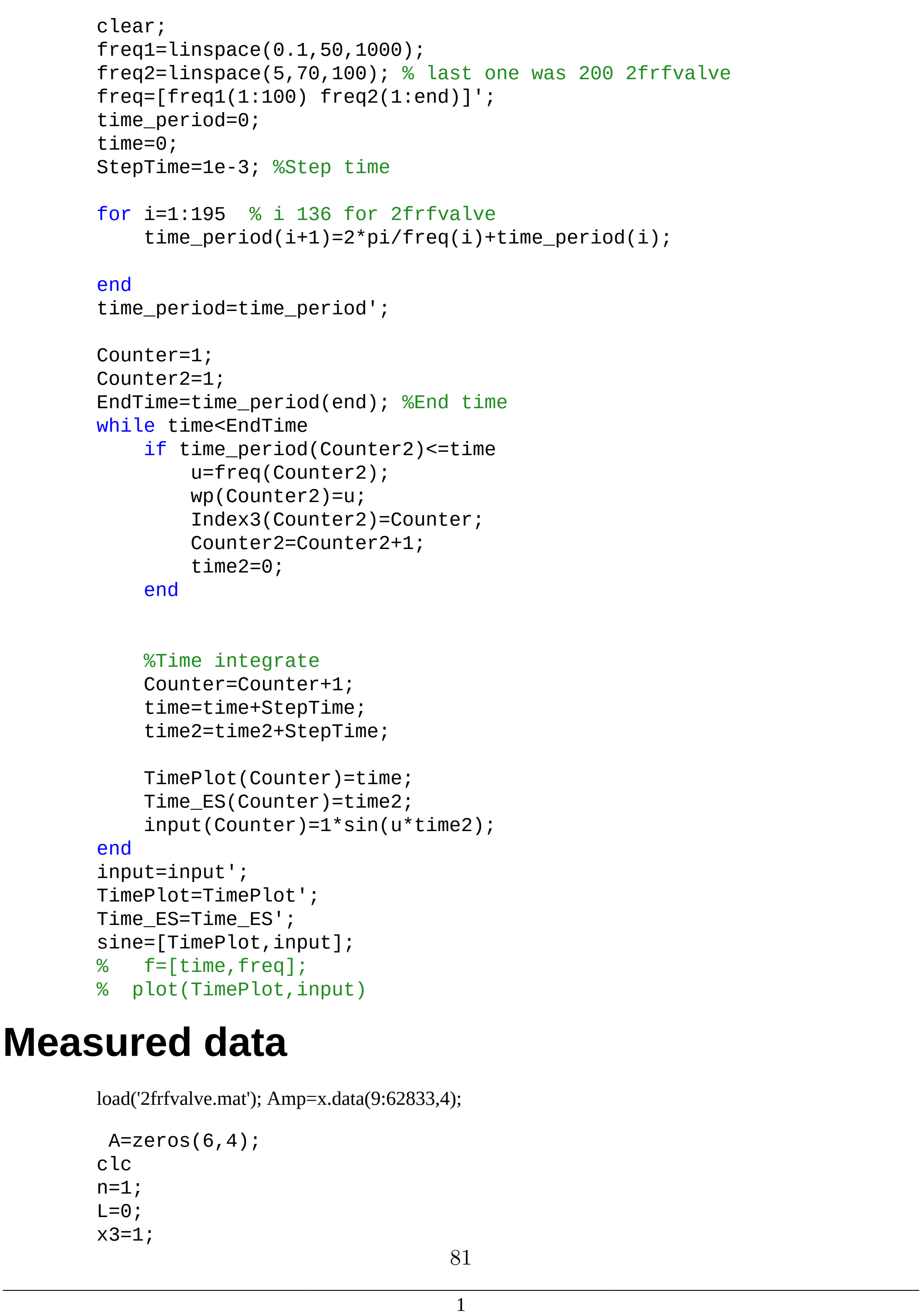
```
    Time_index1(i)=index_total+length(x.data(Index3(i):Index3(i
+1),3));
    time1(i)=x.data(index1(i)+index_total*1,9);
    time22(i)=x.data(index2(i)+index_total2*1,9);
    d_time(i)=time22(i)-time1(i);
    D_phase(i)=-(360/(2*pi/wp(i)))*d_time(i);
%      Mag(i)=20*log10(M2(i)-M22(i));
if i>20
    av=0;
else
    av=0;
end
 Mag(i)=(M2(i))-90*0-M22(i)*av;
 index_total=index_total+length(x.data(Index3(i):Index3(i+1),3));
 index_total2=index_total2+length(x.data(Index3(i):Index3(i+1),4));
end

figure(4)
subplot(2,1,1);
w=log10(wp);
 plot(w(1:end-1),Mag);
 xlabel('frequency log10(\omega)');ylabel('Gain(dB)');title('A')
 subplot(2,1,2);plot(w(1:end-1),D_phase);
xlabel('frequency log10(\omega)');ylabel('Phase(rd)');title('Phase')

P=D_phase';
A=Mag;
wi=freq(1:99);

A1=load('A_100.mat');A2=load('A_75.mat');%A3=load('A_100.mat');
P1=load('P_100.mat');P2=load('P_75.mat');%P3=load('P_100.mat');
A=(A1.A+A2.A)/2;
P=(P1.P+P2.P)/2;

 figure(7)
%     A=20*log10(A);
subplot(211), loglog(wi(1:20),A(1:20)), ylabel('Magnitude(dB)')
title('Bode Diagram'); legend('FRA Test')
grid on
hold on
% subplot(211), loglog(wi,M), title('Amplitude Response')

subplot(212), semilogx(wi,P),ylabel('Phase (deg)')
grid on
hold on
% subplot(212), semilogx(wi,phase1),title('Phase Response')

s=tf('s');
% wi=0.1:0.1:1000;
sys=10/(s^2+4.4*s);
[mag,phase,wout] = bode(sys,wi);
for i=1:length(mag)
   A(i)=20*log10(mag(i));
   P(i)=phase(i);
```

3

```
end
figure(1)
grid on
hold on
bode(sys)
```

Published with MATLAB® R2017b

A.1 Valve code



```
clear;
freq1=linspace(0.1,50,1000);
freq2=linspace(5,70,100); % last one was 200 2frfvalve
freq=[freq1(1:100) freq2(1:end)]';
time_period=0;
time=0;
StepTime=1e-3; %Step time

for i=1:195  % i 136 for 2frfvalve
    time_period(i+1)=2*pi/freq(i)+time_period(i);

end
time_period=time_period';

Counter=1;
Counter2=1;
EndTime=time_period(end); %End time
while time<EndTime
    if time_period(Counter2)<=time
        u=freq(Counter2);
        wp(Counter2)=u;
        Index3(Counter2)=Counter;
        Counter2=Counter2+1;
        time2=0;
    end


    %Time integrate
    Counter=Counter+1;
    time=time+StepTime;
    time2=time2+StepTime;

    TimePlot(Counter)=time;
    Time_ES(Counter)=time2;
    input(Counter)=1*sin(u*time2);
end
input=input';
TimePlot=TimePlot';
Time_ES=Time_ES';
sine=[TimePlot,input];
%   f=[time,freq];
%  plot(TimePlot,input)
```

Measured data

load('2frfvalve.mat'); Amp=x.data(9:62833,4);

```
 A=zeros(6,4);
clc
n=1;
L=0;
x3=1;
```

```
y=0;y3=0;
for i=1:length(Index3)-1
     if i>1
        av=0;
    else
        av=1;
    end
Amp=x.data(8*av+Index3(i)+1:Index3(i+1),6);
% t=x.data(8*av+Index3(i):Index3(i+1),6);
t=Time_ES(8*av+Index3(i)+1:Index3(i+1));
%  fun=@(A,t)A(1)+A(2)*sin(A(3)*t-A(4));
 fun=@(A,t)(A(1)+A(2))/2+((A(1)-A(2))/2)*sin(A(3)*t-A(4));
max1=max(Amp);min1=min(Amp);p=wp(i);
%  x0 = [0 max1 p pi/2];
 x0= [max1 min1 p pi/2];
A(i,:)=lsqcurvefit(fun,x0,t,Amp);
% y=A(1)+A(2)*sin(A(3)*t-A(4));
% y1=(A(i,1)+A(i,2))/2+((A(i,1)-A(i,2))/2)*sin(A(i,3)*t-A(i,4));
y1=(A(i,1)+A(i,2))/2+((A(i,1)-A(i,2))/2)*sin(A(i,3)*t-A(i,4));


L=L+length(y1);
    for j=n:L
      y(j,1)=y1(x3,1);
      x3=x3+1;
    end
    x3=1;
    n=j+1;

end

Undefined variable "x" or class "x.data".

Error in FRF_Valve (line 59)
Amp=x.data(8*av+Index3(i)+1:Index3(i+1),6);

 y3=x.data(8:Index3(195)-2,6);
 t=x.data(8:Index3(195)-2,9);
plot(t,y,'b',t,y3,'r')
 legend('Fitted','Data')

 index_total=0;
index_total2=0;
 for i=1: length(Index3)-1

       if i>1
        av=0;
    else
        av=1;
    end
    [M1(i),index1(i)]=max(8*av+x.data(Index3(i)+1:Index3(i+1),1));
    [M2(i),index2(i)]=max(y(Index3(i):Index3(i+1)-9));
    [M22(i),index22(i)]=min(y(Index3(i):Index3(i+1)-9));
    Time_index1(i)=index_total+length(x.data(Index3(i):Index3(i
+1),1));
    time1(i)=x.data(index1(i)+index_total*1,9);
    time22(i)=x.data(index2(i)+index_total2*1,9);
    d_time(i)=time22(i)-time1(i);
    D_phase(i)=-(360/(2*pi/wp(i)))*d_time(i);
%      Mag(i)=20*log10(M2(i)-M22(i));
 Mag(i)=(M2(i));
 index_total=index_total+length(x.data(Index3(i):Index3(i+1),1));
 index_total2=index_total2+length(x.data(Index3(i):Index3(i+1),6));
end

figure(4)
subplot(2,1,1);
w=log10(wp);
 plot(w(4:end-1),Mag(4:end)/Mag(4));
 xlabel('frequency log10(\omega)');ylabel('Gain(dB)');title('A')
 subplot(2,1,2);plot(w(4:end-1),D_phase(4:end));
xlabel('frequency log10(\omega)');ylabel('Phase(rd)');title('Phase')

P=D_phase(4:end)';
A=Mag(4:end)/1';
% wi=freq(1:198);
wi=wp(4:end-1);
In=x.data(8:Index3(195)-2,1);
Out=y3;

A1=load('A_35.mat');A2=load('A_75.mat');A3=load('A_100.mat');
P1=load('P_35.mat');P2=load('P_75.mat');P3=load('P_100.mat');
A=(A1.A+A2.A+A3.A)/3;
P=(P1.P+P2.P+P3.P)/3;

A=sgolayfilt(A, 11, 131);
P=sgolayfilt(P, 11, 131);

A=20*log10(A);

hold on
figure(7)
subplot(211), loglog(wi,(A)), ylabel('Magnitude(dB)')
title('Bode Diagram'); %legend('FRA Test')
grid on
hold on
% subplot(211), loglog(wi,M), title('Amplitude Response')

subplot(212), semilogx(wi,P),ylabel('Phase (deg)')
grid on
hold on
% subplot(212), semilogx(wi,phase1),title('Phase Response')

s=tf('s');
% wi=0.1:0.1:1000;
% sys=10/(s^2+4.4*s);
sys=tf4;
[mag,phase,wout] = bode(sys,wi);
```

A.2 Estimator code

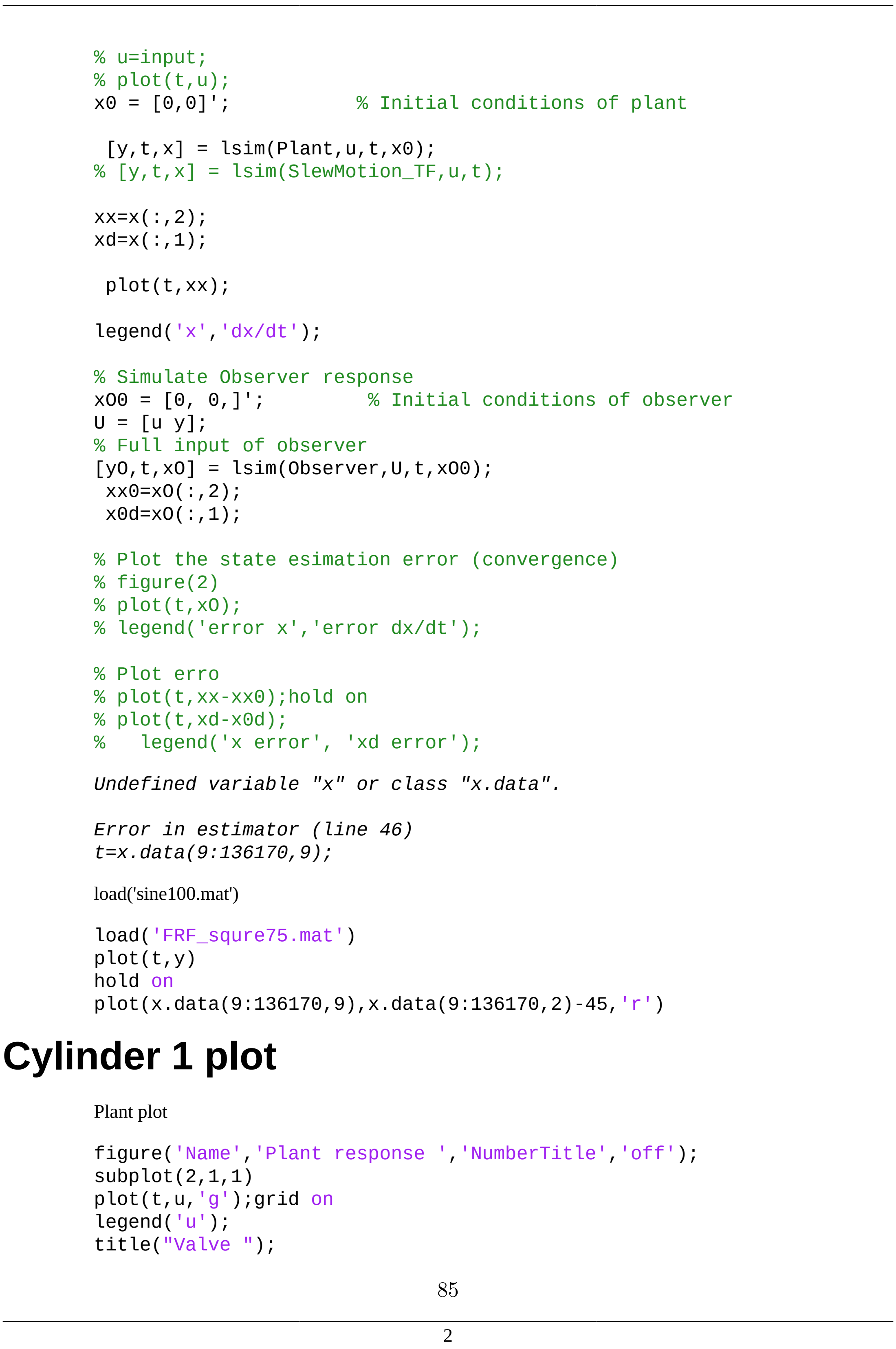
```
% clear
% clc
s=tf('s');
% System transfer function form FRF measurements
  SlewMotion_TF= 10/(s*(s+4.4));

% Transform to the state-space model
SlewMotion_ss = ss(SlewMotion_TF);

% Parameters
A = SlewMotion_ss.a;
B = SlewMotion_ss.b;
C = SlewMotion_ss.c;
D = SlewMotion_ss.d;


% controllability of the models
c=ctrb(A,B);
r=rank(c);
% observability of the models
o=obsv(A,C);
r2=rank(o);
```

State-space model Slew Motion

```
Plant = ss(A,B,C,D);
% Evaluate the "fastest" pole of the plant (abs -> w0 eigenfrequency)
FastPole = max(abs(pole(Plant)));
% Assign Observer poles to be, for example, 3 times "faster" than
 FastPole
ObsPoles = [-3*FastPole, -1.5*FastPole]';
% Observer feedback gains
L = place(A',C', ObsPoles)';
Aobs = A-L*C;
Bobs = [B L];
Cobs = C;
Dobs = D;
Observer = ss(Aobs,Bobs,Cobs,Dobs);

% Compare the pole-zero diagram of plant and observer
pzmap(Plant,Observer);
% pzmap(Plant2,Observer3);
% legend('Plant','Plant2');
 legend('Plant','Observer');
```

Simulate Plant response

```
%t = [0:0.001:63]';          % Time vector
t=x.data(9:136170,9);
% u = zeros(length(t),1);     % Input sequence with square pulses
% u=1*sin(t*0.1);
u=x.data(9:136170,1);
% t=TimePlot;
```



```
% u=input;
% plot(t,u);
x0 = [0,0]';           % Initial conditions of plant

 [y,t,x] = lsim(Plant,u,t,x0);
% [y,t,x] = lsim(SlewMotion_TF,u,t);

xx=x(:,2);
xd=x(:,1);

 plot(t,xx);

legend('x','dx/dt');

% Simulate Observer response
xO0 = [0, 0,]';         % Initial conditions of observer
U = [u y];
% Full input of observer
[yO,t,xO] = lsim(Observer,U,t,xO0);
 xx0=xO(:,2);
 x0d=xO(:,1);

% Plot the state esimation error (convergence)
% figure(2)
% plot(t,xO);
% legend('error x','error dx/dt');

% Plot erro
% plot(t,xx-xx0);hold on
% plot(t,xd-x0d);
%   legend('x error', 'xd error');
```

```
Undefined variable "x" or class "x.data".

Error in estimator (line 46)
t=x.data(9:136170,9);
```

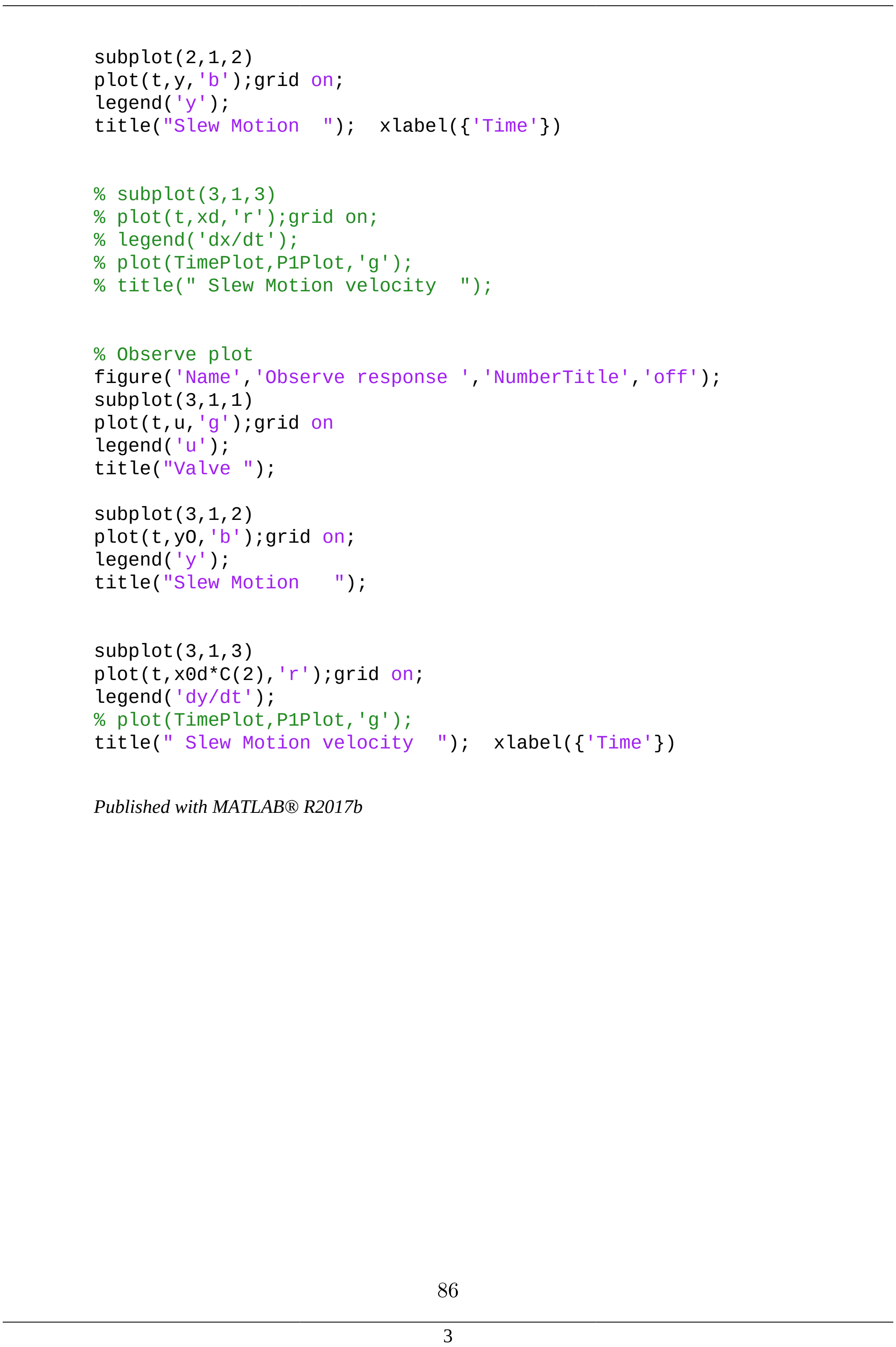
load('sine100.mat')

```
load('FRF_squre75.mat')
plot(t,y)
hold on
plot(x.data(9:136170,9),x.data(9:136170,2)-45,'r')
```

Cylinder 1 plot

Plant plot

```
figure('Name','Plant response ','NumberTitle','off');
subplot(2,1,1)
plot(t,u,'g');grid on
legend('u');
title("Valve ");
```



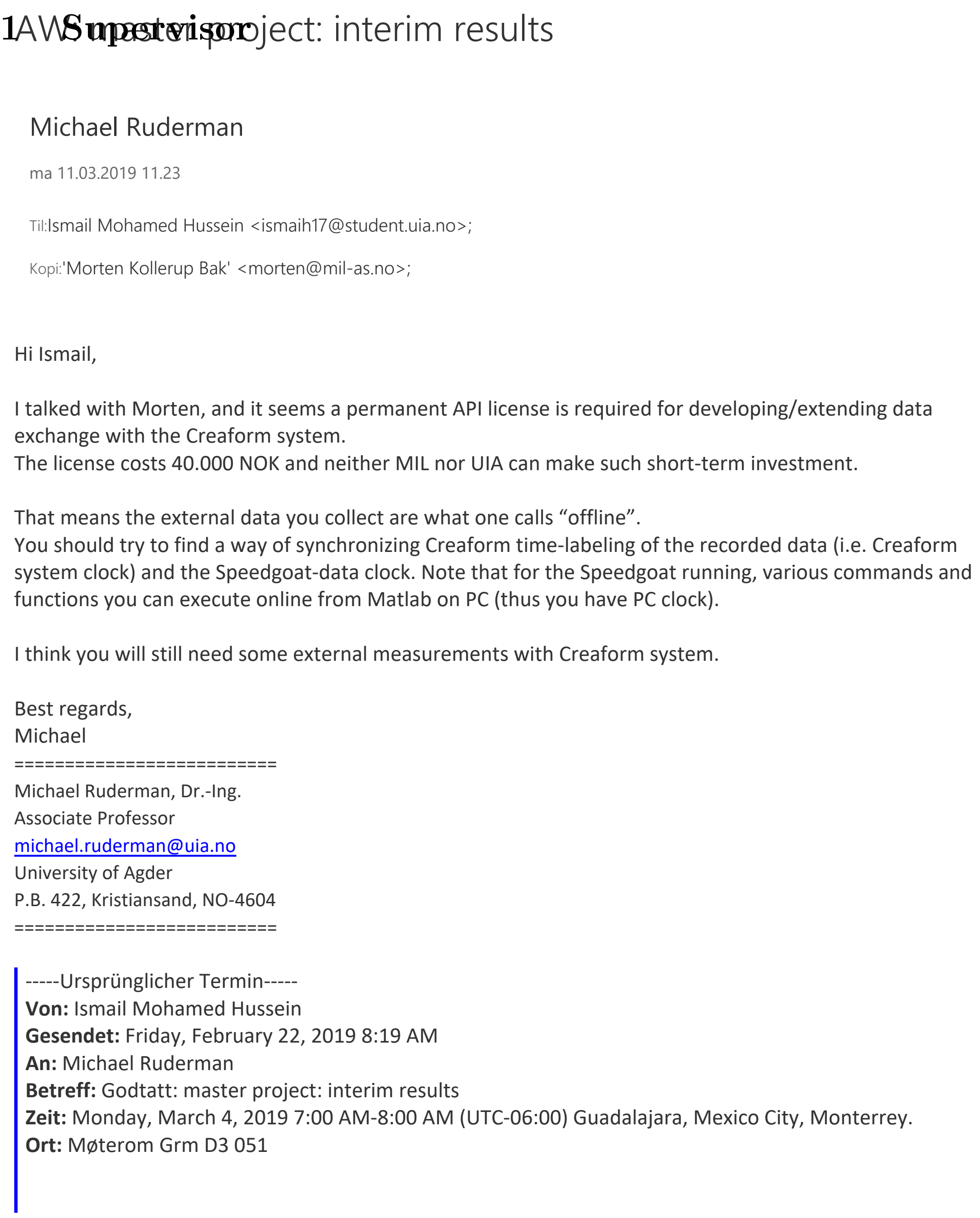
```
subplot(2,1,2)
plot(t,y,'b');grid on;
legend('y');
title("Slew Motion  ");  xlabel({'Time'})


% subplot(3,1,3)
% plot(t,xd,'r');grid on;
% legend('dx/dt');
% plot(TimePlot,P1Plot,'g');
% title(" Slew Motion velocity  ");


% Observe plot
figure('Name','Observe response ','NumberTitle','off');
subplot(3,1,1)
plot(t,u,'g');grid on
legend('u');
title("Valve ");

subplot(3,1,2)
plot(t,y0,'b');grid on;
legend('y');
title("Slew Motion   ");


subplot(3,1,3)
plot(t,x0d*C(2),'r');grid on;
legend('dy/dt');
% plot(TimePlot,P1Plot,'g');
title(" Slew Motion velocity  ");  xlabel({'Time'})
```

Published with MATLAB® R2017b

Appendix B

Communication E-mails

B.1 Supervisor



AW: master project: interim results

Michael Ruderman

ma 11.03.2019 11.23

Til:Ismail Mohamed Hussein <ismaih17@student.uia.no>;

Kopi:'Morten Kollerup Bak' <morten@mil-as.no>;

Hi Ismail,

I talked with Morten, and it seems a permanent API license is required for developing/extending data exchange with the Creaform system.
The license costs 40.000 NOK and neither MIL nor UIA can make such short-term investment.

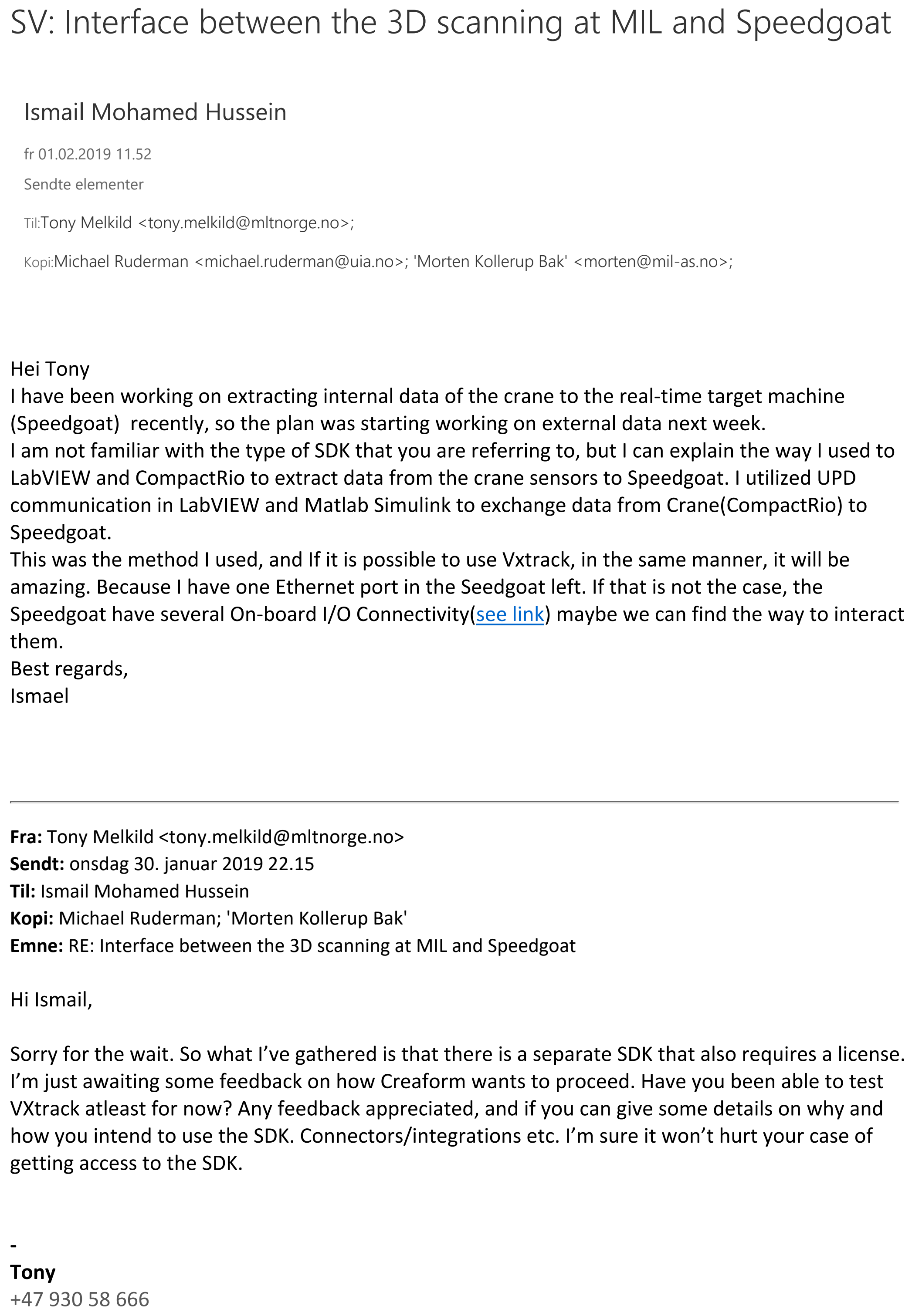
That means the external data you collect are what one calls “offline”.
You should try to find a way of synchronizing Creaform time-labeling of the recorded data (i.e. Creaform system clock) and the Speedgoat-data clock. Note that for the Speedgoat running, various commands and functions you can execute online from Matlab on PC (thus you have PC clock).

I think you will still need some external measurements with Creaform system.

Best regards,
Michael
=========================
Michael Ruderman, Dr.-Ing.
Associate Professor
michael.ruderman@uia.no
University of Agder
P.B. 422, Kristiansand, NO-4604
=========================

> -----Ursprünglicher Termin-----
> **Von:** Ismail Mohamed Hussein
> **Gesendet:** Friday, February 22, 2019 8:19 AM
> **An:** Michael Ruderman
> **Betreff:** Godtatt: master project: interim results
> **Zeit:** Monday, March 4, 2019 7:00 AM-8:00 AM (UTC-06:00) Guadalajara, Mexico City, Monterrey.
> **Ort:** Møterom Grm D3 051

B.2 MLT Norway



SV: Interface between the 3D scanning at MIL and Speedgoat

Ismail Mohamed Hussein

fr 01.02.2019 11.52

Sendte elementer

Til:Tony Melkild <tony.melkild@mltnorge.no>;

Kopi:Michael Ruderman <michael.ruderman@uia.no>; 'Morten Kollerup Bak' <morten@mil-as.no>;

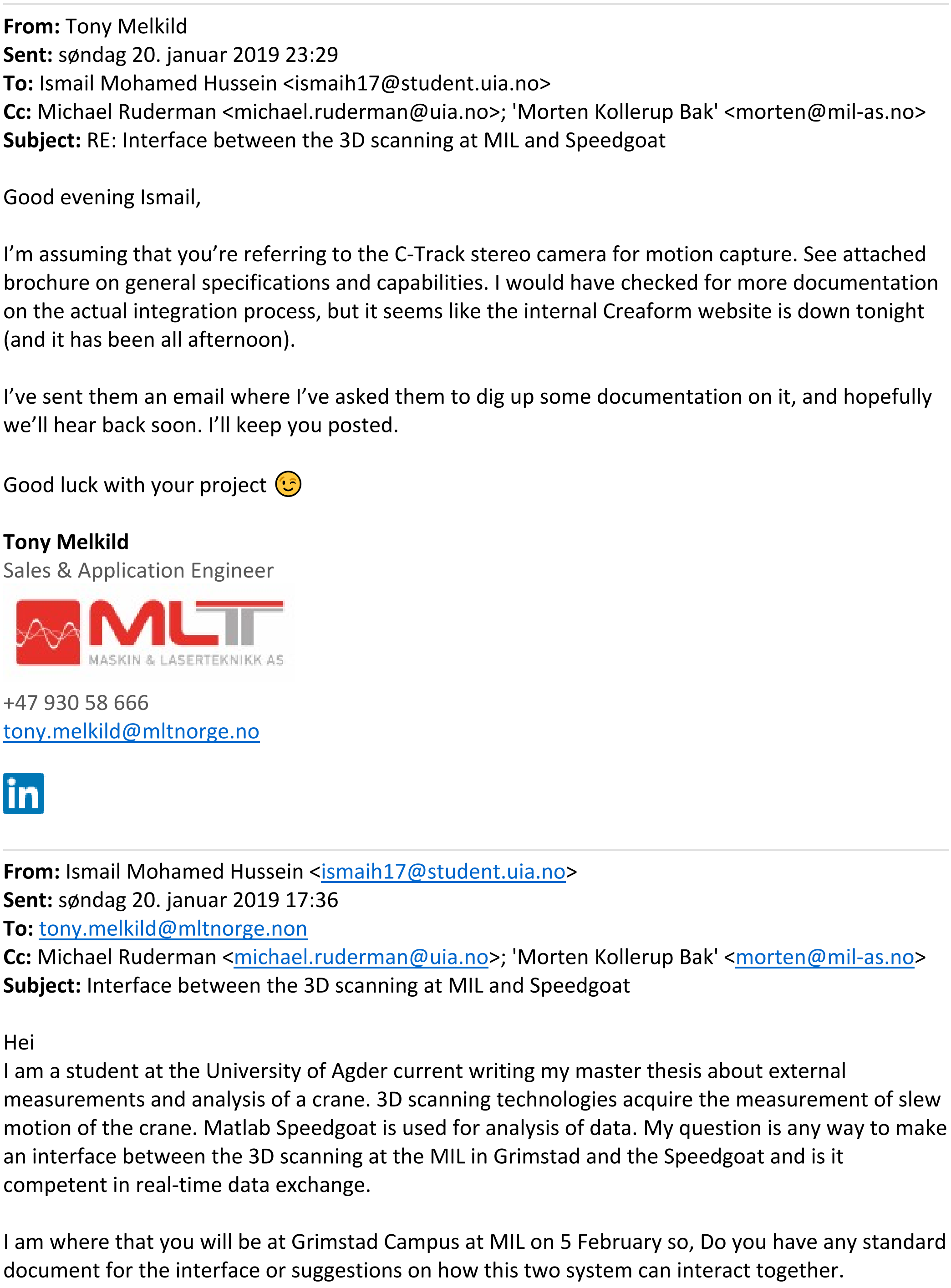
Hei Tony
I have been working on extracting internal data of the crane to the real-time target machine (Speedgoat) recently, so the plan was starting working on external data next week.
I am not familiar with the type of SDK that you are referring to, but I can explain the way I used to LabVIEW and CompactRio to extract data from the crane sensors to Speedgoat. I utilized UPD communication in LabVIEW and Matlab Simulink to exchange data from Crane(CompactRio) to Speedgoat.
This was the method I used, and If it is possible to use Vxtrack, in the same manner, it will be amazing. Because I have one Ethernet port in the Seedgoat left. If that is not the case, the Speedgoat have several On-board I/O Connectivity(see link) maybe we can find the way to interact them.
Best regards,
Ismael

Fra: Tony Melkild <tony.melkild@mltnorge.no>
Sendt: onsdag 30. januar 2019 22.15
Til: Ismail Mohamed Hussein
Kopi: Michael Ruderman; 'Morten Kollerup Bak'
Emne: RE: Interface between the 3D scanning at MIL and Speedgoat

Hi Ismail,

Sorry for the wait. So what I've gathered is that there is a separate SDK that also requires a license. I'm just awaiting some feedback on how Creaform wants to proceed. Have you been able to test VXtrack atleast for now? Any feedback appreciated, and if you can give some details on why and how you intend to use the SDK. Connectors/integrations etc. I'm sure it won't hurt your case of getting access to the SDK.

-
Tony
+47 930 58 666



From: Tony Melkild
Sent: søndag 20. januar 2019 23:29
To: Ismail Mohamed Hussein <ismaih17@student.uia.no>
Cc: Michael Ruderman <michael.ruderman@uia.no>; 'Morten Kollerup Bak' <morten@mil-as.no>
Subject: RE: Interface between the 3D scanning at MIL and Speedgoat

Good evening Ismail,

I'm assuming that you're referring to the C-Track stereo camera for motion capture. See attached brochure on general specifications and capabilities. I would have checked for more documentation on the actual integration process, but it seems like the internal Creaform website is down tonight (and it has been all afternoon).

I've sent them an email where I've asked them to dig up some documentation on it, and hopefully we'll hear back soon. I'll keep you posted.

Good luck with your project 😉

Tony Melkild
Sales & Application Engineer

MLT
MASKIN & LASERTEKNIKK AS

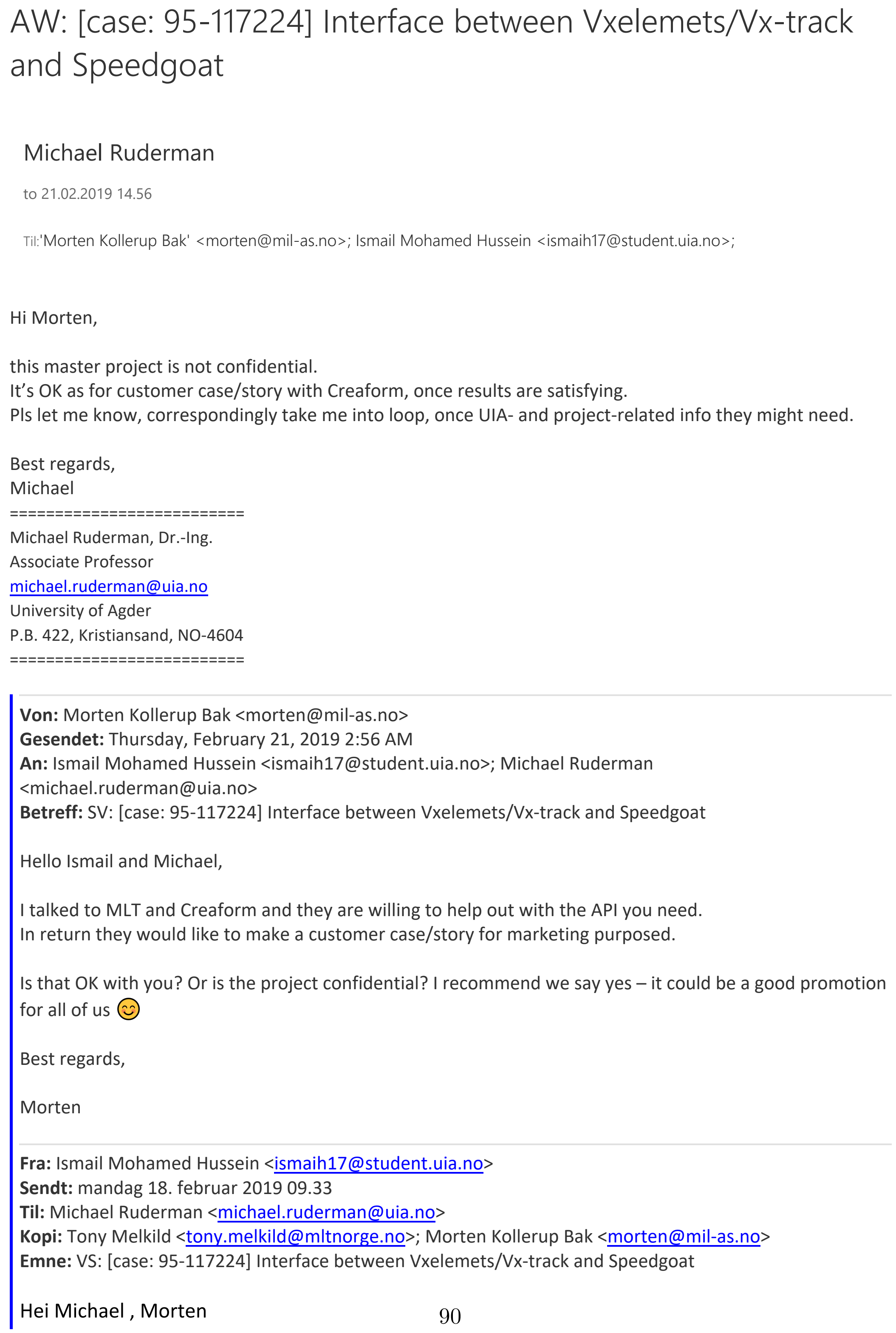
+47 930 58 666
tony.melkild@mltnorge.no

From: Ismail Mohamed Hussein <ismaih17@student.uia.no>
Sent: søndag 20. januar 2019 17:36
To: tony.melkild@mltnorge.non
Cc: Michael Ruderman <michael.ruderman@uia.no>; 'Morten Kollerup Bak' <morten@mil-as.no>
Subject: Interface between the 3D scanning at MIL and Speedgoat

Hei
I am a student at the University of Agder current writing my master thesis about external measurements and analysis of a crane. 3D scanning technologies acquire the measurement of slew motion of the crane. Matlab Speedgoat is used for analysis of data. My question is any way to make an interface between the 3D scanning at the MIL in Grimstad and the Speedgoat and is it competent in real-time data exchange.

I am where that you will be at Grimstad Campus at MIL on 5 February so, Do you have any standard document for the interface or suggestions on how this two system can interact together.

B.3 AMETEK SAS DIVISION CREAFORM France



AW: [case: 95-117224] Interface between Vxelemets/Vx-track and Speedgoat

Michael Ruderman

to 21.02.2019 14.56

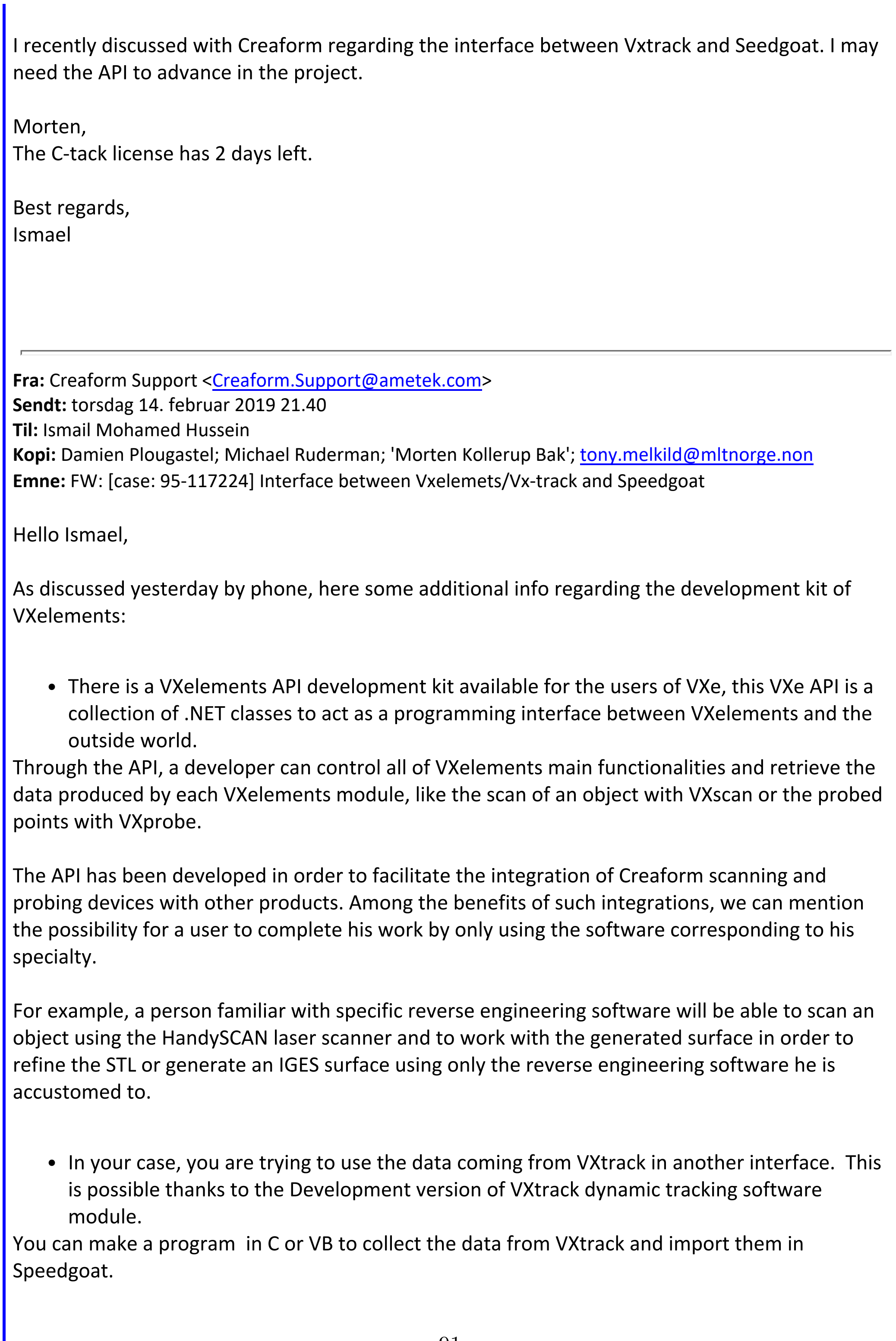
Til:'Morten Kollerup Bak' <morten@mil-as.no>; Ismail Mohamed Hussein <ismaih17@student.uia.no>;

Hi Morten,

this master project is not confidential.
It's OK as for customer case/story with Creaform, once results are satisfying.
Pls let me know, correspondingly take me into loop, once UIA- and project-related info they might need.

Best regards,
Michael
=========================
Michael Ruderman, Dr.-Ing.
Associate Professor
michael.ruderman@uia.no
University of Agder
P.B. 422, Kristiansand, NO-4604
=========================

> **Von:** Morten Kollerup Bak <morten@mil-as.no>
> **Gesendet:** Thursday, February 21, 2019 2:56 AM
> **An:** Ismail Mohamed Hussein <ismaih17@student.uia.no>; Michael Ruderman <michael.ruderman@uia.no>
> **Betreff:** SV: [case: 95-117224] Interface between Vxelemets/Vx-track and Speedgoat
>
> Hello Ismail and Michael,
>
> I talked to MLT and Creaform and they are willing to help out with the API you need.
> In return they would like to make a customer case/story for marketing purposed.
>
> Is that OK with you? Or is the project confidential? I recommend we say yes – it could be a good promotion for all of us 😊
>
> Best regards,
>
> Morten
>
> **Fra:** Ismail Mohamed Hussein <ismaih17@student.uia.no>
> **Sendt:** mandag 18. februar 2019 09.33
> **Til:** Michael Ruderman <michael.ruderman@uia.no>
> **Kopi:** Tony Melkild <tony.melkild@mltnorge.no>; Morten Kollerup Bak <morten@mil-as.no>
> **Emne:** VS: [case: 95-117224] Interface between Vxelemets/Vx-track and Speedgoat
>
> Hei Michael , Morten



I recently discussed with Creaform regarding the interface between Vxtrack and Seedgoat. I may need the API to advance in the project.

Morten,
The C-tack license has 2 days left.

Best regards,
Ismael

Fra: Creaform Support <Creaform.Support@ametek.com>
Sendt: torsdag 14. februar 2019 21.40
Til: Ismail Mohamed Hussein
Kopi: Damien Plougastel; Michael Ruderman; 'Morten Kollerup Bak'; tony.melkild@mltnorge.non
Emne: FW: [case: 95-117224] Interface between Vxelemets/Vx-track and Speedgoat

Hello Ismael,

As discussed yesterday by phone, here some additional info regarding the development kit of VXelements:

- There is a VXelements API development kit available for the users of VXe, this VXe API is a collection of .NET classes to act as a programming interface between VXelements and the outside world.

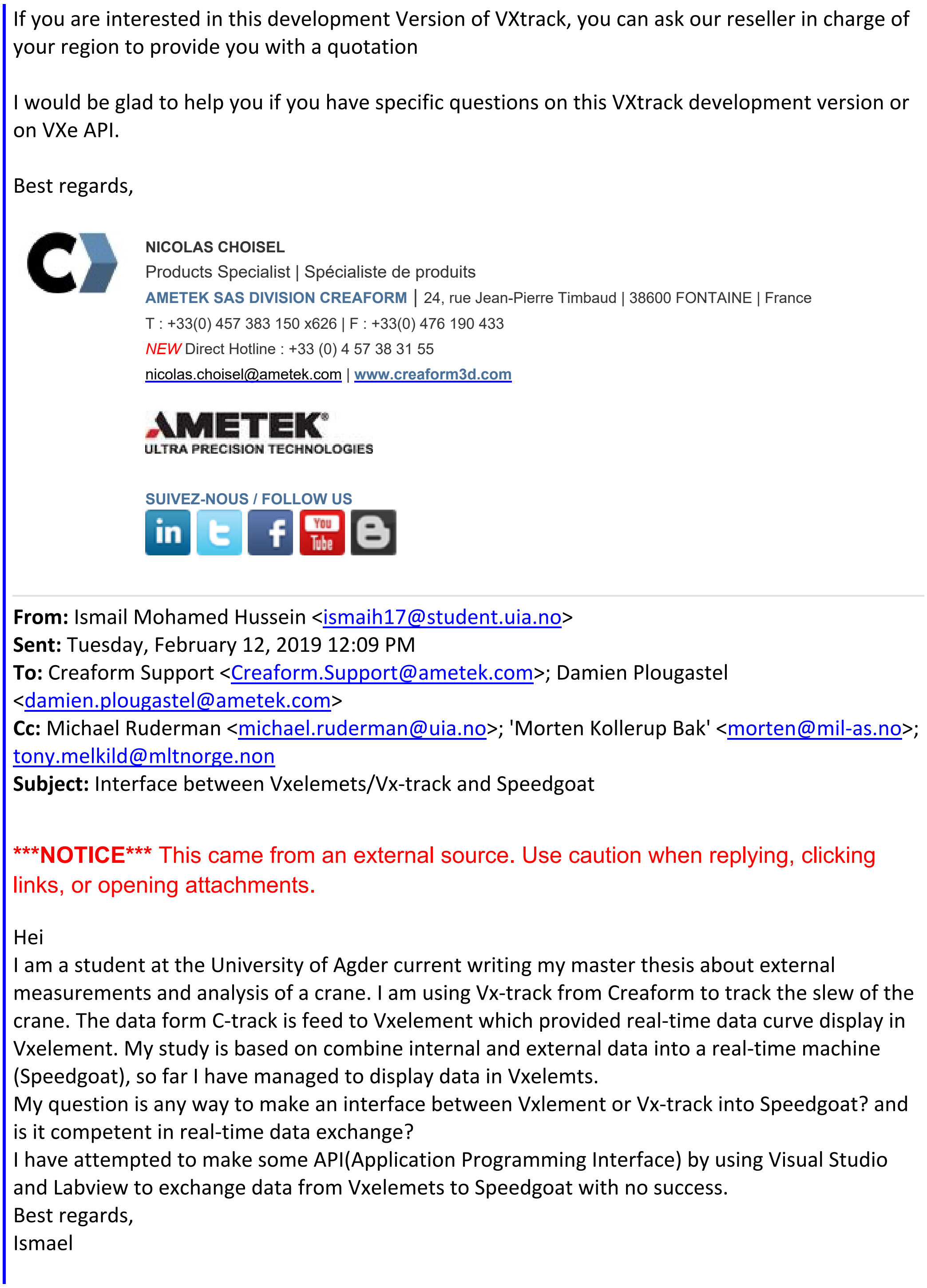
Through the API, a developer can control all of VXelements main functionalities and retrieve the data produced by each VXelements module, like the scan of an object with VXscan or the probed points with VXprobe.

The API has been developed in order to facilitate the integration of Creaform scanning and probing devices with other products. Among the benefits of such integrations, we can mention the possibility for a user to complete his work by only using the software corresponding to his specialty.

For example, a person familiar with specific reverse engineering software will be able to scan an object using the HandySCAN laser scanner and to work with the generated surface in order to refine the STL or generate an IGES surface using only the reverse engineering software he is accustomed to.

- In your case, you are trying to use the data coming from VXtrack in another interface. This is possible thanks to the Development version of VXtrack dynamic tracking software module.

You can make a program in C or VB to collect the data from VXtrack and import them in Speedgoat.



If you are interested in this development Version of VXtrack, you can ask our reseller in charge of your region to provide you with a quotation

I would be glad to help you if you have specific questions on this VXtrack development version or on VXe API.

Best regards,

NICOLAS CHOISEL
Products Specialist | Spécialiste de produits
AMETEK SAS DIVISION CREAFORM | 24, rue Jean-Pierre Timbaud | 38600 FONTAINE | France
T : +33(0) 457 383 150 x626 | F : +33(0) 476 190 433
NEW Direct Hotline : +33 (0) 4 57 38 31 55
nicolas.choisel@ametek.com | **www.creaform3d.com**

AMETEK®
ULTRA PRECISION TECHNOLOGIES

SUIVEZ-NOUS / FOLLOW US

From: Ismail Mohamed Hussein <ismaih17@student.uia.no>
Sent: Tuesday, February 12, 2019 12:09 PM
To: Creaform Support <Creaform.Support@ametek.com>; Damien Plougastel <damien.plougastel@ametek.com>
Cc: Michael Ruderman <michael.ruderman@uia.no>; 'Morten Kollerup Bak' <morten@mil-as.no>; tony.melkild@mltnorge.non
Subject: Interface between Vxelemets/Vx-track and Speedgoat

*****NOTICE***** This came from an external source. Use caution when replying, clicking links, or opening attachments.

Hei
I am a student at the University of Agder current writing my master thesis about external measurements and analysis of a crane. I am using Vx-track from Creaform to track the slew of the crane. The data form C-track is feed to Vxelement which provided real-time data curve display in Vxelement. My study is based on combine internal and external data into a real-time machine (Speedgoat), so far I have managed to display data in Vxelemts.
My question is any way to make an interface between Vxlement or Vx-track into Speedgoat? and is it competent in real-time data exchange?
I have attempted to make some API(Application Programming Interface) by using Visual Studio and Labview to exchange data from Vxelemets to Speedgoat with no success.
Best regards,
Ismael