UNIVERSITY OF AGDER

Modeling, Simulation and Control for Marine Crane Operations

Ilja Boginskis and Hmdoun Abker Ibrahim Hmdoun

Supervisor
Professor Jing Zhou

University of Agder, 2020
Faculty of Engineering and Science
Department of Engineering Sciences

Acknowledgements

We would like to express our special thanks of gratitude to our teacher professor Jing Zhou at the University of Agder who gave us the golden opportunity to do this wonderful project on the topic (Modeling and Control of Dynamics Between Crane and Vessel), which also helped us in doing a lot of Research and i came to know about so many new things I am really thankful to them. Secondly we would also like to thank our families and friends who helped us a lot in finalizing this project within the limited time frame.

Abstract

Marine cranes play an important part in offshore operations. They are expected to perform a wide range of different tasks, such as handling the load safely offshore. Nowadays, the offshore oil fields and wind farms are detected and developed, thus more and more offshore platforms are built. The scale and capacity of crane vessels are developing rapidly to fulfill the construction need of offshore platforms and offshore wind farms. Heavy crane, placed on-board vessel, is affected by vessel motions induced by environmental forces such as wind, waves and current, and vice versa. Therefore, the investigation of coupled dynamics between heavy crane and vessel, position control of vessel, and heave compensation is important and needs to be analyzed thoroughly. This master project focus on Modeling, Simulation and Control for Marine Crane Operations. The model was developed in Matlab, Simulink and Simscape Multi-body.

It is several ways to developed Control designs for an offshore crane, but in this thesis the control task only concerns position control of the crane. The main goal is to determine the most suitable controller design for this tasks.

A trajectory planning scheme is developed that can damp out unexpected payload swing. Crane end-effector is controlled to follow a trajectory planning scheme in x-direction with as small error between desired and measured position as possible. The desired position is a movement from 23.15m which is the x-coordinate when the crane arm is fully extended, to 15m.

It was created two crane models, first model in the Simulink based on crane dynamic and other model was designed by using the Simscape. Various control designs were developed with the task of controlling the end-effector position in horizontal direction, using both crane models as the plant. PID-, PI and PD-controller design were developed to perform control of end-effector in desired position.

In this project a Simscape model for the crane was devolped using tow ways one of them is using On-Shape web-side, and another is used the Simscape library. Through the project, two Simscape crane models were designed:

- A Simscape model for the crane to test and control the crane joints.
- A Simscape model for the crane with fixed cable and payload.

The Simscpae models for the crane were combined with the controllers and the trajectory planning scheme to control end-effector in x-direction.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Project Description

This project is carried out as part of the course MAS500: Master Project and is concerned with Modeling, simulation and Control for marine crane operations. As assistance, support material has been provided by the supervisor for the execution of the project, including; Literature studies of control and modeling of crane and vessel, useful kinematics and previous master thesis in [1, 2].

Figure 1.1: Crane on Vessel

1.2 Assignment Requirements

The tasks of this project are listed below

- Literature studies of control and modeling of crane and vessel.
- Dynamics modeling of coupled dynamics between heavy crane, cable and payload.
- Develop the Simscape model for crane.
- Develop a trajectory planning scheme is developed to put forward to generate a desired trajectory for the payload transportation process.
- Combine the trajectory planning scheme and PID algorithm to control the crane.

1.3 System Description

The hydraulic crane consists of three revolute joints and one extension joint. The first revolute joint is used to rotate the base and is driven by a slew mechanism. The second and third revolute joints are the main joint and jib joint respectively, which allow planar movement of the end payload when input signal is applied to the crane. The last joint is the extension joint which is used to extend the reach of the end payload. To accomplish the requirements in the assignment, it is necessary to have a controlled movement of the crane. In order to control and steer the hydraulic crane, the governing kinematics must be obtained and design a control system. Fig. 1.2 shows the crane bodies.

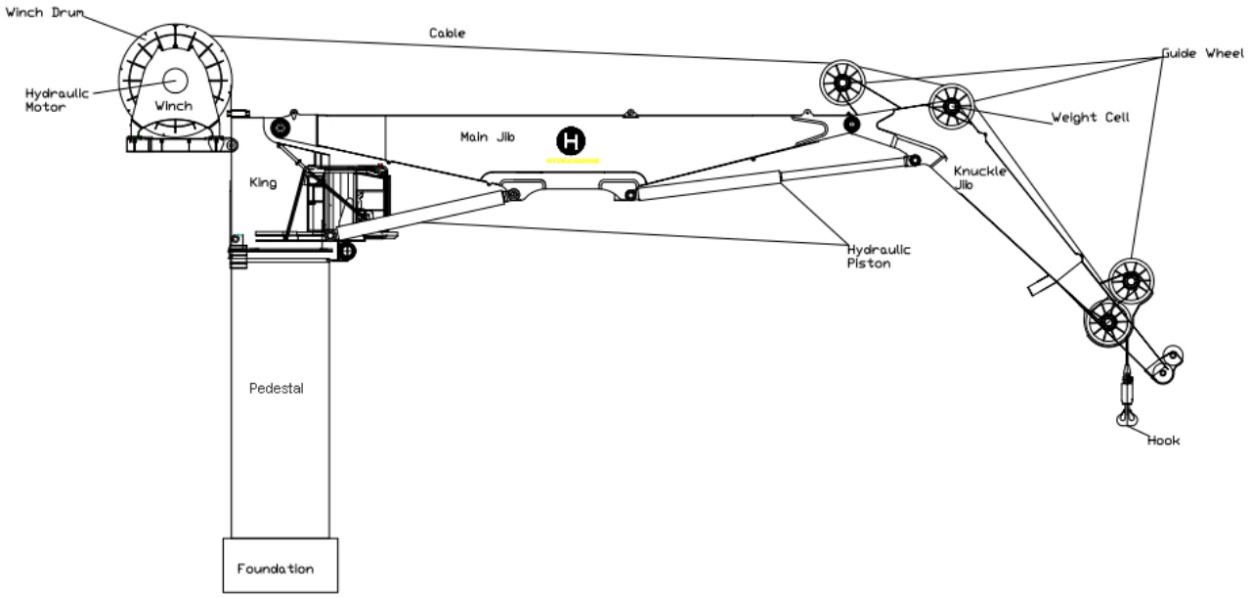

Figure 1.2: Drawing of the knuckle jib crane

1.4 Technical Approach

In Fig. 1.3, the strategy of approaching this project is illustrated as a flowchart. The first step will be to generate a model of the system and then a controller is modeled in Simulink. The second step is to tune the simulation model. When simulation performance and parameter tuning is satisfactory, the control structure is used in virtual reality model (VRML) of the crane. This is a replica of the actual crane. This process is iterated for the duration of the project.

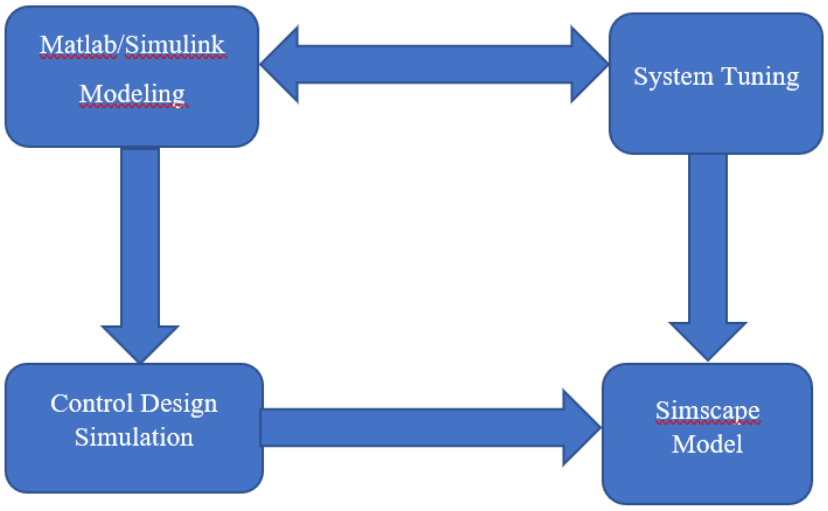

Figure 1.3: Illustration of workflow

1.5 Outline of Report

Chapter 1 - Introduction:
This chapter includes project description, assignment requirements, system description, technical approach, outline of the report and software that used in this project.

Chapter 2 - Literature review:
In this chapter all literature in this thesis are summarized.

Chapter 3 - Preliminary theory of crane modeling and control:
This chapter includes the theory used when developing the crane model is covered.

Chapter 4 - Modeling of the crane:
This chapter covers the development of crane model.

Chapter 5 - Control of the crane:
This chapter includes different system controllers which were used in this project.

Chapter 6 - Simulation results:
This chapter covers the results from the crane model simulation both Simulink model and Simscape model.

Chapter 7 - Conclusion:
This chapter includes conclution of the project.

1.6 Software Used During The Project

There are several software used to carry out this project, these software are Solidworks, Matlab, Simulink and Simscape SimMechanics. This section includes briefs about each software as follow:

1.6.1 SolidWorks

SolidWorks is computer-aided design (CAD) software owned by Dassault Systemes. It uses the principle of parametric design and generates three kinds of interconnected files: the part, the assembly, and the drawing. Solidworks helps you perform 2D and 3D modelling.

Figure 1.4: SolidWorks Logo

1.6.2 Matlab

Matlab is a high-performance language for technical computing. It integrates computation, visualization, and programming in an easy-to-use environment where problems and solutions are expressed in familiar mathematical notation. Typical uses include: ... Data analysis, exploration, and visualization.

Figure 1.5: Matlab Logo

1.6.3 Simulink

Simulink, an add-on product to MATLAB, provides an interactive, graphical environment for modeling, simulating, and analyzing of dynamic systems. It enables rapid construction of virtual prototypes to explore design concepts at any level of detail with minimal effort.

Figure 1.6: Simulink Logo

1.6.4 Simscape SimMechanics

Simscape Multibody (formerly SimMechanics) provides a multibody simulation environment for 3D mechanical systems, such as robots, vehicle suspensions, construction equipment, and aircraft landing gear. You can model multibody systems using blocks representing bodies, joints, constraints, force elements, and sensors.

Chapter 2

Literature Review

Different literature have been used in this project, this chapter includes a review of these literature which are related to the plan trajectory, modelling, simulation and control of the offshore cranes. In this these papers there is an explanation of methods of the path trajectory, the offshore cranes kinematics and dynamics, as well as the control system for offshore cranes.

2.1 Trajectory Planning Method for Overhead Cranes

The paper [5] is written by Ning Sun, Yongchun Fang, Yudong Zhang and Bojun Ma, and the tittle of this paper is a novel kinematic coupling based trajectory planning method for overhead cranes. The writers have tried to achieve a smooth trolley motion and payload swing, a kinematic coupling-based offline trajectory planning method is proposed for 2-D overhead cranes. Specifically, to prevent unexpected payload swing. As shown in figure2.1 an antiswing mechanism is first introduced into an S-shape reference trajectory based on accurate analysis for the coupling behavior between the payload and the trolley. After that, the combined trajectory is further tuned through a novel iterative learning strategy , which guarantees accurate trolley position. The performance of the proposed trajectory is proven by Lyapunov techniques and Barbalat's Lemmas. In this project the same method is used for the offshore cranes.

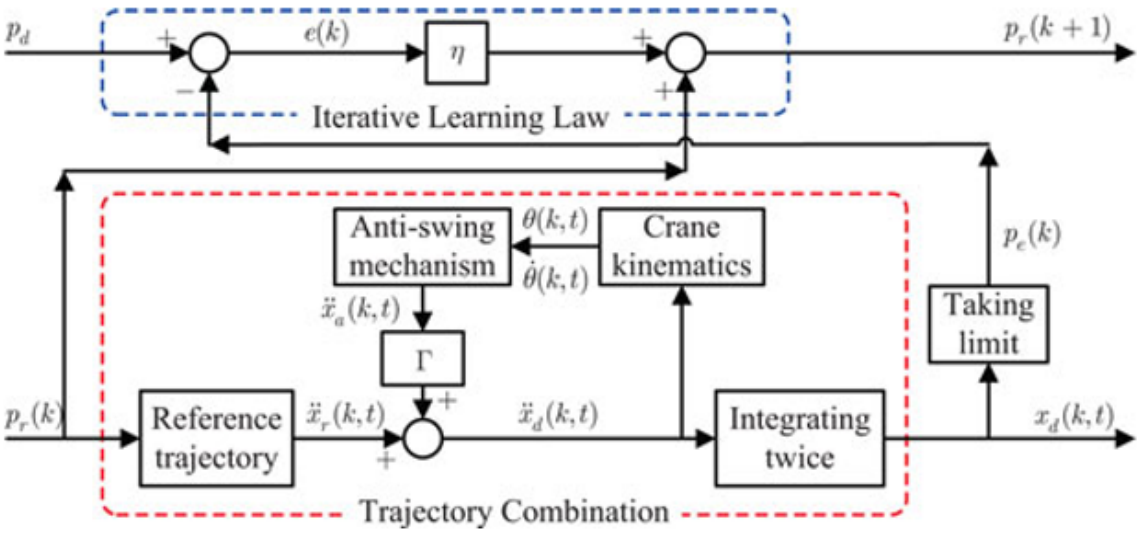

Figure 2.1: Block diagram of the overall trajectory planning process

2.2 Control of Ship Mounted Cranes

Paper [4] shows that the past several decades of the control problem for land-fixed cranes has been deeply studied, and a lot of solutions for both linear and nonlinear control methods have been reported and many control strategies for land-fixed cranes have already been developed. As opposed to the land-fixed cranes, which are fixed and operated in the inertia, the ship mounted cranes present much more complicated nonlinear dynamical characteristics and they are persistently influenced by different mismatched disturbances due to harsh sea environments, e.g, sea waves, ocean current, sea winds and so forth; these unfavorable factors bring about many challenges for the development of effective control schemes. Due to those rough working conditions and the influence by various external disturbance the control problem for offshore cranes is highly demanding when the intention of the control is to place a payload precisely and smoothly to a desired location as fast as possible, without making the payload swing during the operation and avoid residual swing at the end. Writers looked at other literature of work denoted to the control of ship mounted cranes and found some control strategies that were already developed.

In the literature, there are some innovative works devoted to the control of ship-mounted cranes. In particular, for ship mounted cranes with the well known (Maryland rigging) a feedforward control strategy is designed with gain scheduling to suppress the cargo swing caused by ship roll movements. Later on, Al-Sweiti and Söffker derived the mathematical model of an elastic Maryland rigged ship mounted crane and then develop a novel control scheme consisting of a variable-gain observer and a variable-gain controller, which is demonstrated to be effective. In addition, there are also some other meaningful works on both dynamics analysis and advanced controller design, including predictionbased control, preview tracking control, nonlinear feedback control, sliding mode control (SMC), composite control, linear matrix inequality-based control, delayed feedback control, active rate-based control, combination-based control, external modelbased control, and so on.

They found some advantages of using these control strategies, for instance that the control scheme with variable-gain observer and variable-gain controller is proved to be effective and that two nonlinear sliding mod controllers are developed and are proved to be and robust. Despite these advantages, most of these control strategies are based on simplified or reduced crane dynamics, which may cause system instability because it is difficult to avoid a swinging payload, due to the complicated working scenario of an offshore crane.

The paper presents a control design for offshore cranes based on the original nonlinear dynamics of the crane without any simplifications or approximations. It presents how the dynamics is transformed into a form that is more practical for such an approach, where the new control variables are defined. The Lyapunov control law and a closed loop stability analysis are provided, and as far as they know this paper produces the first closed loop control method which attain asymptotic results for an offshore crane, affected by ship roll and heave movement, without needing linearization and approximation of the original nonlinear dynamics. Tis method is compared to the existing method using Matlab/Simulink RTWT to verify that the performance of such a control method is better and more robust against external disturbances than the other control methods.

2.3 Modeling and Control for Rotary Crane Using Matlab/Simulink

In this literature rotary crane has been studied, rotary crane carry a payload from one position to another position. The cart and jib of the crane start to accelerate in linear and rotational motions respectively when input signal to the crane system is applied. This will cause swinging or swaying of the payload. Therefore an antiswing for the crane is proposed in paper [6]. This paper presents the modeling and intelligent control system design for a rotary crane. The modeling and simulation was done using Matlab Simsacpe toolbox which is physical approach of modeling. The mathematical model of the crane was also derived for comparison. The intelligent control system is implemented as Fuzzy-PID controller. The physical modeling approach using Matlab Simsacpe toolbox can ease modeling process since mathematical modeling results in unknown parameters. Jib load system has been discussed in this paper.

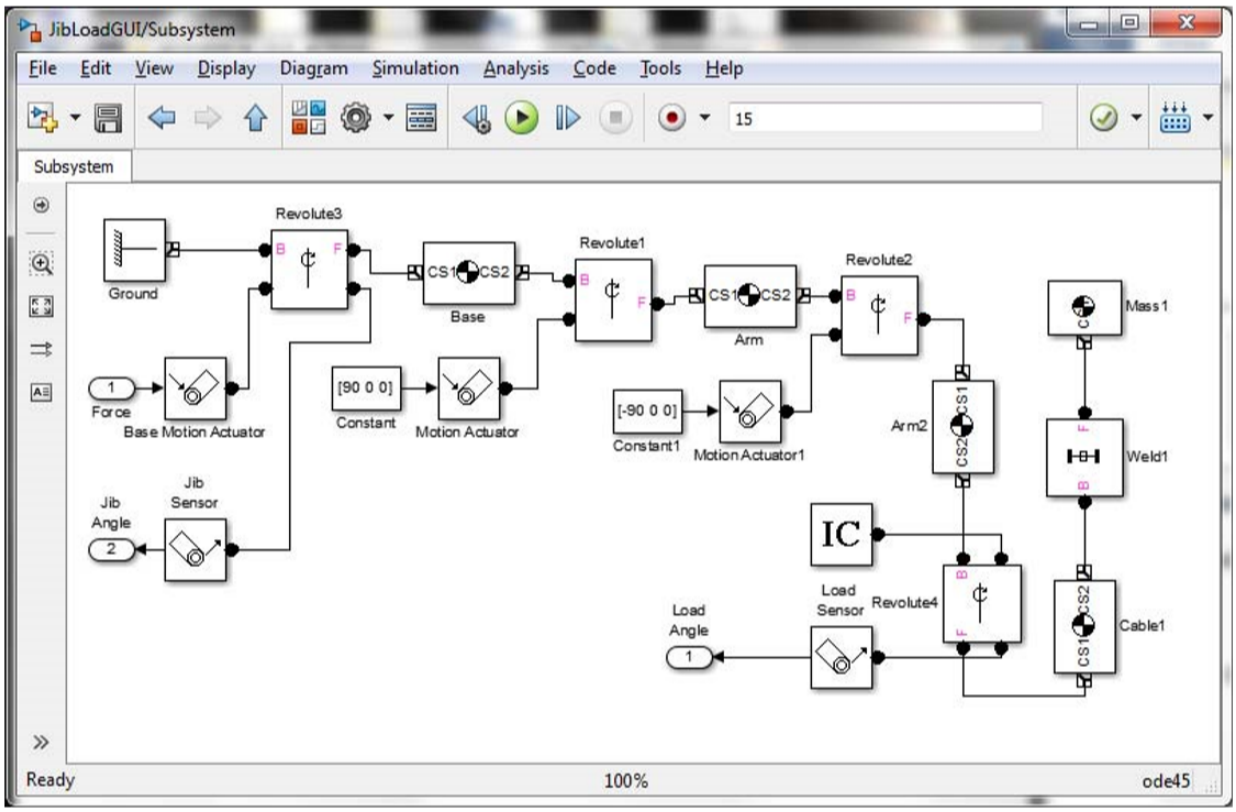

Figure 2.2: Simscape Model of The Jib-load System

Chapter 3

Preliminary theory of crane modeling and control

The purpose of this chapter is to present and define the theoretical expressions that are central to crane modeling and control. In this chapter, the theory of crane modeling is used to help develop kinematics and dynamics equations. Part of the task is to advance the theory behind different control methods such as PID, PI and PD controls for crane.

3.1 Kinematics

3.1.1 Forward and Inverse kinematics

Forward kinematics is the method of calculating the motion of an end-effector from dimensions and states of the system on which it is mounted.

Inverse kinematics is the opposite of forward kinematics. It uses the kinematic equations to calculate the motion of the joints from the motion of the end-effector.

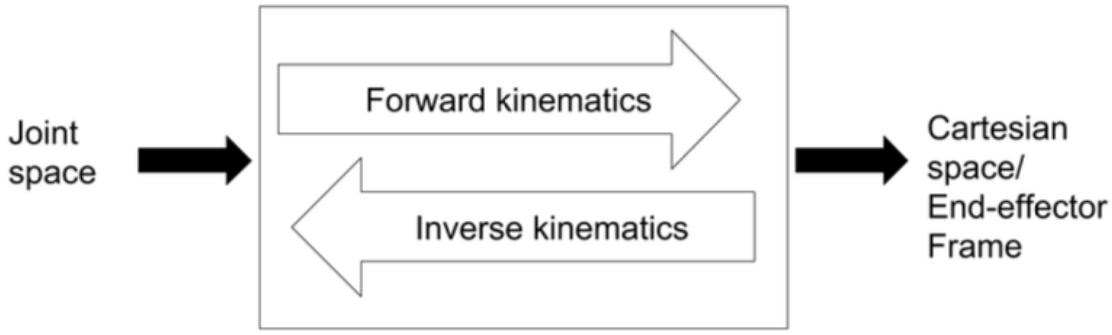

Figure 3.1: Forward and inverse kinematics

From Figure 3.1, we can distinguish between forward kinematics and inverse kinematics.

3.1.2 The Denavit-Hartenberg Convention

The theory presented in this chapter is based on [2].

Jacques Denavit and Richard Hartenberg created a standardized form for finding the forward kinematics of a robot system of succeeding joints and links. This method uses the length of the links, and the angles of the joints to find the position of the end-effector on the end of the last link. The method uses the local coordinate frames of each joint to "travel" through the links towards the tip.

The method is performed by first setting up a DH-table, for then to use this to find the transformation matrix needed to identify the coordinates of the tip. The DH-table is made as shown in table 3.1.

Joint	**Rot Z**	**Trans Z**	**Rot X**	**Trans X**
1	θ_{Z1}	L_{Z1}	θ_{X1}	L_{X1}
2	θ_{Z2}	L_{Z2}	θ_{X2}	L_{X2}
3	θ_{Z3}	L_{Z3}	θ_{X3}	L_{X3}

Table 3.1: DH-table

In the DH-table, every joint get its own row. In the second column, the rotation of the joint about the z-axis is input. In the third column, the length of the following link in z direction is input (for when the rotation is 0°). In the fourth column, the length of the following link in x direction is given (when the rotation is 0°). In the fth column, the rotation of the joint about the x-axis is input. An illustration corresponding to the DH-table is shown in figure 3.2.

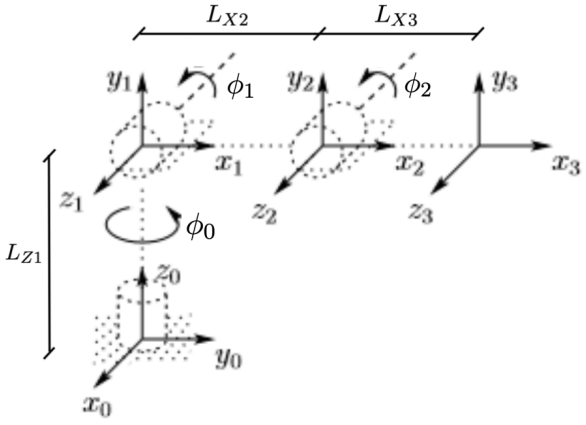

Figure 3.2: System illustration for DH table.

When the DH-table is completed, the values are set into rotation matrices and translation matrices, for then to be combined into a transformation matrix. There is made one transformation matrix for each of the joints, which is found equally for all joints using equation 3.1.

$$A_n^{n-1} = Rot_Z(\theta_{Zn}) \cdot Trans_Z(L_{Zn}) \cdot Rot_X(\theta_{Xn}) \cdot Trans_X(L_{Xn}) \tag{3.1}$$

where $Rot_m(\theta)$ is a rotational matrix using θ as the angle, and $Trans_m(L)$ is a translational matrix using L as the input.

The rotational matrices for rotation about the X, Y, and Z-axes are shown in equations 3.2, 3.3, and 3.4 respectively.

$$Rot_X(\theta_X) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\theta_X) & -\sin(\theta_X) & 0 \\ 0 & \sin(\theta_X) & \cos(\theta_X) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.2}$$

$$Rot_Y(\theta_Y) = \begin{bmatrix} \cos(\theta_Y) & 0 & \sin(\theta_Y) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(\theta_Y) & 0 & \cos(\theta_Y) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.3}$$

$$Rot_Z(\theta) = \begin{bmatrix} \cos(\theta_Z) & -\sin(\theta_Z) & 0 & 0 \\ \sin(\theta_Z) & \cos(\theta_Z) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.4}$$

where θ is the angle of the rotation.

The translational matrix is shown in equation 3.5.

$$Trans_m(L_m) = \begin{bmatrix} 1 & 0 & 0 & L_X \\ 0 & 1 & 0 & L_Y \\ 0 & 0 & 1 & L_Z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.5}$$

where L_m is the translation for the m-axis.

The transformation matrix between the origin of the system and the tip of the outer link is found by:

$$H_3^0 = A_1^0 A_2^1 A_3^2 \tag{3.6}$$

3.1.3 Velocity - The Jacobian

A moving reference frame have both linear and angular velocity components. The manipulator Jacobian relates the link angular and linear velocity with the joint velocities. It can be split into one that relates the joint velocity of link i with the linear velocity v_i, and the other relates the joints angles to the angular velocity w_i
The global velocity for link i may be written

$$v_i = j_{v1}\dot{q_1} + ... + j_{vi}\dot{q_i} = J_v\dot{q} \tag{3.7}$$
$$w_i = j_{w1}\dot{q_1} + ... + j_{wi}\dot{q_i} = J_w\dot{q} \tag{3.8}$$

Where J_v is the linear Jacobian and J_w is the angular Jacobian. The $6 \times n$ Jacobian to consider are then

$$J_v = \begin{bmatrix} j_{v1} & ... & j_{vi} & 0 & ... & 0 \end{bmatrix} \tag{3.9}$$
$$J_w = \begin{bmatrix} j_{w1} & ... & j_{wi} & 0 & ... & 0 \end{bmatrix} \tag{3.10}$$

The total Jacobian for a n-link manipulator can then be written as

$$J_i = \begin{bmatrix} J_{vi} \\ J_{wi} \end{bmatrix} = \begin{cases} \begin{bmatrix} z_{i-1} \times (r_n - r_{i-1}) \\ z_{i-1} \end{bmatrix} & \text{revolute joint} \\ \begin{bmatrix} z_{i-1} \\ 0 \end{bmatrix} & \text{prismatic joint} \end{cases} \tag{3.11}$$

Here, the 3×1 vector z_{i-1} is given by the the first three elements in the third column of H_{i-1}^0, while r_{i-1} is given by the first three elements of the fourth column of H_{i-1}^0.

3.2 Dynamics

Kinematics describes just the motion of a robot manipulator without considering the torque applied and dynamics describes the relationship between torque and motion.

Information and equations regarding the dynamics of a robot manipulator is collected from [1]

3.2.1 Lagrange's Approach

Lagrange's Approach offers a systematic way to formulate the equations of the motion of a mechanical system or a (flexible) structural system with multiple degrees of freedom. A scalar approach is obtained by expressing the scalar quantities of kinetic and potential energy in terms of generalized coordinates. The Lagrangian $\mathcal{L}$ of a mechanical system is described as the difference between the kinetic and the potential energy of the system, it can be calculated from following equation:

$$\mathcal{L} = \mathcal{K} - \mathcal{P} \tag{3.12}$$

Where:
$\mathcal{K}$ and $\mathcal{P}$ are the total kinetic and potential energy.
The use of the Lagrange's equations for any type of mechanical system leads to a system of (n) coupled, second nonlinear ordinary differential equations given by:

$$\frac{d}{dt}\frac{\partial \mathcal{L}}{\partial \dot{q}_i} - \frac{\partial \mathcal{L}}{\partial q_i} = \tau_i \qquad\qquad i = 1, \ldots, i \tag{3.13}$$

Where:
τ_i = is the force associated with link i.

In the equation above we see the relation between the force applied to each joint and joint positions, velocities and accelerations, so it can be possible to drive the dynamic model using kinetic and potential energy of the system.

3.2.2 Kinetic Energy

The total kinetic energy of a n-link manipulator is given by the sum of contribution of kinetic energy relative to the motion of each link and can be written as

$$\mathcal{K} = \sum_{i=1}^{n} \mathcal{K}_i = \sum_{i=1}^{n} (\frac{1}{2} m_i v_i^T v_i + \frac{1}{2} w_i^T I_i w_i) \tag{3.14}$$

Where
m_i: is the total mass of each object.

I_i: is the Inertia tensor given by a 3×3 matrix. It is important to express the inertia tensor in the inertial frame to make it possible to compute the triple product $w_i^T I_i w_i$. This is done in terms of the orientation transformation between the body attached frame and the inertial frame. The Inertia tensor can be written as $R_i I_i R_i^T$.

v_i and w_i: is the linear and angular velocity vector which can be expressed by utilization of the Jacobian matrix and the derivative of the joint angles, and since the joint variables are the generalized coordinates, the linear and angular velocity can be written as $J_{vi}(q)\dot{q}$ and $J_{wi}(q)\dot{q}$
.
By inserting Equation for $\mathcal{K}_i$ I_i, v_i and w_i in Equation 3.16, the total kinetic energy for a n-link robot manipulator can be written as

$$\mathcal{K} = \frac{1}{2} \dot{q}^T \sum_{i=1}^{n} [m_i J_{vi}(q)^T J_{vi}(q) + J_{wi}(q)^T R_i(q) I_i R_i(q)^T J_{wi}(q)] \dot{q} = \frac{1}{2} \dot{q}^T M(q) \dot{q} \tag{3.15}$$

where $M(q)$ is the inertia matrix, and is given by

$$M = m_i J_{vi}(q)^T J_{vi}(q) + J_{wi}(q)^T R_i(q) I_i R_i(q)^T J_{wi}(q) \tag{3.16}$$

3.2.3 Potential Energy

The total potential energy of a n-link manipulator is given by the sum of contribution of kinetic energy relative to the motion of each link and can be written as

$$\mathcal{P} = \sum_{i=1}^{n} \mathcal{P}_i = \sum_{i=1}^{n} g^T r_{ci} m_i \tag{3.17}$$

where:

g: is a 3×1 gravity acceleration vector in the inertial frame. If the z-axis is defined as the vertical axis the gravity acceleration vector can be written as $g = [0 \quad 0 \quad -9.81 m/s^2]^T$

r_{ci}: is the vector of the center of mass of link i.

3.2.4 Equations of Motion

It is necessary to specialized the Lagrange's equation **??** before driving the equations of motion. The kinetic energy can be written as a quadratic function of $\dot{q}$ in the form:

$$\mathcal{K} = \frac{1}{2} \sum_{i,j}^{n} M_{i,j}(q) \dot{q}_i \dot{q}_j = \frac{1}{2} \dot{q}^T M(q) \dot{q} \tag{3.18}$$

Since the potential energy is independent of $\dot{q}$, so it can be written as:

$$\mathcal{P} = \mathcal{P}(q) \tag{3.19}$$

When substituting equation 3.18 and equation 3.19 into equation 3.12 we obtain:

$$\mathcal{L} = \mathcal{K} - \mathcal{P} = \frac{1}{2}\sum_{i,j}^{n} M_{i,j}\,(q)\dot{q}_i\dot{q}_j - \mathcal{P}(q) = \frac{1}{2}\dot{q}^T M(q)\dot{q} - \mathcal{P}(q) \tag{3.20}$$

When solving equation 3.12 with respect to equation 3.20 the Lagrange's equations can be written as:

$$\sum_j M_{kj}\ddot{q} + \sum_{i,j}(\frac{\partial M_{kj}}{\partial q_i} - \frac{1}{2}\frac{\partial M_{ij}}{\partial q_k})\dot{q}_i\dot{q}_j - \frac{\partial \mathcal{P}}{\partial q_k} = \tau_k \tag{3.21}$$

Further, by interchanging the order of the summation and use symmetry we can get the following Lagrange's equations:

$$\sum_j M_{kj}\ddot{q} + \sum_{i,j} C_{ijk}(q)\dot{q}_i\dot{q}_j + g_k(q) = \tau_k \qquad\qquad k = 1, \ldots, n \tag{3.22}$$

Where:
C_{ijk} is the Christoffel symbols and can be calculated from the following equation:

$$C_{ijk} = \frac{1}{2}(\frac{\partial M_{kj}}{\partial \dot{q}_i} + \frac{\partial M_{kj}}{\partial \dot{q}_j} - \frac{\partial M_{kj}}{\partial \dot{q}_k}) \tag{3.23}$$

and

$$g_k = \frac{\partial \mathcal{P}}{\partial \dot{q}_k} \tag{3.24}$$

Rewrite the equations of motion in the matrix form as follow:

$$M(q)\ddot{q} + C(\dot{q}, q)\dot{q} + g(q) = \tau\frac{\partial \mathcal{P}}{\partial \dot{q}_k} \tag{3.25}$$

Where: $C(\dot{q}, q)$ is the Coriolis and Centripetal matrix. The j,k-th matrix is defined as:

$$C_{kj} = \sum_{i=1}^{n} C_{ijk}(q)q_i \tag{3.26}$$

and $M(q)$ is the inertia matrix given by equation 3.16, and g_q is the gravity vector and given by the equation 3.24

3.3 Control Theory

A controller is designed to control the behavior of dynamical systems. This section is about control theory and presents a description of different control methods. In this project a feedback controllers as PID, PI, PD and P are used, explanation and structure of the control methods are shown in the block diagram and equations. In figure3.3 you can see the concept of the feedback controller. These

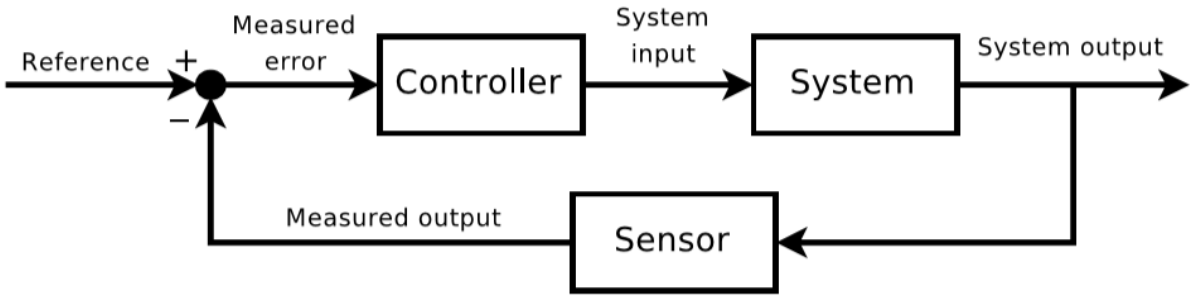

Figure 3.3: The Concept of the Feedback Controller

equations descries the PID,PI, PD and P controllers, and all controllers parameters are described in this section.

3.3.1 PID-controller

The PID controller3.4 is probably the most used feedback control design. PID ia an initialism for Proportional, Integral and Derivative, referring to the three terms operating on the error signal to produce a control signal. If u(t) is the control signal sent to the system, y(t) is the measured output and r(t) is the desired output, and tracking error $e(t) = r(t) - y(t)$, a PID controller has the general form:

$$u(t) = K_p e(t) + K_I \int e(t)dt + K_D \frac{d}{dt} e(t) \tag{3.27}$$

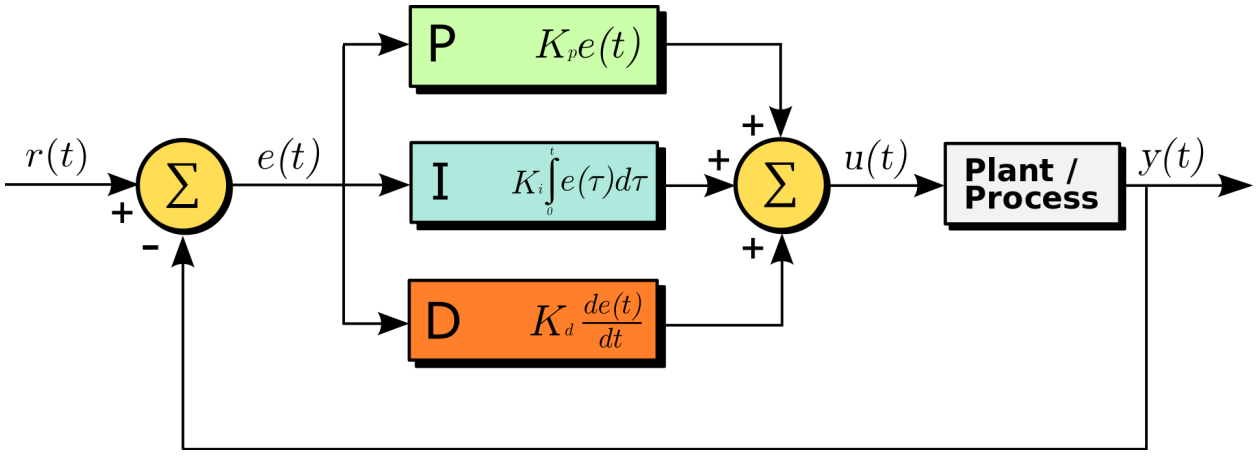

Figure 3.4: The PID Controller

The desired closed loop dynamics is obtained by adjusting the three parameters K_P, K_I and K_D, often iteratively by "tuning" and without specific knowledge of a plant model. Stability can often be ensured using only the proportional term. The integral term permits the rejection of a step disturbance (often a striking specification in process control). The derivative term is used to provide damping or shaping of the response, there are other effects of each controller parameters for a close loop system it shows in table3.2. PID controllers are the most well established class of control systems: however, they cannot be used in several more complicated cases, especially if MIMO systems are considered.

CL Response	**Rise Time**	**Overshoot**	**Settling Time**	**S-S Error**
K_p	Decrease	Increase	Small Change	Decrease
K_i	Decrease	Increase	Increase	Decrease
K_d	Small Change	Decrease	Decrease	No Change

Table 3.2: Effects of Controller Parameters

Applying Laplace transformation results in the transformed PID controller equation:

$$u(s) = K_p e(s) + K_I \frac{1}{s} e(s) + K_D s e(s) \tag{3.28}$$

$$u(s) = (K_p + K_I \frac{1}{s} + K_D s) e(s) \tag{3.29}$$

With the PID controller transfer function:

$$C(s) = (K_p + K_I \frac{1}{s} + K_D s) \tag{3.30}$$

The PID controller transfer function in series form:

$$C(s) = K(1 + \frac{1}{sT_i})(1 + sT_d) \tag{3.31}$$

3.3.2 P-controller

A variation of Proportional Integral Derivative (PID) control is to use only the proportional term as a P-only control. Figure3.5 shows the proportional controller. The output of the controller can be found from the equation:

$$u(t) = K_p e(t) \tag{3.32}$$

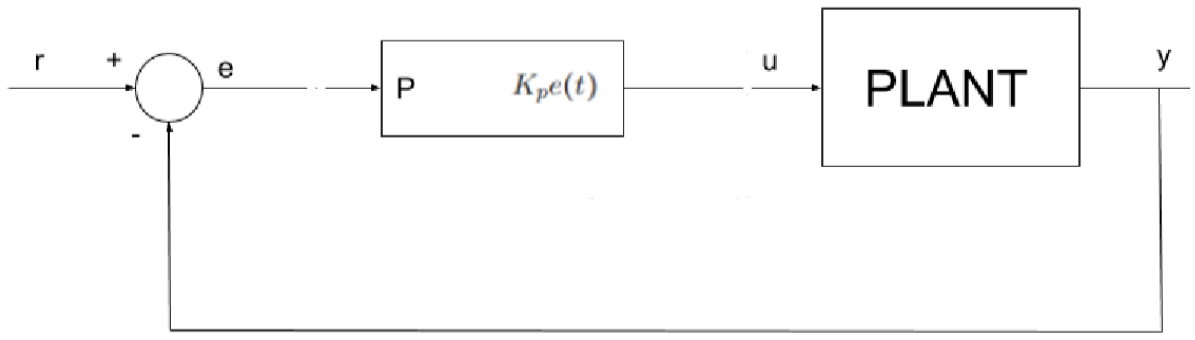

Figure 3.5: The P Controller

A very important application of proportional controller with fixed bias or offset id the zero load process. It means the dynamic characteristics of process will not get any disturbance even if there is no flow through the controller for small duration. Advantages are: easy to implement, low coast and easy to tune the proportional constant. Disadvantages are: responds only change in error, error can be reduced to zero(i.e controlled value cannot reach set point) and fine controlling is not possible.

3.3.3 PI-controller

A Proportional-Integral controller (PI controller) includes a proportional gain and integral gain see figure3.6

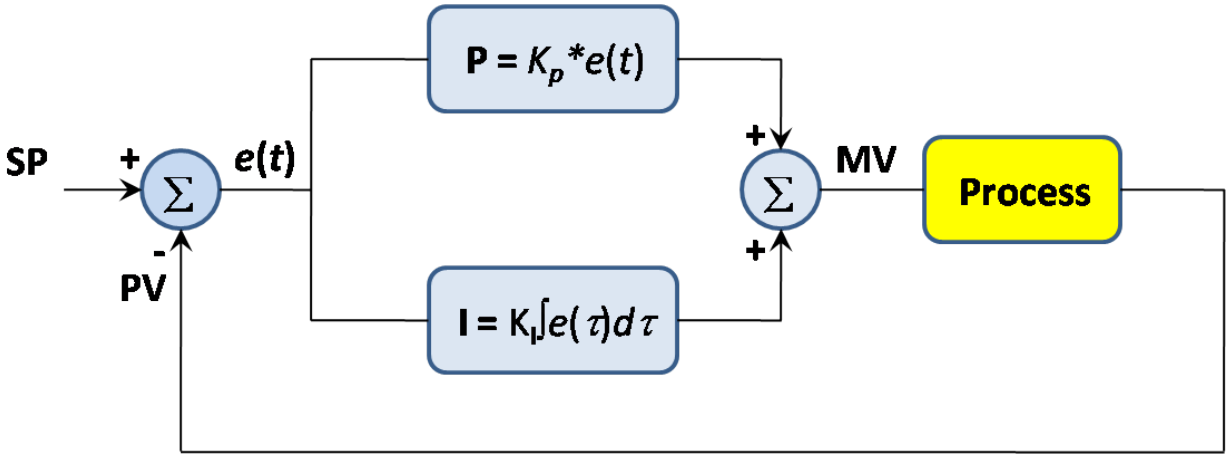

Figure 3.6: The PI Controller

The PI controller has a general form as:

$$u(t) = K_p e(t) + K_I \int e(t)dt \tag{3.33}$$

Applying Laplace transformation results in the transformed PI controller equation:

$$u(s) = K_p e(s) + K_I \frac{1}{s} e(s) \tag{3.34}$$

$$u(s) = (K_p + K_I \frac{1}{s}) \tag{3.35}$$

With the PI controller transfer function:

$$C(s) = \frac{K_p s + K_i}{s} \tag{3.36}$$

The PID controller transfer function in series form:

$$C(s) = K(1 + \frac{1}{T_i s}) \tag{3.37}$$

Advantage of the PI controller is to eliminate the steady state error from the proportional controller, disadvantage of this type of controller is the integral part has a negative effect on the speed of the response and stability of the system, for this reason this controller use when the speed of system response is not a problem.

3.3.4 PD-controller

A PD controller is a controller consist of proportional gain and derivative gain as shown in figure3.7

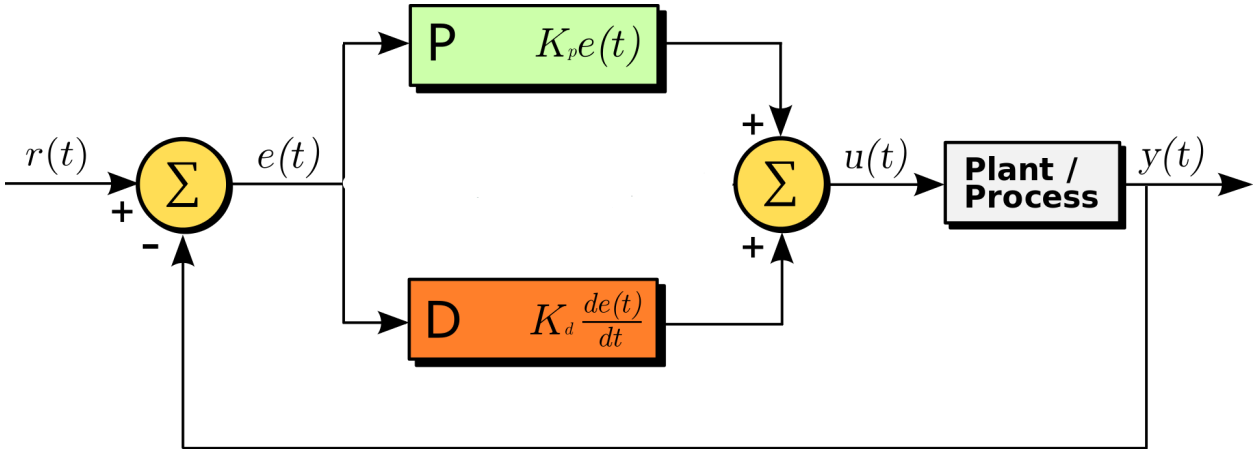

Figure 3.7: The PD Controller

A PD controller has the general form:

$$u(t) = K_p e(t) + K_D \frac{d}{dt} e(t) \tag{3.38}$$

Applying Laplace transformation results in the transformed PD controller equation:

$$u(s) = K_p e(s) + K_D s e(s) \tag{3.39}$$

$$u(s) = (K_p + K_I \frac{1}{s} + K_D s) e(s) \tag{3.40}$$

With the PD controller transfer function:

$$C(s) = K_p + K_D s = \frac{K_D s^2 + K_p s}{s} \tag{3.41}$$

The PD controller transfer function in series form:

$$C(s) = K(1 + sT_d) \tag{3.42}$$

The PD controller uses to increase the stability of the system since the derivative part of the controller has ability to predict the future errors of the system response, but the derivative pert can amplify the system noise.

3.3.5 Ziegler-Nichols Tuning

There are different methods to find parameters for the PID, PD and PI controllers. The Ziegler-Nichols method is one of these tuning methods, which is used in this project. The Ziegler Nichols tuning method is a heuristic method of tuning a PID controller. It was developed by John G. Ziegler and Nathaniel B. Nichols, they developed tow techniques to control tuning one of them is Ziegler Nichols open loop tuning and second is Ziegler Nichols closed loop tuning.

The open-loop method is depends on the open-loop step response of the dynamic system, the open-loop method is performed by applying an input step signal to get this step response. By using Ziegler-Nichols open-loop tuning with dead time L, reaction rate R and amplitude U of step input, the parameters of the controller P, PI and PID can be found from the table 3.3.

Type	K_p	$T_i = \frac{K_p}{K_i}$	$T_d = \frac{K_d}{K_i}$
P	$\frac{1}{LR/U}$	∞	0
PI	$\frac{0.9}{LR/U}$	3.3L	0
PID	$\frac{1.2}{LR/U}$	2L	0.5L

Table 3.3: Ziegler Nichols Method Open loop

The closed-loop method is performed by setting the I (integral) and D (derivative) gains to zero. The "P" (proportional) gain, K_p is then increased (from zero) until it reaches the ultimate gain K_u which is the largest gain at which the output of the control loop has stable and consistent oscillations; higher gains than the ultimate gainK_u have diverging oscillation. K_u and the oscillation period P_u are then used to set the P, I, and D gains depending on the type of controller used and behaviour desired, see Ziegler Nichols method tables bellow for both open loop 3.3 and closed loop 3.4:

Type	K_p	$T_i = \frac{K_p}{K_i}$	$T_d = \frac{K_d}{K_i}$
P	$0.5K_u$	∞	0
PI	$0.45K_u$	$\frac{P_u}{1.2}$	0
PD	$0.8K_u$	∞	$\frac{P_u}{8}$
PID	$0.6K_u$	$\frac{P_u}{2}$	$\frac{P_u}{8}$

Table 3.4: Ziegler Nichols Method Closed loop

Chapter 4

Modeling of the crane

In this chapter it is designed a mathematical model of crane. The crane model is created based on the crane dynamics equation developed by the crane kinematics. Finally the dynamic crane model is made in Simulink In addition, a crane model is also created in simscape using the multi-body program.

4.1 Description of the Crane

The figures 4.1 shows assembly of the crane with was create in Solidworks. All crane measurements are chosen to be similar to real MacGregor cran. Actually the MacGregor crane consists of hundreds of parts, but it was simplified to be assembled of four bodies.

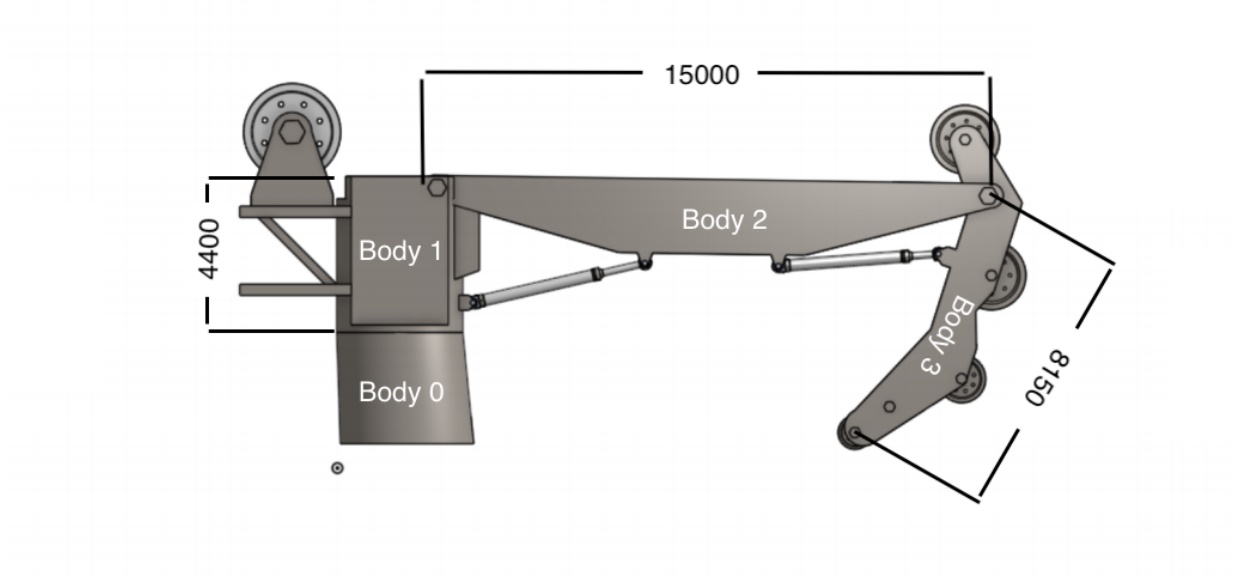

Figure 4.1: Assembly of the crane

Body 0: Foundation
Body 1: King assembly
Body 2: Main Jib
Body 4: Knuckle Jib

The inertia tensors, mass and center of mass of each of the four bodies are calculated from Simscape crane model which is made in chapter 4.5. The calculation results are presented in Table 4.1

Body	Mass [t]	Center of mass [m]	Ixx [kg/m2]	Iyy [kg/m2]	Izz [kg/m2]
Foundation	229875	[-4.04418e-05, 1.54655,-0.0759071]	348523	347834	351494
King assembly	267880	[-0.671535, -1.08361, 2.08881]	534248	1.01755e+06	707533
Main Jib	134427	[-8.48317, 8.02294, 1.58674]	65191.9	1.83276e+06	1.85226e+06
Knuckle Jib	25380.7	[7.34661, 2.06517, -0.36866]	20975.5	171449	170786

Table 4.1: Parameters of the crane

All the crane parameters are used further to create the dynamic crane model in Simulink.

4.2 Crane Kinematics

4.2.1 Crane Geometry Simplification

The crane was simplified as shown in Fig. 4.2. Adding a local coordinate system, as defined in the Eqs. 4.1 and 4.2, makes it slightly easier to find the joint angles through inverse kinematics in chapter 4.2.3.

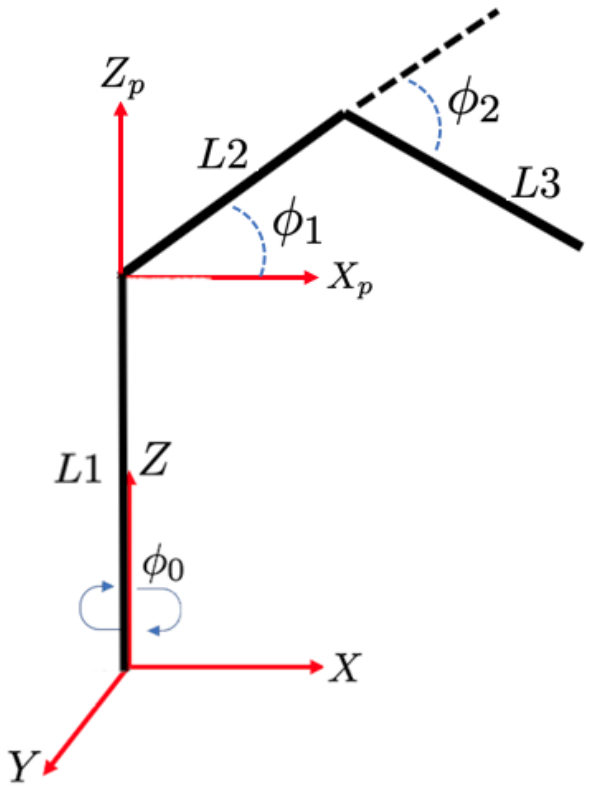

Figure 4.2: A simplification of the crane

Local coordinate system can be defined as:

$$X_p = \sqrt{X^2 + Y^2} \tag{4.1}$$
$$Z_p = Z - L1 \tag{4.2}$$

4.2.2 Forward Kinematics: Denavit-Hartenberg

To model the forward kinematics, the Denavit-Hartenberg convention was used. The method was applied to the simplified model explained in chapter 4.2.1.

Table 5.1 shows Denavit-Hartenberg parameter convention for three joints.

Joint	**Rot Z**	**Trans Z**	**Rot X**	**Trans X**
1	ϕ_0	L_1	$\frac{\pi}{2}$	0
2	ϕ_1	0	0	L_2
3	ϕ_2	0	0	L_3

Table 4.2: DH parameters of a 3-DOF robot

Equation 4.3 represent rotational matrix of the first joint.

$$Rot_Z(\phi_0) = \begin{bmatrix} \cos(\phi_0) & -\sin(\phi_0) & 0 & 0 \\ \sin(\phi_0) & \cos(\phi_0) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.3}$$

Equation 4.4 represent translational matrix from the first joint to the second joint.

$$Trans_Z(L_1) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & L_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.4}$$

The second joint is oriented different with respect to the first joint. Thus the z axis must be rotated by rotational matrix from equation 4.5 so that it aligns with the rotational axis of the second joint.

$$Rot_X(\frac{\pi}{2}) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\frac{\pi}{2}) & -\sin(\frac{\pi}{2}) & 0 \\ 0 & \sin(\frac{\pi}{2}) & \cos(\frac{\pi}{2}) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.5}$$

Equation 4.6 represent rotational matrix of the second joint.

$$Rot_Z(\phi_1) = \begin{bmatrix} \cos(\phi_1) & -\sin(\phi_1) & 0 & 0 \\ \sin(\phi_1) & \cos(\phi_1) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.6}$$

Equation 4.7 represent translational matrix from the second joint to the third joint.

$$Trans_X(L2) = \begin{bmatrix} 1 & 0 & 0 & L_2 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.7}$$

Equation 4.8 represent rotational matrix of the third joint.

$$Rot_Z(\phi_2) = \begin{bmatrix} \cos(\phi_2) & -\sin(\phi_2) & 0 & 0 \\ \sin(\phi_2) & \cos(\phi_2) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.8}$$

Equation 4.9 represent translational matrix from the third joint to the end-effector.

$$Trans_X(L3) = \begin{bmatrix} 1 & 0 & 0 & L_3 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.9}$$

The DH procedure determines the transformation of the model from Frame 0 to Frame 3 with four elementary homogeneous transformations, where the resulting transformation matrices are computed as:

$$A_1^0 = Rot_Z(\phi_0) \cdot Trans_Z(L_1) \cdot Rot_X(\frac{\pi}{2}) = \begin{bmatrix} \cos(\phi_0) & 0 & \sin(\phi_0) & 0 \\ \sin(\phi_0) & 0 & -\cos(\phi_0) & 0 \\ 0 & 1 & 0 & L_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.10}$$

$$A_2^1 = Rot_Z(\phi_1) \cdot Trans_X(L2) = \begin{bmatrix} \cos(\phi_1) & -\sin(\phi_1) & 0 & L_2\cos(\phi_1) \\ \sin(\phi_1) & \cos(\phi_1) & 0 & L_2\sin(\phi_1) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.11}$$

$$A_3^2 = Rot_Z(\phi_2) \cdot Trans_X(L3) = \begin{bmatrix} \cos(\phi_2) & \sin(\phi_2) & 0 & L_3\cos(\phi_2) \\ \sin(\phi_2) & \cos(\phi_2) & 0 & L_3\sin(\phi_2) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.12}$$

Then the total transformation from Frame 0 to Frame 3 can be computed as:

$$H_3^0 = A_1^0 A_2^1 A_3^2 = \begin{bmatrix} c_0c_{12} & -c_0s_{12} & s_0 & c_0(L_2c_1 + L_3c_{12}) \\ s_0c_{12} & -s_0s_{12} & -c_0 & s_0(L_2c_1 + L_3c_{12}) \\ s_{12} & c_{12} & 0 & L_1 + L_2s_1 + L_3s_{12} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.13}$$

where:

$$c_0 = \cos(\phi_0) \tag{4.14}$$
$$c_1 = \cos(\phi_1) \tag{4.15}$$
$$s_0 = \sin(\phi_0) \tag{4.16}$$
$$s_1 = \sin(\phi_1) \tag{4.17}$$
$$c_{12} = \cos(\phi_1 + \phi_2) \tag{4.18}$$
$$s_{12} = \sin(\phi_1 + \phi_2) \tag{4.19}$$

The position of the end-effector can be found from the fourth column and three first rows of the total transformation H_3^0. The position of the end-effector has represented by X,Y and Z

$$X = c_0(L_2c_1 + L_3c_{12}) \tag{4.20}$$
$$Y = s_0(L_2c_1 + L_3c_{12}) \tag{4.21}$$
$$Z = L_1 + L_2s_1 + L_3s_{12} \tag{4.22}$$

4.2.3 Inverse Kinematics

Inverse kinematics were applied to the end-effector position in order to acquire the joint angles. The inverse kinematics are found by using geometry and provided supporting material from the lecturer Michael Ruderman. [3]

Joint angle ϕ_2 can be described by the following equation:

$$\phi_2 = atan2(\sin(\phi_2), \cos(\phi_2)) \tag{4.23}$$

Where $\cos(\phi_2)$ and $\sin(\phi_2)$ can be computed as:

$$\cos(\phi_2) = \frac{X_p^2 + Z_p^2 - L2^2 - L3^2}{2 \cdot L2 \cdot L3} \tag{4.24}$$

$$\sin(\phi_2) = -\sqrt{1 - \cos(\phi_2)^2} \tag{4.25}$$

Joint angle ϕ_1 can be described by the following equation:

$$\phi_1 = atan2(Z_p, X_p) - atan2(L3 \cdot \sin(\phi_2), L2 + L3 \cdot \cos(\phi_2)) \tag{4.26}$$

Joint angle ϕ_0 can be described by the following equation:

$$\phi_0 = atan2(Y, X) \tag{4.27}$$

4.2.4 Validation of Forward Kinematics

It is important that Forward kinematics is correct because it is used further to find Jacobian. The way to check if kinematics is correct is to compare the joint angles which is used in Forward kinematics with the joint angles which is found from inverse kinematics.

To make it easier, the sine wave trajectory is used in kinematics. The figures 4.3 shows that the joint angles are converted into end-effector position by using Forward kinematics. Then the end-effector position is converted back into the joint angles by using Inverse kinematics.

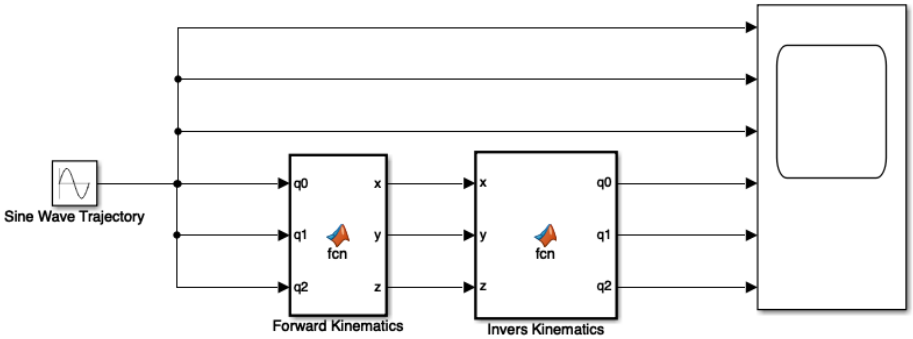

Figure 4.3: Validation of Forward Kinematics Simulink model

The figure 4.4 shows the joint angels of trajectory is identical with the joint angles that are found from the forward kinematics. This means that Forward kinematics is correct

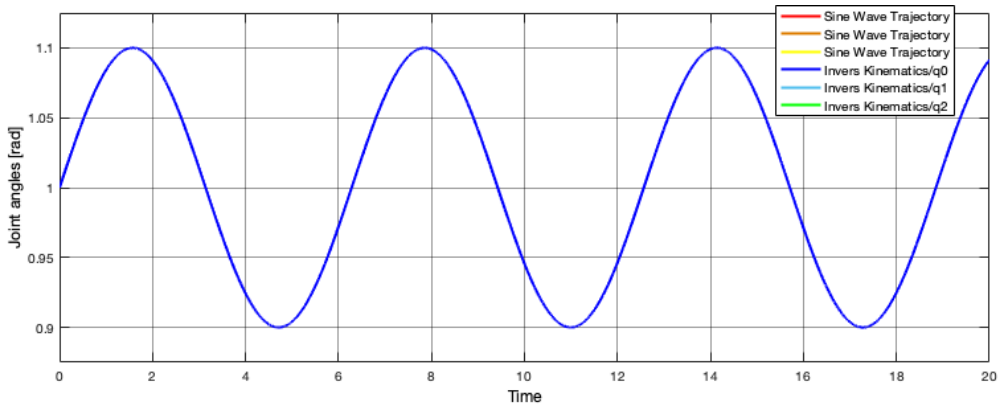

Figure 4.4: Joint angles comparison

The MATLAB script for Inverse and Forward kinematics is attached in appendices 8.1

4.2.5 Geometric Jacobian between Frame 3 and Joints

The first step to find the geometric Jacobian is to find the transformation from Frame 0 to Frame 3. The homogeneous transformation matrices H_i^0 are needed to obtain the geometric Jacobian from Frame 0 to 1, Frame 0 to 2 and Frame 0 to 3.

From Frame 0 to Frame 1:

$$H_1^0 = A_1^0 = \begin{bmatrix} c_0 & 0 & s_0 & 0 \\ s_0 & 0 & -c_0 & 0 \\ 0 & 1 & 0 & L_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.28}$$

From Frame 0 to Frame 2:

$$H_2^0 = A_1^0 A_2^1 = \begin{bmatrix} c_0c_1 & -c_0s_1 & s_0 & L_2c_0c_1 \\ s_0c_1 & -s_0s_1 & -c_0 & L_2s_0c_1 \\ s_1 & c_1 & 0 & L_1 + L_2s_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.29}$$

From Frame 0 to Frame 3:

$$H_3^0 = A_1^0 A_2^1 A_3^2 = \begin{bmatrix} c_0c_{12} & -c_0s_{12} & s_0 & c_0(L_2c_1 + L_3c_{12}) \\ s_0c_{12} & -s_0s_{12} & -c_0 & s_0(L_2c_1 + L_3c_{12}) \\ s_{12} & c_{12} & 0 & L_1 + L_2s_1 + L_3s_{12} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.30}$$

From the transformation matrices can be found the vectors r_{i-1} and z_{i-1}
The vectors r_{i-1} are found from the fourth column and three first rows of H_{i-1}^0, yielding

$$r_0 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \quad r_1 = \begin{bmatrix} 0 \\ 0 \\ L_1 \end{bmatrix} \quad r_2 = \begin{bmatrix} L_2c_0c_1 \\ L_2s_0c_1 \\ L_1 + L_2s_1 \end{bmatrix} \quad r_3 = \begin{bmatrix} c_0(L_2c_1 + L_3c_{12}) \\ s_0(L_2c_1 + L_3c_{12}) \\ L_1 + L_2s_1 + L_3s_{12} \end{bmatrix} \tag{4.31}$$

For the z_{i-1} vectors, we use the third column and first three rows of H_{i-1}^0, yielding

$$z_0 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \quad z_1 = \begin{bmatrix} s_0 \\ -c_0 \\ 1 \end{bmatrix} \quad z_2 = \begin{bmatrix} s_0 \\ -c_0 \\ 1 \end{bmatrix} \tag{4.32}$$

There are only revolute joints, so that geometric Jacobian can be written as

$$J_i = \begin{bmatrix} J_{vi} \\ J_{wi} \end{bmatrix} = \begin{bmatrix} z_{i-1} \times (r_3 - r_{i-1}) \\ z_{i-1} \end{bmatrix} \tag{4.33}$$

Then the geometric Jacobian for the joints can be computed as follows
Joint variable 1:

$$J_1 = \begin{bmatrix} z_0 \times (r_3 - r_0) \\ z_0 \end{bmatrix} = \begin{bmatrix} -s_0(L_2c_1 + L_3c_12) \\ c_0(L_2c_1 + L_3c_12) \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \tag{4.34}$$

Joint variable 2:

$$J_1 = \begin{bmatrix} z_1 \times (r_3 - r_1) \\ z_1 \end{bmatrix} = \begin{bmatrix} -c_0(L_2s_1 + L_3s_12) \\ -s_0(L_2s_1 + L_3s_12) \\ L_2c_1 + L_3c_12 \\ s_0 \\ -c_0 \\ 0 \end{bmatrix} \tag{4.35}$$

Joint variable 3:

$$J_1 = \begin{bmatrix} z_2 \times (r_3 - r_2) \\ z_2 \end{bmatrix} = \begin{bmatrix} -L_3c_0c_12 \\ -L_3s_0s_12 \\ L_3c_12 \\ s_0 \\ -c_0 \\ 0 \end{bmatrix} \tag{4.36}$$

Finally, the total geometric Jacobian can be written as

$$J = \begin{bmatrix} J_v \\ J_w \end{bmatrix} = \begin{bmatrix} J_1 & J_2 & J_3 \end{bmatrix} = \begin{bmatrix} -s_0(L_2c_1 + L_3c_12) & -c_0(L_2s_1 + L_3s_12) & -L_3c_0c_12 \\ c_0(L_2c_1 + L_3c_12) & -s_0(L_2s_1 + L_3s_12) & -L_3s_0s_12 \\ 0 & L_2c_1 + L_3c_12 & L_3c_12 \\ 0 & s_0 & s_0 \\ 0 & -c_0 & -c_0 \\ 1 & 0 & 0 \end{bmatrix} \tag{4.37}$$

4.3 Crane Dynamics

The crane dynamics can be described by the dynamic equations based on the kinetic and potential energy of the crane. Dynamic equations are calculated from the capital chapter 3.2

The manipulator equation, can be written matrix form as

$$M(q)\ddot{q} + C(\dot{q}, q)\dot{q} + g(q) = \tau \tag{4.38}$$

Where

$M(q)\ddot{q}$: the Inertia matrix
$C(\dot{q}, q)$: the Coriolis and centripetal matrix
$g(q)$: the potential energy consisting of the gravity vector
τ: the joint torque vector, which is the input to the dynamic crane model

4.3.1 Kinetic Energy

The Inertia matrix for this crane is found from equation 3.16 and can be written as

$$M(q) = \begin{bmatrix} M_{11} & M_{12} & M_{13} \\ M_{21} & M_{22} & M_{23} \\ M_{31} & M_{32} & M_{33} \end{bmatrix} \tag{4.39}$$

where

$$M_{11} = I_{1y} + I_{2x}s_1^2 + I_{2y}c_1^2 + I_{3x}s_{12}^2 + I_{3y}c_{12}^2 + m_2(\frac{1}{2}L_2)^2c_1^2 + m_3(\frac{1}{2}c_{12} + L_2c_1)^2 \tag{4.40}$$

$$M_{12} = M_{21} = 0 \tag{4.41}$$

$$M_{13} = M_{31} = 0 \tag{4.42}$$

$$M_{22} = I_{2z} + I_{3z} + m_2(\frac{1}{2}L_2)^2 + m_3((\frac{1}{2}L_2)^2 + L_2^2 + L_2L_3c_2) \tag{4.43}$$

$$M_{23} = M_{32} = I_{3z} + m_3((\frac{1}{2}L_2)^2 + \frac{1}{2}L_2L_3c_2) \tag{4.44}$$

$$M_{33} = I_{3z} + m_3(\frac{1}{2}L_3)^2 \tag{4.45}$$

Further, the Inertia matrix can be used to derive the Coriolis and centripetal matrix. The Coriolis and centripetal matrix for this crane is found from equation 3.26 and can be written as

$$C(q, \dot{q}) = \begin{bmatrix} C_{11} & C_{12} & C_{13} \\ C_{21} & C_{22} & C_{23} \\ C_{31} & C_{32} & C_{33} \end{bmatrix} \tag{4.46}$$

where

$$
\begin{aligned}
C_{11} &= \frac{1}{2}(\frac{\partial M_{11}}{\partial q_1}\dot{q_1} + \frac{\partial M_{11}}{\partial q_2}\dot{q_2}) && (4.47)\\
&= (c_1 s_1(I_{2x} - I_{2y}) + c_{12}s_{12}(I_{3x} - I_{3y}) - m_2 L_2^2 c_1 s_1 && (4.48)\\
&- m_3(L_{3c}c_{12} + L_2 c_1)(L_{3c}s_{12} + L_2 s_1))\dot{q_1} && (4.49)\\
&+ (c_{12}s_{12}(I_{3x} - I_{3y}) - m_3 L_{3c}s_{12}(L_{3c}c_{12} + L_2 c_1))\dot{q_2} && (4.50)\\
C_{12} &= \frac{1}{2}\frac{\partial M_{11}}{\partial q_1}\dot{q_0} && (4.51)\\
&= (c_1 s_1(I_{2x} - I_{2y}) + c_{12}s_{12}(I_{3x} - I_{3y}) - m_2 L_2^2 c_1 s_1 && (4.52)\\
&- m_3(L_{3c}c_{12} + L_2 c_1)(L_{3c}s_{12} + L_2 s_1))\dot{q_0} && (4.53)\\
C_{13} &= \frac{1}{2}\frac{\partial M_{11}}{\partial q_2}\dot{q_0} = (c_{12}s_{12}(I_{3x} - I_{3y}) - m_3 L_{3c}s_{12}(L_{3c}c_{12} + L_2 c_1))\dot{q_0} && (4.54)\\
C_{21} &= -\frac{1}{2}\frac{\partial M_{11}}{\partial q_1}\dot{q_1} = -C_{12} && (4.55)\\
C_{22} &= \frac{1}{2}\frac{\partial M_{22}}{\partial q_2}\dot{q_2} = -\frac{1}{2}m_3 L_2 L_3 s_2 \dot{q_2} && (4.56)\\
C_{22} &= \frac{1}{2}(\frac{1}{2}\frac{\partial M_{22}}{\partial q_2}\dot{q_1} + 2\frac{1}{2}\frac{\partial M_{23}}{\partial q_2}\dot{q_1}) = -\frac{1}{2}m_3 L_2 L_3 s_2 \dot{q_1} - m_3 L_2 L_3 s_2 \dot{q_2} && (4.57)\\
C_{31} &= -\frac{1}{2}\frac{\partial M_{11}}{\partial q_2}\dot{q_0} = -C_{13} && (4.58)\\
C_{32} &= -\frac{1}{2}\frac{\partial M_{22}}{\partial q_2}\dot{q_1} = \frac{1}{2}m_3 L_2 L_3 s_2 \dot{q_1} && (4.59)\\
C_{33} &= 0 && (4.60)
\end{aligned}
$$

In the Coriolis and centripetal matrix, L_{ci} represents $\frac{L_i}{2}$.

4.3.2 Potential Energy

The potential energy equation 3.17 is a function of the gravity and can be written as

$$P = m_1\frac{L_1}{2} + m_2 g(\frac{L_2}{2} + L_1) + m_3 g(\frac{L_3}{2}s_{12} + L_2 s_1 + L_1) \tag{4.61}$$

The gravity vector for this crane is found from equation 3.24 and can be written as

$$g(q) = \begin{bmatrix} 0 \\ m_2 g L_{2c} c_1 + m_3 g L_{3c} c_{12} + L_2 c_1 \\ m_3 g L_{3c} c_{12} \end{bmatrix} \tag{4.62}$$

In in the gravity vector, L_{ci} represents $\frac{L_i}{2}$.

4.4 Dynamic Crane Model in Simulink

By utilizing equation ?? Equation of motion can also be written as

$$\ddot{q} = M^{-1}(\tau - C(\dot{q}, q)\dot{q} - g(q)) \tag{4.63}$$

The figures 4.5 shows the dynamic crane model in the form of a block diagram based on equation 4.63

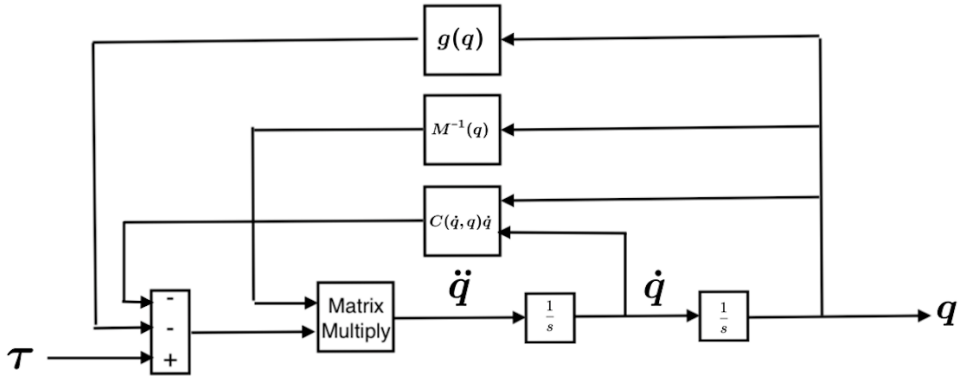

Figure 4.5: Block diagram of the dynamic crane model

The dynamic crane model is create in simulink which is shown in Figure 4.6 This model consists of Matlab function blocks of the Inertia matrix, Coriolis and centripetal matrix and Gravity vector.

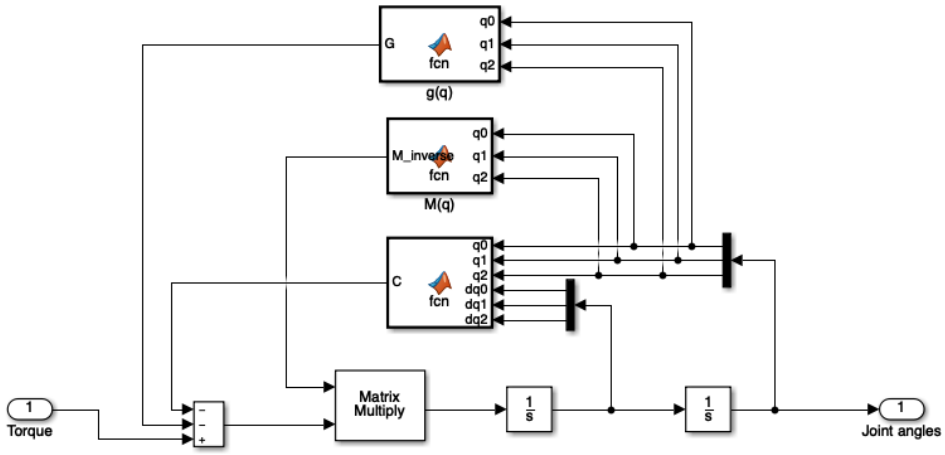

Figure 4.6: Dynamic crane model in Simulink

The MATLAB script for the Inertia matrix, Coriolis and centripetal matrix and Gravity vector can be found in Appendix 8.3

4.5 Crane Model in Simscape

Tow ways were applied to create the crane Simscape model first using the browser based CAD modelling tool Onshape, and second using Simscape Multibody (SimMechanics).

4.5.1 Browser Based CAD Modelling Tool Onshape

A 3D CAD model for the crane is created in SolidWorks as shown in figure 4.7. Matlab includes functions to import 3D models from the browser based CAD modelling tool Onshape. This functionality helps importing mechanical 3D objects for further analysis in Matlab/Simscape environment. This section presents an overview on how to import objects from Onshape to Matlab. The model consist of multiple different parts, figure 4.8 shows the 3D model in the Onshape.

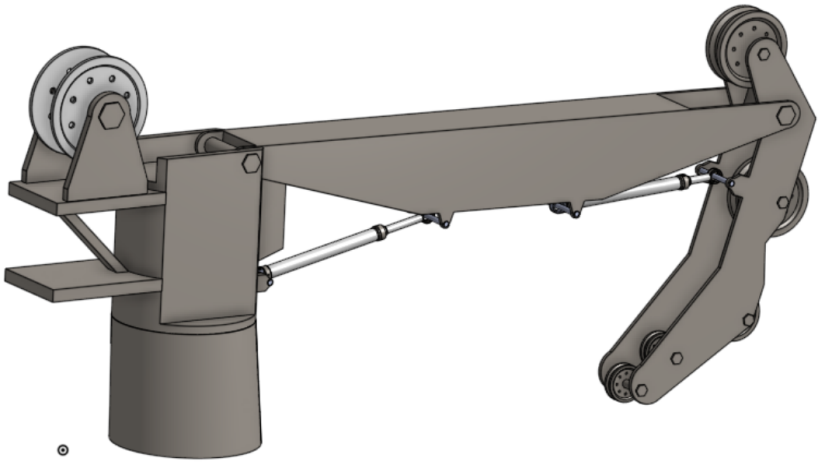

Figure 4.7: 3D Model of the crane

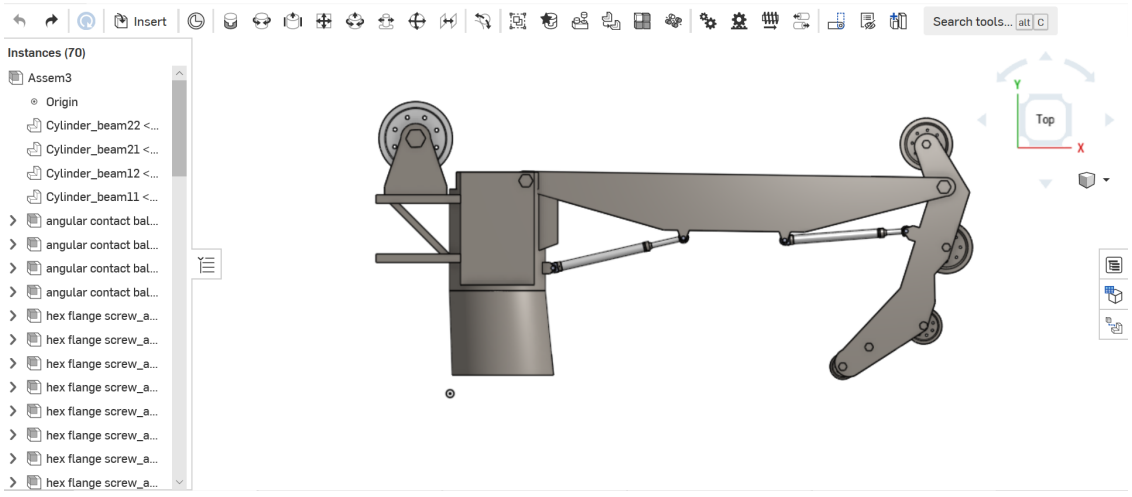

Figure 4.8: 3D Model in Onshape

The model consists of multiple different joints. We will see how the in-built OnShape functions will automatically register and convert the joints in to the Simscape environment.

A link has been created by copy the model link from adress bar to import it into Simscape, then command (smexportonshape("link")) was ran from MATLAB terminal to import Onshape model to MATLAB as shown in figure 4.9.
The next step was to run the following command (smimport(".xml")) to translate the .xml file to a Simscape object, as show in figure 4.10. The command automatically open a new Simulink window with the blocks from Onshape translated to Simscape. Figure 4.11 shows the crane Simscape model imoported from Onshape, and in figure 4.14 you can see the crane in 3D in the mechanics explorer.

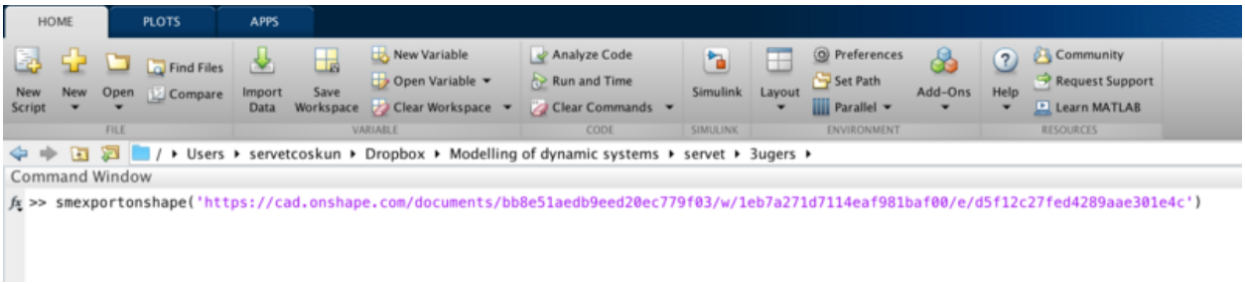

Figure 4.9: Importing Onshape Mdel to MATLAB

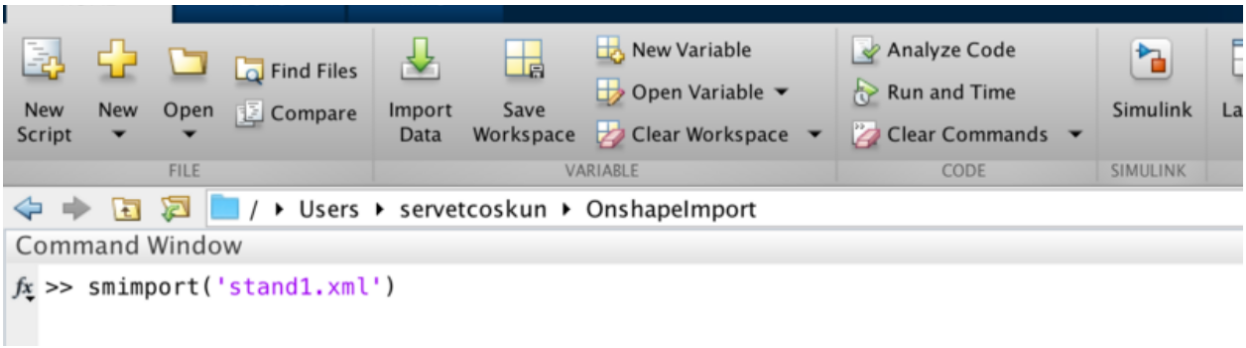

Figure 4.10: Translate .xml file to Simscape Object

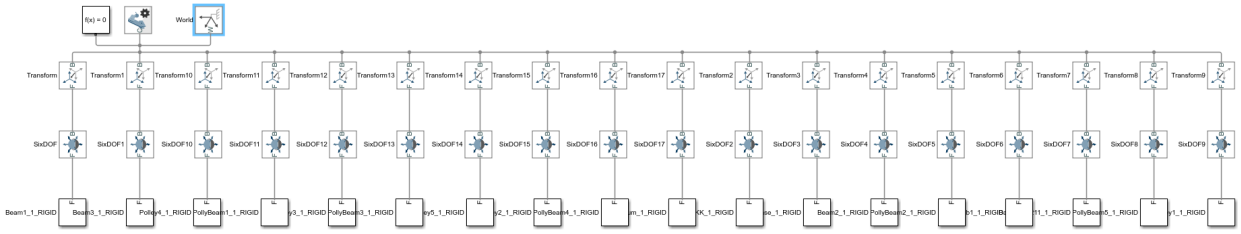

Figure 4.11: Simscape Model Imported from Onshape

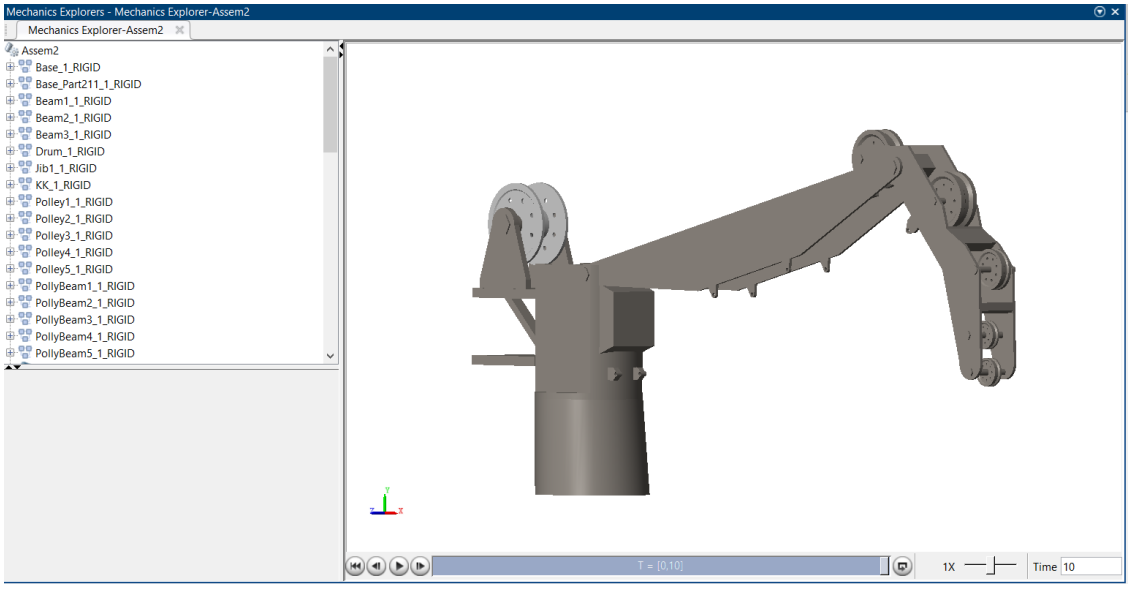

Figure 4.12: Simscape Model Imported from Onshape in The Mechanics Explorer

4.5.2 Simscape Multibody (SimMechanics)

The crane model was designed by using the Simscape library, different blocks from the library were used to make the crane model by using CAD model, as you can see in figure 4.13. Blocks from belts and cables library are used to model the pulleys that control lifting the payload.

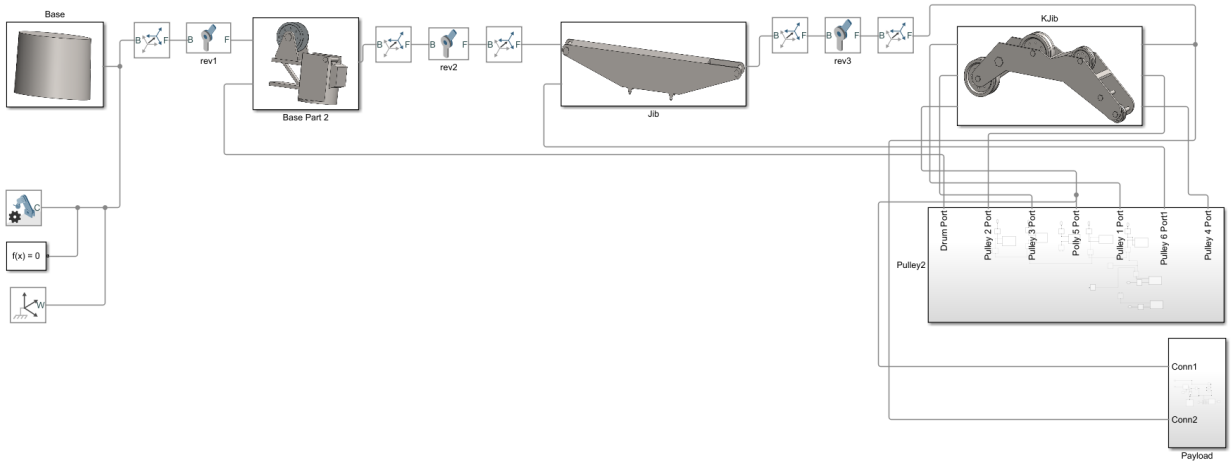

Figure 4.13: Crane Model Using Simscape Library

Figure 4.14 shows the simulation of the crane in the mechanics explorer with the payload.

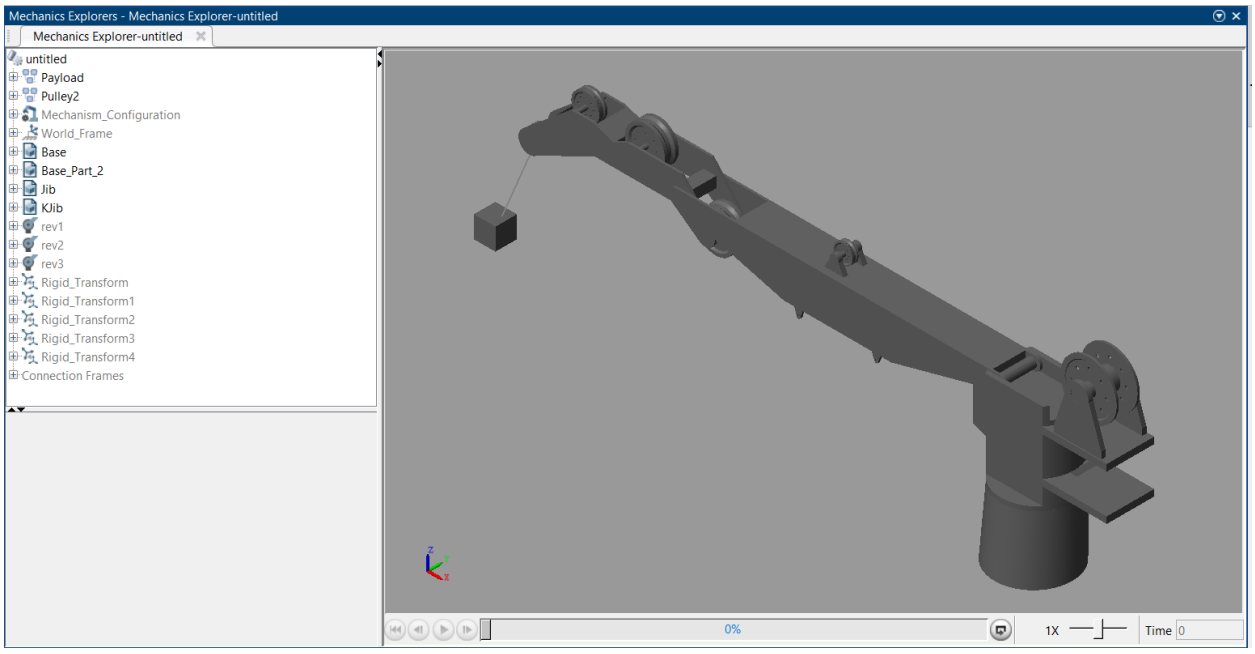

Figure 4.14: Crane from the Mechanics Explorer

Chapter 5

Control of the crane

This chapter describes how to control the end-effector in desired planned trajectory with small payload swing. Controllers are designed for both the crane model in simulink described in chapter 4.4 and the crane model in simscape described in chapter 4.5. The controller designs that will be presented in this chapter are:

PID-controller
PD-controller
PI-controller

5.1 Trajectory Planning

The control task regarding control of crane end effector is to get the crane end effector to follow a desired trajectory movement in horizontal direction with small payload swing. To solve that problems there was used trajectory planning method for 2-D overhead cranes from paper [5]. It will be create to simulink model of Trajectory. One of the model will be reference trajectory $x_r(t)$ and the other simulink model will be planned trajectory $x_d(t)$ with anti-swing mechanism. After that simulation results of this two models will be compared.

The paper [5] is focused on the trajectory planning problem for overhead cranes during the transportation process. The planar model of a 2-D overhead crane (with a payload suspended) is shown in Figure 5.1 where l represents the rope length, x and θ are the trolley displacement and the payload swing with respect to the vertical, respectively, which are regarded as the states of the system, F , F_r , and mg denote the control force, the rail friction, and the payload gravity, respectively

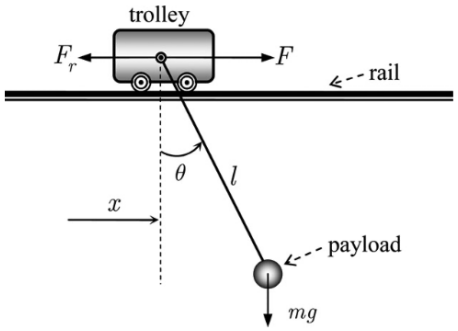

Figure 5.1: Schematic illustration of a planar 2-D overhead crane.

With air resistance being ignored, the crane dynamics with constant rope length can be depicted as follows:

$$(M+m)\ddot{x} + ml\ddot{\theta}\cos\theta - ml\dot{\theta}^2\sin\theta = F - F_r \tag{5.1}$$

$$ml^2\ddot{\theta} + ml\ddot{x}\cos\theta + mgl\sin\theta = 0 \tag{5.2}$$

where
M: is the mass of trolley
m: is the mass of payload

Assumption 1: The rope is massless and inflexible. Moreover, the rope length is kept constant during a specific transportation process.
Assumption 2: During the overall transferring process, the payload is always beneath the trolley, namely

$$\theta(t) \in (-\frac{\pi}{2}, \frac{\pi}{2}) \tag{5.3}$$

The dynamic model includes the actuated part (the first equation) and the underactuated part (the second equation), where the latter one is the kinematic equation of the overhead crane system which describes the coupling behavior between the trolley acceleration $\ddot{x}(t)$ and the payload swing $\theta(t)$. Dividing both sides of the kinematic equation by ml, we obtain

$$l\ddot{\theta} + \ddot{x}\cos\theta + g\sin\theta = 0 \tag{5.4}$$

This relationship captures the kinematic features of all 2-D overhead cranes and it is the basis of the subsequent trajectory development.

Kinematic equation 5.5 can also be written as

$$\ddot{\theta} = \frac{-\ddot{x}\cos\theta - g\sin\theta}{l} \tag{5.5}$$

Figure 5.2 shows simulink model of kinematic equation 5.5

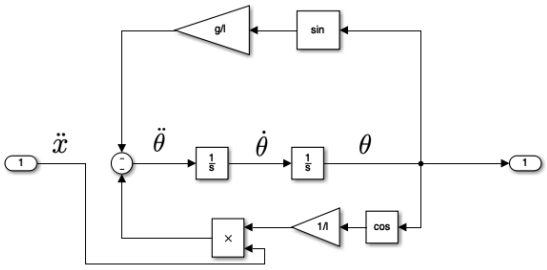

Figure 5.2: kinematic model in simulink

The trolley reference trajectory $x_r(t)$ is given by equation 5.6

$$x_r(t) = \frac{p_r}{2} + \frac{1}{2k_2} ln[\frac{\cosh k_1 t - \varepsilon}{\cosh k_1 t - \varepsilon - k_2 p_r}] \tag{5.6}$$

where $\varepsilon \in \mathbb{R}^+$ is a coefficient introduced to adjust the initial acceleration while k_1, k_2 are two auxiliary parameters and p_r are trolley trajectory distance.

Equation 5.7 shows trolley acceleration reference trajectory $\ddot{x}_r(t)$ which was found by double derivatives of trolley position trajectory equation 5.6. This acceleration is used in simulink model of kinematic 5.2 to find payload swing angle.

$$\ddot{x}_r(t) = \frac{k_1^2((\tanh(k_2 p_r - k_1 t + \varepsilon))^2 - (\tanh(k_1 t - \varepsilon))^2)}{2k_2} \tag{5.7}$$

Figure 5.3 shows simulink model of trolley reference trajectory $x_r(t)$. This model consists of Matlab function block of the trolley acceleration trajectory from equation 5.7 and block diagram of kinematic equation 5.5. To go from acceleration trajectory to position trajectory it is used two integrator blocks. The simulation results of reference trajectory and swing angle of payload in degrees is shown in 5.1.

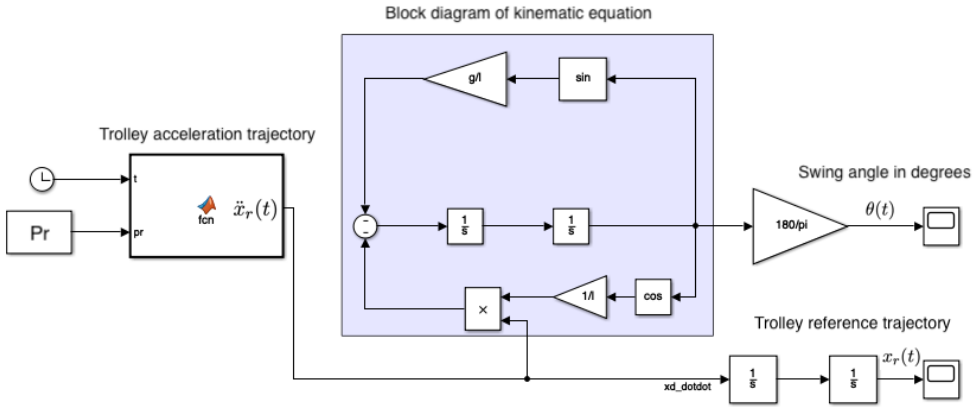

Figure 5.3: Simulink model of reference trajectory

In next step it will be create simulink model of trolley planned trajectory $\ddot{x}_d(t)$ which can be found by euation 5.8

$$\ddot{x}_d(t) = \ddot{x}_r(t) + \Gamma \ddot{x}_a(t) \tag{5.8}$$

where $\Gamma \in [1/(\beta+\alpha l), 1]$ is a coefficient introduced to adjust the weighings of the reference trajectory and the anti-swing mechanism. $\ddot{x}_r(t)$ is reference trajectory from equation 5.6 and $\ddot{x}_a(t)$ can be computed by using anti-swing mechanism equation 5.9.

$$\ddot{x}_a = \frac{-g\sin\theta + \alpha l\dot{\theta} + \beta(\dot{\theta} + \alpha\theta)}{\cos\theta} \tag{5.9}$$

where $\beta \gg g$ is an adjusting parameter and $\alpha \in \mathbb{R}^+$ is a positive gain satisfying $\alpha \gg g$.

When the anti-swing mechanism is brought into the reference trajectory, it leads to positioning error for the trolley. To solve this problem, an iterative learning strategy is then put forward to further revise the planned trajectory.

Figure 5.4 shows iterative learning strategy create in simulink.

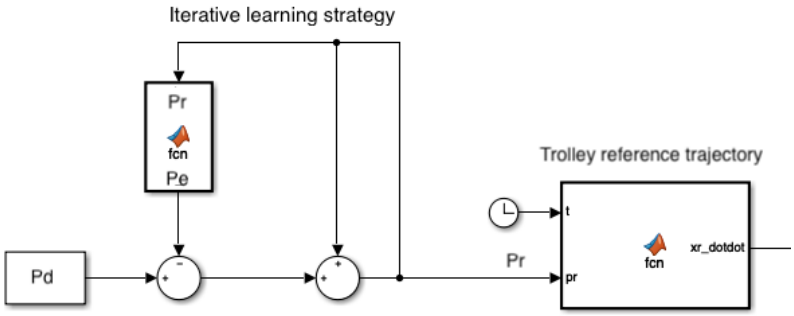

Figure 5.4: Iterative learning strategy

Where p_d is trolley trajectory distance and p_e are computed in Matlab function block by equation 5.10

$$p_e = p_r + \frac{\Gamma\alpha\beta}{g + \Gamma\alpha\beta} p_r \tag{5.10}$$

Figure 5.5 shows simulink model of trolley planned trajectory $x_d(t)$. This model consists of iterative learning strategy and Matlab function block of the trolley acceleration trajectory from equation 5.7. Angles $\dot{\theta}$ and θ from block diagram of kinematic equation 5.5 is used in Matlab function block of anti-swing mechanism to compute $\ddot{x}_a$. Finally to go from acceleration trajectory to position trajectory it is used two integrator blocks. The simulation results of reference trajectory and swing angle of payload in degrees is shown in 5.1.

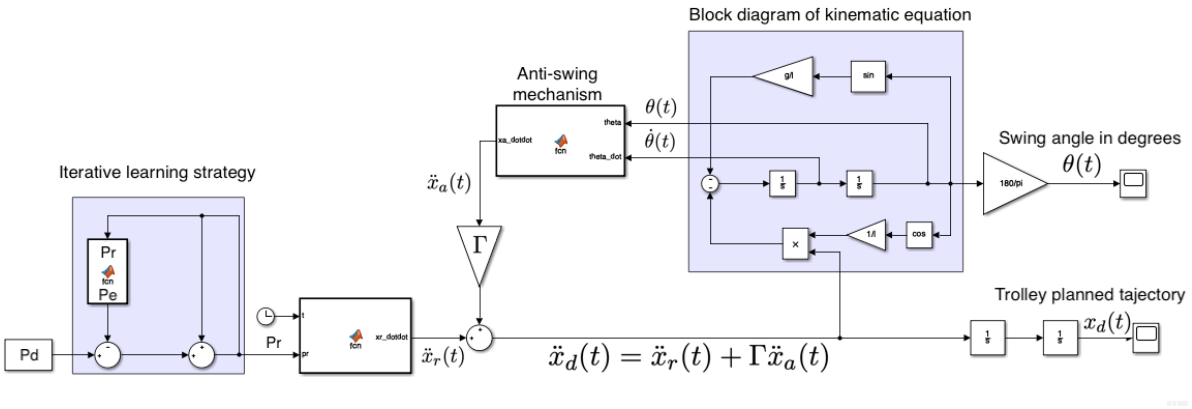

Figure 5.5: Simulink model of planned trajectory

Noting the fact that the planned trajectory does not include any terms related to m, we can conclude that the proposed method is robust against payload weight variations. This merit is of significant importance for practical applications.

All parameters for trolley reference and planned trajectory is shown below and are taken from paper [5].

$$p_r(1) = 0.6m \quad p_d = 0.6m \quad g = 9.8m/s^2 \quad \Gamma = 0.015 \tag{5.11}$$

$$\varepsilon = 3.5 \quad \alpha = \beta = 50 \quad k_1 = 1.2 \quad k_2 = 0.48 \tag{5.12}$$

Figure 5.6 and 5.7 shows simulation results comparison of trolley Trajectory and swing angle from both simulink models. The simulation results show that the swing angel for payload is reduced by small change of reference trajectory using anti-swing mechanism.

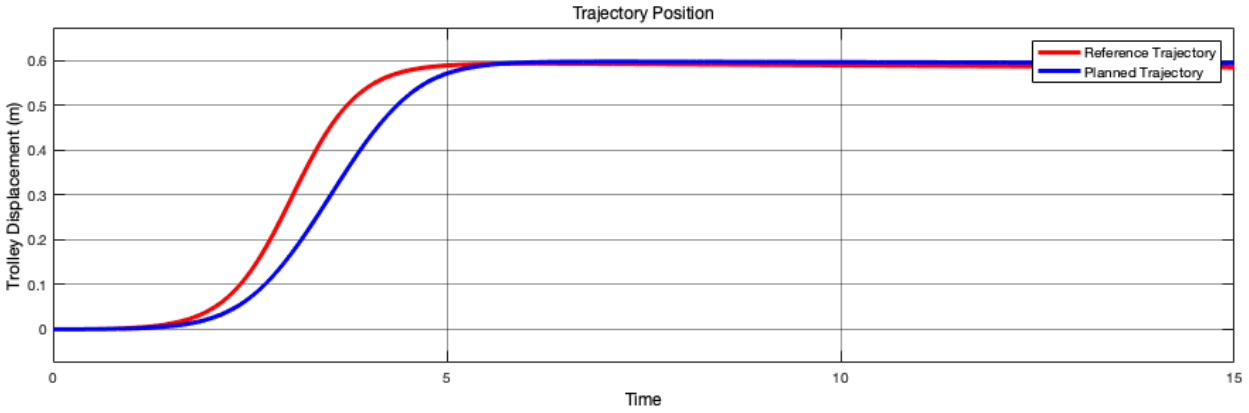

Figure 5.6: Comparison of trolley trajectory. Blue curve is the reference trajectory and the red curve is the planned trajectory.

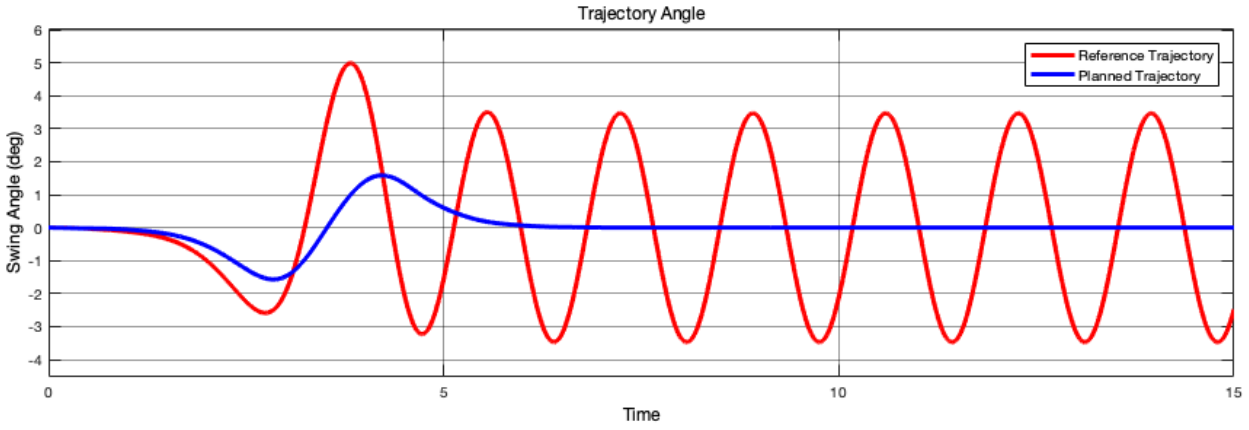

Figure 5.7: Comparison of payload swing angel trolley from trajectory. Blue curve is the swing angle from reference trajectory and the red curve is the swing angle from planned trajectory.

The planned trajectory from simulation is used further for control of Crane end-effector.

5.2 Control of End-effector for crane model in Simulink

The control task concerning control of crane end-effector is to get the crane end-effector to follow a desired trajectory movement in horizontal direction. The desired trajectory is to move the end-effector from an initial point to another desired point in x-direction. The initial location is set to 23.15 m, which is the x-coordinate when the crane arm is fully extended, and the desired location is set to 15 m. After the end-effector has reached the desired location, the aim is to keep the end-effector steady at this point.

Figure 5.8 shows the desired trajectory of the crane end-effector. This trajectory is the same as planned trajectory for 2-D overhead cranes from chapter 5.1 to damp out unexpected payload swing. The only difference is that the start and end point of the end-effector was changed.

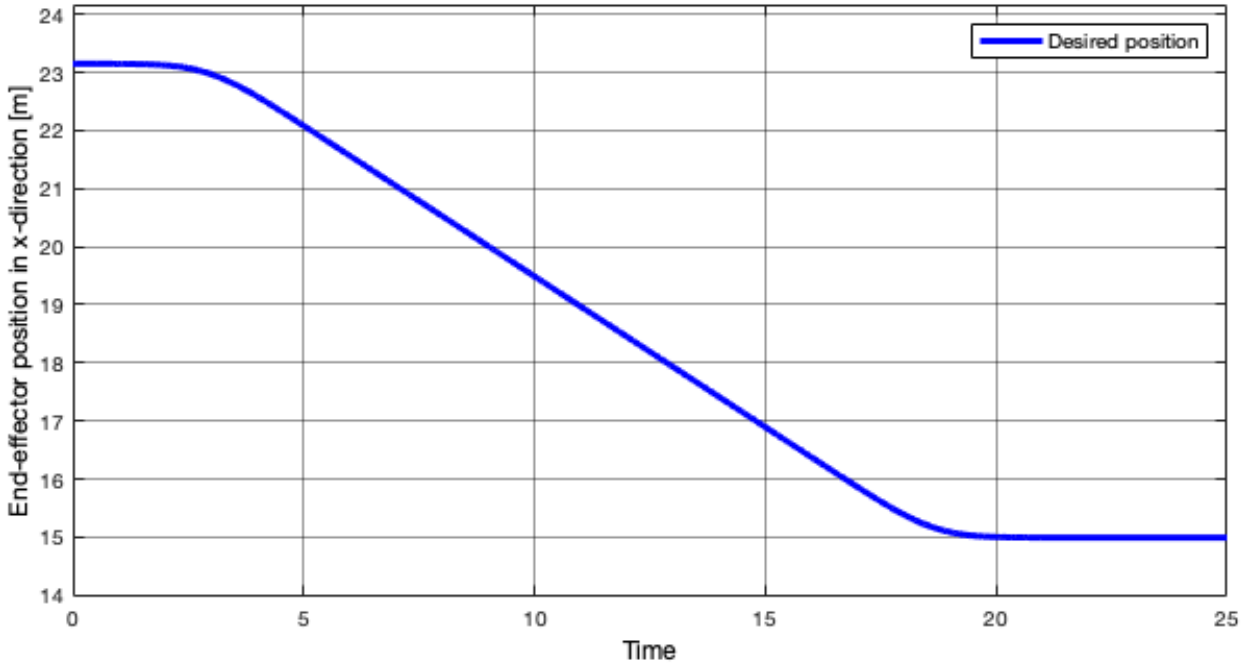

Figure 5.8: Desired position in x-direction

5.2.1 Zigler Nichols Closed Loop Tuning

There are different methods to find parameters for the PID, PD and PI controllers. The Ziegler-Nichols method is one of these tuning methods, which is used in this project. It is performed by setting the I (integral) and D (derivative) gains to zero. The "P" (proportional) gain, K_p is then increased (from zero) until it reaches the ultimate gain K_u which is the largest gain at which the output of the control loop has stable and consistent oscillations as shown in figure 5.9, the value of the ultimate period P_u can be found by running the Simulink model and used "Peak Finder" tool in Simulink as shown in figure 5.10.

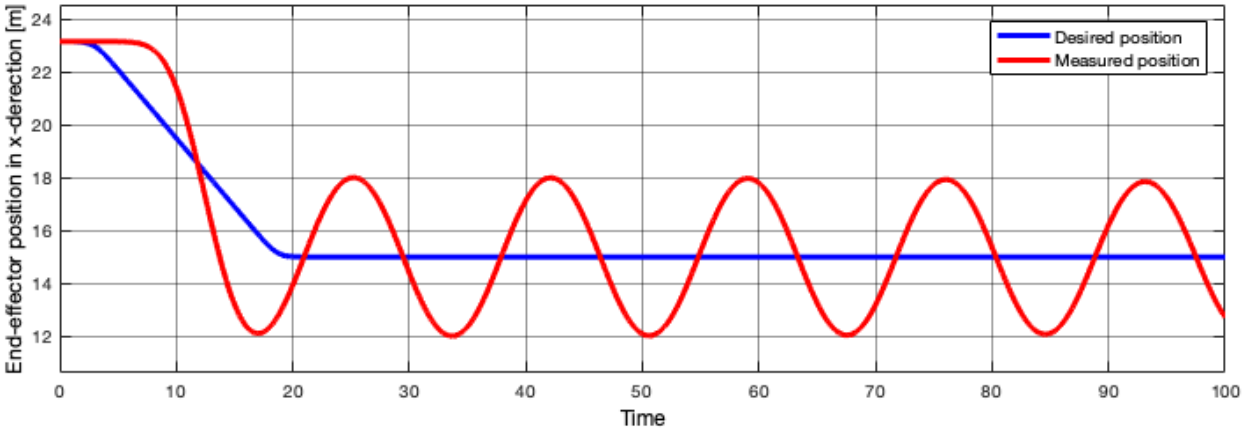

Figure 5.9: Procedure to find the ultimate gain K_u

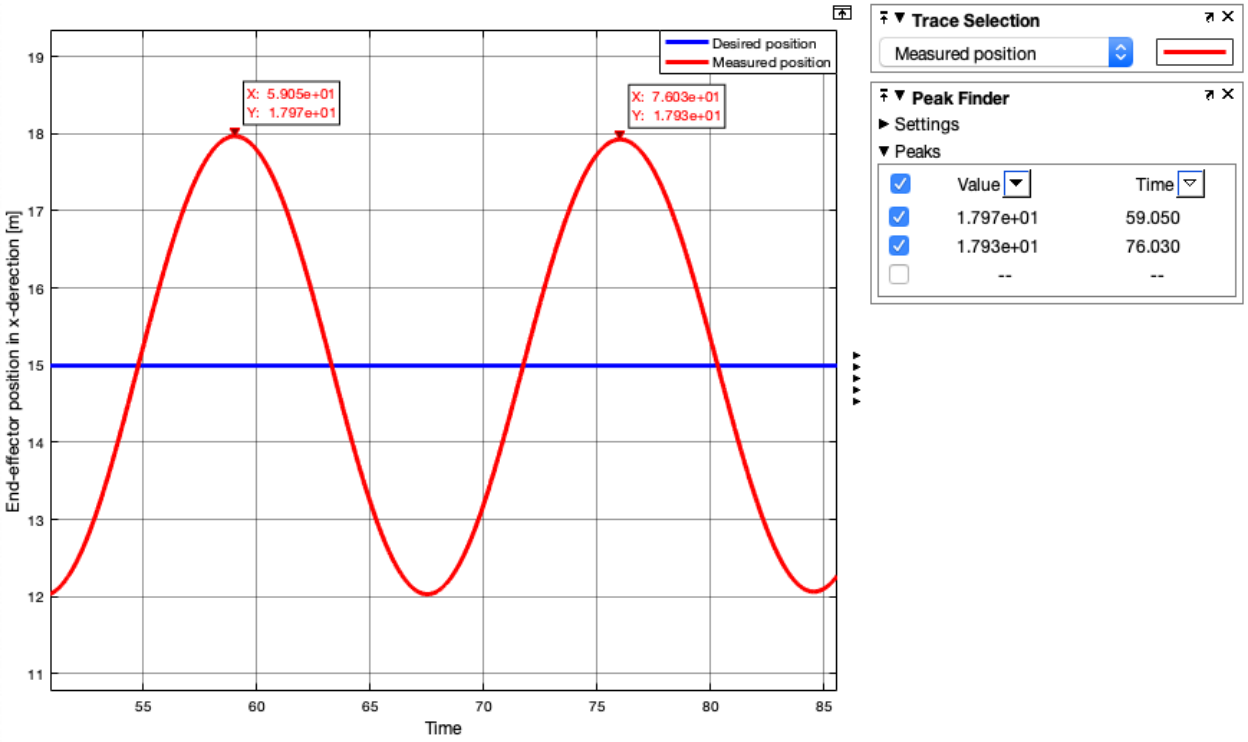

Figure 5.10: Using peak finder to determine the ultimate period P_u

From this method the ultimate gain became $K_p = 10000$ and ultimate period became $P_u = 17$. The controllers parameters can be calculated by using equations from table 3.4. The calculation results of controllers parameters are presented in table 5.2.1

Type	Kp	Kd	Ki
PI	4500	0	317
PD	8000	17000	0
PID	6000	12750	706

Table 5.1: Ziegler-Nichols closed-loop controller parameters

5.2.2 PID Controller

A proportional derivative (PID) controller is implemented to control the crane end-effector position and make sure that it follow the desired planned trajectory with very small error,the PD controller is designed with feedback system. The PID controller is used to improve the dynamic response as well as to reduce or eliminate the steady-state error. The derivative controller improve the transient response. The integral controller reduces the steady-state error.

The PID controller design is developed, and taking into account gravity compensating because of the crane weight. Figure 5.11 shows the design of the PID controller with gravity compensation for the crane.

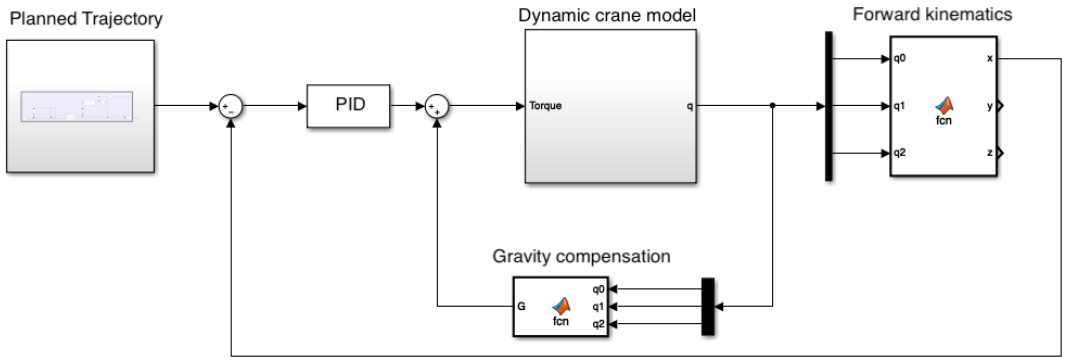

Figure 5.11: Control of the crane end-effector position using PID-controller with gravity compensation

5.2.3 PD controller

A proportional derivative (PD) controller is implemented to control the crane end-effector position and make sure that it follow the desired planned trajectory with very small error,the PD controller is designed with feedback system. The proportional part has ability to reduce the error, and make the closed loop system faster, but at same time can make the closed loop system to overshoot more. Figure 5.12 shows the PD controller design with gravity compensation.

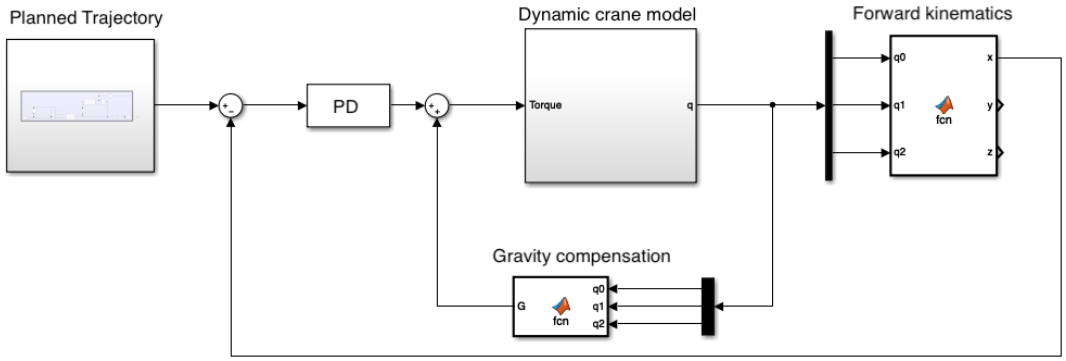

Figure 5.12: Control of the crane end-effector position using PD-controller with gravity compensation

5.2.4 PI Controller

A proportional Integral (PI) controller is implemented to control the crane end-effector position and make sure that it follow the desired planned trajectory with very small error, The PI controller is designed with feedback system. The proportional part has ability to reduce the error, and make the closed loop system faster, but at same time can make the closed loop system to overshoot more. While the integral controller reduces the steady-state error. Figure 5.13 shows the PI controller design with gravity compensation.

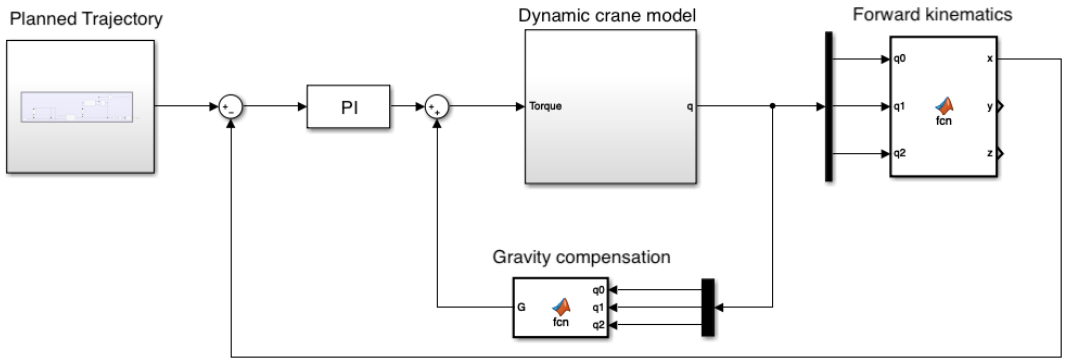

Figure 5.13: Control of the crane end-effector position using PI-controller with gravity compensation

5.3 Control of Joints for crane model in Simscape

A PID controller is implemented to control the crane joints position with sine wave as input signal and make sure that it follow the desired sine wave with very small error, The PID controller is designed with feedback system, the values of the controller parameters gains are very big. The proportional part has ability to reduce the error, and make the closed loop system faster, but at same time can make the closed loop system to overshoot more. While the integral controller reduces the steady-state error. Figure 6.13 shows the PID controller design with gravity compensation.

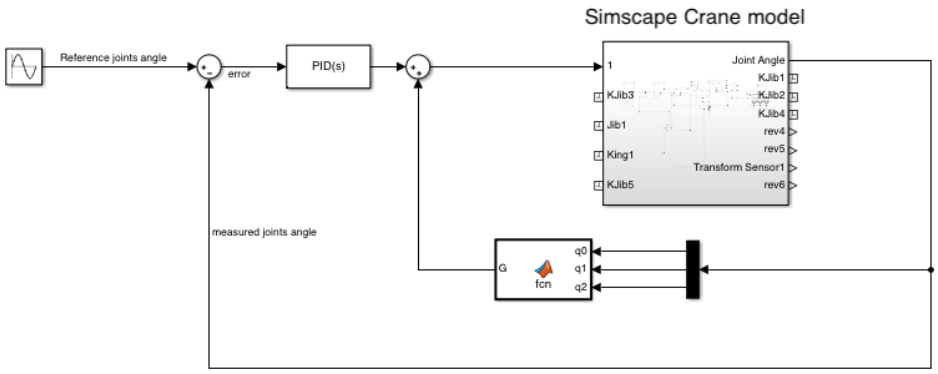

Figure 5.14: Control of the crane joints angle using PID-controller with gravity compensation

5.4 Control of End-effector for crane model in Simscape

To control the end-effector for the crane in the Simscape model a PID controller was used. The trajectory planning used as reference input signal and PID controller with large controller parameters gains, to be sure that the desired end-effector and measured end-effector position in x direction have small error. Figure 5.15 shows the PID controller design with gravity compensation.

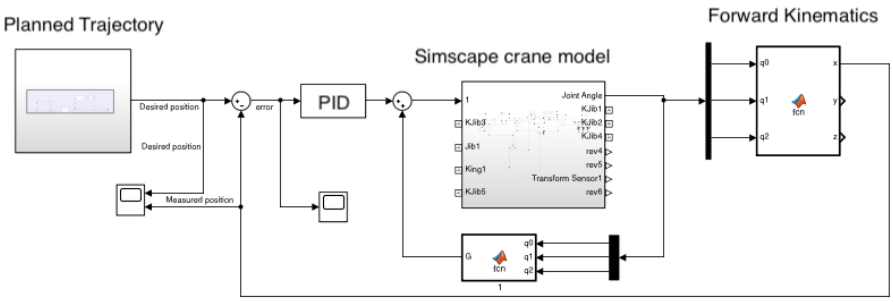

Figure 5.15: Control of the crane end-effector using PID-controller with gravity compensation

In chapter 6.5 it is explained why it was not choose to create controller design as PI and PD for simscape crane model.

Chapter 6

Simulation results

In this chapter simulation results from the control designs will be presented. This includes results from the control designs concerning control of end-effector position in horizontal directing.

6.1 Control of End-effector for crane model in Simulink

Simulation results will be presented by using following controllers:

PID-controller
PD-controller
PI-controller

Simulation results of each of controller will be presented as the desired end-effector position in x-direction compared to the measured end-effector position in x-direction. Error between desired and measured end-effector position in x-direction will be presented in form of a graph with respect to time and with a maximal value.

6.1.1 PID Controller

From Ziegler Nichols closed-loop tuning method the ultimate gain became $K_u = 10000$ and the ultimate period became $P_u = 17$, which resulted in the following controller parameter for a PID:

$K_p = 6000$
$K_i = 706$
$K_d = 12750$

Figure 6.1 shows simulation results of the measured and desired position in x-direction by using the PID-controller design with the calculated controller parameters and Figure 6.2 shows simulation result of the error between desired and measured position.

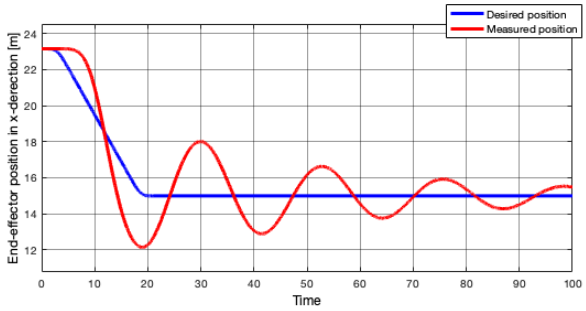

Figure 6.1: Comparison of desired and measured end-effector position in x-direction using Ziegler-Nichols parameters for PID-control. The blue curve is the desired and the red curve is the measured end-effector position in x-direction.

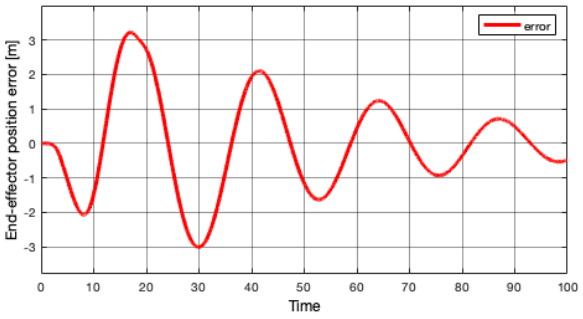

Figure 6.2: Error between desired and measured end-effector position in x-direction using Ziegler-Nichols parameters for PID-control.

From figure 6.2, the maximal error between desired and measured end-effector position is 3.2m.

Figure 6.3 and 6.4 shows new results where K_p and K_d have been increased. It was done to minimize error between desired and measured end-effector position.

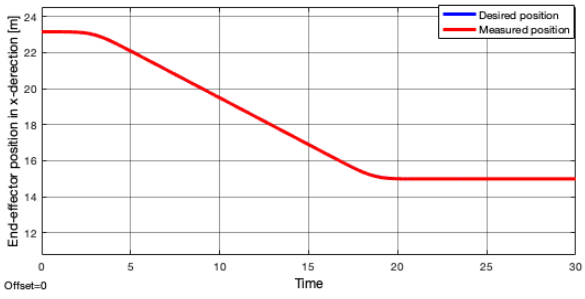

Figure 6.3: Comparison of desired and measured end-effector position in x-direction using increased gains for PID-control.

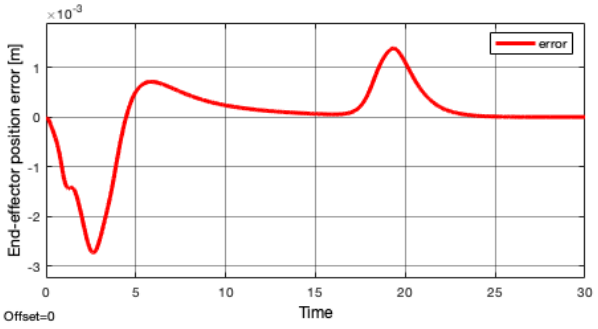

Figure 6.4: Error between desired and measured end-effector position in x-direction using increased gains for PID-control.

From figure 6.4, the maximal error between desired and measured end-effector position is 2.735mm.

6.1.2 PD Controller

From Ziegler Nichols closed-loop tuning method the ultimate gain became $K_u = 10000$ and the ultimate period became $P_u = 17$, which resulted in the following controller parameter for a PD:

$K_p = 8000$
$K_d = 17000$

Figure 6.5 shows simulation results of the measured and desired position in x-direction by using the PD-controller design with the calculated controller parameters and Figure 6.6 shows simulation result of the error between desired and measured position.

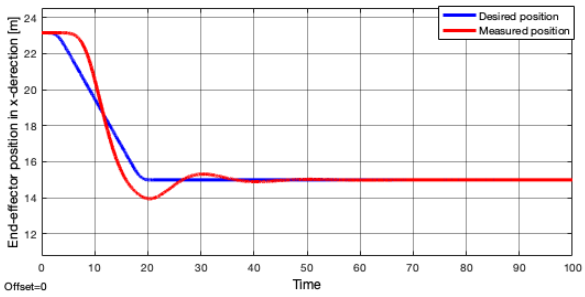

Figure 6.5: Comparison of desired and measured end-effector position in x-direction using Ziegler-Nichols parameters for PD-control. The blue curve is the desired and the red curve is the measured end-effector position in x-direction.

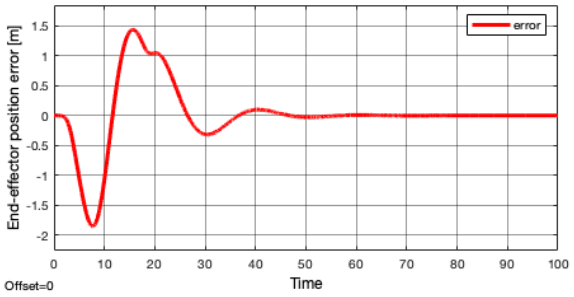

Figure 6.6: Error between desired and measured end-effector position in x-direction using Ziegler-Nichols parameters for PD-control.

From figure 6.6, the maximal error between desired and measured end-effector position is 1.8m.

Figure 6.7 and 6.8 shows new results where K_p and K_d have been increased. It was done to minimize error between desired and measured end-effector position.

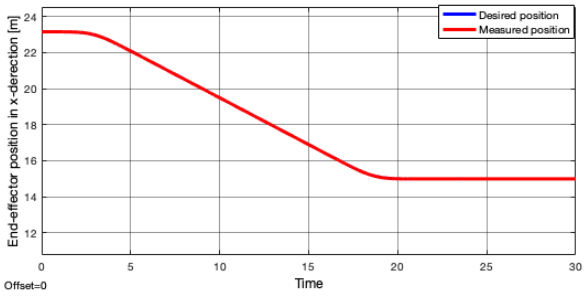

Figure 6.7: Comparison of desired and measured end-effector position in x-direction using increased gains for PD-control.

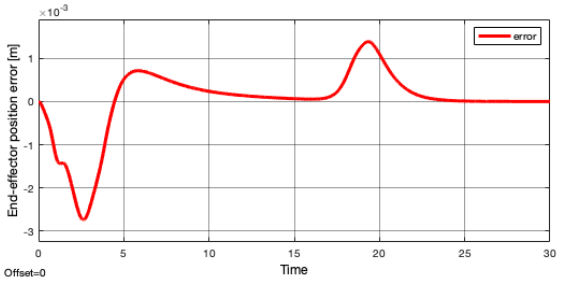

Figure 6.8: Error between desired and measured end-effector position in x-direction using Ziegler-Nichols parameters for PD-control.

From figure 6.8, the maximal error between desired and measured end-effector position is 2.735mm
.

6.1.3 PI Controller

From Ziegler Nichols closed-loop tuning method the ultimate gain became $K_u = 10000$ and the ultimate period became $P_u = 17$, which resulted in the following controller parameter for a PI:

$K_p = 4500$
$K_i = 317$

Figure 6.9 shows simulation results of the measured and desired position in x-direction by using the PI-controller design with the calculated controller parameters and Figure 6.10 shows simulation result of the error between desired and measured position.

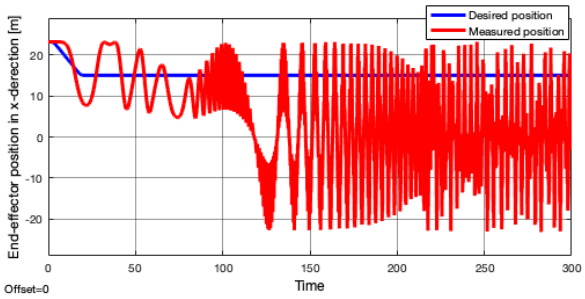

Figure 6.9: Comparison of desired and measured end-effector position in x-direction using Ziegler-Nichols parameters for PI-control. The blue curve is the desired and the red curve is the measured end-effector position in x-direction.

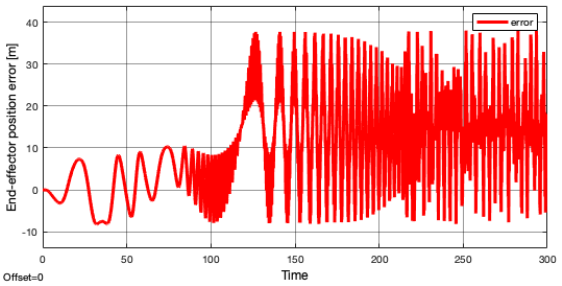

Figure 6.10: Error between desired and measured end-effector position in x-direction using Ziegler-Nichols parameters for PI-control.

The figures 6.10 shows error between desired and measured end-effector position which begins to increase rapidly after 100s and reaches 38m.

Figure 6.11 and 6.12 shows new results where K_p have been increased. It was done to minimize error between desired and measured end-effector position.

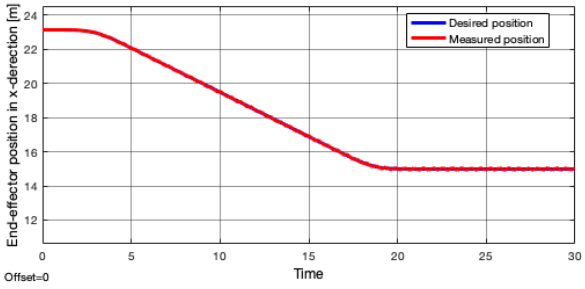

Figure 6.11: Comparison of desired and measured end-effector position in x-direction using increased gains for PI-control.

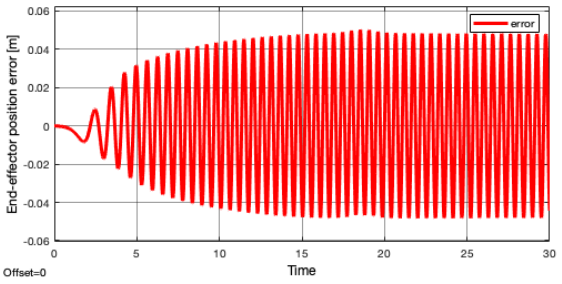

Figure 6.12: Error between desired and measured end-effector position in x-direction using Ziegler-Nichols parameters for PI-control.

From figure 6.12, the maximal error between desired and measured end-effector position is 50.22mm.

6.2 Discussion of control results for crane model in Simulink

The design of the PID controller succeed to track the desired end-effector position, and keeps the error between the desired and measured position very small. When using the design of the PD controller, the results of controlling the tracking of the desired end-effector position were better than the results from PID controller. The reason was the integral part was removed which leads to reduce the error. The results from both the PID when running simulation shows that the maximum error is $3.2m$ see figure 6.2, and when the PID controller parameters were increased with a huge values, the error becomes very small $2.735mm$ see figure 6.4. The results from the PD shows that the maximum error is $1.8m$ see figure 6.6, and when the PD controller parameters were increased with a huge values, the error becomes very small $2.735mm$ see figure 6.8. It is obvious that the error from PD is smaller than the error from the PID.

When the PI controller was used to control the end-effector and when at 100 second the system starts to oscillate as it seen in figure 6.9, in addition of the oscillations the error increases as the time increases see the figure 6.10 (larger than others controllers), this means the system might be unstable when time increasing. When the PI controller parameters were increased with a huge values, the error becomes very small $50.22mm$ see figure 6.12 but it still are much higher than PID and PD results.

6.3 Control of Joints for crane model in Simscape

Results from simulation of the PID-controller design include measured joint angles versus desired joint angles and the error between desired and measured joint angles. Controller parameters for the PID-controller design are found from testing several values. The best results were obtained when K_p, K_i and K_d where increased to their limits, which eventually resulted in very high gains.

Figure 6.13 shows simulation results of the measured and desired joints angles by using the PD-controller design with very high gains and Figure 6.14 shows simulation result of the error between desired and measured position.

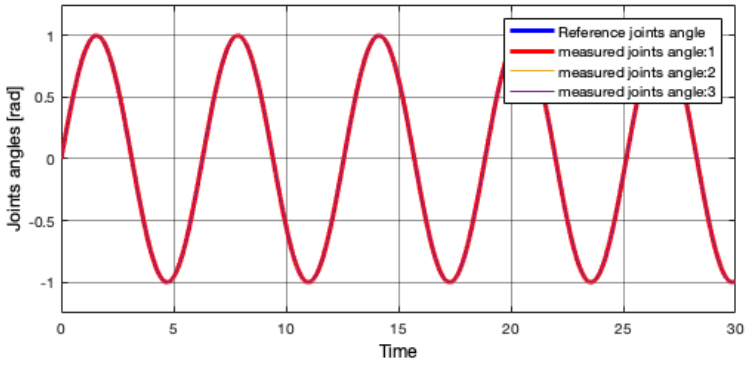

Figure 6.13: Measured joint angles versus desired joint angles using PID-controller with gravity compensation

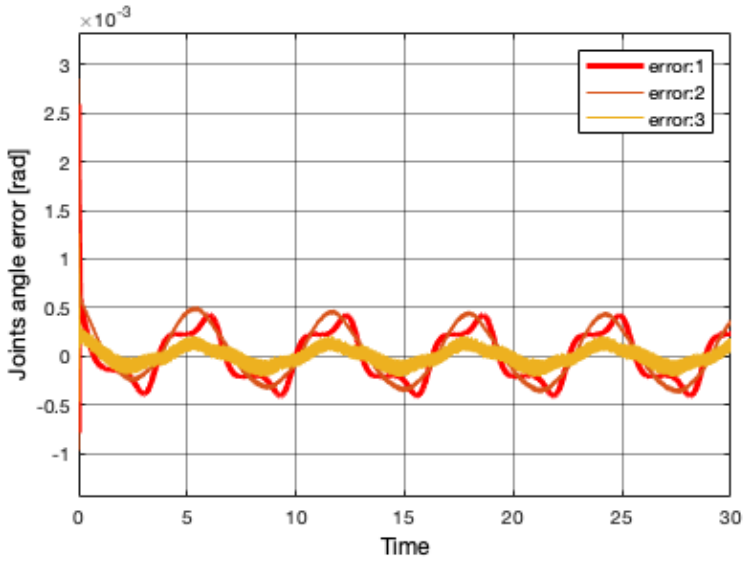

Figure 6.14: Error between desired and measured joint angles using PID-controller with gravity compensation

The maximum error for the joint 1 is 2.6mm, joint 2 2.8mm and joint 3 1.2mm, as it can be seen in figure 6.14.

6.4 Control of End-effector for crane model in Simscape

A proportional, Integral and Derivative (PID) is used to control the end-effector of the crane mdel in Simscape. The reference input signal is s-shape trajectory planning to simulate the system. A huge controller gains were used to verify that the desired end-effector and measured end-effector poisition and keep the error as small as possible.

Figure 6.15 shows simulation results of the measured and desired position in x-direction by using the PID-controller design with very high gains and Figure 6.16 shows simulation result of the error between desired and measured position.

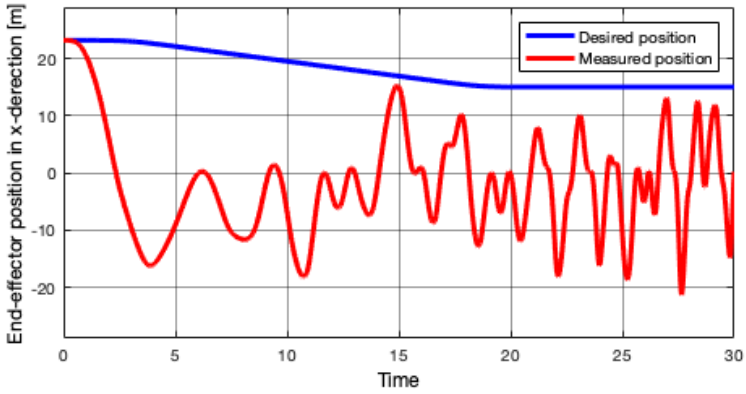

Figure 6.15: Measured versus desired end-effector position using PID-controller with gravity compensation

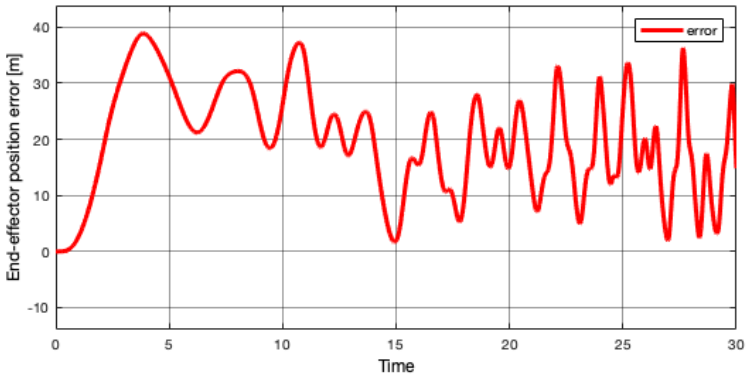

Figure 6.16: Error between desired and measured end-effector position using PID-controller with gravity compensation

6.5 Discussion of control results for crane model in Simscape

The results from the crane joints were the same as the sine wave input single with PID controller, this means that the Simscape model verified see figure 6.13. The error was $2.6mm$, $2.8mm$ and $1.2mm$ for the joint 1,2 and 3 respectively as seen in figure 6.14.

When controlling the end-effector by using trajectory planning and PID controller the error between the desierd and measured end-effector position was large see figure 6.16, this could be because of PID controller and because of lack of time we were not able to design or use another control method to control the crane and get better result, and using of Ziegler-Nichols Tuning method was not possible.

The trajectory planning could the other reason because it designed in horizontal direction(x), it may help if the trajectory planning was in three directions.

The proportional, derivative (PD) and proportional, integral (PI) controllers were used to control the system with different parameters gains and the results were the same with no change.

Chapter 7

Conclusions and Future Work

7.1 Conclusions

The aims of this project are to model, simulate and control a marine crane operations. Through the project a mathematically model of the crane were developed using robot modeling theory.

It was created two crane models: first, Simulink model and second, Simscape model. The Simulink model was developed based on crane dynamic equations and the Simscape model was developed using the Simmechanics library (Simscape Multi-body). The tow crane models was utilized as a plant for different control systems (PID, PD and PI).

A trajectory planning scheme is developed to put forward to generate a desired trajectory for the payload transportation process.

The trajectory planning scheme is combined with the PID controller algorithm to control the crane in x-direction.

Crane end-effector was controlled in dynamic crane simulink model and best simulation results with small error was received using the PID and PD controllers to control of end-effector.

A Simscape model for the crane was developed using simMechanics (Simscape Multibody), and this model was used as a virtual prototype software. The Simscape crane model includes body0, body1, body3, pulleys, cable and payload.

The crane joints angels was controlled in simscape model by using PID-controller with high gain and simulation results showed that were typically same, that means the Simscape model is able to be controlled.

The simulation results from controlling end-effector in Simscape model using PID, PD and PI controllers were not good enough because the error between the desired end-effector position and measured end-effector was big.

7.2 Further Work

Improvements on this thesis could to try to control the crane in 3D directions (x,y,z) with different operational positions and conditions.

The only PID, PD and PI controllers were tested in this project, thus the more advanced controller would be interesting.

In reality an offshore cranes work in rough conditions and effected by other forces and in this project we did not get access to software that can model the marine craft and vessel, further work could be to combine the Simscape model with the vessel model and marine craft model.

The payload with fixed cable length is controlled in this project, further can payload with variable cable investigate and all able forces.

Chapter 8

Appendix

8.1 Matlab script for Forward Kinematics

```
function [x,y,z]= fcn(q0,q1,q2)

%____________________Data Given_____________________________

L1=4.4; %Length of the first link
L2=15; %Length of the second link
L3=8.15; %Length og the third link

c0=cos(q0);
c1=cos(q1);
c12=cos(q1+q2);

s0=sin(q0);
s1=sin(q1);
s12=sin(q1+q2);

%____________________End-effector position_____________________________

x=c0.*(L2.*c1+L3.*c12);
y=s0.*(L2.*c1+L3.*c12);
z=L1+L2.*s1+L3.*s12;
```

8.2 Matlab script for Inverse Kinematics

```
function [q0,q1,q2]= fcn(x,y,z)

%____________________Data Given_____________________________

L1=4.4; %Link offset; variable joint parameter for joint 1
L2=15; %Length of the second link
L3=8.15; %Length og the third link

%_____________joint2 Kinematic_____________________________
x_p=sqrt(x.^2+y.^2);
z_p=z-L1;

c2=(x_p.^2+z_p.^2-L2.^2-L3.^2)/(2*L2*L3);
s2=-sqrt(1-c2.^2);

```

```
16 q2=atan2(s2,c2);
17
18
19 %_____joint1 Kinematic_____
20 q1=atan2(z_p,x_p)- atan2((L3*sin(q2)),(L2+L3*cos(q2)));
21
22
23
24
25 %_____joint0 Kinematic_____
26 q0=atan2(y,x);
```

8.3 Matlab script for Inertia matrix

```
1  function M_inverse = fcn(q0,q1,q2)
2
3  %%%%%%%_Data_%%%%%%%%
4
5  c1=cos(q0);
6  s1=sin(q0);
7  c2=cos(q1);
8  s2=sin(q1);
9  c3=cos(q2);
10 s3=sin(q2);
11 c23=cos(q1+q2);
12 s23=sin(q1+q2);
13
14 % Inertia Tensor and mass of Body 1
15 m1=267880;
16 I1x=534248;
17 I1y=1.01755e+06;
18 I1z=707533;
19
20
21 % Inertia Tensor and mass of Body 2
22 m2=134427;
23 I2x=65191.9;
24 I2y=1.83276e+06;
25 I2z=1.85226e+06;
26
27 % Inertia Tensor and mass of Body 3
28 m3=25380.7;
29 I3x=20975.5;
30 I3y=171449;
31 I3z=170786;
32
33 % Length of link 2 and 3
34 l2=15;
35 l3=8.15;
36
37
38 % Equations for the inertia matrix
39 M11=I1y+I2x*s2^2+I2y*c2^2+I3x*s23^2+I3y*c23^2+m2*(1/2*l2)^2*c2^2+m3
       *(1/2*c23+l2*c2)^2;
```

```
M12=0;
M21=0;
M13=0;
M31=0;
M22=I2z+I3z+m2*(1/2*l2)^2+m3*((1/2*l3)^2+l2^2+l2*l3*c3);
M23=I3z+m3*((1/2*l3)^2+1/2*l2*l3*c3);
M32=M23;
M33=I3z+m3*(1/2*l3)^2;

% Inertia matrix
M=[M11 M12 M13;...
     M21 M22 M23;...
     M31 M32 M33];

% Inverse Inertia matrix
M_inverse=inv(M);
```

8.4 Matlab script for Coriolis and centripetal matrix

```
function C= fcn(q0,q1,q2,dq0,dq1,dq2)

%%%%%%%_Data_%%%%%%%%

c1=cos(q0);
s1=sin(q0);
c2=cos(q1);
s2=sin(q1);
c3=cos(q2);
s3=sin(q2);
c23=cos(q1+q2);
s23=sin(q1+q2);

% Inertia Tensor and mass of Body 1
m1=267880;
I1x=534248;
I1y=1.01755e+06;
I1z=707533;


% Inertia Tensor and mass of Body 2
m2=134427;
I2x=65191.9;
I2y=1.83276e+06;
I2z=1.85226e+06;

% Inertia Tensor and mass of Body 3
m3=25380.7;
I3x=20975.5;
I3y=171449;
I3z=170786;

% Length of link 2 and 3
l2=15;
l3=8.15;
```

```
36
37
38
39 % Equations for the Coriolis and centripetal matrix
40 C11=(s2*c2*(I2x-I2y)+c23*s23*(I3x-I3y)-m2*(1/2*l2)^2*c2*s2-m3*((1/2*l3)
       *c23+l2*c2)*((1/2*l3)*s23+l2*s2))*dq1+(c23*s23*(I3x-I3y)-m3*(1/2*l3)
       *s23*((1/2*l3)*c23+l2*c2))*dq2;
41 C12=(s2*c2*(I2x-I2y)+c23*s23*(I3x-I3y)-m2*(1/2*l2)^2*c2*s2-m3*((1/2*l3)
       *c23+l2*c2)*((1/2*l3)*s23+l2*s2))*dq0;
42 C13=(c23*s23*(I3x-I3y)-m3*(1/2*l3)*s23*((1/2*l3)*c23+l2*c2))*dq0;
43 C21=-C12;
44 C22=-1/2*m3*l2*l3*s3*dq2;
45 C23=-1/2*m3*l2*l3*s3*dq1-m3*(1/2*l3)*l2*s3*dq2;
46 C31=-C13;
47 C32=1/2*m3*l2*l3*s3*dq1;
48 C33=0;
49
50 % Coriolis and centripetal matrix
51 C=[C11*dq0+C12*dq1+C13*dq2;...
52       C21*dq0+C22*dq1+C23*dq2;...
53       C31*dq0+C32*dq1+C33*dq2];
```

8.5 Matlab script for Gravity vector

```
1  function G = fcn(q0,q1,q2)
2
3
4  %%%%%%_Data_%%%%%%%%
5
6  c1=cos(q0);
7  s1=sin(q0);
8  c2=cos(q1);
9  s2=sin(q1);
10 c3=cos(q2);
11 s3=sin(q2);
12 c23=cos(q1+q2);
13 s23=sin(q1+q2);
14
15 % The gravity acceleration
16 g=-9.81;
17
18 % Mass of Body 2 and 3
19 m2=134427;
20 m3=25380.7;
21
22 % Length of link 2 and 3
23 l2=15;
24 l3=8.15;
25
26 % The gravity vector
27 G=[0;...
28       m2*g*1/2*l2*c2+m3*g*1/2*l3*c23+l2*c2;...
29       m3*g*1/2*l3*c23];
```

Bibliography

[1] Syvertsen, P.G.Modeling and Control of Crane on Offshore Vessel. NTNU, Trondheim, Norway. -, 2011.

[2] *Forward kinematics*. Retrieved 27.11.19, from https://www.codeproject.com/Articles/756070/Simulator-Axis-Articulated-Robotsl. 2019.

[3] Magnus Berthelsen Kjelland. *MAS501: Crane Kinematics*. -, 2013.

[4] Yiming Fu Ning Sun Yongchun Fang and Biao Lu. "Nonlinear Stabilizing Control for Ship Mounted Cranes with Ship Roll and Heave Movements: Design, Analysis and Experiments." In: *IEEE/ASME Transactions on Mechatronics* 48.10 (2018), pp. 1781–1793.

[5] Yudong Zhang Ning Sun Yongchun Fang and Bojun Ma. "a novel kinematic coupling based trajectory planning method for overhead cranes." In: *IEEE/ASME Transactions on Mechatronics* 17.01 (2012), pp. 166–173.

[6] F. Heltha R.S. Arvin M.I Solihin. "Modeling and Control Desgin for Rotary Crane System Using Matlab Smuscape Toolbox." In: *IEEE 5th control and system graduate research colloquium* 11.05 (2014), pp. 170–175.