Control of Non-Holonomic Mobile Robot

Vaibhav Gupta 2013ME10735
B. Nalin 2013ME10652

Under the guidance of

Prof. I. N. Kar
Prof. S. K. Saha

Indian Institute of Technology, Delhi

This report is submitted for
Mini-Project in Robotics (JRD-301)

May 2017

Abstract

The use, design and fabrication of Wheeled Mobile Robots (WMRs) is on a surge both in industry and academia. Students at IIT-Delhi also build various mobile robots each year, for performing a variety of tasks. But, often a lot of time is wasted on control and trajectory tracking of the robot rather than on its core functionality. Moreover, the control scheme used by students is generally based on achieving controller parameters by trial-and-error and hence the parameters have to be tuned for each different trajectory. In this project, we intend to build a general adaptive controller which could be used to control a non-holonomic robot with suitable sensors.

This report summarizes the course of action taken till the end-semester evaluation of this project. This involves formulation of kinematic and dynamic model of the robot and subsequent computed-torque controller design. 'Pure Pursuit Tracking' methodology was used for trajectory tracking. The controller has 3 essential components, an outer controller which operates on the tracking error, an inner controller which does PID control on the velocity, and a non-linear compensator which applies inverse dynamics control to calculate motor torques. A Simulink Model was developed for the robot system, the controller and the input trajectory generator assuming ideal sensors. The model behaviour was also acceptable when random errors were introduced in the computed torque. A physical model has also been manufactured to test our control algorithms.

Further, we intend to make the controller robust and adaptive and use a suitable filter to make the sensor data more usable.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

A mobile robot is an uncertain nonlinear dynamic multiple-input multiple-output (MIMO) system which suffers from structured and unstructured uncertainties. Hence, the real-time control of WMRs is a challenging task. A non-holonomic robot system, characterized by non-integrable kinematic constraints makes the dynamic formulation and controller design even more challenging.

1.1 Scope and Flow of the Report

This document presents the work carried out till the end-semester presentation in the 'Control of Non-Holonomic Mobile Robot' project, which has been undertaken as a part of the course JRD-301. The CAD models developed as part of the project can be readily used to fabricate the robot. The simulation model can be used for any mobile robot with the assumed geometrical model. The accuracy of path-tracking and other results are discussed in later chapters.

Chapter 1 discusses the key outcomes of various books and publications related to the topic and specifies how they have been used in this work. It also describes the basic geometric model used to visualize and later fabricate the robot.

Chapter 2 defines the generalized coordinates and corresponding kinematic constraints. It details-out the kinematics and dynamics of the robot. Lagrangian dynamics is used to derive an expression for required torques in terms of the generalized coordinates.

Chapter 3 formulates the trajectory tracking problem for a set of discrete path points.

Chapter 4 presents the proposed control structure which includes an outer controller for tracking control, an inner controller for velocity control and a non-linear compensator for inverse dynamic control.

Chapter 5 presents the simulation model and simulation results for two dynamic parameter sets each for two separate trajectories.

Chapter 6 finally concludes the report with a summary of the simulation results and the work remaining to be done in future.

1.2 Literature Review

Kinematics and dynamics of industrial serial-chain robots was discussed in [1]–[3]. Several of these concepts were applicable for mobile robots as well.

[4] discussed wheeled mobile robots in detail and presented their kinematics and several control techniques. It also presented techniques to create a sensor model and use filtering on sensor data.

[5] discusses dynamic model of WMR with toroidal wheel on uneven surface. Many of the concepts presented here were used to develop constrained dynamic equations for our model of the robot.

We used computed-torque (inverse dynamics) control for our robot. This involves control-law partitioning, where the model is used to calculate torque using the control input and the control input is generated independent of the model using desired and sensor inputs. The stability of the controller is proved using Lyapunov direct method. This was presented in [1]–[3], [6]. Principles of adaptive and robust controllers are also discussed here.

Modeling of non-holonomic WMR is done in [7]–[9]. They presented kinematic constraints and lagrangain dynamic formulation. [7] proposes an adaptive controller based on fuzzy logic, whereas [9] provides the controller design for a state-space feedback controller. [8] solves the tracking problem using a back-stepping technique.

For formulating the trajectory tracking problem, we have used the Pure Pursuit Algorithm given in [10]. [11] provides a stable tracking control rule for non-holonomic robots by dividing the linear control scheme into outer control (K_x, K_y K_θ) and inner control (K_p, K_d).

1.3 Geometric Model

Here we consider the following geometric model of a non-holonomic WMR.

The robot is assumed to move in a 2-D plane. X-Y is the global coordinate-system and x-y is the body-fixed coordinate system, with origin at A. The y-axis is defined as the line passing through the axes of the rear wheels, and the x-axis is defined along the perpendicular from the front wheel to y-axis. Directions are as shown in figure. The center of mass of the robot lies at C.

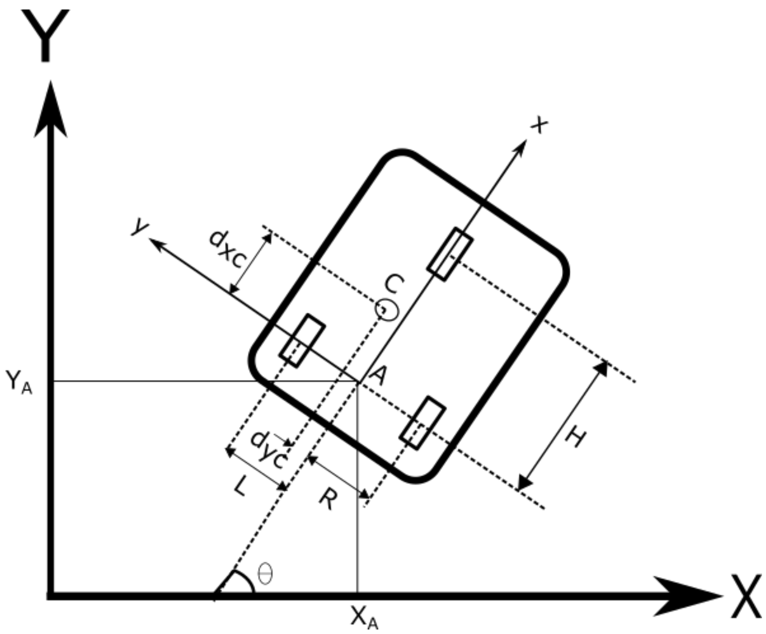

Figure 1.1 Geometric model of the robot system

Two motors are used to actuate the two rear non-holonomic wheels. Since this is a non-holonomic system, the robot can reach a given configuration (given by X_A, Y_A and θ) with the help of just these two actuators.

The front wheel is non-actuated, and should also not be interpreted as a steering wheel. It is a holonomic wheel (omni-wheel), mounted on a shaft parallel to the y-axis, such that an encoder mounted on such a shaft gives us the velocity of the robot in x-direction. Note that, the robot doesn't have any velocity in the (body-fixed) y-direction.

1.4 Physical Model

The solid modeling was done on SolidWorks. Fig.**??** shows the isometric view of the solid model assembly.

The different parts of the assembly were made out of acrylic using laser-cutting and fixed together using threaded fasteners as well as the inherent locking system of the parts.

1.4.1 Feedback Sensors

Two sensors are used for feedback. An encoder mounted on the shaft of the front holonomic wheel gives the angular position of the shaft, which when multiplied by the radius of front wheel (and divided by time) would give us the velocity of the front wheel (and hence that

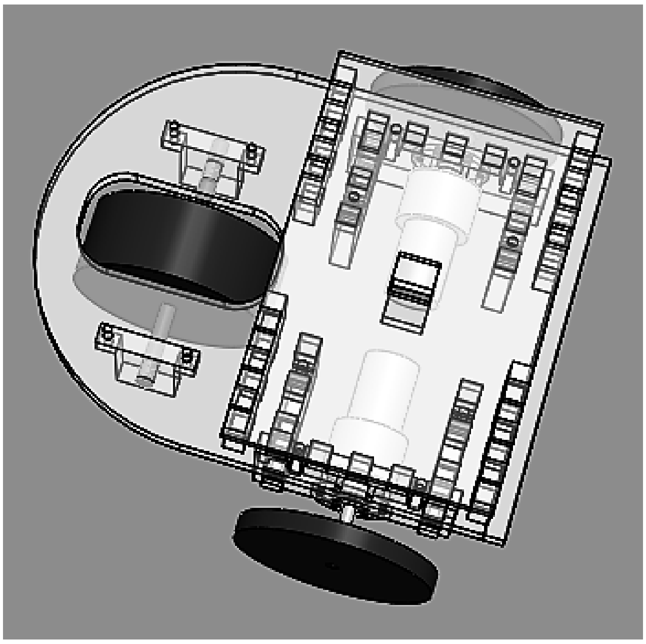

Figure 1.2 Solid model of the robot assembly

of the robot) in x-direction. This choice for the placement of the sensor seems appropriate considering the fact that a slip in non-actuated wheel will be less than that in an actuated one. The second sensor would be a gyro, mounted preferably at the center of mass of the system. The gyro would provide angular velocity of the system.

The sensors and motors used are the following:

Motors RKI-1188 High torque 12V DC 300rpm (ROBOKITS)
Encoder AMT112S-2048-5000-W (CUI Inc.)
Gyro MPU-6050 Break-out board (InvenSense)

Chapter 2

Model of Non-Holonomic Robot

2.1 Kinematics

Let $\dot{x}_A$ represent the velocity of point A on the robot in x direction and $\dot{y}_A$ represent that in y. Similarly let $\dot{X}_A$ and $\dot{Y}_A$ be the denote the velocity in the global frame. The relation between these quantities can be specified using a rotation matrix:

$$\begin{bmatrix} \dot{x}_A \\ \dot{y}_A \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} \dot{X}_A \\ \dot{Y}_A \end{bmatrix} \tag{2.1}$$

We know that the system doesn't have any velocity in the body-fixed y axis. Hence we get the following kinematic constraint:

$$-\dot{X}_A \sin\theta + \dot{Y}_A \cos\theta = 0 \tag{2.2}$$

Assuming no-slip conditions at all three wheels, we get the following constraints corresponding to left, right and front wheels respectively:

$$\dot{x}_L = \dot{x} - L\dot{\theta} = \dot{X}_A \cos\theta + \dot{Y}_A \sin\theta - L\dot{\theta} = R_L \dot{\theta}_L \tag{2.3}$$

$$\dot{x}_R = \dot{x} + R\dot{\theta} = \dot{X}_A \cos\theta + \dot{Y}_A \sin\theta + R\dot{\theta} = R_R \dot{\theta}_R \tag{2.4}$$

$$\dot{x}_3 = \dot{x} = \dot{X}_A \cos\theta + \dot{Y}_A \sin\theta = R_3 \dot{\theta}_3 \tag{2.5}$$

Here, R_L, R_R and R_3 are the radii of the left, right and front wheels respectively. L and R are distances to the left and right wheels from point A.

Using the the four kinematic constraints given above, we can form the constraint Jacobian $\boldsymbol{A}(\boldsymbol{q})$ such that $\boldsymbol{A}(\boldsymbol{q})\dot{\boldsymbol{q}} = 0$:

$$\begin{bmatrix} -\sin\theta & \cos\theta & 0 & 0 & 0 & 0 \\ \cos\theta & \sin\theta & -L & -R_L & 0 & 0 \\ \cos\theta & \sin\theta & R & 0 & -R_R & 0 \\ \cos\theta & \sin\theta & 0 & 0 & 0 & -R_3 \end{bmatrix} \begin{bmatrix} \dot{X}_A \\ \dot{Y}_A \\ \dot{\theta} \\ \dot{\theta}_L \\ \dot{\theta}_R \\ \dot{\theta}_3 \end{bmatrix} = \mathbf{0} \tag{2.6}$$

Since our sensor only provide the velocity of the robot in its body-fixed x-axis and the angular velocity of the robot, i.e. $\dot{x}_A$ and θ, we need to be able to express $\boldsymbol{q}$ in terms of these two quantities. This can be done, again by using the kinematic constraints.

$$\dot{\boldsymbol{q}} = \begin{bmatrix} \dot{X}_A \\ \dot{Y}_A \\ \dot{\theta} \\ \dot{\theta}_L \\ \dot{\theta}_R \\ \dot{\theta}_3 \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \\ \frac{1}{R_L} & \frac{-L}{R_L} \\ \frac{1}{R_R} & \frac{R}{R_R} \\ 0 & R_3 \end{bmatrix} \begin{bmatrix} \dot{x}_A \\ \dot{\theta} \end{bmatrix} = \boldsymbol{S}(\boldsymbol{q})\boldsymbol{\eta} \tag{2.7}$$

For doing dynamics of the system, we would also require the position and velocity of the center of mass of system. Let the center of mass be fixed at a point C whose coordinates are (d_{xc}, d_{yc}) in the body fixed coordinate system. Its position in the global frame is given by:

$$X_C = X_A + d_{xc}\cos\theta - d_{yc}\sin\theta \tag{2.8}$$

$$Y_C = Y_A + d_{xc}s\theta + d_{yc}s\theta \tag{2.9}$$

and its velocity in global frame will be:

$$\dot{X}_C = \dot{X}_A - d_{xc}\dot{\theta}\sin\theta - d_{yc}\dot{\theta}\cos\theta \tag{2.10}$$

$$\dot{Y}_C = \dot{Y}_A + d_{xc}\dot{\theta}\cos\theta - d_{yc}\dot{\theta}\sin\theta \tag{2.11}$$

2.2 Dynamics

Euler-Lagrange equation is used to create the dynamic model of the robot.

$$\frac{d}{dt}\left(\frac{\partial \mathscr{L}}{\partial \dot{\boldsymbol{q}}}\right) - \frac{\partial \mathscr{L}}{\partial \boldsymbol{q}} = \boldsymbol{\tau} \tag{2.12}$$

where,

$$\mathscr{L} = \mathscr{T} - \mathscr{U}$$
$$\mathscr{T} = \text{Kinetic Energy of the system}$$
$$\mathscr{U} = \text{Potential Energy of the system}$$

If kinematic constraint of the form 2.13 is applied to the system then the equation 2.12 gets modified to the form 2.14

$$\boldsymbol{A}(\boldsymbol{q})\dot{\boldsymbol{q}} = \boldsymbol{0} \tag{2.13}$$

$$\frac{d}{dt}\left(\frac{\partial \mathscr{L}}{\partial \dot{\boldsymbol{q}}}\right) - \frac{\partial \mathscr{L}}{\partial \boldsymbol{q}} = \boldsymbol{B}(\boldsymbol{q})\boldsymbol{\tau} - \boldsymbol{A}^T(\boldsymbol{q})\boldsymbol{\lambda} \tag{2.14}$$

From section 2.1, we know the value of $\boldsymbol{A}(\boldsymbol{q})$ with the generalized coordinates of the form,

$$\boldsymbol{q} = \begin{bmatrix} X_a & Y_a & \theta & \theta_L & \theta_R & \theta_3 \end{bmatrix}^T \tag{2.15}$$

Kinetic energy of the system could be written as,

$$\mathscr{T} = \mathscr{T}_p + \mathscr{T}_L + \mathscr{T}_R + \mathscr{T}_3$$

where,

$$\mathscr{T}_p = \frac{1}{2}m_p\left(\dot{X}_A^2 + \dot{Y}_A^2 + \left(d_{x_c}^2 + d_{y_c}^2\right)\dot{\theta}^2 - 2d_{y_c}\dot{\theta}\left(\dot{X}_A\cos\theta + \dot{Y}_A\sin\theta\right)\right) + \frac{1}{2}I_p\dot{\theta}^2$$
$$\mathscr{T}_L = \frac{1}{2}m_L\left(\dot{X}_A^2 + \dot{Y}_A^2 - 2L\dot{\theta}\left(\dot{X}_A\cos\theta + \dot{Y}_A\sin\theta\right)\right) + \frac{1}{2}I_L\dot{\theta}^2 + \frac{1}{2}I_{w_L}\dot{\theta}_L^2$$
$$\mathscr{T}_R = \frac{1}{2}m_R\left(\dot{X}_A^2 + \dot{Y}_A^2 + 2R\dot{\theta}\left(\dot{X}_A\cos\theta + \dot{Y}_A\sin\theta\right)\right) + \frac{1}{2}I_R\dot{\theta}^2 + \frac{1}{2}I_{w_R}\dot{\theta}_R^2$$
$$\mathscr{T}_3 = \frac{1}{2}m_3\left(\dot{X}_A^2 + \dot{Y}_A^2 + 2H^2\dot{\theta}\right) + \frac{1}{2}I_3\dot{\theta}^2 + \frac{1}{2}I_{w_3}\dot{\theta}_3^2$$

On simplification,

$$\mathscr{T} = \frac{1}{2}m\left(\dot{X}_A^2 + \dot{Y}_A^2\right) + \frac{1}{2}I\dot{\theta}^2 + \bar{m}\dot{\theta}\left(\dot{X}_A\cos\theta + \dot{Y}_A\sin\theta\right) + \frac{1}{2}I_{w_L}\dot{\theta}_L^2 + \frac{1}{2}I_{w_R}\dot{\theta}_R^2 + \frac{1}{2}I_{w_3}\dot{\theta}_3^2 \tag{2.16}$$

where,

$$\begin{aligned} m &= m_p + m_L + m_R + m_3 \\ I &= I_p + I_L + I_R + I_3 + m_p\left(d_{x_c}^2 + d_{y_c}^2\right) + m_3H^2 \\ \bar{m} &= -m_pd_{y_c} - m_LL + m_RR \end{aligned}$$

On a flat surface, we can assume potential energy of the system to be constant (say 0), i.e., $\mathscr{U} = 0$ which results in $\mathscr{L} = \mathscr{T}$. On differentiation we get,

$$\frac{d}{dt}\left(\frac{\partial\mathscr{L}}{\partial\dot{\boldsymbol{q}}}\right) = \begin{bmatrix} m\ddot{X}_A + \bar{m}\ddot{\theta}\cos\theta - \bar{m}\dot{\theta}^2\sin\theta \\ m\ddot{Y}_A + \bar{m}\ddot{\theta}\sin\theta + \bar{m}\dot{\theta}^2\cos\theta \\ I\ddot{\theta} + \bar{m}\left(\ddot{X}_A^2\cos\theta + \ddot{Y}_A^2\sin\theta - \dot{X}_A\dot{\theta}\sin\theta + \dot{Y}_A\dot{\theta}\cos\theta\right) \\ I_{w_L}\ddot{\theta}_L \\ I_{w_R}\ddot{\theta}_R \\ I_{w_3}\ddot{\theta}_3 \end{bmatrix}$$

$$\frac{\partial\mathscr{L}}{\partial\boldsymbol{q}} = \begin{bmatrix} 0 \\ 0 \\ \bar{m}\dot{\theta}\left(-\dot{X}_A\sin\theta + \dot{Y}_A\cos\theta\right) \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \{\text{Refer section 2.1}\}$$

On simplification, we get the dynamic model of the robot in the form,

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{V}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} = \boldsymbol{B}(\boldsymbol{q})\boldsymbol{\tau} - \boldsymbol{A}^T(\boldsymbol{q})\boldsymbol{\lambda} \tag{2.17}$$

where,

$$\boldsymbol{M}(\boldsymbol{q}) = \begin{bmatrix} m & 0 & \bar{m}\cos\theta & 0 & 0 & 0 \\ 0 & m & \bar{m}\sin\theta & 0 & 0 & 0 \\ \bar{m}\cos\theta & \bar{m}\sin\theta & I & 0 & 0 & 0 \\ 0 & 0 & 0 & I_{w_L} & 0 & 0 \\ 0 & 0 & 0 & 0 & I_{w_R} & 0 \\ 0 & 0 & 0 & 0 & 0 & I_{w_3} \end{bmatrix}$$

$$\boldsymbol{V}(\boldsymbol{q},\dot{\boldsymbol{q}}) = \begin{bmatrix} 0 & 0 & -\bar{m}\dot{\theta}\sin\theta & 0 & 0 & 0 \\ 0 & 0 & \bar{m}\dot{\theta}\cos\theta & 0 & 0 & 0 \\ -\dot{\theta}\sin\theta & \dot{\theta}\cos\theta & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\boldsymbol{B}(\boldsymbol{q}) = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}$$

$$\boldsymbol{A}^T(\boldsymbol{q}) = \begin{bmatrix} -\sin\theta & \cos\theta & \cos\theta & \cos\theta \\ \cos\theta & \sin\theta & \sin\theta & \sin\theta \\ 0 & -L & R & 0 \\ 0 & -R_L & 0 & 0 \\ 0 & 0 & -R_R & 0 \\ 0 & 0 & 0 & -R_3 \end{bmatrix}$$

However, it was found that for the definition of the tracking problem, it is much easier to use the following generalized coordinates. (Refer chapter 3)

$$\boldsymbol{q} = \begin{bmatrix} X_A & Y_A & \theta \end{bmatrix}^T \tag{2.18}$$

To find this we replace $\dot{\theta}_L$, $\dot{\theta}_R$ and $\dot{\theta}_3$ in terms of X_A, Y_A and θ in equation 2.16 to give,

$$\mathscr{T} = \frac{1}{2}m\left(\dot{X}_A^2 + \dot{Y}_A^2\right) + \frac{1}{2}k\left(\dot{X}_A\cos\theta + \dot{Y}_A\sin\theta\right)^2 + \frac{1}{2}I\dot{\theta}^2 + \bar{m}\dot{\theta}\left(\dot{X}_A\cos\theta + \dot{Y}_A\sin\theta\right) \tag{2.19}$$

where,

$$
\begin{aligned}
m &= m_p + m_L + m_R + m_3 \\
I &= I_p + I_L + I_R + I_3 + m_p\left(d_{x_c}^2 + d_{y_c}^2\right) + m_3 H^2 + \frac{L^2 I_{w_L}}{R_L^2} + \frac{R^2 I_{w_R}}{R_R^2} \\
\bar{m} &= -m_p d_{y_c} - m_L L + m_R R - \frac{L I_{w_L}}{R_L^2} + \frac{R I_{w_R}}{R_R^2} \\
k &= \frac{I_{w_L}}{R_L^2} + \frac{I_{w_R}}{R_R^2} + \frac{I_{w_3}}{R_3^2}
\end{aligned}
$$

On simplification, we get the dynamic model of the robot in the form,

$$
\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{V}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} = \boldsymbol{B}(\boldsymbol{q})\boldsymbol{\tau} - \boldsymbol{A}^T(\boldsymbol{q})\boldsymbol{\lambda} \tag{2.20}
$$

where,

$$
\begin{aligned}
\boldsymbol{M}(\boldsymbol{q}) &= \begin{bmatrix} m + k\cos^2\theta & k\cos\theta\sin\theta & \bar{m}\cos\theta \\ k\cos\theta\sin\theta & m + k\sin^2\theta & \bar{m}\sin\theta \\ \bar{m}\cos\theta & \bar{m}\sin\theta & I \end{bmatrix} \\
\boldsymbol{V}(\boldsymbol{q},\dot{\boldsymbol{q}}) &= \begin{bmatrix} -k\dot{\theta}\cos\theta\sin\theta & -k\dot{\theta}\sin^2\theta & -\bar{m}\dot{\theta}\sin\theta \\ k\dot{\theta}\cos^2\theta & k\dot{\theta}\cos\theta\sin\theta & \bar{m}\dot{\theta}\cos\theta \\ -\dot{\theta}\sin\theta & \dot{\theta}\cos\theta & 0 \end{bmatrix} \\
\boldsymbol{B}(\boldsymbol{q}) &= \begin{bmatrix} \frac{\cos\theta}{R_L} & \frac{\cos\theta}{R_R} \\ \frac{\sin\theta}{R_L} & \frac{\sin\theta}{R_R} \\ -\frac{L}{R_L} & \frac{R}{R_R} \end{bmatrix} \\
\boldsymbol{A}(\boldsymbol{q}) &= \begin{bmatrix} -\sin\theta & \cos\theta & 0 \end{bmatrix}
\end{aligned}
$$

Now, we consider a smooth matrix $\boldsymbol{S}(\boldsymbol{q})$ spanning over the null space of the matrix $\boldsymbol{A}(\boldsymbol{q})$ such that,

$$
\boldsymbol{S}^T(\boldsymbol{q})\boldsymbol{A}^T(\boldsymbol{q}) = \boldsymbol{0} \tag{2.21}
$$

For the current model,

$$
\boldsymbol{S}(\boldsymbol{q}) = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix} \tag{2.22}
$$

It can further shown that,

$$\dot{\boldsymbol{q}} = \boldsymbol{S}(\boldsymbol{q})\boldsymbol{\eta}_{(2\times 1)} = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{x}_A \\ \dot{\theta} \end{bmatrix} \tag{2.23}$$

where, $\dot{x}_A$ is the velocity of the WMR in forward direction and $\dot{\theta}$ is its angular velocity.

$\boldsymbol{S}^T(\boldsymbol{q})$ is left-multiplied to the equation 2.20 to get,

$$\boldsymbol{S}^T(\boldsymbol{q})\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{S}^T(\boldsymbol{q})\boldsymbol{V}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} = \boldsymbol{S}^T(\boldsymbol{q})\boldsymbol{B}(\boldsymbol{q})\boldsymbol{\tau} \tag{2.24}$$

The last term of equation 2.20 vanishes after pre-multiplication due to equation 2.21. This way we can avoid Lagrangian multipliers from our control structure.

Chapter 3

Tracking Problem Definition

For tracking, the algorithm being used is the ’Pure Pursuit Tracking Algorithm’ proposed in [10]. In this algorithm, we use the concept of look ahead to find the next position the robot has to reach.

We first find the point on the curve nearest to the robot. Then, another point on the curve is taken at a distance d from the previous point in forward direction. Now, the robot tries to reach this point during the current iteration of the curve.

Now, the problem has been converted into a set of destination problems where destination is defined in terms of $\boldsymbol{q}_d$, $\dot{\boldsymbol{q}}_d$ and $\ddot{\boldsymbol{q}}_d$. Further, this can simplified by using equation 2.23.

$$\dot{\boldsymbol{q}} = \boldsymbol{S}(\boldsymbol{q})\boldsymbol{\eta} \tag{3.1}$$

$$\ddot{\boldsymbol{q}} = \dot{\boldsymbol{S}}(\boldsymbol{q})\boldsymbol{\eta} + \boldsymbol{S}(\boldsymbol{q})\dot{\boldsymbol{\eta}} \tag{3.2}$$

So, the required parameters become $\boldsymbol{q}$, $\dot{x}_A$ (velocity in forward direction), $\dot{\theta}$ (angular velocity), $\ddot{x}_A$ (acceleration in forward direction) and $\ddot{\theta}$ (angluar acceleration).

Chapter 4

Controller Design

We are currently using inverse dynamics controller for the robot. The scheme for the same is shown in figure 4.1.

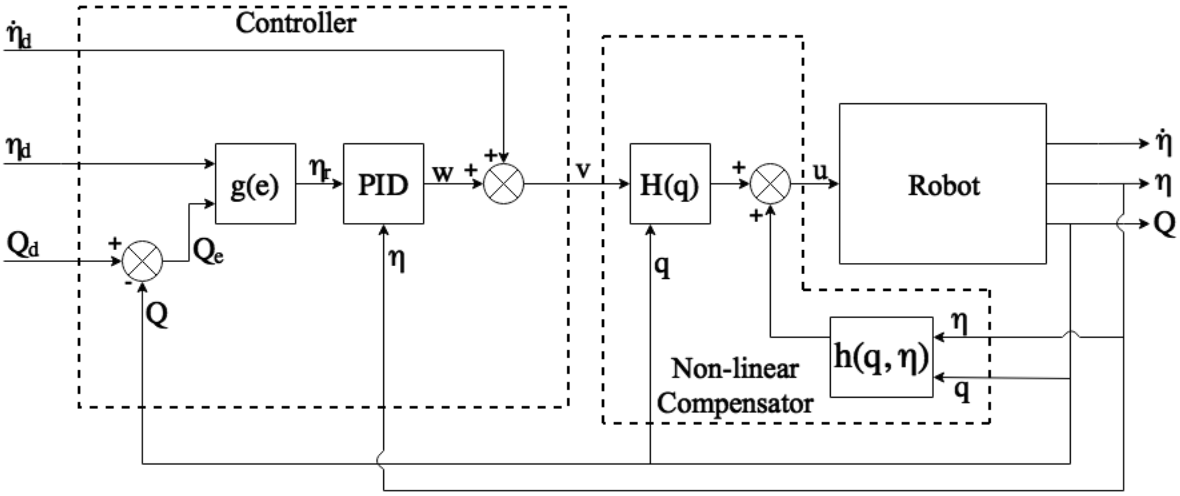

Figure 4.1 Scheme for Inverse Dynamics Controller

4.1 Outer Controller

The error in body-fixed frame can be expressed in terms of global current and desired coordinates as follows:

$$\boldsymbol{q_e} = \begin{bmatrix} x_e \\ y_e \\ \theta_e \end{bmatrix} = \begin{bmatrix} (X_d - X)cos\theta + (Y_d - Y)sin\theta \\ -(X_d - X)sin\theta + (Y_d - Y)cos\theta \\ \theta_d - \theta \end{bmatrix} \tag{4.1}$$

where the subscript d has been used to denote desired value.

The time derivative of the error can be computed as follows:

$$\boldsymbol{\dot{q}_e} = \begin{bmatrix} (\dot{X}_d - \dot{X})cos\theta + (\dot{Y}_d - \dot{Y})sin\theta - (X_d - X)\dot{\theta}sin\theta + (Y_d - Y)\dot{\theta}cos\theta \\ -(\dot{X}_d - \dot{X})sin\theta + (\dot{Y}_d - \dot{Y})cos\theta - (X_d - X)\dot{\theta}cos\theta - (Y_d - Y)\dot{\theta}sin\theta \\ \dot{\theta}_d - \dot{\theta} \end{bmatrix} = \begin{bmatrix} y_e\omega - v + v_d cos\theta_e \\ -x_e\omega + v_d sin\theta_e \\ \omega_d - \omega \end{bmatrix} \tag{4.2}$$

where v_d and v are the desired and actual velocities respectively in the local frame, whereas ω_d and ω are the desired and actual angular velocities respectively.

Now we state our control law as follows:

$$\boldsymbol{\eta} = \begin{bmatrix} v \\ \omega \end{bmatrix} = \begin{bmatrix} v_d cos\theta_e + K_x x_e \\ \omega_d + v_d(K_y y_e + K_\theta sin\theta_e) \end{bmatrix} \tag{4.3}$$

Using Eqn.4.2 and Eqn.4.3, we can derive a new expression for the time derivative of error vector $\boldsymbol{\dot{q}_e}$

$$\boldsymbol{\dot{q}_e} = \begin{bmatrix} (\omega_d + v_d(K_y y_e + K_\theta sin\theta_e))y_e - K_x x_e \\ -(\omega_d + v_d(K_y y_e + K_\theta sin\theta_e))x_e + v_d sin\theta_e \\ -v_d(K_y y_e + K_\theta sin\theta_e) \end{bmatrix} \tag{4.4}$$

Now, we propose the following scalar function Vas our Liapunov function candidate:

$$V = \frac{1}{2}(x_e^2 + y_e^2) + \frac{1 - cos\theta_e}{K_y} \tag{4.5}$$

Clearly, $V \geq 0$ and $V = 0 iff q_e = 0$

Further, using Eqn.4.4

$$\dot{V} = \dot{x_e}x_e + \dot{y_e}y_e + \dot{\theta}_e sin\theta_e/K_y = -K_x x_e^2 - v_d K_\theta sin^2\theta_e/K_y \tag{4.6}$$

Clearly $\dot{V} \leq 0. \therefore V$ becomes a Liapunov function. This also proves the stability of our control law.

We've chosen the following values for the constants:

$$K_x = 10/s \tag{4.7}$$

$$K_y = 64/m^2 \tag{4.8}$$

$$K_\theta = 16/m \tag{4.9}$$

4.2 Inner Controller and Non-linear Compensator

Equation 2.24 is written in form of,

$$\boldsymbol{H}(\boldsymbol{q})\dot{\boldsymbol{\eta}} + \boldsymbol{h}(\boldsymbol{q},\boldsymbol{\eta}) = \boldsymbol{u} \tag{4.10}$$

where,

$$\begin{aligned}
\boldsymbol{H}(\boldsymbol{q}) &= \left(\boldsymbol{S}^T(\boldsymbol{q})\boldsymbol{B}^{-1}(\boldsymbol{q})\right)\boldsymbol{S}^T(\boldsymbol{q})\boldsymbol{M}(\boldsymbol{q})\\
\boldsymbol{h}(\boldsymbol{q},\boldsymbol{\eta}) &= \left(\boldsymbol{S}^T(\boldsymbol{q})\boldsymbol{B}^{-1}(\boldsymbol{q})\right)\boldsymbol{S}^T(\boldsymbol{q})\boldsymbol{V}(\boldsymbol{q},\boldsymbol{\eta})\boldsymbol{\eta}\\
\boldsymbol{u} &= \boldsymbol{\tau}
\end{aligned}$$

Let estimates of $\boldsymbol{H}(\boldsymbol{q})$ and $\boldsymbol{h}(\boldsymbol{q})$ be $\hat{\boldsymbol{H}}(\boldsymbol{q})$ and $\hat{\boldsymbol{h}}(\boldsymbol{q})$ respectively.

$$\hat{\boldsymbol{H}}(\boldsymbol{q}) = \boldsymbol{H}(\boldsymbol{q}) + \Delta\boldsymbol{H}(\boldsymbol{q}) \tag{4.11}$$

$$\hat{\boldsymbol{h}}(\boldsymbol{q},\boldsymbol{\eta}) = \boldsymbol{h}(\boldsymbol{q},\boldsymbol{\eta}) + \Delta\boldsymbol{h}(\boldsymbol{q},\boldsymbol{\eta}) \tag{4.12}$$

Let,

$$\boldsymbol{u} = \hat{\boldsymbol{H}}(\boldsymbol{q})\boldsymbol{v} + \hat{\boldsymbol{h}}(\boldsymbol{q},\boldsymbol{\eta}) \tag{4.13}$$

Then,

$$\ddot{\boldsymbol{q}} = \boldsymbol{v} + \boldsymbol{H}^{-1}(\boldsymbol{q})\left(\Delta\boldsymbol{H}(\boldsymbol{q})\boldsymbol{v} + \Delta\boldsymbol{h}(\boldsymbol{q},\boldsymbol{\eta})\right) \tag{4.14}$$

We take the tracking error $\boldsymbol{e}$ as,

$$\boldsymbol{e} = \begin{bmatrix}\boldsymbol{\eta}_d - \boldsymbol{\eta}\end{bmatrix} \tag{4.15}$$

So,

$$\boldsymbol{v} = \dot{\boldsymbol{\eta}}_d + f(\boldsymbol{e}) \tag{4.16}$$

$$\dot{\boldsymbol{\eta}} = \dot{\boldsymbol{\eta}}_d + f(\boldsymbol{e}) + Z(\boldsymbol{q},\boldsymbol{\eta},\dot{\boldsymbol{\eta}},\boldsymbol{e}) \tag{4.17}$$

where,

$$Z(\boldsymbol{q},\boldsymbol{\eta},\dot{\boldsymbol{\eta}},\boldsymbol{e}) = \boldsymbol{H}^{-1}(\boldsymbol{q})\left(\Delta\boldsymbol{H}(\boldsymbol{q})\left(\dot{\boldsymbol{\eta}}_d + f(\boldsymbol{e})\right) + \Delta\boldsymbol{h}(\boldsymbol{q},\boldsymbol{\eta})\right)$$

Derivative of the tracking error $\dot{\boldsymbol{e}}$ is given by,

$$\dot{\boldsymbol{e}} = \left[\dot{\boldsymbol{\eta}}_d - \dot{\boldsymbol{\eta}}\right] = -\left(f(\boldsymbol{e}) + Z(\boldsymbol{q}, \boldsymbol{\eta}, \dot{\boldsymbol{\eta}}, \boldsymbol{e})\right) \tag{4.18}$$

If the system is defined such that there is no modeling error then,

$$Z(\boldsymbol{q}, \boldsymbol{\eta}, \dot{\boldsymbol{\eta}}, \boldsymbol{e}) = 0 \tag{4.19}$$

Under the aforementioned assumption, it could be easily proven that if $f(\boldsymbol{e})$ chosen in the form shown in equation 4.20 then the final state of the system is stable with zero error.

$$f(\boldsymbol{e}) = \boldsymbol{K}_P\boldsymbol{e} + \boldsymbol{K}_I \int \boldsymbol{e} + \boldsymbol{K}_D\dot{\boldsymbol{e}} \tag{4.20}$$

Using equations 4.17, 4.19 and 4.20,

$$\begin{aligned} \dot{\boldsymbol{\eta}} &= \dot{\boldsymbol{\eta}}_d + \boldsymbol{K}_P\boldsymbol{e} + \boldsymbol{K}_I \int \boldsymbol{e} + \boldsymbol{K}_D\dot{\boldsymbol{e}} \\ \Rightarrow 0 &= (1 + \boldsymbol{K}_D)\,\dot{\boldsymbol{e}} + \boldsymbol{K}_P\boldsymbol{e} + \boldsymbol{K}_I \int \boldsymbol{e} \end{aligned}$$

In Laplace domain,

$$\begin{aligned} 0 &= (1 + \boldsymbol{K}_D)\,s\boldsymbol{e} + \boldsymbol{K}_P\boldsymbol{e} + \boldsymbol{K}_I\frac{\boldsymbol{e}}{s} \\ \Rightarrow 0 &= (1 + \boldsymbol{K}_D)\,s^2 + (\boldsymbol{K}_P)\,s + \boldsymbol{K}_I \end{aligned}$$

This equation is of the form,

$$0 = s^2 + 2\zeta\omega_0 s + \omega_0^2$$

Using these relationships we can say that,

$$\begin{aligned} \omega_0 &= \sqrt{\frac{\boldsymbol{K}_I}{(1 + \boldsymbol{K}_D)}} \\ \zeta &= \frac{\boldsymbol{K}_P}{2\sqrt{(1 + \boldsymbol{K}_D)\,\boldsymbol{K}_I}} \end{aligned}$$

Assuming critically damped condition ($\zeta = 1$), we get,

$$\boldsymbol{K}_P^2 = 4\,(1 + \boldsymbol{K}_D)\,\boldsymbol{K}_I \tag{4.21}$$

Chapter 5

Simulation

5.1 Simulation Model

First, a trajectory function generator was created. It takes two signals of v and ω, which represent linear velocity and angular velocity respectively, as input and gives sampled points on the trajectory of the path.

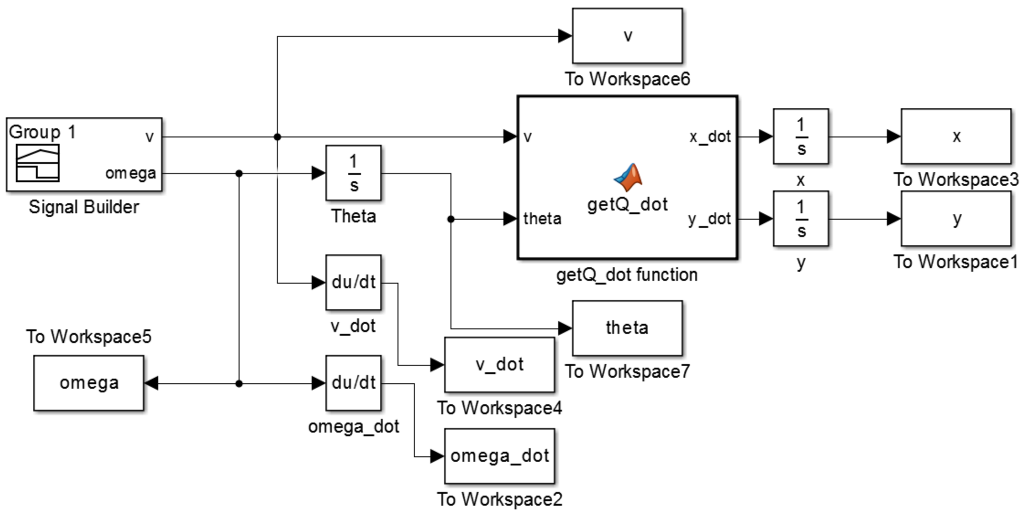

Figure 5.1 Simulink model of the trajectory generator

Then the control loops were created which tells the motors what torque to apply. Then using Forward Dynamics, a robot is simulated to simulate the control action of the controller. To account for wheel slips, a Gaussian noise with an application probability of 25% is added to the torque.

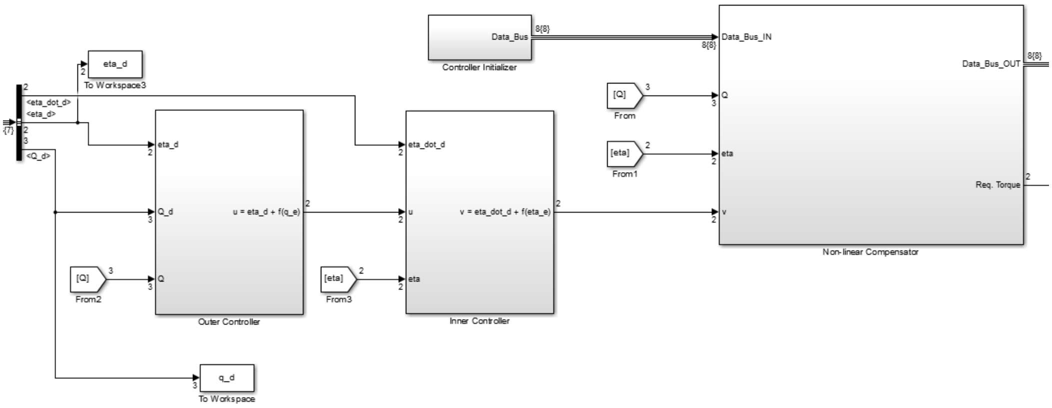

Figure 5.2 Simulink model of the controller with two loops. Outer loop ensures that the positional error is asymptotically zero while inner loop ensures the error in velocity is asymptotically zero. The non-linear compensator gives torque needed to move the robot with given acceleration in the given state.

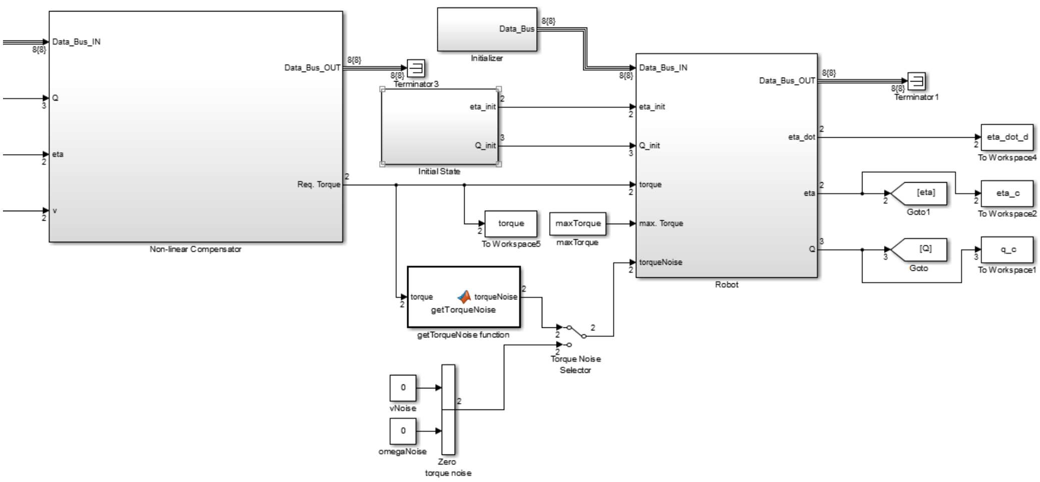

Figure 5.3 Simulink model of the robot showing the noise added to torque as a representative modeling for the wheel slip.

5.2 Parameters used for simulation

Table 5.1 Robot parameters

Robot parameter	Actual Value	Estimate used in controller
H	115 mm	115 mm
L	108 mm	107.8 mm
R	108 mm	107.8 mm
R_L	49 mm	50 mm
R_R	51 mm	50 mm
R_3	52 mm	50 mm
d_{x_c}	26 mm	25.79 mm
d_{y_c}	0 mm	0 mm
m_p	0.8 kg	0.75 kg
m_L	1.2 kg	1.15 kg
m_R	1.2 kg	1.15 kg
m_3	0.6 kg	0.55 kg
I_p	6430 kg mm^2	6428.68 kg mm^2
I_L	350 kg mm^2	351.882 kg mm^2
I_R	350 kg mm^2	351.882 kg mm^2
I_3	655 kg mm^2	653.657 kg mm^2
I_{w_L}	629 kg mm^2	628.378 kg mm^2
I_{w_R}	629 kg mm^2	628.378 kg mm^2
I_{w_3}	629 kg mm^2	628.378 kg mm^2

5.3 Simulation Results

Simulation result for various types of path are discussed in this section. Also, the effect of estimating the robot parameters is characterized in the chapter.

5.3.1 Circular Path

A circular path of radius 30 mm is given to the controller to track. To understand the effect of parameter estimation on the control of the robot, the control parameters of the robot are slightly different from the actual robot parameters which are used to simulate the robot. A noise is added to the torque to simulate slip conditions on the robot.

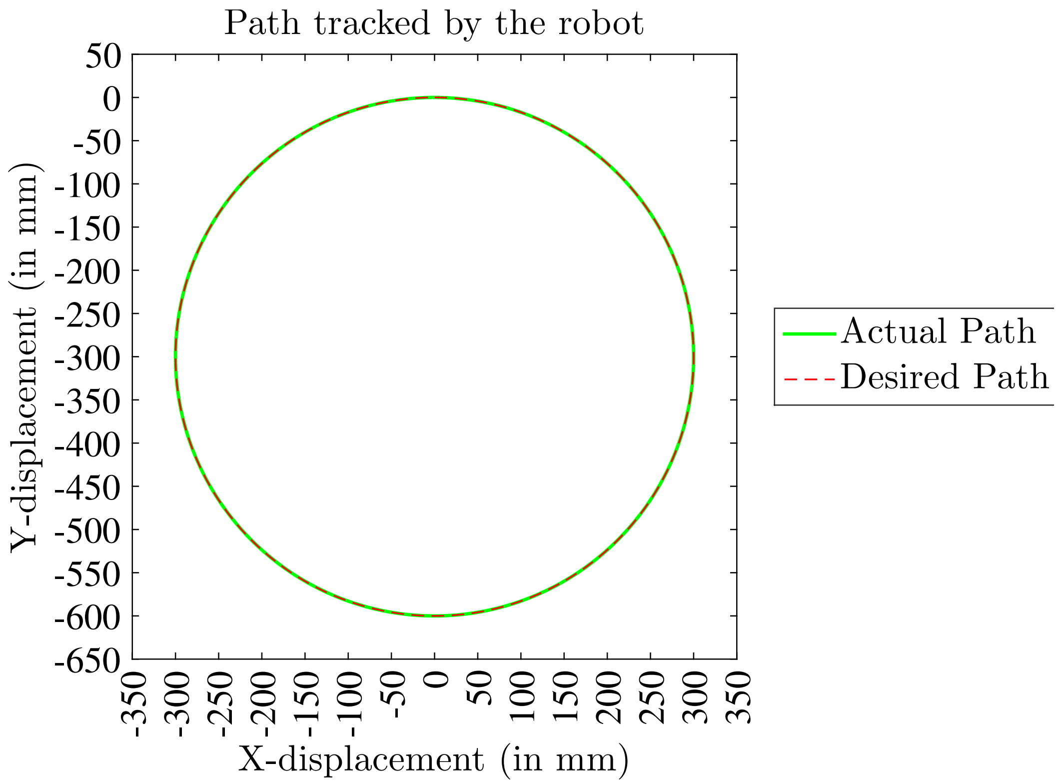

Figure 5.4 Graph of robot tracking a circular path of 300mm

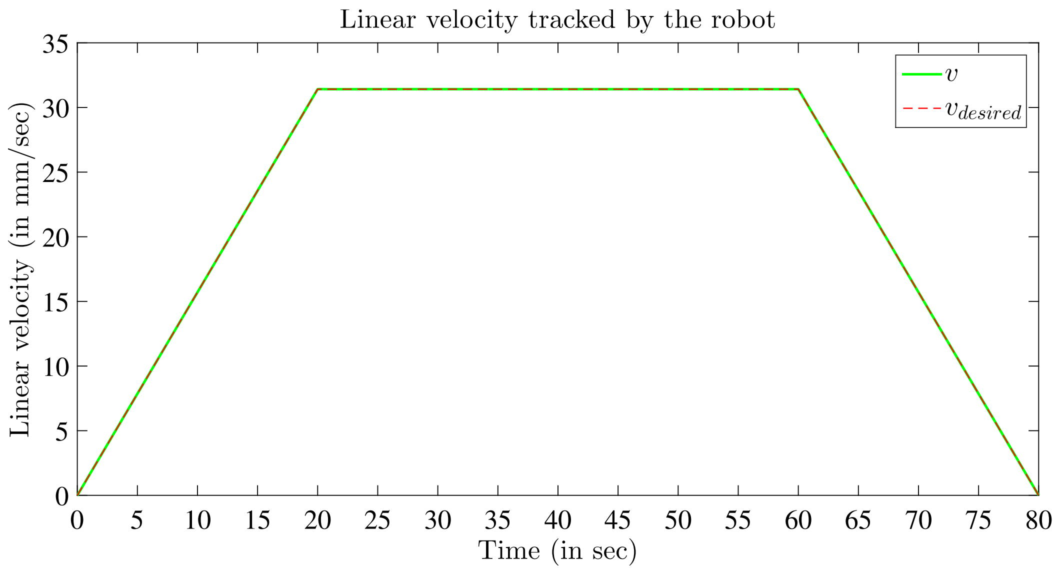

Figure 5.5 Graph of linear velocity of the robot when it is tracking a circular path

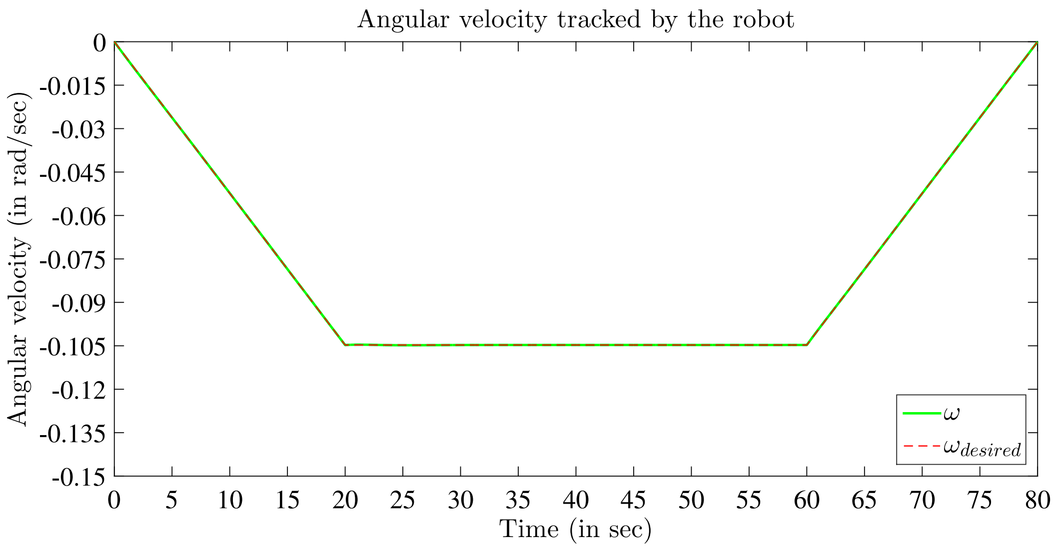

Figure 5.6 Graph of angular velocity of the robot when it is tracking a circular path

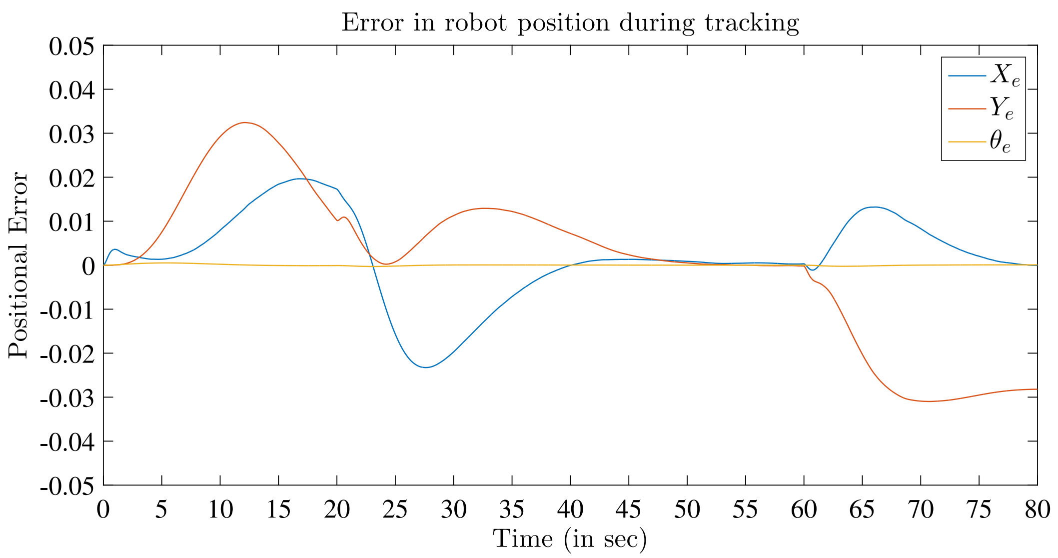

Figure 5.7 Error in the position of the robot when it is tracking a circular path

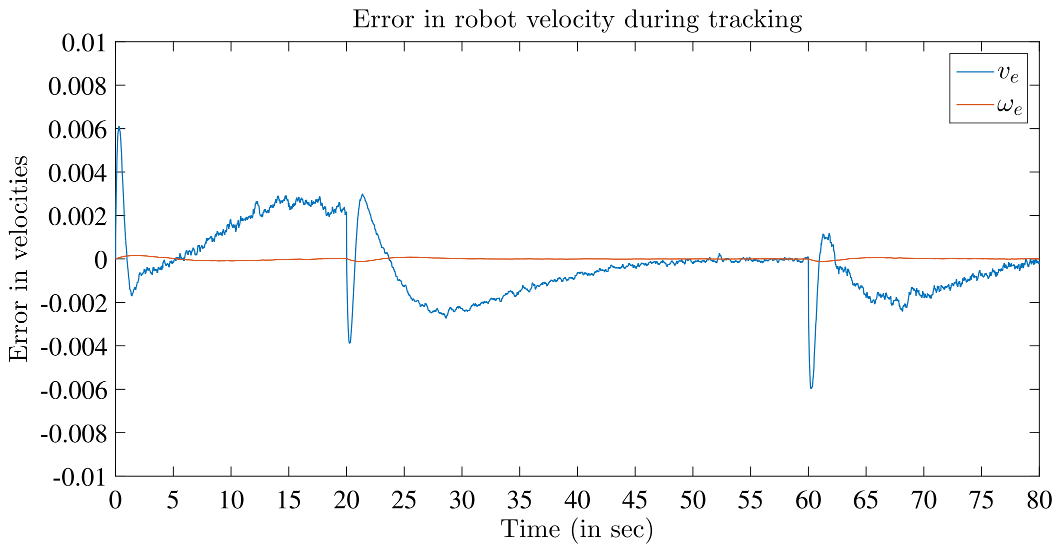

Figure 5.8 Error in the velocities of the robot when it is tracking a circular path

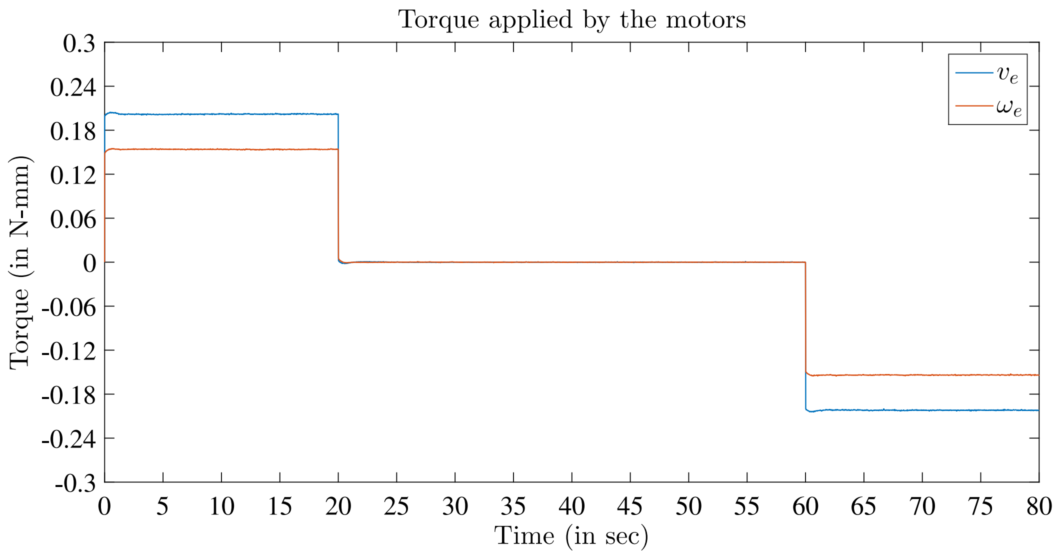

Figure 5.9 Torque applied by each motor of the robot when it is tracking a circular path. Noise added to the torque can be interpreted as slip.

5.3.2 Straight Path

A straight path making an angle of 45° with the x-axis is given to the controller to track. To understand the effect of parameter estimation on the control of the robot, the control parameters of the robot are slightly different from the actual robot parameters which are used to simulate the robot. A noise is added to the torque to simulate slip conditions on the robot.

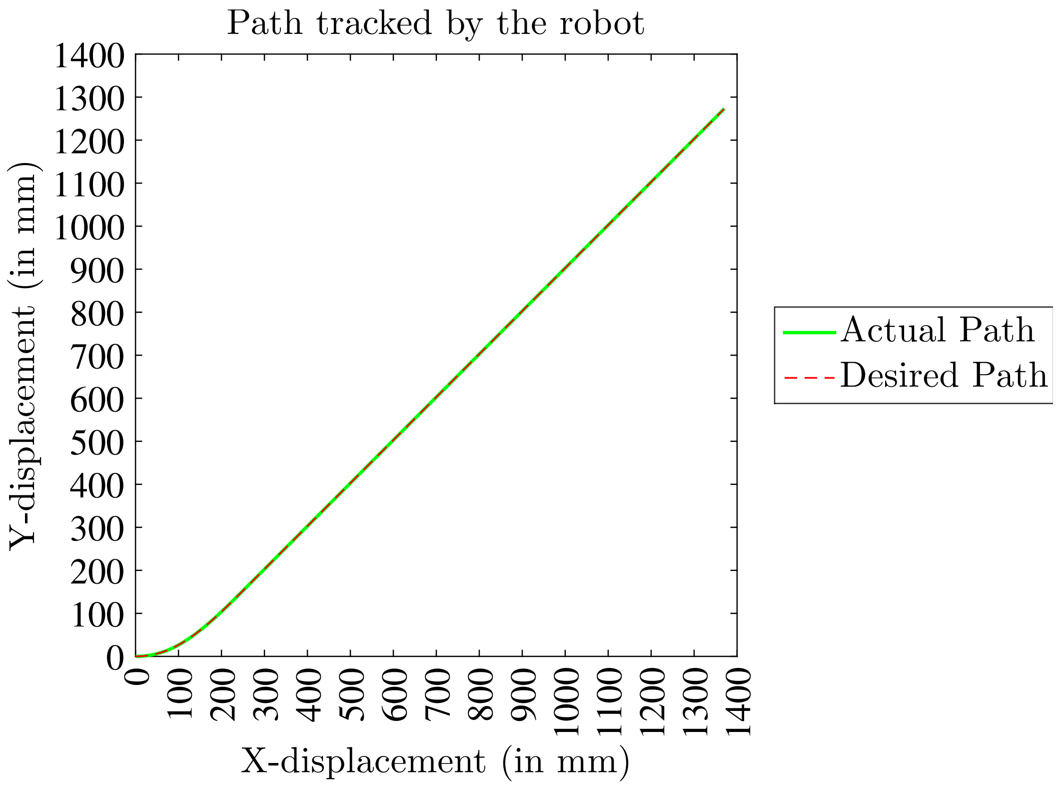

Figure 5.10 Graph of robot tracking a straight path

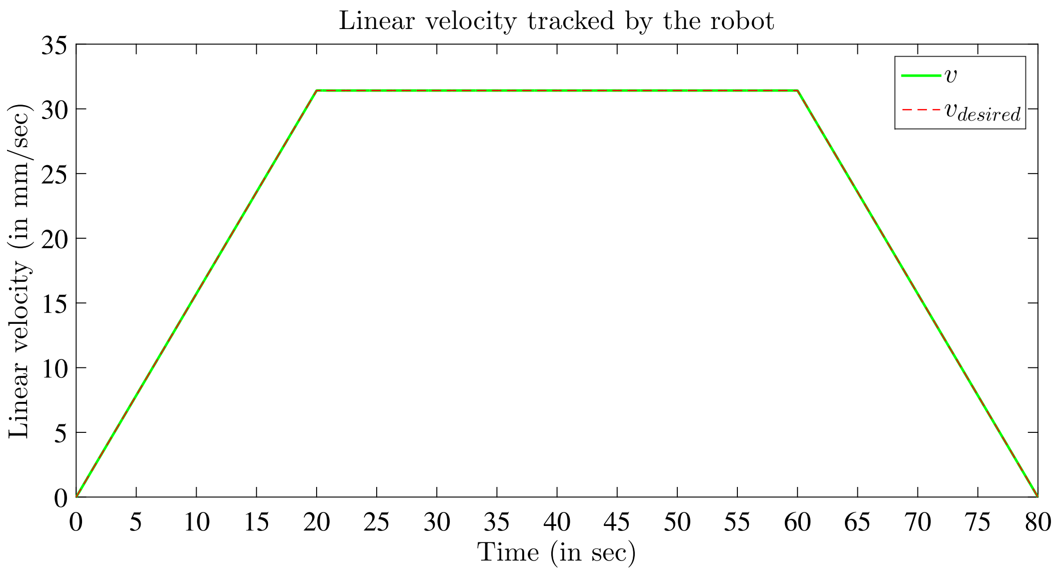

Figure 5.11 Graph of linear velocity of the robot when it is tracking a straight path

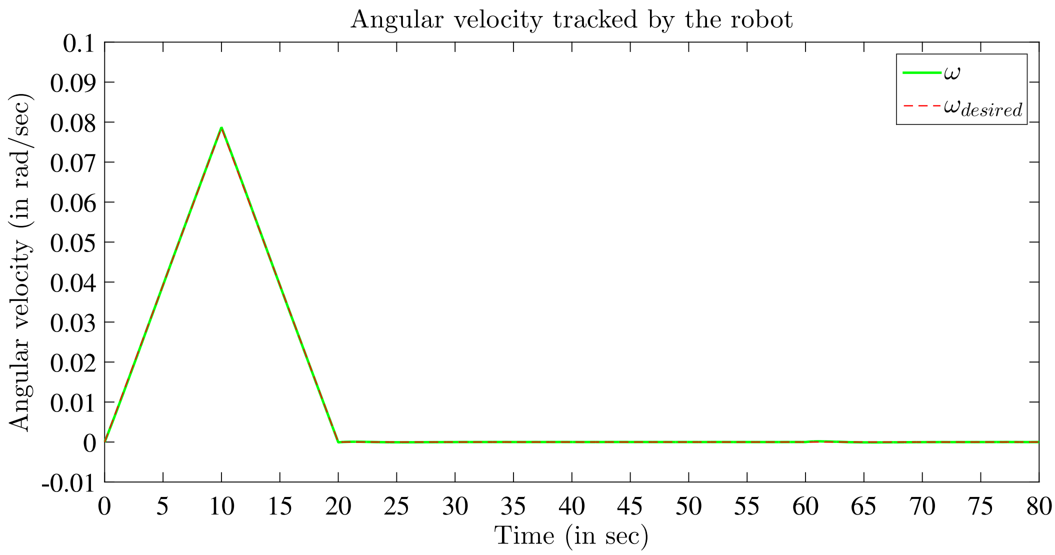

Figure 5.12 Graph of angular velocity of the robot when it is tracking a straight path

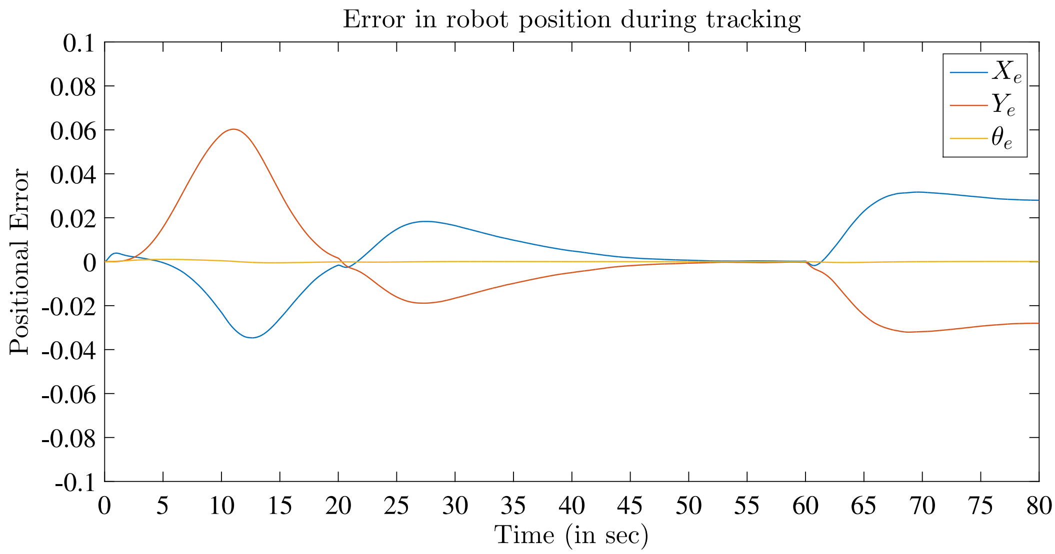

Figure 5.13 Error in the position of the robot when it is tracking a straight path

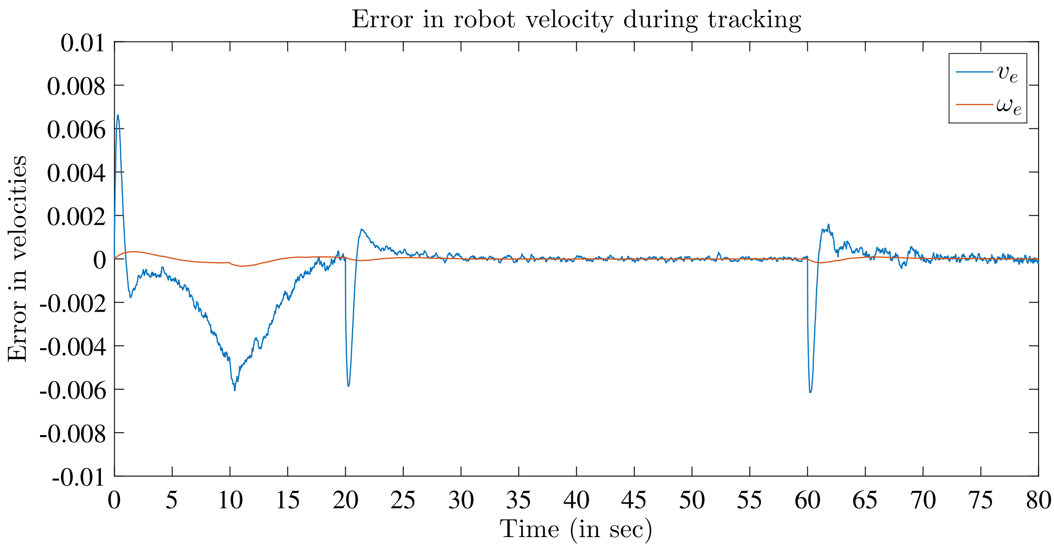

Figure 5.14 Error in the velocities of the robot when it is tracking a straight path

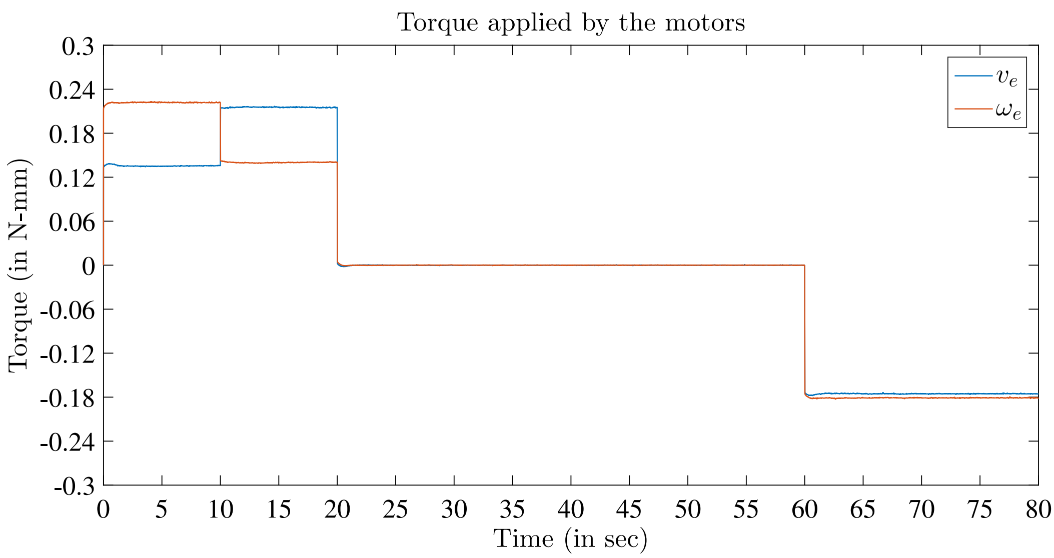

Figure 5.15 Torque applied by each motor of the robot when it is tracking a straight path. Noise added to the torque can be interpreted as slip.

5.3.3 Circular path with an offset mass

Here, we assume that an additional mass of 2 kg (Note: the added mass is about half as heavy as the actual robot) is added to the robot such that its center of mass changes but the controller is not updated to reflect this change.

Table 5.2 Robot parameters with an offset mass

Robot parameter	Actual Value	Estimate used in controller
H	115 mm	115 mm
L	108 mm	107.8 mm
R	108 mm	107.8 mm
R_L	49 mm	50 mm
R_R	51 mm	50 mm
R_3	52 mm	50 mm
d_{x_c}	56 mm	25.79 mm
d_{y_c}	10 mm	0 mm
m_p	2.8 kg	0.75 kg
m_L	1.2 kg	1.15 kg
m_R	1.2 kg	1.15 kg
m_3	0.6 kg	0.55 kg
I_p	6430 kg mm^2	6428.68 kg mm^2
I_L	350 kg mm^2	351.882 kg mm^2
I_R	350 kg mm^2	351.882 kg mm^2
I_3	655 kg mm^2	653.657 kg mm^2
I_{w_L}	629 kg mm^2	628.378 kg mm^2
I_{w_R}	629 kg mm^2	628.378 kg mm^2
I_{w_3}	629 kg mm^2	628.378 kg mm^2

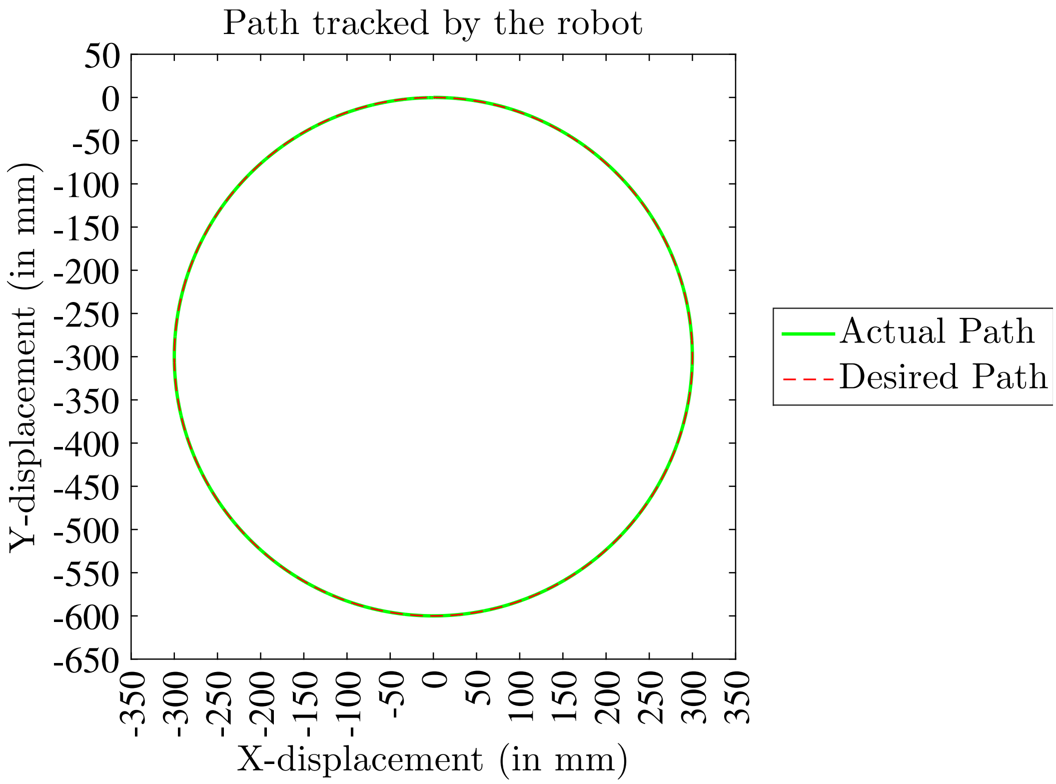

Figure 5.16 Graph of robot with an offset mass tracking a circular path of 300mm

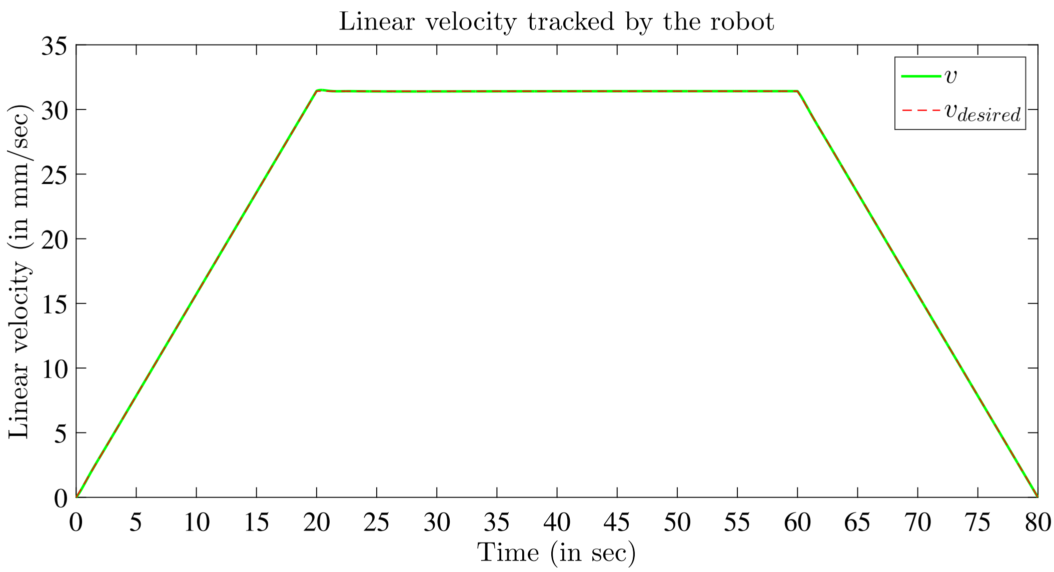

Figure 5.17 Graph of linear velocity of the robot with an offset mass when it is tracking a circular path

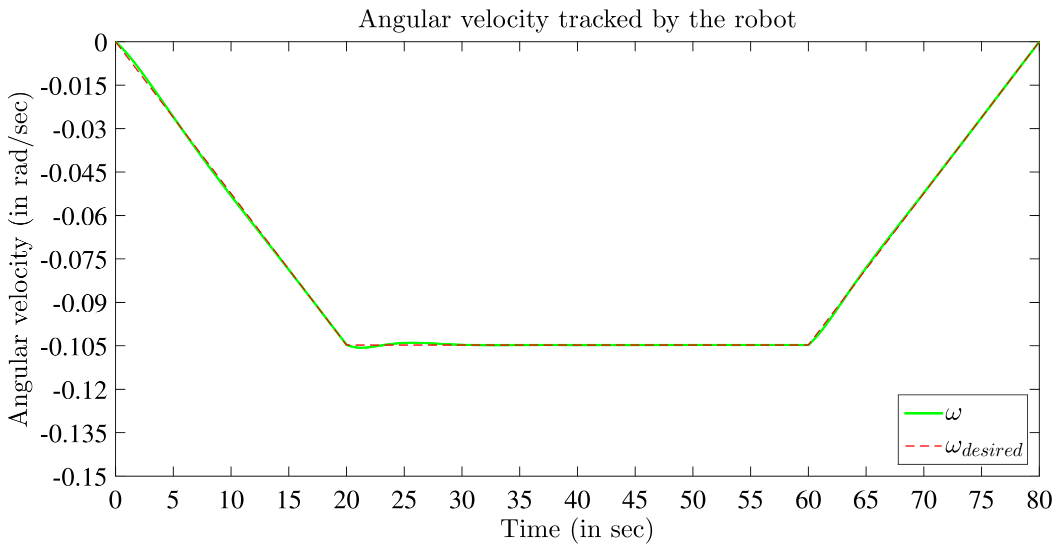

Figure 5.18 Graph of angular velocity of the robot with an offset mass when it is tracking a circular path

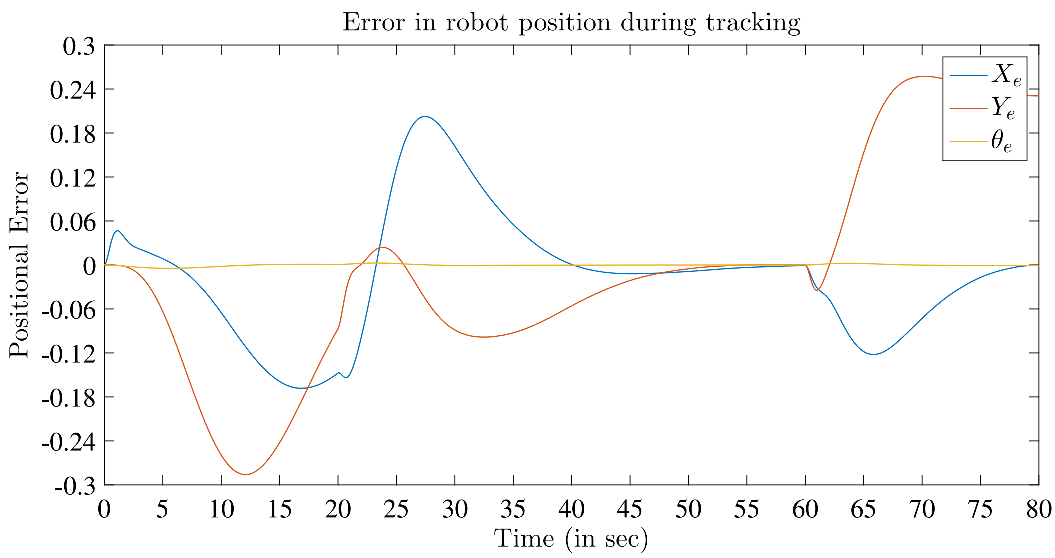

Figure 5.19 Error in the position of the robot with an offset mass when it is tracking a circular path

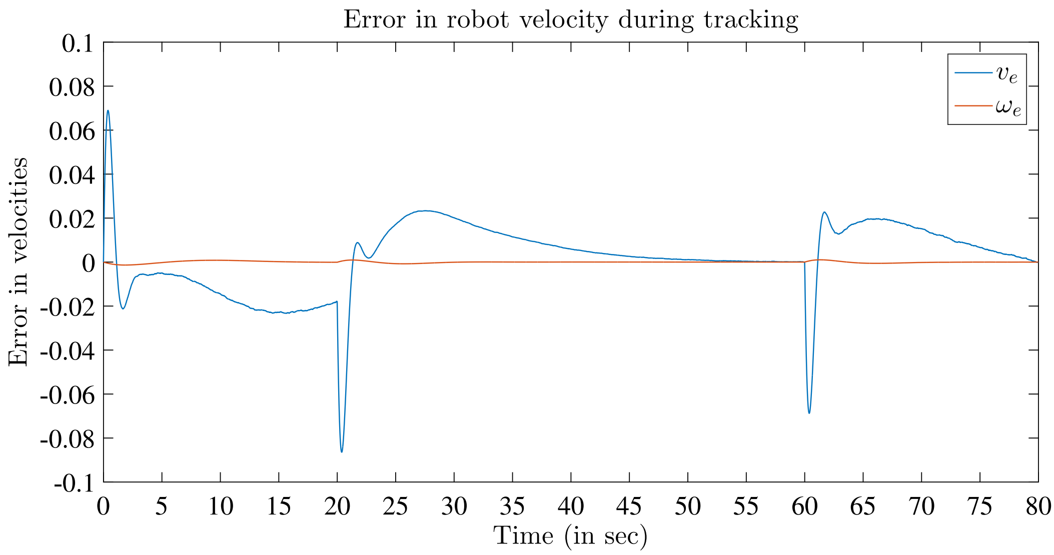

Figure 5.20 Error in the velocities of the robot with an offset mass when it is tracking a circular path

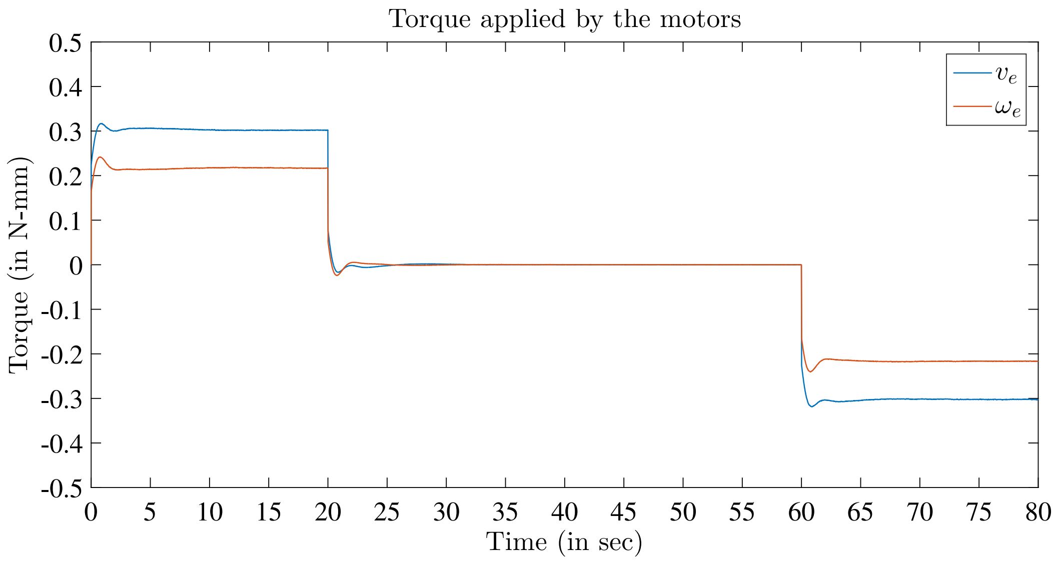

Figure 5.21 Torque applied by each motor of the robot with an offset mass when it is tracking a circular path. Noise added to the torque can be interpreted as slip.

5.3.4 Straight path with offset mass

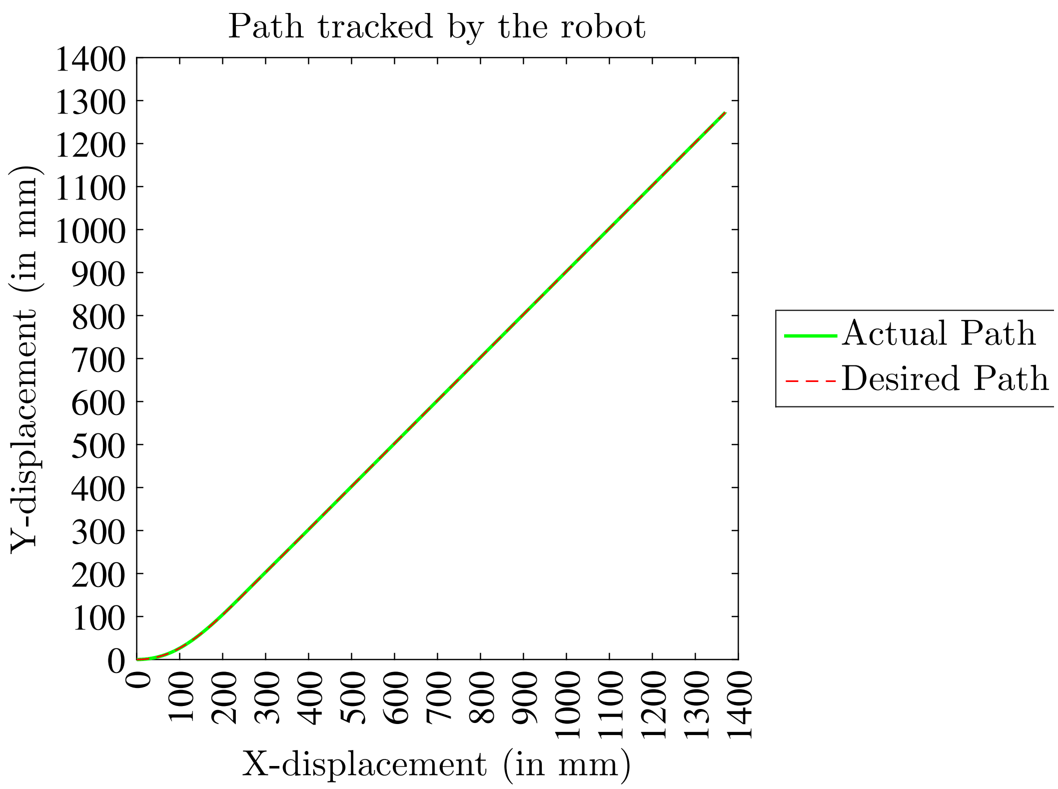

Figure 5.22 Graph of robot with an offset mass tracking a straight path

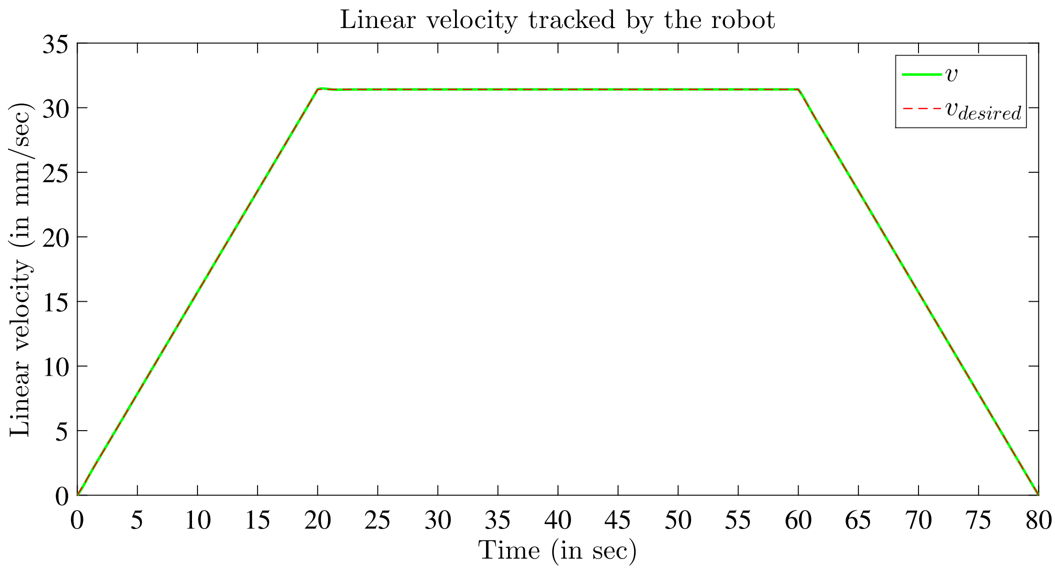

Figure 5.23 Graph of linear velocity of the robot with an offset mass when it is tracking a straight path

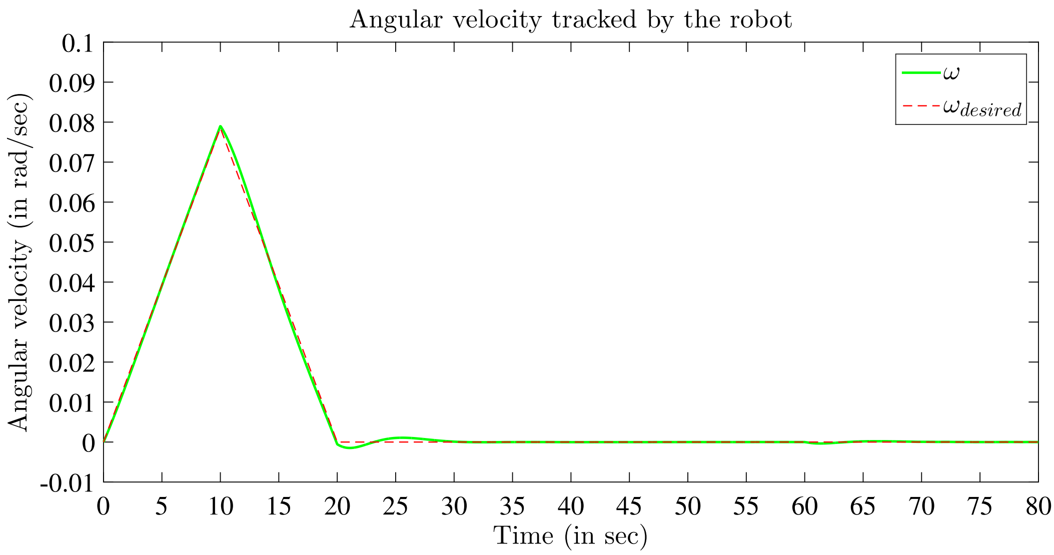

Figure 5.24 Graph of angular velocity of the robot with an offset mass when it is tracking a straight path

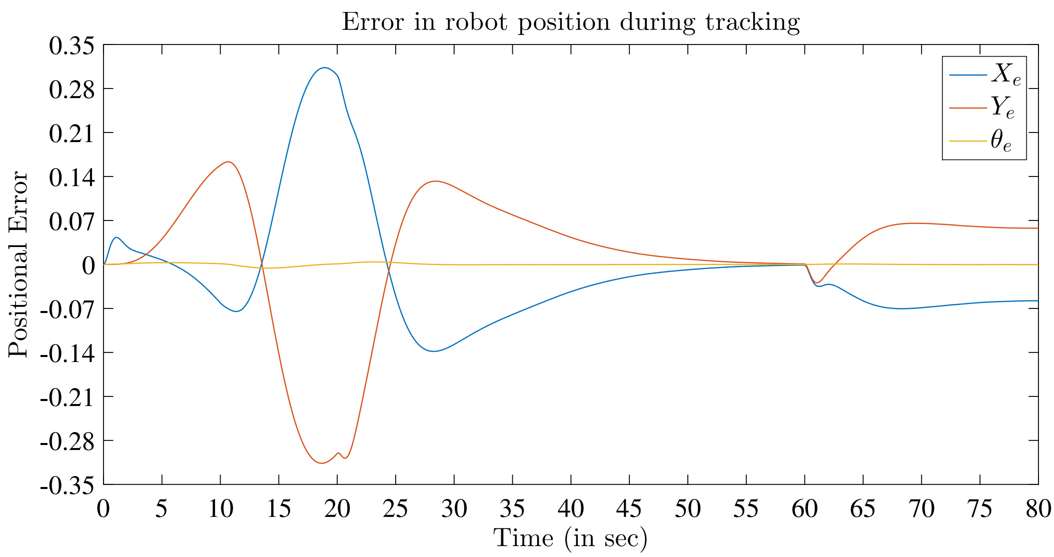

Figure 5.25 Error in the position of the robot with an offset mass when it is tracking a straight path

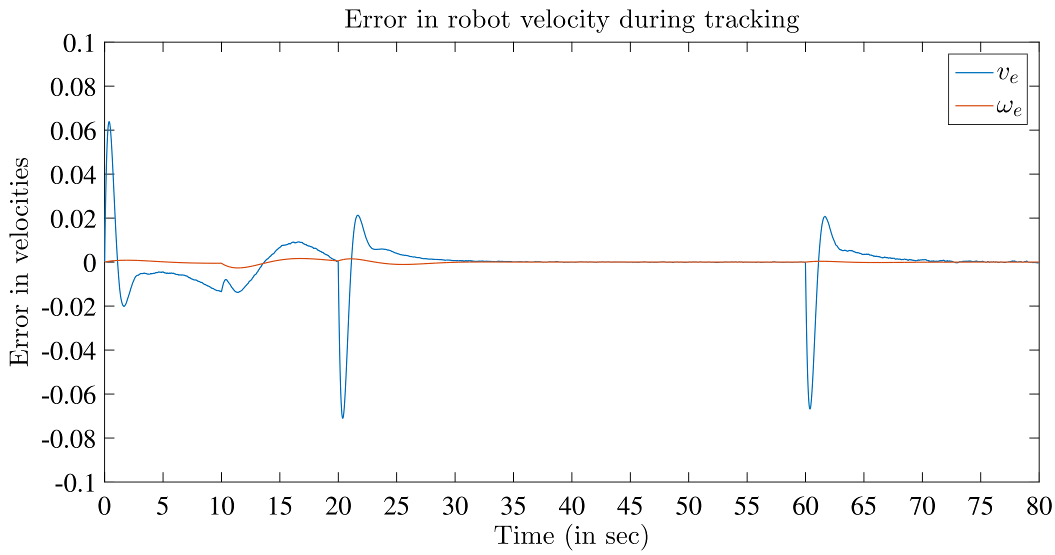

Figure 5.26 Error in the velocities of the robot with an offset mass when it is tracking a straight path

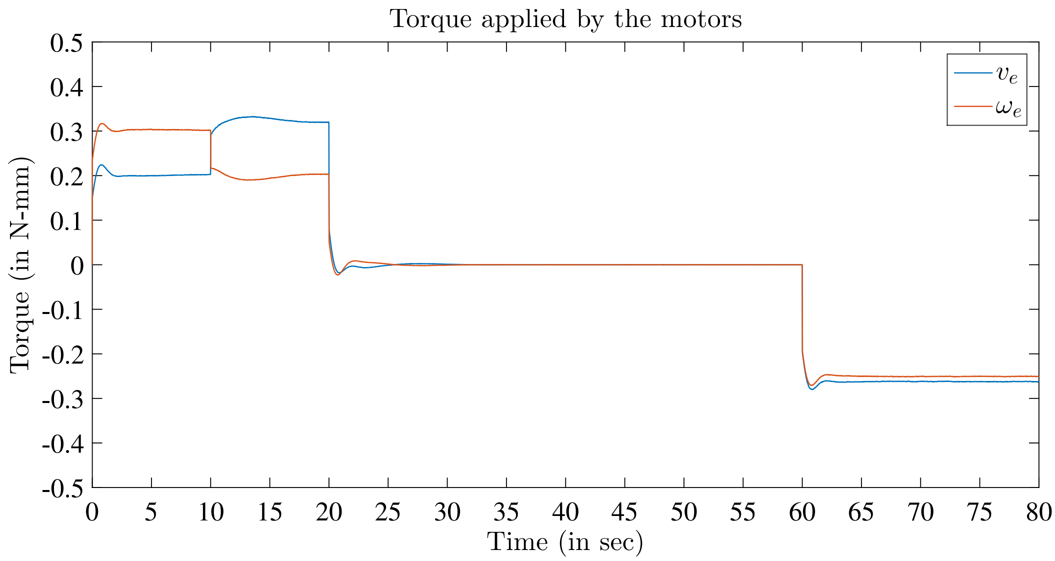

Figure 5.27 Torque applied by each motor of the robot with an offset mass when it is tracking a straight path. Noise added to the torque can be interpreted as slip.

Chapter 6

Conclusion and Future Work

In the project, kinematic and dynamic model for a general differential drive robot is evaluated and validated. A control structure for a differential drive robot is proposed and simulated in the report.

Simulation results shows high accuracy and high precision of the control structure for the tracking problem. Simulations also show high degree of robustness of the robot in circumstances where the estimate of the robot's parameters vary from the actual parameters.

A physical model for the verification of the proposed control structure has been manufactured and is under testing. It would be used to verify the correctness of the assumptions made during the design of the control structure and to serve as a Proof of Concept for the controller.

Bibliography

[1] M. W. Spong, S.Hutchinson, and M. Vidyasagar, "Robot modeling and control", in, 1^{st}. John Wiley & Sons, Inc., 2005, ch. 6, 8, pp. 187–225, 263–279.

[2] J. J. Craig, "Introduction to robotics mechanics and control", in, 3^{rd}. Pearson Education Inc., 2005, ch. 6, 10, pp. 165–200, 290–316.

[3] S. K. Saha, "Introduction to robotics", in, 2^{nd}. Tata McGraw-Hill, 2005, ch. 5–6, pp. 116–208.

[4] A. Kelly, "Mobile robotics: Mathematics, models and methods", in, 1^{st}. Cambridge, 2013, ch. 4–7, pp. 173–513.

[5] N. Chakraborty and A. Ghosal, "Dynamic modeling and simulation of a wheeled mobile robot for traversing uneven terrain without slip", *Journal of Mechanical Design*, vol. 127, no. 5, pp. 901–909, 2004.

[6] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, "Robotics: Modelling, planning and control", in, 1^{st}. Springer, 2009, ch. 7–8, pp. 247–362.

[7] T. Das and I. N. Kar, "Design and implementation of an adaptive fuzzy logic-based controller for wheeled mobile robots", *IEEE Transactions on Control Systems Technology*, vol. 14, no. 3, pp. 501–510, 2006.

[8] R. Fierro and F. L. Lewis, "Control of a non-holonomic mobile robot: Backstepping kinematics into dynamics", *Journal of Robotic Systems*, vol. 14, no. 3, pp. 149–163, 1997.

[9] G. Campion, B. d'Andrea Novel, and G. Bastin, "Modelling and state feedback control of non-holonomic mechanical systems", in *Conference on Decision and Control*, Brighton, England, 1991.

[10] R. C. Coulter, "Implementation of the pure pursuit path tracking algorithm", The Robotics Institute, Carnegie Mellon University, Tech. Rep., 1992.

[11] Y. Kanayama, Y. Kimura, F. Miyazaki, and T. Noguchi, "A stable tracking control method for an autonomous mobile robot", in *IEEE Intnl. Conf. on Robotics and Automation*, Cincinnati, USA, 1990.

Appendix A

Code & CAD

All the source code used in the simulation along with the CAD model of the prototype can be found on GitHub.

https://github.com/guptavaibhav0/differential-drive-controller